NASA Contractor Report 181626

ICASE REPORT NO. 88-14

ICASE

DATA TRAFFIC REDUCTION SCHEMES FOR SPARSE CHOLESKY FACTORIZATIONS

Vijay K. Naik

Merrell L. Patrick

Contract No. NAS1-18107
February 1988

INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING
NASA Langley Research Center, Hampton, Virginia 23665

Operated by the Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

Data Traffic Reduction Schemes for Sparse Cholesky Factorizations

Vijay K. Naik
and
Merrell L. Patrick

Institute for Computer Applications in Science & Engineering
NASA Langley Research Center, Hampton, VA 23665

and

Computer Science Dept., Duke University
Durham, NC 27706.

ABSTRACT

Load distribution schemes are presented which minimize the total data traffic in the Cholesky factorization of dense and sparse, symmetric, positive definite matrices on multiprocessor systems with local and shared memory. The total data traffic in factoring an n x n sparse, symmetric, positive definite matrix representing an n-vertex regular 2-D grid graph using n^{α}, $\alpha \leq 1$, processors is shown to be $O(n^{1+\frac{\alpha}{2}})$. It is $O(n^{\frac{3}{2}})$, when n^{α}, $\alpha \geq 1$, processors are used. Under the conditions of uniform load distribution these results are shown to be asymptotically optimal. The schemes allow efficient use of up to $O(n)$ processors before the total data traffic reaches the maximum value of $O(n^{\frac{3}{2}})$. The partitioning employed within the scheme, allows a better utilization of the data accessed from shared memory than those of previously published methods.

Research supported by the National Acronautics and Space Administration under NASA contract No. NAS1-18107 while the authors were in residence at ICASE, NASA Langley Research Center, Hampton, VA 23665.

1. Introduction

Consider the problem of solving a system of linear equations

$$Ax = b$$

where A is an n x n sparse, symmetric, positive definite matrix, x is an n x 1 vector of variables, and b is an n x 1 vector of constants on a multiprocessor system with both shared and local memory. In this paper the total data traffic involved in factoring the matrix A using the column version of Cholesky decomposition is analyzed. The analysis is restricted to systems for 2-D regular grid graphs. The results developed here, however, can be applied to a wider class of problems.[1]

In [2], George et al. have given a parallel sparse factorization scheme for local memory multiprocessor systems that has a total data traffic of $O(n^{1+\alpha}\log_2 n)$ using n^{α} processors. This result was improved to $O(n^{1+\alpha})$ in [4]. In this paper a scheme that has a total data traffic of $O(n^{1+\frac{\alpha}{2}})$ using n^{α}, $\alpha \leq 1$, processors is presented; with n^{α}, $\alpha \geq 1$, processors the resulting data traffic is $O(n^{\frac{3}{2}})$. Although a multiprocessor system with both shared and local memory is assumed as the model of computation, the results developed here are equally applicable to systems with only local memories.

In Section 2, some background work is discussed and the model of computation is specified. In Section 3, expressions for the total data traffic required in factoring dense matrices are developed. Using these results the total data traffic for sparse systems is analyzed in Section 4. Conclusions are given in the final section.

---

[1] V. K. Naik and M. L. Patrick, *Communication requirements of parallel sparse Cholesky factorizations*, in preparation.

2. Some Preliminaries

The model of computation used in our analysis is that of a multiprocessor system with a two level memory hierarchy such that each processor has local memory and all processors have access to a shared memory. The access cost per nonzero element in the shared memory is assumed to be unity for any processor. The total number of shared memory accesses from the beginning to the end of an algorithm is defined as the *communication requirement* or *data traffic* of that algorithm implemented on the multiprocessor system.

The $n \times n$ matrix A considered here is assumed to represent a system corresponding to a 2-D regular grid graph, the vertices of which are ordered using the nested dissection method [3]. Figure 1 shows an ordering scheme for a 7 x 7 grid.

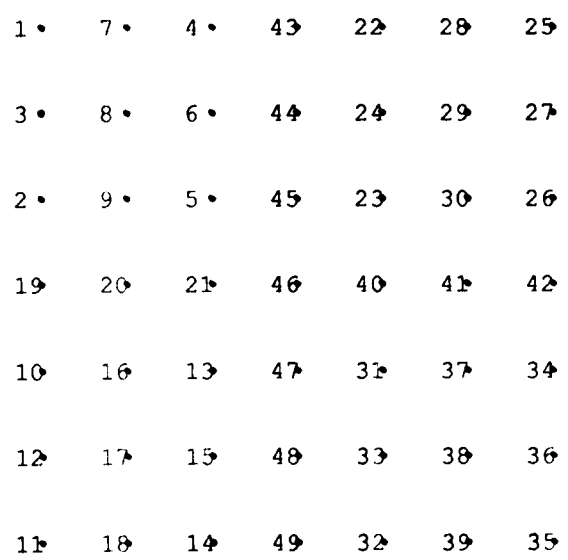

Figure 1: A 7 x 7 grid with nested dissection ordering.

Such an ordering has an optimal sequential operation count and fill-in.

The basic algebraic scheme used in this paper to factor A is the column version of the Cholesky decomposition method [6]. An outline of this algorithm for factoring the n x n matrix A is as follows. Let $A = LL^T$; $a_{ij} \in A$ and $l_{ij} \in L$.

```
for j = 1 until n do
    begin
        Initialize l_ij = a_ij,  i = j, ··· ,n.
        for k = 1 until j–1 do
            for i = j until n do
                l_ij = l_ij – l_ik * l_jk ;
        l_jj = √l_jj ;
        for k = j+1 until n do
            l_kj = l_kj / l_jj;
    end
```

In the above algorithm for clarity, the values of l_{ij} are shown separately from those of a_{ij}. In practice l_{ij} may overwrite a_{ij}.

In the next two sections, parallel schemes for the above algorithm are developed for the distribution of work among processors of the multiprocessor system and their communication requirements are analyzed. The analysis assumes some familiarity with graph theory concepts related to matrix representations of systems of equations, in particular, the notion of vertices, edges, separators, subgraphs of a graph, and the correspondences between the vertices and the rows and columns of the matrix, the edges and the nonzero elements, and addition of edges and fill-in during the factorization of A. For details on these concepts see [3] and [7] and the references therein.

Finally, the standard notations "O", "o", and "Ω" are used to characterize the asymptotic growth rates of computation and communication requirements. Precise

definitions of these notations can be found in elementary textbooks on the analysis of algorithms, e.g. [9].

3. Communication Requirements of a SPOCC Factorization Scheme for Dense Matrices

First, a scheme based on the column version of the Cholesky decomposition [6] for factoring the lower triangular part of an m x m symmetric, positive definite, dense matrix using p processors is described. Without loss of generality, further assume that $p = \frac{1}{2}(r^2 + r)$ where r is an integer. The lower triangular matrix is partitioned into p partitions such that all, except r partitions, are s x s square blocks, where $s = \frac{m}{r}$. The remaining r partitions which lie on the diagonal of the matrix are s x s triangular blocks. Each of these partitions is assigned to a unique processor. Initially, each processor reads the data for its partition from shared memory into its local memory. The r processors in charge of the partitions containing the left most s x s blocks of the matrix commence the computations of their part of the factorization. As soon as an element of the factor is computed, it is written into shared memory for access by other processors. As the necessary data becomes available, the remaining processors initiate computations on the blocks assigned to them. Each processor accesses from shared memory only those elements that are needed for local computation and each element is read no more than once by any processor. This parallel factorization scheme is termed as the *submatrix-partition oriented column Cholesky* (SPOCC) factorization scheme.

Next the communication requirements of the above scheme are analyzed. In the following two theorems it shown that the total data traffic involved is $O(m^2 \sqrt{p})$ and that

this result is asymptotically optimal. First consider the data traffic associated with a generic square block I shown in Figure 2. In that figure, the darkened area represents the data elements that are required for the computations in block I. Let block I be bounded by columns $(i-1)s + 1$ and $i \cdot s$, and by rows $(j-1)s + 1$ and $j \cdot s$ where $1 \leq i \leq r$ and $1 \leq j \leq r$. The following lemma provides bounds for the communication cost of block I.

Let $a_{k,l}$ be any element in block I. During the factorization this element undergoes a sequence of modifications before it arrives at a final value. The computations leading to the final value of this element require the final resultant values of certain other elements and its own initial value. For determining the communication requirements the actual value, either final or initial, of an element accessed is not important. Only an account of this access to the shared memory is necessary. To keep the discussion brief, in the analysis of the data traffic, the term appropriate or required value of an

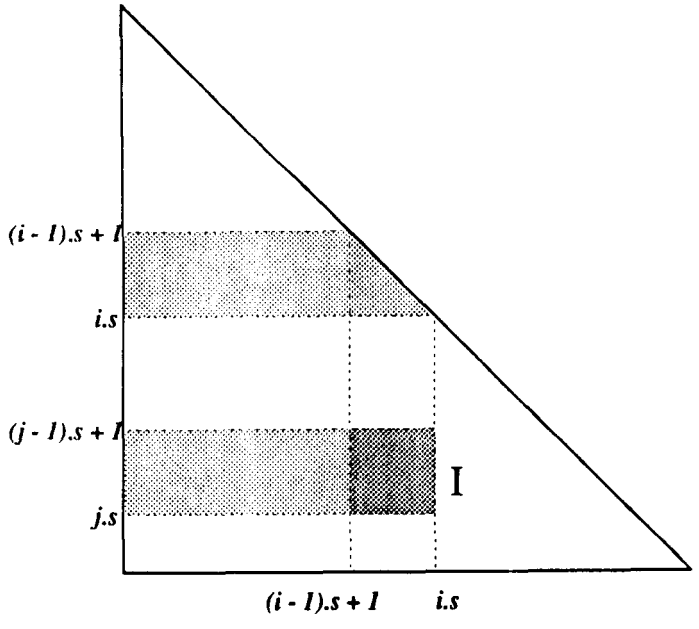

Figure 2: The data traffic associated with block I.

element is used. An *appropriate* or *required value* of an element at a particular step of the computations is defined to be the value of that element that is required by the data dependencies of the algorithm at that step of computation. In the proof given below, $a_{l,*}$ represents all the elements in row l of the lower triangular part of the matrix under consideration.

LEMMA 1 : *A total data traffic of* $(2i - \frac{1}{2})\cdot s^2 + \frac{1}{2}\cdot s$ *is necessary and sufficient for factoring the square block I; it is the same for all square blocks bounded by the columns* $(i-1)s + 1$ *and* $i\cdot s$. *The data traffic associated with a* s *x* s *triangular diagonal block bounded by the columns* $(i-1)s + 1$ *and* $i\cdot s$, *is* $(i - \frac{1}{2})\cdot s^2 + \frac{1}{2}\cdot s$.

PROOF: Consider the computations at any element $a_{k,l}$ in block I. For these computations the appropriate values of the elements $a_{l,*}$ and $a_{k,l'}$ for all $l' \le l$ are required; i.e., the data required are all the elements in row l and all the elements in row k on and to the left of column l. Now $a_{k,l}$ is any element in block I and hence, $(j-1)s + 1 \le k \le j\cdot s$ and $(i-1)s + 1 \le l \le i\cdot s$. Thus, the computations corresponding to any element in block I depend on all the elements in some row l of the factor, $(i-1)s + 1 \le l \le i\cdot s$, and on the elements to the left of column $i\cdot s+1$ in some row k, $(j-1)s + 1 \le k \le j\cdot s$. Furthermore, no other information is needed in performing the factorization at that element. Therefore all the information in the above mentioned rows is sufficient for performing the computations at all the elements of block I. Conversely, the block I contains all the elements $a_{k,l}$ such that $(j-1)s + 1 \le k \le j\cdot s$ and $(i-1)s + 1 \le l \le i\cdot s$. The completion of the computations at these elements requires all the elements in rows l of the factor such that $(i-1)s + 1 \le l \le i\cdot s$ and all the elements to the left of column $i\cdot s+1$ in all the rows k such

that $(j-1)s + 1 \leq k \leq j \cdot s$. Thus, it is necessary as well as sufficient to read the values of these specific elements for completing the factorization of the square block I. A count of the number of such elements gives a bound on the data traffic involved in factoring block I. It should be noted that this count includes the elements in block I which are locally computed, but the initial value of each of these elements has to be read prior to performing computations at that point and thus the communication cost remains the same.

Thus, the data traffic that is both necessary and sufficient to factor block I is

$$= 2(i-1) \cdot s^2 + \frac{3}{2} \cdot s^2 + \frac{1}{2} \cdot s$$

$$= (2i - \frac{1}{2}) \cdot s^2 + \frac{1}{2} \cdot s.$$

The above bound on the data traffic is independent of j and so it is the same for all the square blocks within the columns $(i-1)s + 1$ and $i \cdot s$.

Similarly, we can show that the data traffic associated with a s x s triangular diagonal block bounded by the columns $(i-1)s + 1$ and $i \cdot s$ is, $(i - \frac{1}{2}) \cdot s^2 + \frac{1}{2} \cdot s$.

□

Using these results we get a bound on the total data traffic as stated in the following Theorem.

THEOREM 1 : *The total data traffic involved in the SPOCC factorization of an m x m dense matrix using p processors is* $O(m^2 \sqrt{p})$.

PROOF: The total data traffic associated with all the blocks bounded by columns $(i-1)s + 1$ and $i \cdot s$, $1 \le i \le r$, is given by,

$(r-i)$ x *data traffic associated with a square block*

\+ *data traffic associated with a triangular block* bounded by the given columns.

From Lemma 1 and the fact that there are r such column partitions, we get the total data traffic involved in factoring the m x m dense matrix using the SPOCC factorization as

$$\sum_{i=1}^{r}\left[(r-i)((2i - \frac{1}{2})\cdot s^2 + \frac{1}{2}\cdot s) + ((i - \frac{1}{2})\cdot s^2 + \frac{1}{2}\cdot s)\right].$$

Ignoring the lower order terms, the total data traffic is

$$= \sum_{i=1}^{r}(2r\cdot i - 2i^2)\cdot s^2$$

$$\le \frac{1}{3}r^3\cdot s^2.$$

Since $p = \frac{1}{2}(r^2 + r)$ and $s = \frac{m}{r}$, the total data traffic is $O(m^2\sqrt{p})$.

□

The above theorem gives an upper bound on the data traffic involved in factoring the m x m matrix using the submatrix assignment scheme. In the following theorem we prove that this result is optimal in the order of magnitude sense. Before giving the proof, we again consider the data dependencies involved in computing an element of the factor. Consider the computations at any element $a_{i,j}$. To compute the value of the corresponding element in the factor, values at all the elements in row j and the values of elements in column 1 through j of row i are needed. Thus, any off-diagonal ele-

ment $a_{i,j}$ requires values of $2{\cdot}j$ elements for completing the computations and any diagonal element (when $i = j$) requires values of j elements. We now have the following two obvious observations:

(1) The values at all the elements in any row i can be used to complete the computations at exactly one element, namely, the diagonal element $a_{i,i}$.

(2) If i and j are any two rows, such that $i > j$, then the values of the elements in these two rows can be used to complete computations at exactly one off-diagonal element $a_{i,j}$. No other information is needed to complete the computations at that element.

These observations lead to the following lemma on the number of elements at which the computations can be completed by acquiring information from exactly k rows.

LEMMA 2 : *If the required values of all the elements in any k rows are available then the computations of at most* $\frac{k^2}{2} + \frac{k}{2}$ *elements can be completed; conversely, the computations corresponding to any* $\frac{k^2}{2} + \frac{k}{2}$ *elements depend on at least one element from each row of at least k distinct rows.*

PROOF: We prove the first part of the lemma by induction on the number of rows.

When $k = 1$, $\frac{k^2}{2} + \frac{k}{2} = 1$, which is true from Observation (1). This gives the basis for the inductive proof. Now assume that the result is true for k–1 rows; i.e., assume that with the values of all elements in k–1 rows computations of at most $\frac{(k-1)^2}{2} + \frac{(k-1)}{2}$ elements can be completed. In addition to the values of all the elements in the k–1 rows, let the values of all the elements in the kth row be known. For the kth row there is a

unique diagonal element, whose value can be computed using only the information from that row (Observation (1)). From Observation (2), using the information from each of the previous k–1 rows and the kth row, computations at one unique new element can be completed. Thus, by knowing the appropriate values at all the elements in the kth row, computations of at most k extra elements can be completed; i.e., the values of all elements in the k rows can be used to complete computations of at most $\frac{(k-1)^2}{2} + \frac{(k-1)}{2} + k = \frac{k^2}{2} + \frac{k}{2}$ elements. This completes the inductive proof of the first part of the lemma.

To prove the second part, assume that there is a set of $\frac{k^2}{2} + \frac{k}{2}$ elements such that all the computations at these elements depend on values from less that k rows. Since the computations at any element depends on its own initial value, it is obvious that for the above assertion to hold, these $\frac{k^2}{2} + \frac{k}{2}$ elements must lie on less than k rows. Assume that $k - \varepsilon$ rows hold these elements and that information from $k - \varepsilon'$ rows is sufficient to complete the computations at these elements, for $\varepsilon \geq \varepsilon' \geq 1$. From the first part of the lemma, all the information from any $k - \varepsilon'$ rows can be used to complete computations of at most $\frac{(k-\varepsilon')^2}{2} + \frac{(k-\varepsilon')}{2}$ elements. This implies that there are at least $\varepsilon'\cdot(k - \frac{\varepsilon'}{2} + \frac{1}{2}) > 0$ elements in $k - \varepsilon$ rows where the computations cannot be completed using the information in the $k - \varepsilon'$ rows - a contradiction. Therefore information from at least k rows is needed in order to complete the computations at $\frac{k^2}{2} + \frac{k}{2}$ elements.

□

Next the above described scheme with a total data traffic of $O(m^2 \sqrt{p})$ in factoring the m x m matrix using p processors is shown to be optimal in its communication requirements.

THEOREM 2 : *Assuming that the computational load is to be uniformly distributed among p processors, the data traffic involved in computing the Cholesky factor of an m x m dense matrix is* $\Omega(m^2 \sqrt{p})$.

PROOF: Consider the computations corresponding to the elements in the set $S = \{a_{ij} \mid i, j > \frac{m}{2}\}$. In Figure 3 the gray area *FGH* denotes this set of elements. The total computational work in factoring the m x m matrix is $\frac{m^3}{6} + o(m^3)$ and that in factoring the part corresponding to area *FGH* is $\frac{m^3}{12} + o(m^3)$. This implies that out of p pro-

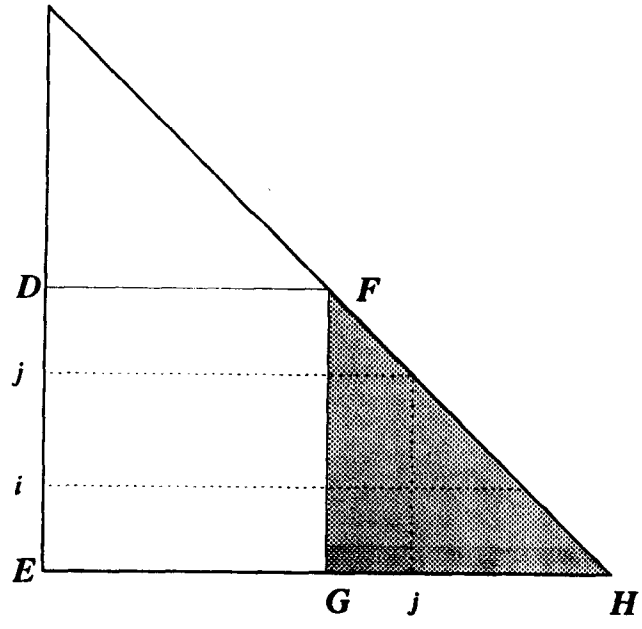

Figure 3.

cessors, $c_1 \cdot p$ processors must be assigned to complete the factorization of area *FGH*, for some constant $0 < c_1 < 1$, to achieve a uniform load distribution of $c_2 \cdot \frac{m^3}{p} + o(\frac{m^3}{p})$ work per processor.

Let Π be some partitioning of elements, that partitions the elements in the set S in $c_1 \cdot p$ subsets such that each subset has the same amount of computational work. It can be shown that for any such partitioning Π there are constants $c', c'' > 0$, such that the number of elements in any subset are in the interval $\left[c' \cdot \frac{m^2}{p},\ c'' \cdot \frac{m^2}{p}\right]$. Ignoring the lower order terms each subset defined by the partitioning Π has $c_2 \cdot \frac{m^3}{p}$ amount of work. Now the element $a_{m,m}$ in the set S has the maximum amount of computational work associated with it, which is proportional to m since $a_{m,m}$ is modified m times. Therefore, the number of elements in any subset must be greater than or equal to $c' \cdot \frac{m^2}{p}$, for some constant c'. Similarly, for any element $a_{*,(\frac{m}{2}+1)}$, i.e., an element in the column $\frac{m}{2} + 1$, the computational work is the minimum over the set S. This minimum computational work is proportional to $\frac{m}{2}$. Hence the number of elements in any subset defined by Π must be less than or equal to $c'' \cdot \frac{m^2}{p}$ for some constant c''.

Let $c_i \cdot \frac{m^2}{p}$, $c' \le c_i \le c''$, be the number of elements in the ith partition defined by Π. From Lemma 2, computations at any $c_i \cdot \frac{m^2}{p}$ elements depend on information from at least $\left\lceil \sqrt{2c_i} \cdot \frac{m}{\sqrt{p}} - \frac{1}{2} \right\rceil \approx c_i' \cdot \frac{m}{\sqrt{p}}$ rows where c_i' is a constant. Now for any element $a_{i,j} \in S$, j is greater than $\frac{m}{2}$. Hence the values of at least $\frac{m}{2}$ elements from each of the $c_i' \cdot \frac{m}{\sqrt{p}}$

rows is required in completing the computation of any subset of elements defined by Π. Thus the data traffic associated with each partition is at least $\frac{c_i'}{2} \cdot \frac{m^2}{\sqrt{p}}$. Since there are $c_1 \cdot p$ such partitions, the data traffic involved is at least $c_1 \cdot \frac{c_i'}{2} \cdot m^2 \cdot \sqrt{p}$.

Therefore under the condition of uniform load distribution, the data traffic involved in computing the Cholesky factor of an m x m dense matrix using p processors is $\Omega(m^2 \sqrt{p})$.

□

It should be noted that, in an order of magnitude sense, the computational work associated with each block in the SPOCC factorization scheme presented earlier is the same as that of a partition under the uniform load distribution scheme. Thus the data traffic associated with that scheme is optimal in an order of magnitude sense.

4. Communication Requirements of a SPOCC Factorization Scheme for Sparse Matrices

In this section the total data traffic of a SPOCC factorization scheme in factoring a sparse, symmetric, positive definite matrix A corresponding to a regular n-vertex 2-D grid graph is analyzed. A 9-point stencil, unless otherwise stated, is used. In the following discussion L denotes the lower triangular Cholesky factor of the matrix A. First, the worst case communication requirement complexity result of $O(n^{\frac{3}{2}})$, which is independent of the number of processors used, is obtained and then a scheme that reduces the communication requirement complexity to $O(n^{1+\frac{\alpha}{2}})$, when n^{α}, $\alpha \leq 1$,

processors are used, is presented. The result is shown to be optimal under the condition of uniform load distribution.

Clearly, the communication requirement is the worst when the use of local memory is not allowed. Now consider the computations involved in computing a nonzero element $l_{ij} \in L$. Recall that the expression $a_{ij} - \sum_{k=1}^{j-1} l_{i,k} \cdot l_{j,k}$ is computed first, followed by a single division by l_{jj}. Thus, for each multiplication there is one subtraction operation, at most one division and three memory references and a constant overhead such as index computation. Let the values of all the elements of the lower triangular part of matrix A and those of L be stored in the shared memory. Any number of processors are allowed to participate in computing a nonzero element of the factor provided that no single operation is performed by more than one processor. The following theorem gives the bound on the total data traffic under these assumptions. Although the result is obvious, it is useful because it is independent of the number of processors used and gives the worst case bound on the data traffic even for the models of computation that are more restrictive.

THEOREM 3 : *The worst case data traffic associated with factoring the matrix A is* $O(n^{\frac{3}{2}})$.

PROOF: For the assumed model of computation, each multiplication operation requires at most a constant number of memory references. Thus, the total data traffic is

$$\leq \textit{const} \cdot \textit{number of multiplication operations.}$$

From [1], the number of multiplication operations associated with factoring matrix A is $O(n^{\frac{3}{2}})$. Hence, the total worst case data traffic is $O(n^{\frac{3}{2}})$.

□

Returning to the model of computation used to get bounds for the dense matrix case, the above worst case bound on the data traffic is improved when the number of processors is n^{α} where $\alpha < 1$. This is possible because now the local memory is used to store the data that is needed in more than one computations.

First, bounds are obtained on the data traffic associated with computing the elements of the Cholesky factor along the columns corresponding to a generic "+" shaped separator in the nested dissection ordering of a 2-D regular grid graph. Let the ordering scheme be such that the vertices on vertical part of a "+" separator are ordered after those on the horizontal part. Let each of the two horizontal parts of the "+" separator be referred to as the *horizontal sub-separator* and the single vertical part be referred to as the *vertical sub-separator.*

Let $\eta_i^j = \{ k \mid k \leq j \text{ and } l_{i,k} \neq 0, l_{i,k} \in L \}$; i.e., η_i^j is the set of all columns of the factor L to the left of the column j+1 such that the elements in row i of these columns are nonzero. Let $\overline{\eta}_{i,k}^j = \bigcup_{s=i}^{k} \eta_s^j$; i.e., $\overline{\eta}_{i,k}^j$ is the set of all the columns to left of column j+1 such that on each of these columns there is a nonzero element in at least one of the i through k rows of the factor. Let Γ represent any m-vertex sub-separator. It is assumed that all the vertices in any sub-separator are ordered consecutively. Let $low(\Gamma)$ and $high(\Gamma)$ be the indices of the lowest and the highest ordered vertices, respectively, on the sub-separator Γ. Note that $high(\Gamma) - low(\Gamma) + 1 = m$. The following lemma establishes some basic sub-separator related properties that are useful in analyzing the communication requirements.

LEMMA 3 : *Let* Γ *be any* m*-vertex sub-separator.* (i) *Corresponding to the vertices of* Γ *there is a dense* $m \times m$ *triangular diagonal block in the Cholesky factor.* (ii) *In the factor* L*, the columns* $low(\Gamma)$ *through* $high(\Gamma)$ *contain at most four off-diagonal rectangular blocks with nonzero elements. Each of these blocks is of size at most* $(c_1 \cdot m + c_2) \times m$ *where* $c_1 \leq 2$ *and* $c_2 \leq 3$ *are integer constants. Any nonzero element in these columns is in one of these four blocks or in the diagonal triangular block and no where else.*

PROOF: The first part of the lemma is obvious. In Figure 4, the sub-separator Γ separates the vertices in regions R_1 and R_2. Since the vertices in these two regions are ordered ahead of those of Γ, the fill-in due to the elimination of vertices in regions R_1 and R_2 ensures a dense $m \times m$ triangular diagonal block bounded by columns $low(\Gamma)$ and $high(\Gamma)$ as shown in Figure 5.

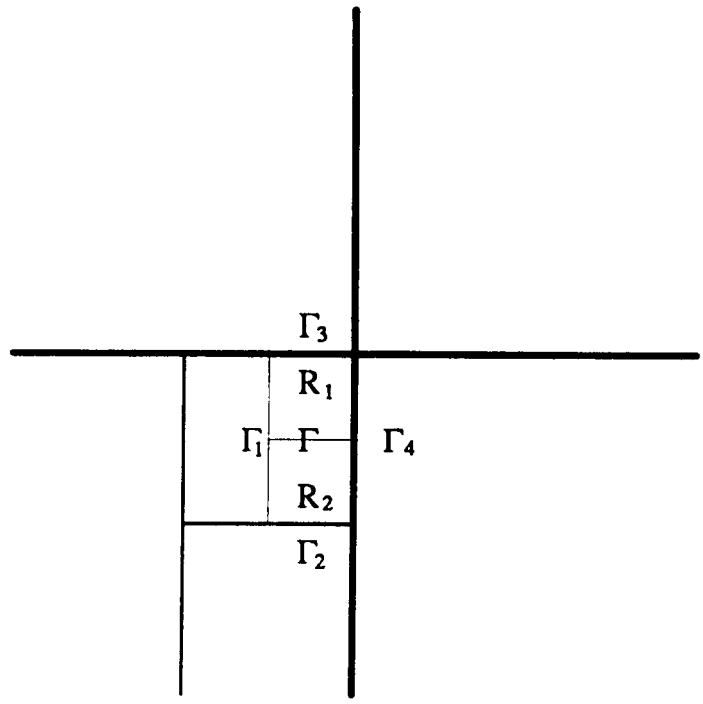

Figure 4: Sub-separator Γ with four surrounding sub-separators.

To prove the second part of the lemma, consider Figure 4 again. In that figure the thickness of the lines qualitatively indicate the separator levels in the nested dissection ordering. Let Γ_1, Γ_2, Γ_3, and Γ_4 be the four partial sub-separators that surround the sub-separator Γ. Because of the nature of the nested dissection ordering the vertices of Γ are "connected" to only those higher ordered vertices that lie on Γ_1, Γ_2, Γ_3, and Γ_4 and to no other vertices.† Thus, all the nonzeros on columns *low*(Γ) through *high*(Γ) in rows below *high*(Γ) are confined to only the rows corresponding to the vertices on Γ_1, Γ_2, Γ_3, and Γ_4. Furthermore, each vertex in Γ is "connected" to every vertex on these four partial sub-separators and hence, the four rectangular blocks are dense. This is shown schematically in Figure 5. It can be verified that if Γ is a horizontal m-vertex

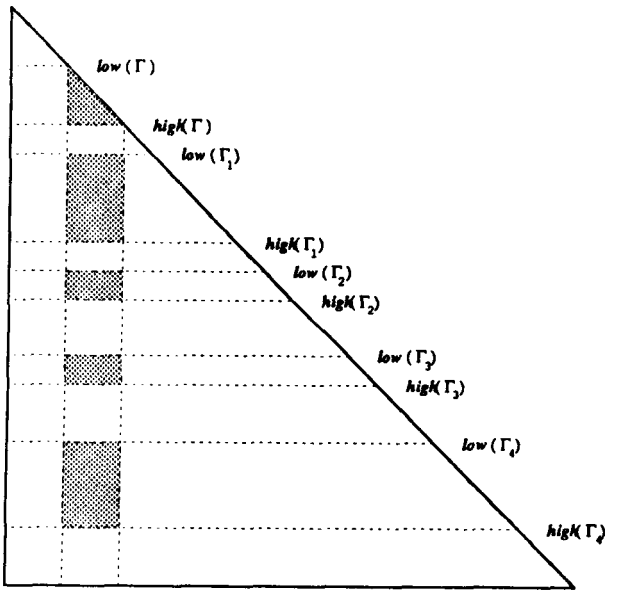

Figure 5: Off-diagonal blocks with nonzeros corresponding to sub-separator Γ.

† Vertex u is said to be "connected" to vertex v, if there exists a path $[u, u_1, u_2, \cdots, u_k, v]$ of length one or more in the grid graph such that $index(u_r) < \min(index(u), index(v))$, for $1 \le r \le k$; i.e, if $l_{i,j} \neq 0$, $l_{i,j} \in L$ where $i = index(u)$, $j = index(v)$.

sub-separator, then the surrounding box of vertices is of dimension $(2m+3)$ x $(m+2)$. Therefore there are two rectangular off-diagonal dense blocks of dimension at most $(2m+3)$ x m and the other two of dimension at most $(m+2)$ x m. Similarly, if Γ is a vertical m-vertex sub-separator, there are four off-diagonal rectangular blocks of dimension at most $(m+2)$ x m in the factor. If Γ is not surrounded on all four sides then some these blocks will be missing.

□

Using the above stated sub-separator properties, the following lemma gives the data dependencies of the nonzero elements in any of the five blocks specified in Lemma 3. The lemma shows that the number of nonzero elements in any row i of the factor L is less than $c \cdot m$ where c is an integer constant and m is the size of the sub-separator to which the vertex corresponding to row i belongs. It is then shown that for any row i, the computations at all the elements $l_{i,j} \in L$, $low(\Gamma) \leq j \leq high(\Gamma)$, for some m-vertex sub-separator Γ require a total of less than $c \cdot m$ nonzero elements from that row. Note that this count is independent of the sub-separator to which the vertex corresponding to row i belongs. Thus, the computations at all the elements in a row of any of the five blocks specified in Lemma 3, require only $c \cdot m$ elements from that row irrespective of the relative location of the off-diagonal blocks in the factor.

LEMMA 4 : *Let Γ be any m-vertex sub-separator. The nonzero elements from row i, $i \geq low(\Gamma)$, required in completing the computations of all elements $l_{i,j} \in L$ such that $low(\Gamma) \leq j \leq high(\Gamma)$, are those elements in row i on the columns in the set given by,* $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)}$. *For all $i \geq low(\Gamma)$, the number of such useful nonzero elements*

in row i is $\leq c \cdot m$ for some constant c.

PROOF: Any nonzero element $l_{i,j} \in L$, $i \geq low(\Gamma)$ and $low(\Gamma) \leq j \leq high(\Gamma)$, is in one of the five blocks specified in Lemma 3. Hence to prove the result of this lemma only the rows that intersect one of these blocks need to be considered. The result for $low(\Gamma) \leq i \leq high(\Gamma)$ is first proved followed by that for $i > high(\Gamma)$.

When $low(\Gamma) \leq i \leq high(\Gamma)$, $\bar{\eta}_{low(\Gamma),high(\Gamma)}^{high(\Gamma)} \cap \eta_i^{high(\Gamma)} = \eta_i^{high(\Gamma)}$, since, $\eta_i^{high(\Gamma)} \subset \bar{\eta}_{low(\Gamma),high(\Gamma)}^{high(\Gamma)}$. By definition, the set $\eta_i^{high(\Gamma)}$ contains all the columns that have a nonzero element in row i. Clearly, the nonzero elements from the row i required in completing the computations at all the elements $l_{i,j} \in L$, $low(\Gamma) \leq j \leq high(\Gamma)$, are on columns in the set $\eta_i^{high(\Gamma)}$.

To measure the size of the set $\eta_i^{high(\Gamma)}$ note that it is bounded by the number of vertices ordered ahead of the vertex i and which are "connected" to vertex i. Using the recursive nature of the nested dissection ordering it can be verified that in the case of constant degree grid graphs and when $low(\Gamma) \leq i \leq high(\Gamma)$, the size of the set $\eta_i^{high(\Gamma)}$ is bounded by $c \cdot m$, where c is a constant dependent on the degree of the graph and on whether Γ is a horizontal or vertical sub-separator. If Γ is a horizontal m-vertex sub-separator, then for a 5-point stencil, $c = 7$ and for a 9-point stencil, $c = 11$. When Γ is a vertical m-vertex sub-separator, the values of c are 5 and 7, respectively. This completes the proof when $low(\Gamma) \leq i \leq high(\Gamma)$.

The case where $i > high(\Gamma)$ is considered next. As shown above, $\| \eta_i^{high(\Gamma)} \|$ depends on the size of the sub-separator to which the vertex i belongs and hence when

$i > high(\Gamma)$, $\| \eta_i^{high(\Gamma)} \|$ can be much greater than $O(m)$ where m is size of Γ.† However, when the computation of only those elements in row i that lie on columns $low(\Gamma)$ through $high(\Gamma)$ are of concern, each of these computations consists of a product of a nonzero element in row i and a nonzero element in one of the rows $low(\Gamma)$ through $high(\Gamma)$ in the column $high(\Gamma)$ or in some other column to the left of it. Thus, for these computations only the columns that have a nonzero element in row i and in row j where $low(\Gamma) \le j \le high(\Gamma)$, are of interest. Now the set $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)}$ consists of all the columns that have a nonzero element in at least one of rows $low(\Gamma)$ through $high(\Gamma)$. Similarly, $\eta_i^{high(\Gamma)}$ consists of all the columns that have a nonzero element in row i. Clearly, the set $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)}$ consists of all the columns which contain all the pairs of nonzero elements that can be usefully employed in completing the computations at all the elements l_{ij}, $low(\Gamma) \le j \le high(\Gamma)$. Thus, the nonzero elements from row i, $i > high(\Gamma)$, required in completing computations at all the elements $l_{ij} \in L$ such that $low(\Gamma) \le j \le high(\Gamma)$, are those elements in the row i on the columns in the set given by, $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)}$.

To get a bound on the size of $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)}$, consider the m-vertex horizontal sub-separator Γ shown in Figure 4. It is surrounded by sub-separators Γ_1, Γ_2, Γ_3, and Γ_4. Suppose that $low(\Gamma_1) \le i \le high(\Gamma_1)$. Now, the set $\overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)}$ consists of columns corresponding to vertices on Γ or corresponding to those vertices ordered ahead of them which are "connected" to at least one vertex in Γ and to the vertex corresponding to row i. Using the recursive ordering of the nested dissection

---

† $\| U \|$ denotes the cardinality or the number of elements in set U.

scheme it can be shown that the number of such vertices is less than $7m$. Thus, $\| \overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)} \| \leq 7m$, for $low(\Gamma_1) \leq i \leq high(\Gamma_1)$. The same bound is obtained when $low(\Gamma_4) \leq i \leq high(\Gamma_4)$. If $low(\Gamma_2) \leq i \leq high(\Gamma_2)$ or $low(\Gamma_3) \leq i \leq high(\Gamma_3)$ then it can be verified that, $\| \overline{\eta}^{high(\Gamma)}_{low(\Gamma),high(\Gamma)} \cap \eta_i^{high(\Gamma)} \| \leq 3m$. If Γ is vertical sub-separator the two bounds are $5m$ and $\frac{5}{2}m$ respectively.

□

The scheme for completing the computations in all five dense blocks associated with an m-vertex sub-separator Γ using p processors is now described. If $p = 1$, that processor accesses the necessary values and completes all the computations. If $p > 1$, the factorization corresponding to the m x m triangular diagonal block is first completed using all the processors and then the factorization corresponding to the four off-diagonal blocks. For the first part, as stated in the previous section for the dense matrices, the m x m dense diagonal block is partitioned into $\frac{r^2}{2} - \frac{r}{2}$ square blocks and r diagonal triangular blocks each of size $\frac{m}{r}$ x $\frac{m}{r}$ where $p = \frac{r^2}{2} + \frac{r}{2}$, and each of these p partitions is assigned to a unique processor. Each processor completes the computations corresponding to its partition by accessing the required data from the shared memory. For the purpose of factoring, the off-diagonal blocks are treated as if they were adjacent, and the resultant rectangular block is partitioned into p sub-blocks each of size $\frac{c \cdot m}{\sqrt{p}}$ x $\frac{m}{\sqrt{p}}$, where $c \leq 6$ for a horizontal sub-separator and $c \leq 4$ for a vertical sub-separator. Again each partition is assigned to a unique processor.

Using the results from the previous lemma, a bound is obtained on the data traffic associated with the computations performed as outlined in the above scheme.

LEMMA 5 : *The total data traffic associated with completing the computations, using p processors, in all the dense blocks within the columns low(Γ) through high(Γ), for some m-vertex horizontal sub-separator Γ, is bounded by* $(53 + 11\sqrt{2})m^2 \cdot \sqrt{p}$. *It is bounded by* $(28 + 8\sqrt{2})m^2 \cdot \sqrt{p}$ *for an m-vertex vertical sub-separator.*

PROOF: Let Γ be an m-vertex horizontal sub-separator. Assume that this sub-separator is surrounded on all four sides by vertices ordered higher than any vertex on the sub-separator. Such a sub-separator has the worst case communication requirements among all the m-vertex sub-separators.

First, compute the data traffic associated with factoring the triangular diagonal block using $p = \frac{r^2}{2} + \frac{r}{2}$ processors. Each of the sub-blocks requires nonzero elements from at most $\frac{2m}{r}$ rows out of the m rows in the range *low*(Γ) through *high*(Γ) of the factor. No other information is needed. From Lemma 4, each of these rows has at most $11m$ nonzeros. Thus the communication requirement of each partition is at most $11m \cdot 2\frac{m}{\sqrt{2p}}$ and the total communication requirement of the triangular block is, bounded by $11\sqrt{2}m^2 \cdot \sqrt{p}$.

Now consider the data traffic associated with the off-diagonal blocks. Each partition is of size $\frac{6m}{\sqrt{p}}$ x $\frac{m}{\sqrt{p}}$. Thus, each partition requires nonzero elements from $\frac{6m}{\sqrt{p}}$ rows which are below the row *high*(Γ) in the factor. From Lemma 4, each of these rows has at most $7m$ nonzeros that are useful in completing the computations in any of

the partitions. Each partition also requires information from $\frac{m}{\sqrt{p}}$ rows from the region *low*(Γ) through *high*(Γ). Each of these rows has at most $11m$ nonzeros. Thus, the communication requirement of each partition is at most $7m \cdot \frac{6m}{\sqrt{p}} + 11m \cdot \frac{m}{\sqrt{p}} = \frac{53m^2}{\sqrt{p}}$ and the total communication requirement of completing the computations at the off-diagonal blocks using p processors is less than or equal to $53m^2\sqrt{p}$.

Adding the communication costs corresponding to the diagonal and the off-diagonal blocks we get the total data traffic associated with Γ to be less than or equal to $(53 + 11\sqrt{2})m^2 \cdot \sqrt{p}$.

A similar analysis can be used to compute the data traffic when Γ is an m-vertex vertical sub-separator and can be shown to be bounded by $(28 + 8\sqrt{2})m^2 \cdot \sqrt{p}$.

□

Applying the results from the above lemma, a bound is obtained on the total data traffic in factoring the sparse matrix A associated with the 2-D grid graph. First a description of the overall load distribution scheme among the processors and some notation are given. For the sake of simplicity, assume that the n-vertex 2-D grid graph is a $n^{\frac{1}{2}}$ x $n^{\frac{1}{2}}$ square grid and that there are n^{α} processors, $\alpha \leq 1$. The vertices of the grid are ordered using the nested dissection ordering scheme such that the vertices in each square sub-grid are consecutively ordered. In Figure 1, such an ordering for a 7 x 7 grid is shown. This particular ordering is chosen only for the sake of describing the scheme. The results presented here are applicable to other nested dissection ordering schemes as well. This ordering results in n^{α} sub-grids each of size $n^{\frac{1}{2}-\frac{\alpha}{2}}$ x $n^{\frac{1}{2}-\frac{\alpha}{2}}$.

The computations at all the nonzero elements corresponding to the vertices of each such sub-grid is assigned to a unique processor. After completing the work, two processors from adjacent sub-grids combine to complete the work corresponding to the $n^{\frac{1}{2}-\frac{\alpha}{2}}$-vertex horizontal sub-separator that separates the two sub-grids. This is followed by the work on the $2n^{\frac{1}{2}-\frac{\alpha}{2}}$-vertex vertical sub-separator. Four processors work on each such vertical sub-separator. This process is continued until the factorization of the entire matrix is completed. Whenever more than one processor is working on any sub-separator, the scheme presented earlier for a generic m-vertex sub-separator is applied.

Let $\tau_h(m, p, k)$ represent the data traffic using p processors in completing the computations at all the nonzero elements $l_{ij} \in L$ in the columns corresponding to an m-vertex horizontal sub-separator that is surrounded by higher ordered vertices on k sides. Let $\tau_v(m, p, k)$ represent the same for an m-vertex vertical sub-separator. From Lemma 5, $\tau_h(m, p, 4) \leq (53 + 11\sqrt{2})m^2 \cdot \sqrt{p}$ and $\tau_v(m, p, 4) \leq (28 + 8\sqrt{2})m^2 \cdot \sqrt{p}$. Let $\tau_g(m, p, k)$ represent the total data traffic, using p processors, in completing the computations corresponding to all the sub-separators within an m-vertex sub-grid that is surrounded by higher ordered vertices on k sides.

The following theorem gives an upper bound on the total data traffic in factoring the matrix A associated with an n-vertex 2-D regular grid graph using n^α processors with the scheduling scheme as described above.

THEOREM 4 : *The total data traffic in factoring the* $n \times n$ *sparse matrix* A *using* n^α *processors is* $O(n^{1+\frac{\alpha}{2}})$; *i.e.,* $\tau_g(n, n^\alpha, 0) = O(n^{1+\frac{\alpha}{2}})$.

PROOF: On a $n^{\frac{1}{2}}$ x $n^{\frac{1}{2}}$ regular grid there is an $n^{\frac{1}{2}}$-vertex vertical sub-separator and two $\frac{1}{2}n^{\frac{1}{2}}$-vertex horizontal sub-separators (ignoring the additive constant -1). The vertical sub-separator is not surrounded by any vertices that are ordered after the vertices on the vertical sub-separator. Each of the two horizontal sub-separators are surrounded on one side. These three sub-separators subdivide the n-vertex grid graph into four sub-grids of size $\frac{1}{2}n^{\frac{1}{2}}$ x $\frac{1}{2}n^{\frac{1}{2}}$, each surrounded on two sides by higher ordered vertices. Thus, the total data traffic in factoring the corresponding matrix A is given by,

$$\tau_g\,(n, n^{\alpha}, 0) \;=\; \tau_v\,(n^{\frac{1}{2}}, n^{\alpha}, 0) \;+\; 2\tau_h\,(\tfrac{1}{2}n^{\frac{1}{2}}, \tfrac{1}{2}n^{\alpha}, 1) \;+\; 4\tau_g\,(\tfrac{1}{4}n, \tfrac{1}{4}n^{\alpha}, 2).$$

A recursive expansion of the above expression contains data traffic terms for vertical sub-separators of different sizes that are surrounded on zero sides, two sides, three sides (in two different ways), and on all four sides by higher ordered vertices. It also contains data traffic expressions for horizontal sub-separators of different sizes surrounded in five different ways. To keep the analysis simple, it is assumed that all the four sub-grids of size $\frac{1}{2}n^{\frac{1}{2}}$ x $\frac{1}{2}n^{\frac{1}{2}}$ are surrounded on all four sides. This simplification results in a conservative expression for the data traffic, but affects only the constant terms in the bound. Thus,

$$\tau_g\,(n, n^{\alpha}, 0) \;\leq\; \tau_v\,(n^{\frac{1}{2}}, n^{\alpha}, 0) \;+\; 2\tau_h\,(\tfrac{1}{2}n^{\frac{1}{2}}, \tfrac{1}{2}n^{\alpha}, 1) \;+\; 4\tau_g\,(\tfrac{1}{4}n, \tfrac{1}{4}n^{\alpha}, 4).$$

Now,

$$\tau_g\,(\tfrac{1}{4}n, \tfrac{1}{4}n^{\alpha}, 4) \;=\; \tau_v\,(\tfrac{1}{2}n^{\frac{1}{2}}, \tfrac{1}{4}n^{\alpha}, 4) \;+\; 2\tau_h\,(\tfrac{1}{4}n^{\frac{1}{2}}, \tfrac{1}{8}n^{\alpha}, 4) \;+\; 4\tau_g\,(\tfrac{1}{16}n, \tfrac{1}{16}n^{\alpha}, 4).$$

From Lemma 5, it follows that,

$$\tau_g\left(\frac{1}{4}n, \frac{1}{4}n^{\alpha}, 4\right) \leq \frac{1}{16}(134 + 85\sqrt{2})\cdot n^{1+\frac{\alpha}{2}}.$$

An analysis similar to that given in Lemma 5 yields

$$\tau_v\left(n^{\frac{1}{2}}, n^{\alpha}, 0\right) \leq 8\sqrt{2}\cdot n^{1+\frac{\alpha}{2}}.$$

and

$$\tau_h\left(\frac{1}{2}n^{\frac{1}{2}}, \frac{1}{2}n^{\alpha}, 1\right) \leq \frac{1}{8}(22 + 25\sqrt{2})\cdot n^{1+\frac{\alpha}{2}}.$$

Thus, we get,

$$\tau_g\left(n, n^{\alpha}, 0\right) \leq \frac{1}{2}(78 + 71\sqrt{2})\cdot n^{1+\frac{\alpha}{2}}.$$

□

In the following theorem the communication bound of the above presented scheme is shown to be optimal in an order of magnitude sense, by giving a lower bound proof.

THEOREM 5 : *Under the condition of uniform load distribution, the data traffic in factoring the* n x n *sparse matrix A, using* n^{α} *processors,* $\alpha \leq 1$, *is* $\Omega(n^{1+\frac{\alpha}{2}})$.

PROOF: For a regular 2-D grid graph with n vertices, the separator size for nested dissection ordering is $n^{\frac{1}{2}}$ [7]. From Lemma 3, it follows that the factor L has an $n^{\frac{1}{2}}$ x $n^{\frac{1}{2}}$ dense triangular diagonal block incorporated in it. From Theorem 2, the data traffic involved in completing the computations associated with the elements of this dense triangular block, under the condition of uniform load distribution using n^{α} pro-

cessors, is $\Omega(n^{1+\frac{\alpha}{2}})$. Since the factorization of A cannot be completed without completing the factorization of this dense block, the result follows.

□

From Theorems 4 and 5, it is clear that the load assignment scheme described here for factoring the n x n sparse matrix using n^α processors is optimal in an order of magnitude sense. Note that when n^α, $\alpha > 1$, processors are used, the data traffic bound given in Theorem 3 holds.

5. Conclusions

In this paper load distribution schemes for multiprocessor systems with local and shared memory are presented for factoring dense and sparse symmetric, positive definite matrices. The communication requirements of these schemes are analyzed and are shown to be asymptotically optimal for systems corresponding to regular 2-D problems where the vertices of the grid graph are ordered using the nested dissection ordering methods. For these schemes it is shown that the total data traffic in factoring an n x n dense, symmetric, positive definite matrix is $O(n^{2+\frac{\alpha}{2}})$ when n^α processors are used. The total data traffic in factoring the n x n sparse, symmetric, positive definite matrix corresponding to a 2-D grid problem, is shown to be $O(n^{1+\frac{\alpha}{2}})$ when n^α, $\alpha \leq 1$, processors are used. When n^α, $\alpha > 1$, processors are used the total data traffic is $O(n^{\frac{3}{2}})$. Under the condition of uniform load distribution, these results are shown to be optimal in an order of magnitude sense.

In [4], George et al. have given a load assignment scheme for factoring the matrix A that has a total data traffic of $O(n^{1+\alpha})$ when $O(n^{\alpha})$ processors are used. In this assignment scheme, the computational work corresponding to an entire column of the matrix A is performed by no more than one processor. In other words the work associated with any column is treated as indivisible. Under the condition of uniform load distribution and a column-level indivisibility, in [5] and [8] it is shown that the total data traffic of that scheme is asymptotically optimal. The scheme presented in this paper reduces the communication requirement to $O(n^{1+\frac{\alpha}{2}})$ by removing the constraint of column-level indivisibility. Here the indivisible work unit is the computation corresponding to a nonzero element in the factor. Note that this does not necessarily enforce the use of a finer grain computational model. However, the SPOCC factorization scheme allows an efficient use of up to $O(n)$ processors before the total data traffic reaches the maximum value of $O(n^{\frac{3}{2}})$, whereas in the former scheme only up to $O(n^{\frac{1}{2}})$ processors may be efficiently used before the total data traffic reaches the maximum.

Another notable aspect of the scheme presented in this paper is that the reduction in the communication requirements is brought about by improving the utilization of the data accessed from shared memory by each processor. Consider the factorization of an m x m dense matrix. Let the *data utilization* of a data element accessed by a processor be defined as the number of computations in which that element is used by that processor divided by m. Since an element in the factor is needed in at most m computations, the maximum utilization of any data accessed is one. Let the *aggregate data utilization* for a processor be defined as the average utilization of the individual data

elements accessed by that processor. In the SPOCC factorization of an m x m dense matrix, each processor accesses at most $\frac{2m^2}{\sqrt{p}}$ elements from the shared memory and each element is used in at least $\frac{m}{\sqrt{p}}$ computations. Thus, the utilization of each data accessed is $\frac{1}{\sqrt{p}}$ and so is the aggregate utilization of all the data accesses. On the other hand, with the column-level work assignment scheme, each processor accesses $O(m^2)$ elements from the shared memory. Of these, only $O(\frac{m}{p})$ elements have an utilization of one and the data utilization for the remaining elements is $\frac{1}{p}$ which gives an aggregate data utilization of approximately $\frac{1}{p}$. Similar improvements in data utilizations are obtained in factoring a sparse matrix.

It should be noted that the square shape of the submatrix partitions produce the best possible aggregate utilizations. For the algorithm considered here, the data dependencies are such that rectangular and square partitions give rise to high data utilizations. Since the square partitions have the minimum perimeter for a given area, the number of data elements accessed (which is proportional to the perimeter of the partition) for a given work load (which is proportional to the area enclosed), is also a minimum for the square partitions.

An effect of the improvement in the aggregate utilization of data and the resulting reduction in the communication requirements is the segregation of the accesses to the shared data. Since the total data traffic in factoring an m x m dense matrix using p processors is $O(m^2 \cdot \sqrt{p})$, on an average each processor accesses only $O(\frac{m^2}{\sqrt{p}})$ data. Note

that the total shared data is $O(m^2)$. Thus, on an average each element in the shared memory is accessed by $O(\sqrt{p})$ processors. The column-level assignment scheme, however, has a total data traffic of $O(m^2 \cdot p)$ and thus, on an average each processor accesses $O(m^2)$ data or on an average each element in the shared memory is accessed by $O(p)$ processors. An obvious implication of this observation is that for the scheme presented here, not only is the total data traffic reduced but also the requests at individual shared addresses. This can have considerable impact on the performance of the systems with a large number of processors.

In the factorization scheme presented here, the work assigned to each processor is balanced in an order of magnitude sense which is sufficient for carrying out the analysis. For practical purposes it is necessary to consider the effect of the constants involved on the load imbalance. Schemes for reducing the load imbalance caused by the constant terms in the column-level assignment scheme and in the SPOCC scheme are presented elsewhere.[2] Some implementation aspects are also considered there. Finally, the scheme developed here for factoring a sparse matrix is applicable to a wider class of problems. The the communication requirements for these problems are found to be optimal as well.[2]

In summary, we have presented schemes that reduce the data traffic involved in factoring both dense and sparse, symmetric, positive definite matrices on multiprocessor systems with a two level memory hierarchy. These reductions are made possible by improving the utilization of each data element accessed.

---

[2] V. K. Naik and M. L. Patrick, *Communication requirements of parallel sparse Cholesky factorizations*, in preparation.

ACKNOWLEDGMENTS

The authors would like to thank Bob Voigt for his support and encouragement.

REFERENCES

(1) J. A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal., vol. 10, pp. 345-363, 1973.

(2) J. A. George, M. T. Heath, J. W. H. Liu, and E. Ng, *Sparse Cholesky factorization on a local-memory multiprocessor*, Technical report ORNL/TM-9962, Oak Ridge National Laboratory, Oak Ridge, Tenn, 1986.

(3) J. A. George and J. W. H. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Inc., Englewood Cliff, NJ, 1981.

(4) J. A. George, J. W. H. Liu, and E. Ng, *Communication reduction in parallel sparse Cholesky factorization on a hypercube*, in *Hypercube Multiprocessors 1987*, ed. M. T. Heath, SIAM Publication, 1987.

(5) J. A. George, J. W. H. Liu, and E. Ng, *Communication results for parallel sparse Cholesky factorization on a hypercube*, Submitted to Parallel Computing, 1988.

(6) G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

(7) R. J. Lipton, D. J. Rose, and R. E. Tarjan, *Generalized nested dissection*, SIAM J. Numer. Anal., vol. 16, pp. 346-358, 1979.

(8) V. K. Naik and M. L. Patrick, *Analysis of communication requirements of sparse Cholesky factorization with nested dissection ordering*, Proc. of the third SIAM

Conference on Parallel Processing for Scientific Computing, Los Angles, Dec. 1-4, 1987.

(9) E. M. Reingold, J. Nievergeld, and N. Deo, *Combinatorial Algorithms: Theory and Practice*, Prentice-Hall, Inc., Englewood Cliff, NJ, 1977.

NASA

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-181626 ICASE Report No. 88-14		

4. Title and Subtitle	5. Report Date
DATA TRAFFIC REDUCTION SCHEMES FOR SPARSE CHOLESKY FACTORIZATIONS	February 1988
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Vijay K. Naik and Merrell L. Patrick	88-14
	10. Work Unit No. 505-90-21-01

9. Performing Organization Name and Address	11. Contract or Grant No.
Institute for Computer Applications in Science and Engineering Mail Stop 132C, NASA Langley Research Center Hampton, VA 23665-5225	NAS1-18107
	13. Type of Report and Period Covered Contractor Report

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
National Aeronautics and Space Administration Langley Research Center Hampton, VA 23665-5225	

15. Supplementary Notes

Langley Technical Monitor:
Richard W. Barnwell

Final Report

Submitted to Int. Supercomputing Conference

16. Abstract

Load distribution schemes are presented which minimize the total data traffic in the Cholesky factorization of dense and sparse, symmetric, positive definite matrices on multiprocessor systems with local and shared memory. The total data traffic in factoring an $n \times n$ sparse, symmetric, positive definite matrix representing an n-vertex regular 2-D grid graph using n^{α}, $\alpha \leq 1$, processors is shown to be $O(n^{1+\frac{\alpha}{2}})$. It is $O(n^{\frac{3}{2}})$, when n^{α}, $\alpha \geq 1$, processors are used. Under the conditions of uniform load distribution these results are shown to be asymptotically optimal. The schemes allow efficient use of up to $O(n)$ processors before the total data traffic reaches the maximum value of $O(n^{\frac{3}{2}})$. The partitioning employed within the scheme, allows a better utilization of the data accessed from shared memory than those of previously published methods.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
communication requirements, shared and local memory, multiprocessor systems, sparse Cholesky factorizations	59 - Mathematical and Computer Sciences (General) 62 - Computer Systems Unclassified - unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	34	A03