Incomplete Nested Dissection with
Implicit Storage and Solution Schemes*

Alan George
and
Hamza Rashwan

Department of Computer Science
University of Waterloo
Waterloo, Ontario , Canada

Research Report CS-81-01

January 1981

Faculty

of

Mathematics

University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

Incomplete Nested Dissection with Implicit Storage and Solution Schemes

Abstract

It is well known that nested dissection orderings are very effective in reducing storage and computational requirements for solving sparse symmetric systems of linear equations. Recently, it has been shown that *incomplete nested dissection* orderings are competitive in their storage and arithmetic requirements with nested dissection orderings. In this paper we study the effect of using a so-called *implicit* storage and solution method on the storage and computational demands of incomplete nested dissection. Analysis of this strategy for n by n grid problems suggests that we can achieve a significant reduction in storage requirements at the expense of slightly increasing the operation count. We also provide some numerical experiments which show the performance of the new method, along with some comparisons with other methods.

§1. Introduction

In this paper we consider the direct solution of the system of linear equations

$$Ax = b \tag{1.1}$$

where A is a sparse N by N symmetric and positive definite matrix arising in finite element problems. The system (1.1) is solved using Cholesky's method by factoring A into LL^T, where L is a lower triangular matrix, and then solving the triangular systems $Ly = b$ and $L^T x = y$. When the factorization of A is carried out, it usually suffers from fill-in; that is, the triangular factors have nonzero components in positions which are zero in A. Thus, we might consider the equivalent system

$$(P\ A\ P^T)\ Px = Pb \tag{1.2}$$

where P is a permutation matrix chosen to reduce fill-in or operation count or both among other objectives.

The so-called nested dissection ordering [2,4] is an effective method for reducing storage and arithmetic for solving the system (1.1). It has been proved that this ordering applied to n by n grid problems is optimal in the asymptotic sense [2,8]. The incomplete nested dissection ordering [7] was shown to be competitive with nested dissection orderings in both storage and arithmetic requirements for solving n by n grid problems.

In this paper, we consider the use of the implicit storage and solution ideas [3,5] in conjunction with incomplete nested dissection. The motivation for this study is that the (incomplete) nested dissection ordering is very effective in reducing storage and computational requirements, and the implicit storage scheme can be used to further reduce the storage requirements.

The outline of the paper is as follows. In §2 we briefly review the nested dissection and incomplete nested dissection orderings. In §3 we review the basic ideas of implicit storage and factorization schemes, and then derive storage and operation counts for incomplete nested dissection employing these ideas. Section 4 contains a brief description of the storage scheme used, some numerical experiments, and our conclusions.

The following notations are used throughout this paper.

Θ_F - the number of operations (multiplicative operations) required to compute the Cholesky factorization.

Θ_S - the number of operations required for the upper or lower solvers, given the Cholesky factorization.

η_L - The number of nonzero elements of L which must be stored, including some of those of A which must be stored as well, in order to execute the algorithm. (For the method we consider here, this is less than the number of nonzero elements of L since a part of L is not stored.)

§2. Review of nested dissection and incomplete nested dissection orderings.

2.1 Nested dissection ordering.

Following George [4], let V be the set of nodes of the n by n mesh and let C_1 be the set of nodes on a vertical mesh line which as nearly as possible divides the mesh into two equal parts R_1^1 and R_1^2, where $V - C_1 = R_1^1 \cup R_1^2 = R_1$. Numbering the nodes in R_1^1 followed by those in R_1^2 and finally those in C_1, induces the following block structure in the reordered matrix A.

$$(2.1) \qquad A = \begin{bmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{13}^T & A_{23}^T & A_{33} \end{bmatrix}$$

Now choose vertex sets $S_1^\ell \subset R_1^\ell$, $\ell = 1,2$, consisting of nodes lying on a horizontal mesh line which as nearly as possible divides R_1^ℓ into two equal parts. If we number the variables associated with the nodes in $R_1^\ell - S_1^\ell$ before those associated with S_1^ℓ, $\ell = 1,2$, and then the remaining nodes as before, we induce the 7 by 7 partitioning on A shown in (2.2)

$$(2.2) \qquad A = \begin{bmatrix} A_{11} & & & & A_{15} & & A_{17} \\ & A_{22} & & & A_{25} & & A_{27} \\ & & A_{33} & & & A_{36} & A_{37} \\ & & & A_{44} & & A_{46} & A_{47} \\ A_{15}^T & A_{25}^T & & & A_{55} & & A_{57} \\ & & A_{36}^T & A_{46}^T & & A_{66} & A_{67} \\ A_{17}^T & A_{27}^T & A_{37}^T & A_{47}^T & A_{57}^T & A_{67}^T & A_{77} \end{bmatrix}$$

The partitioning in (2.2) corresponds to a <u>one-level dissection</u> of the mesh as depicted in figure 2.1.

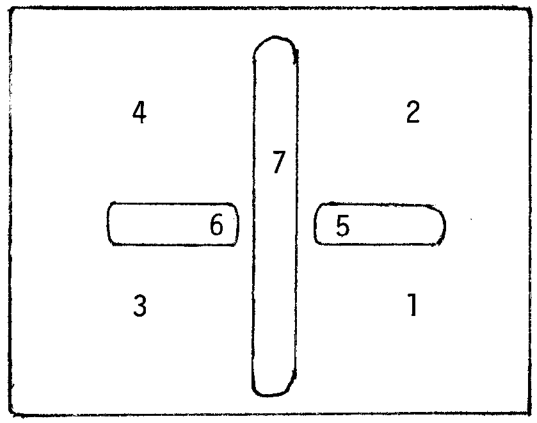

Figure 2.1. A one-level dissection of the mesh.

Applying the dissection strategy recursively on the resulting smaller grids until we can no longer dissect them results in a <u>nested dissection ordering</u>. The following estimates for storage and operation counts are taken from [4].

(2.3) $\Theta_F \simeq 9.88\, n^3 - 17\, n^2 \log(n + 1) + 16.06\, n^2 + O(n \log_2 n)$

(2.4) $\eta_L \simeq 7.75\, n^2 \log_2 (n + 1) - 24\, n^2 + O\,(n \log_2 n)$

(2.5) $\Theta_S = \eta_L$

2.2 Incomplete nested dissection

Consider the n by n grid problem. If $n = 2^k - 1$, where k is a positive integer, then the dissection strategy described in section 2.1 can be carried out $k - 1$ $(= \log_2 (n + 1) - 1)$ times.

Suppose we stop the dissection process sooner than necessary, say at the ℓ - th level, and simply number the $n^2/2^{2\ell}$ ℓ-level nodes in a row by row (or column by column) fashion. The resulting ordering is called an <u>incomplete nested dissection</u> ordering. Figure 2.1 shows a one-level incomplete nested dissection ordering. Figure 2.2 displays the structure of the matrix A and its Cholesky factor corresponding to the ordering in Figure 2.1.

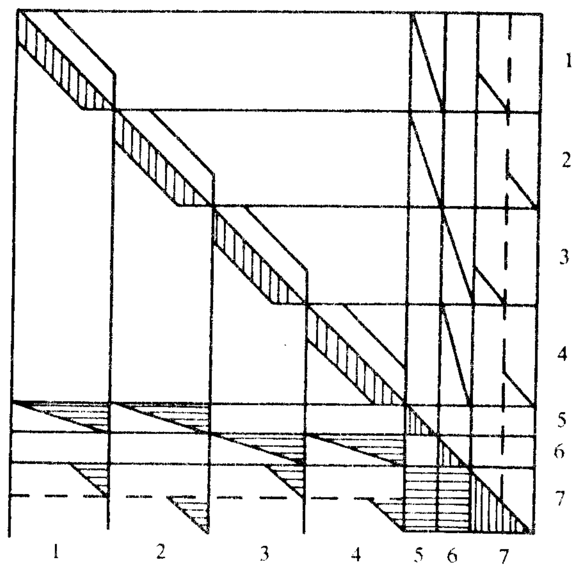

Figure 2.2 The structure of L is shown in the lower part; while that of A is in the upper part.

Note that an ℓ-level incomplete nested dissection ordering partitions the node set into μ subsets, where

$$\mu = 2 \times 2^{2\ell} - 1 \tag{2.6}$$

The corresponding reordered matrix has $2^{2\ell}$ leading diagonal blocks, each of size $\left[\frac{n - (2^{\ell} - 1)}{2^{\ell}}\right]^2$ and bandwidth $\frac{n - (2^{\ell} - 1)}{2^{\ell}}$. The remaining diagonal blocks correspond to the separators.

Now consider the two-level incomplete nested dissection of figure 2.3. The last-level blocks (blocks 1 through 16) are classified as corner, side, and interior blocks for obvious reasons. The structure of the corresponding block columns of the Cholesky factor L is shown in figure 2.4.

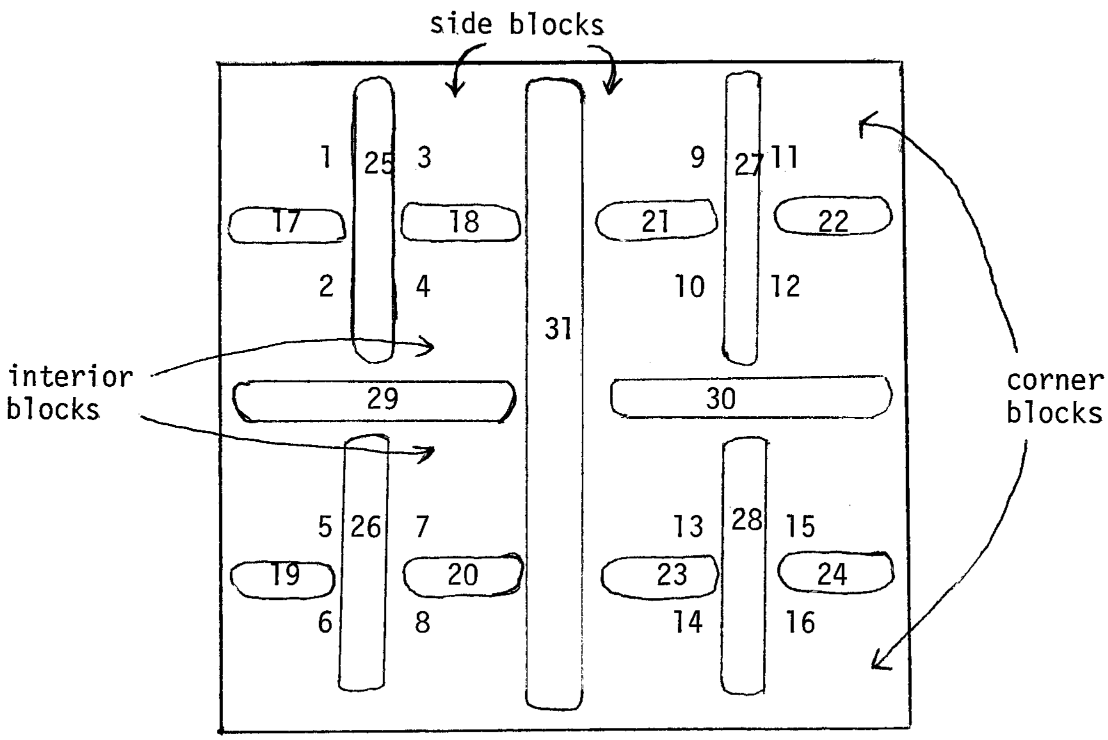

Figure 2.3 A two-level incomplete nested dissection.

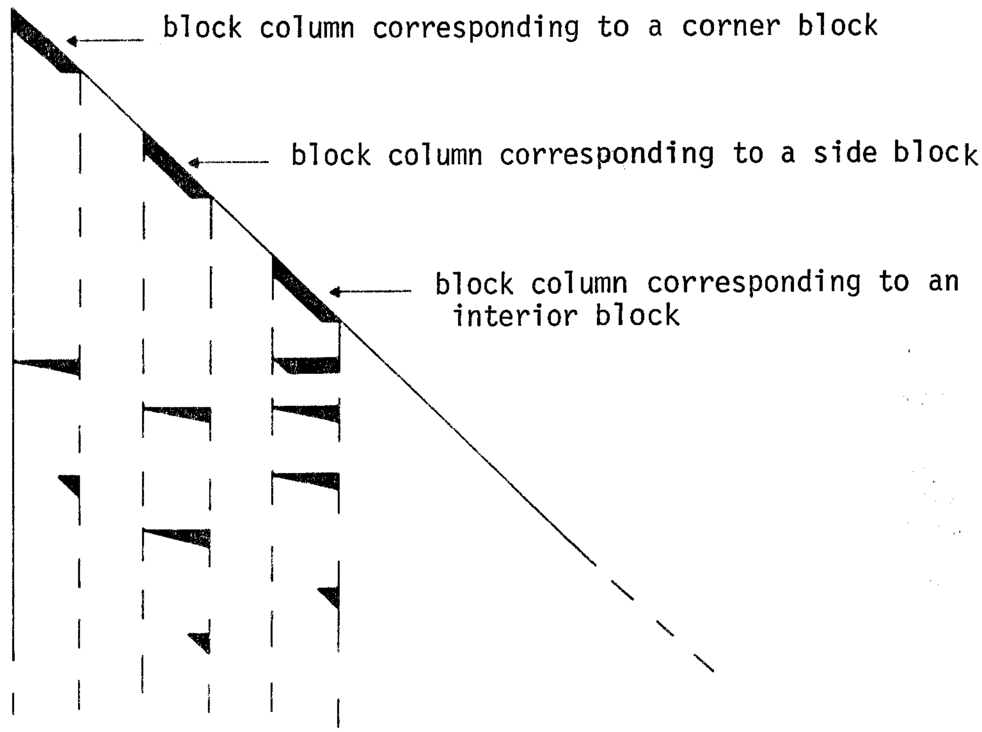

Figure 2.4 Structure of the block columns of the Cholesky factor L, corresponding to leading diagonal blocks.

The expressions for storage and operation counts are derived under the assumption that the banded character of the leading diagonal blocks and the leading zeros of the off-diagonal blocks are exploited. The following estimates for storage and operation counts are taken from [7].

$$
\begin{aligned}
(2.7)\qquad \eta_L(n,\ell) \simeq\ & n^3\,(3 \times 2^{-\ell} - 4 \times 2^{-2\ell} + 2 \times 2^{-3\ell}) \\
& + n^2\,(\tfrac{31}{4}\ell - \tfrac{35}{2} + 31 \times 2^{-\ell} - 18 \times 2^{-2\ell} + 6 \times 2^{-3\ell}) \\
& + n\,(-\tfrac{21}{2} \times 2^{\ell} + \tfrac{31}{2}\ell - 13 + 39 \times 2^{-\ell} - 24 \times 2^{-2\ell} + 6 \times 2^{-3\ell}) \\
& + (2^{2\ell} + \tfrac{3}{2} \times 2^{\ell} - \tfrac{17}{4} \times \ell - \tfrac{11}{2} + 11 \times 2^{-\ell} - 10 \times 2^{-2\ell} + 2 \times 2^{-3\ell})
\end{aligned}
$$

$$
\begin{aligned}
(2.8)\qquad \Theta_F(n,\ell) \simeq\ & n^4\,(\tfrac{14}{3} \times 2^{-2\ell} - 10 \times 2^{-3\ell} + 6 \times 2^{-4\ell}) \\
& + n^3(\tfrac{829}{84} - \tfrac{353}{12} \times 2^{-\ell} + \tfrac{148}{3} \times 2^{-2\ell} - \tfrac{1048}{21} \times 2^{-3\ell} + 24 \times 2^{-4\ell}) \\
& + n^2(-17 \times \ell + \tfrac{2003}{42} - \tfrac{963}{7} \times 2^{-\ell} + 86 \times 2^{-2\ell} - \tfrac{664}{7} \times 2^{-3\ell} \\
& \qquad + 36 \times 2^{-4\ell}) \\
& + n(-\tfrac{17}{6} \times 2^{\ell} + 6 \times \ell - \tfrac{1657}{84} + \tfrac{325}{12} \times 2^{-\ell} + \tfrac{128}{3} \times 2^{-2\ell} \\
& \qquad - \tfrac{524}{7} \times 2^{-3\ell} + 24 \times 2^{-4\ell}) \\
& + (2 \times 2^{2\ell} - \tfrac{17}{6} \times 2^{\ell} + 23 \times \ell - \tfrac{2459}{41} + \tfrac{935}{12} \times 2^{-\ell} + \tfrac{4}{3} \times 2^{-2\ell} \\
& \qquad - \tfrac{454}{21} \times 2^{-3\ell} + 6 \times 2^{-4\ell}).
\end{aligned}
$$

$$(2.9)\qquad \Theta_S(n,\ell) = \eta_L(n,\ell)\ ,$$

where $0 < \ell \leq \log_2 (n + 1) - 1$

§3. Incomplete nested dissection with the implicit storage scheme.

3.1 Factorization of a block 2 by 2 matrix.

The basic ideas of implicit storage and solution of partitioned matrices are illustrated with a block 2 by 2 symmetric, positive definite matrix. Following [4, p.171], let A be partitioned as

$$(3.1)\qquad A = \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} .$$

The first key observation is that the following two factorizations of A can be computed

$$(3.2)\qquad \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} = \begin{bmatrix} L_1 & 0 \\ W^T & L_2 \end{bmatrix} \begin{bmatrix} L_1^T & W \\ 0 & L_2^T \end{bmatrix} ,$$

$$(3.3)\qquad \begin{bmatrix} A_{11} & A_{12} \\ A_{12}^T & A_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ A_{12}^T & \tilde{A}_{22} \end{bmatrix} \begin{bmatrix} I & \tilde{W} \\ 0 & I \end{bmatrix} ,$$

where $A_{11} = L_1L_1^T$ and $L_2L_2^T = \tilde{A}_{22} = A_{22} - A_{12}^T A_{11}^{-1} A_{12}$. The off-diagonal blocks are defined by

$$(3.4)\qquad W = L_1^{-1} A_{12}$$

and

$$(3.5)\qquad \tilde{W} = L_1^{-T} W = A_{11}^{-1} A_{12}.$$

The factorization (3.3) is as useful as (3.2), since we compute and store L_1 and L_2 rather than retaining A_{11} and $\tilde{A}_{22}$. The factorization (3.2) is called a symmetric factorization and (3.3) is called an asymmetric factorization. The essential difference between (3.2) and (3.3) is whether $\tilde{A}_{22}$ is computed as $A_{22} - (A_{12}^T L_1^{-T})(L_1^{-1} A_{12})$ or $A_{22} - A_{12}^T (L_1^{-T} (L_1^{-1} A_{12}))$. These factorizations in general require different amounts of arithmetic to compute, and the asymmetric factorization (3.3) may be cheaper than (3.2). A particularly important point for our purpose in this paper is that the calculation of $\tilde{A}_{22}$ in the asymmetric factorization can be done column by column, so that only one column of the matrix $A_{12}^T A_{11}^{-1} A_{12}$ needs to be stored at any one time. On the other hand, the first form of the computation seems to require the storage of the whole matrix $L_1^{-1} A_{12}$.

The second key observation is that we may not wish to retain the off-diagonal blocks of the factorization. Bunbh and Rose [1] observe that in performing the solution, given the triangular factorization (3.2), it may require fewer arithmetic operations to compute $\tilde{x}_1 = W x_2$ by computing $\bar{x}_1 = A_{12} x_2$ and then solving $L_1 \tilde{x}_1 = \bar{x}_1$, than by simply multiplying x_2 by W. Similar remarks apply to the use of W^T, W and $\tilde{W}^T$.

Returning to our incomplete nested dissection, our basic strategy is to treat all the $2^{2\ell}$ leading diagonal blocks as just one block corresponding to A_{11}, and all the remaining $2^{2\ell} - 1$ blocks as one block corresponding to A_{22}.

3.2 Estimates for storage and arithmetic operations.

The following observations are helpful in obtaining estimates of storage and operation count. For an ℓ-level incomplete nested dissection we have $2^{2\ell}$ leading diagonal blocks each of size $\left[\frac{n - (2^{\ell} - 1)}{2^{\ell}}\right]^2$ and bandwidth $\left[\frac{n - (2^{\ell} - 1)}{2^{\ell}}\right]$. Four of these are corner blocks, $4(2^{\ell} - 2)$ are side blocks, and $(2^{2\ell} - 4 \times 2^{\ell} + 4)$ are interior blocks.

Our estimate of the storage required is obtained from (2.7) by subtracting the storage for the off-diagonal blocks of L and adding the storage for the corresponding blocks of the original matrix. It is straight forward, but lengthy, to obtain the following estimate η_L'. Letting $\beta = \left[\frac{n - (2^{\ell} - 1)}{2^{\ell}}\right]$, we have

$$
\begin{aligned}
(3.6) \quad \eta_L'(n,\ell) &\approx \eta_L(n,\ell) \\
&+ [\, 2^{2\ell}(12\beta - 4) - 2^{\ell}(12\beta) + 4] \\
&- [\, 2^{2\ell}(2\beta^3 + 4\beta^2 + 2\beta) - 2^{\ell}(4\beta^3 + 6\beta^2 + 2\beta) + (2\beta^3 + 2\beta)]
\end{aligned}
$$

where $\eta_L(n,\ell)$ is given by (2.7).

We are interested in finding the value of ℓ which minimizes (3.6). Letting $\alpha = 2^{\ell}$, and assuming n to be sufficiently large so that lower order terms may be ignored in (3.6), we obtain

$$
\begin{aligned}
(3.7) \quad \eta_L'(n,\ell) &\simeq 3\frac{n^3}{\alpha} + \frac{31}{4} n^2 \log_2 \alpha - \frac{21}{2} n\alpha \\
&- \frac{35}{2} n^2 - \alpha^2 (2\beta^3 + 4\beta^2 - 10\beta + 3).
\end{aligned}
$$

It is a simple exercise to find out that $\ell = \tilde{\ell}_n$ which minimizes (3.7) is given by solving (3.8).

$$(3.8) \qquad \left(\frac{n}{\alpha}\right)^3 - \frac{31}{4}\left(\frac{n}{\alpha}\right)^2 \log_2 e - \frac{23}{2}\left(\frac{n}{\alpha}\right) + 30 = 0$$

The solution of (3.8) is given by $\frac{n}{\alpha} \simeq 11.95$, and hence

$$(3.9) \qquad \tilde{\ell}\eta \simeq \log_2 n - 3.58 \ .$$

It is interesting to notice that this value of $\tilde{\ell}\eta$ implies that the leading diagonal blocks are of size 120 by 120 and bandwidth 11 approximately. The estimate of $\eta_L'(n,\tilde{\ell}_\eta)$ is given by

$$(3.10) \qquad \eta_L'(n,\tilde{\ell}\eta) \simeq 7.75\, n^2 \log_2 n - 31.3\, n^2 + 0\,(n \log_2 n).$$

Note that equations (2.4) and (3.10) are asymptotically similar. We can also see that we have saved about 7 n^2 in the storage required. Although it is clear that the relative reduction in storage decreases as n increases, it is still a significant saving for quite large systems. It is also noteworthy that the function $\eta_L'(n,\ell)$ changes slowly near $\tilde{\ell}\eta$, so it is only necessary to obtain an approximate value for $\tilde{\ell}\eta$.

The estimate for the operation count for the upper{lower}solve is given by

$$(3.11) \qquad \begin{aligned} \theta_S'(n,\ell) \simeq\ & \theta_S(n,\ell) \\ & + [2^{2\ell}(\beta^3 + \beta^2 + 10\beta - 4) - 2^{\ell}(12\beta) + 4] \\ & - [2^{2\ell}(2\beta^3 + 4\beta^2 + 2\beta) - 2^{\ell}(4\beta^3 + 6\beta^2 + 2\beta) \\ & \quad + (2\beta^3 + 2\beta)] \ , \end{aligned}$$

where $\theta_S(n,\ell)$ is given by (2.9). The value of $\ell = \tilde{\ell}_S$ which approximately minimizes (3.11) is given by

$$(3.12) \qquad \tilde{\ell}_S \simeq \log_2 n - 2.36 \ .$$

Notice that for this value of ℓ the leading diagonal blocks are 16 by 16 with bandwidth 4. The estimate of $\theta_S'(n,\tilde{\ell}_S)$ is given by

$$(3.13) \qquad \theta_S'(n,\tilde{\ell}_S) \simeq 7.75\ n^2 \log_2 n - 25.9\ n^2 + O(n \log_2 n).$$

Again, we notice that (3.13) and (2.5) are asymptotically similar, and the operation count can be reduced by 1.9 n^2 by choosing ℓ equal to $\tilde{\ell}_S$. However, if we choose $\ell = \tilde{\ell}_n$, we have

$$(3.14) \qquad \theta_S'(n,\tilde{\ell}_n) \simeq 7.75\ n^2 \log_2 n - 21.4\ n^2 + O(n \log_2 n),$$

which means that the operation count is slightly higher than (2.5).

The estimate of the operation count for the factorization is given by

$$\begin{aligned}
(3.15) \qquad \theta_F'(n,\ell) \simeq\ & \theta_F(n,\ell) \\
& + [2^{2\ell}(4\beta^4 + 12\beta^3 + 36\beta^2 + 4\beta) \\
& \quad - 2^{\ell}(8\beta^4 + 20\beta^3 + 58\beta^2 + 4\beta) + 4(\beta^4 + \beta^3 + 4\beta^2 + \beta)] \\
& - [2^{2\ell}(\tfrac{25}{6}\beta^4 + \tfrac{23}{3}\beta^3 + 6\beta^2 + 2\beta) \\
& \quad - 2^{\ell}(10\beta^4 + 12\beta^3 + 8\beta^2 + 2\beta) + (6\beta^4 + 2\beta^3 + 2\beta^2 + 2\beta)] \ .
\end{aligned}$$

$\theta_F'(n,\ell)$ is minimized when we fully dissect the mesh.

Table 3.1 shows the relative difference in storage and operations for solution and factorization between nested dissection and the new scheme for the n by n grid. For the incomplete nested dissection, we used $\ell = \log_2(n+1) - 3$; which means that the leading diagonal blocks are of size 49 by 49. The estimates in the table do suggest that if n is moderately large, we can achieve significant reduction in storage at the expense of a modest increase in operations for factorization.

Table 3.1

Difference between incomplete nested dissection and the new scheme. In the table, index 1 refers to nested dissection and index 2 refers to the new scheme.

n	N	$\left(\frac{\eta_2 - \eta_1}{\eta_1}\right) \times 100$	$\left(\frac{\theta_{S_2} - \theta_{S_1}}{\theta_{S_1}}\right) \times 100$	$\left(\frac{\theta_{F_2} - \theta_{F_1}}{\theta_{F_1}}\right) \times 100$
31	961	-24.2	10.9	35.1
63	3,969	-22.6	2.0	22.8
127	16,129	-19.4	- .8	12.8
255	65,025	-16.5	- 1.6	7.7
511	261,121	-14.2	- 1.7	3.9
1023	1,046,529	-12.2	- 1.4	1.9

§4. Storage scheme for L and some numerical experiments.

4.1 The storage structure for L.

The storage scheme for the matrix A (overwritten by the Cholesky factor L during the factorization process) is briefly considered here. The elements of the leading diagonal blocks are stored using the envelope storage scheme, which is described in chapter 4 of [5]. The elements of the last diagonal block, that is, the block corresponding to all the separators, are stored using a general sparse storage method, as described in chapter 5 of [5]. The elements of the off-diagonal blocks of L are not stored; instead, we store the corresponding elements of the original matrix, as explained in detail in chapter 6 of [5]. An array XBLK of length $\nu + 1$, where ν is the number of diagonal blocks, is used to record the partitioning information. XBLK (i) records the beginning of block i. For convenience of programming, XBLK $(\nu + 1)$ is set to $N + 1$, where N is the number of equations.

4.2 Numerical experiments.

In order to obtain some results about the performance of the suggested ordering and the related solution and storage schemes, and **to compare with other implementations, we applied them to a problem** arising from the graded L mesh [5, p.279]. The numerical experiments were performed on an IBM 3031 computer, using the Fortran H extended compiler. The times reported are all in seconds. Most of the programs used are minor modifications of those in SPARSPAK, a sparse matrix package developed at the University of Waterloo [6].

The mesh was dissected until the size of the blocks remaining to be dissected dropped below 100. In Table 4.1, we report some statistics about the ordering and storage allocation programs.† Also included are some statistics about the partitioning. Table 4.2 contains some statistics about the total storage used for the linear equations solver, and the overhead storage. The overhead storage is that used for pointers for the data structures of L, as well as working storage used by the solver.

Table 4.1

Performance statistics for the ordering and storage allocation, together with some partitioning statistics.

N	ordering storage	ordering time	allocation storage	allocation time	number of blocks	number of leading blocks
265	3,346	0.167	7,120	0.153	7	4
406	5,155	0.383	11,924	0.263	9	5
577	7,354	0.447	17,846	0.373	13	7
778	9,943	0.740	23,857	0.490	19	10
1009	12,922	0.910	31,396	0.627	25	13
1270	16,291	1.147	40,205	0.810	31	16
1561	20,050	1.443	49,800	1.020	39	20
1882	24,199	1.813	59,022	1.190	49	25
2233	28,730	2.100	71,581	1.443	55	28
2614	33,667	2.560	80,311	1.650	73	37
3025	38,986	3.020	96,751	2.000	77	39
3466	44,695	3.533	112,883	2.323	85	43

† The allocation storage reported in Table 4.1 is much larger than necessary. This can be reduced by about 50% without changing the allocation time by using the recently published algorithm, "An Optimal Algorithm for Symbolic Factorization of Symmetric Matrices" by George and Liu in SIAM J. COMPUT., Vol.9, No.3, August 1980.

Table 4.2

Storage used for the solution.

N	Total Storage	Overhead Storage	$\frac{\text{Total}}{N \log N}$	$\frac{\text{Overhead}}{N}$	$\frac{\text{Overhead}}{\text{Total}}$
265	4,447	1,854	2.08	7.00	.414
406	7,479	2,840	2.13	7.00	.380
577	11,545	4,131	2.18	7.16	.356
778	16,691	5,726	2.23	7.36	.343
1009	22,626	7,492	2.25	7.43	.331
1270	29,823	9,464	2.28	7.45	.317
1561	38,174	11,761	2.31	7.53	.308
1882	47,759	14,400	2.33	7.65	.302
2233	58,760	17,072	2.37	7.65	.291
2614	71,673	20,424	2.42	7.81	.285
3025	84,816	23,456	2.42	7.75	.277
3466	98,733	26,887	2.42	7.76	.272

The results of Table 4.2 suggest that the total storage is of order $N \log N$, as expected. It also seems that the overhead storage grows linearly with N. It is interesting to notice that the ratio of (overhead storage)/(total storage) $\to 0$ as $N \to \infty$.

In Table 4.3 we consider some statistics about the performance of the linear equations solver. The times and operation counts for factorization and solution are reported.

Table 4.3

Statistics about the performance of the linear equations solver.

	Factorization				Solution			
N	operations	time	$\frac{\text{operations}}{N\sqrt{N}}$	$\frac{\text{operations}}{\text{time}}$	operations	time	$\frac{\text{operations}}{N \log N}$	$\frac{\text{operations}}{\text{time}}$
265	5.084	0.786	11.79	6.47	0.757	0.090	3.55	8.42
406	10.280	1.487	12.57	6.91	1.374	0.163	3.91	8.43
577	18.282	2.633	13.19	6.94	2.113	0.243	4.00	8.70
778	27.716	3.916	12.77	7.08	2.973	0.343	3.98	8.67
1009	40.336	5.769	12.59	6.99	3.944	0.453	3.92	8.71
1270	56.715	7.919	12.53	7.16	5.335	0.583	4.07	9.15
1561	77.058	10.786	12.49	7.14	6.797	0.757	4.10	8.98
1882	98.387	13.948	12.05	7.05	8.349	0.933	4.08	8.95
2233	130.574	19.654	12.37	6.64	10.413	1.216	4.20	8.56
2614	160.059	22.610	11.98	7.08	12.520	1.373	4.22	9.12
3025	205.905	28.550	12.38	7.21	15.150	1.643	4.32	9.22
3466	252.527	35.056	12.38	7.20	17.649	1.903	4.33	9.27
	$x10^4$			$x10^4$	$x10^4$			$x10^4$

We may conclude from Table 4.3 that the operations for factorization and solution are of $O(N\sqrt{N})$ and $O(N \log_2 N)$ respectively, as predicted by our analysis. Similar observations are true for the execution times. It is also clear that the operation counts for both factorization and solution do reflect the actual computation performed, since in both cases the ratio operations/time do not change much.

Table 4.4 shows the distribution of primary storage of L between the leading diagonal blocks and the off-diagonal blocks and the last block.

Table 4.4

Distribution of primary storage for L.

N	Total Storage	MAXLNZ	ENVSZE	MAXNZ	$\frac{\text{ENVSZE}}{\text{N}}$	$\frac{\text{MAXLNZ}}{\text{N log N}}$	$\frac{\text{MAXLNZ}}{\text{Total}}$
265	2,328	676	1,235	152	4.66	0.32	.29
406	4,233	1,311	2,292	224	5.65	0.37	.31
577	6,837	2,664	3,246	350	5.63	0.50	.39
778	10,187	4,855	4,040	514	5.19	0.65	.48
1009	14,125	7,620	4,780	716	4.74	0.76	.54
1270	19,089	10,397	6,550	872	5.16	0.79	.54
1561	24,852	14,325	7,870	1,096	5.04	0.87	.58
1882	31,477	19,465	8,760	1,370	4.65	0.95	.62
2233	39,455	24,789	10,815	1,618	4.84	1.00	.63
2614	48,635	32,108	11,905	2,008	4.55	1.08	.66
3025	58,335	38,063	15,006	2,241	4.96	1.09	.65
3466	68,380	45,271	17,098	2,545	4.93	1.11	.66

In Table 4.4, MAXLNZ is the primary storage for the last diagonal block, ENVSZE is the primary storage for all the leading diagonal blocks, and MAXNZ is the number of the elements in the off-diagonal blocks of the original matrix. The following observations are clear from Table 4.4. First, the storage used by the leading diagonal blocks appear to grow linearly with N. Second, the storage for the last diagonal block appear to grow as N log N, but is approaching that limit slowly due to the existence of large sub-dominant terms. Finally, the ratio of MAXLNZ/(total storage) tends to 1 as N tends to increase.

In order to compare this new scheme with other methods, we report in Table 4.5 some of the important statistics of the one way dissection, nested dissection, and the new incomplete nested dissection. The statistics about the one way dissection and the nested dissection were taken from [9].

Table 4.5

Comparison between 1WD, ND, and Incomplete ND.

	Total Storage			Factorization operations			Solution operations		
N	ND	1WD	IND	ND	1WD	IND	ND	1WD	IND
265	7,005	4,204	4,447	3.254	4.383	5.084	0.718	0.690	0.757
406	11,721	6,852	7,479	6.926	9.254	10.280	1.267	1.180	1.374
577	17,695	10,412	11,545	12.558	18.107	18.282	1.992	1.869	2.113
778	25,032	14,524	16,691	20.101	29.220	27.716	2.879	2.675	2.973
1009	33,850	19,402	22,626	31.111	44.905	40.336	3.998	3.659	3.944
1270	43,896	25,331	29,823	44.454	65.999	56.715	5.270	4.866	5.335
1561	55,804	32,451	38,174	62.571	97.139	77.058	6.821	6.362	6.797
1882	69,043	40,006	47,759	83.851	131.132	98.387	8.543	7.903	8.349
2233	84,246	48,943	58,760	111.183	177.168	130.574	10.566	9.910	10.413
2614	100,269	58,777	71,673	140.707	232.319	160.059	12.692	11.995	12.520
3025	118,590	70,087	84,816	178.443	301.070	205.905	15.160	14.462	15.150
3466	138,549	81,702	98,733	220.695	373.641	252.527	17.843	17.024	17.649
				$\times 10^4$	$\times 10^4$	$\times 10^4$	$\times 10^4$	$\times 10^4$	$\times 10^4$

It seems safe to conclude that the new scheme is able to reduce the storage requirements from that of the nested dissection, although the one way dissection scheme still requires the least storage. Considering the operations for factorization, our new method is more attractive than the one way dissection method. The operations for solution are all very close.

In order to have some insight on the relative performance of the new method in comparison with the nested dissection and the one way dissection methods we consider the following cost function.

$$COST(S,T) = S \times T ,$$

where S = storage used, and T = execution time. We will use the number of operations required for the numerical factorization and solution as an estimate for the execution time (see the comments following Table 4.3). In the cost function mentioned above we will ignore the cost of ordering and storage allocation because they are similar for the methods under comparison. This assumption is also justified in situations where many problems with identical matrix structure but different numerical values must be solved.

In Table 4.6 we report the value of the cost function for the three methods under comparison.

Table 4.6

COST Function = (Total storage) x (Factorization operations + solution operations) $\times 10^{-9}$

N	ND	1WD	IND
265	.278	.213	.260
406	.960	.715	.872
577	2.575	2.080	2.355
778	5.752	4.632	5.122
1009	11.884	9.422	10.019
1270	21.827	17.951	18.505
1561	38.724	33.587	32.011
1882	63.792	55.622	50.976
2233	102.569	91.562	82.844
2614	153.812	143.600	123.693
3025	229.594	221.147	187.490
3466	330.492	319.181	266.753

The results in Table 4.6 show that for moderately large problems the new method is a viable choice to use in situations where many problems with the same matrix structure must be solved.

We have presented and analysed a new scheme for solving sparse systems of linear equations, based on incomplete nested dissection and an implicit storage scheme. Our analysis and numerical experiments show that this new method enjoys the same asymptotic behaviour as nested dissection orderings, which are known to be optimal. The new ordering and solution scheme requires significantly less storage than nested dissection, but its operation count for factorization is slightly higher for moderately large problems.

§5. References

[1] James R. Bunch and D.J. Rose, Partitioning, tearing and modification of sparse linear systems, J. MATH. ANAL. and APPL., 48 (1974), pp. 574-593.

[2] Alan George, Nested dissection of a regular finite element mesh, SIAM J. NUMER. ANAL., 10 (1973), pp. 345-363.

[3] __________, On block elimination for sparse linear systems, SIAM J. NUMER. ANAL., 11 (1974), pp. 585-603.

[4] __________, Numerical experiments using dissection methods to solve n by n grid problems, SIAM J. NUMER. ANAL., 14 (1977), pp. 161-179.

[5] Alan George and J.W.H. Liu, Computer solution of large sparse symmetric positive definite systems of linear equations, to be published by Prentice Hall, 1981.

[6] __________, The design of a user interface for a sparse matrix package, TOMS, 5 (1979), pp. 139-162.

[7] Alan George, William G. Poole, Jr. and Robert G. Voigt, Incomplete nested dissection for solving n by n grid problems, SIAM J. NUMER. ANAL., 15 (1978), pp. 662-673.

[8] Alan J. Hoffman, Michael S. Martin and Donald J. Rose, Complexity bounds for regular finite difference and finite element grids, SIAM J. NUMER. ANAL., 10 (1973), pp. 364-369.

[9] E.G. Ng, On one-way dissection schemes, M. MATH. Thesis, University of Waterloo (1979).