U.S. DEPARTMENT OF COMMERCE
National Technical Information Service

AD-A028 709

A VARIANT OF NESTED DISSECTION FOR SOLVING n BY n GRID PROBLEMS

College of William and Mary

August 1976

A Variant of Nested Dissection for Solving n by n Grid Problems

A. George, W. G. Poole, Jr., and R. G. Voigt

Technical Report 12

August 1976

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* College of William and Mary Department of Mathematics Williamsburg, Virginia 23185	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

A Variant of Nested Dissection for Solving n by n Grid Proglems

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Report 12, August 1976

5. AUTHOR(S) *(First name, middle initial, last name)*

A. George, W. G. Poole, Jr., and R. G. Voigt

6. REPORT DATE August 1976	7a. TOTAL NO. OF PAGES 21	7b. NO. OF REFS 6

8a. CONTRACT OR GRANT NO. N00014-75-C-0879 b. PROJECT NO. NR044-459 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report 12 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

Approved for public release;
distribution unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Mathematics Program Office of Naval Research Arlington, Virginia 22217

13. ABSTRACT

Nested dissection orderings are known to be very effective for solving the sparse positive definite linear systems which arise from n by n grid problems. In this paper nested dissection is shown to be the final step of incomplete nested dissection, an ordering which corresponds to the premature termination of dissection. Analyses of the arithmetic and storage requirements for incomplete nested dissection are given and the ordering is shown to be competitive with nested dissection under certain conditions.

PRICES SUBJECT TO CHANGE

4. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
NESTED DISSECTION FINITE ELEMENT SYSTEMS						

A Variant of Nested Dissection for Solving n by n Grid Problems

A. George,[+] W. G. Poole, Jr.,[≠] and R. G. Voigt[*]

ABSTRACT

Nested dissection orderings are known to be very effective for solving the sparse positive definite linear systems which arise from n by n grid problems. In this paper nested dissection is shown to be the final step of incomplete nested dissection, an ordering which corresponds to the premature termination of dissection. Analyses of the arithmetic and storage requirements for incomplete nested dissection are given and the ordering is shown to be competitive with nested dissection under certain conditions.

[+] Department of Computer Science, University of Waterloo, Waterloo, Ontario.

[≠] Department of Mathematics, College of William and Mary, Williamsburg, Virginia 23185.

[*] Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, Virginia 23665.

This paper was prepared as a result of work performed under NASA Contract No. NAS1-14101 at ICASE, NASA Langley Research Center, Hampton, VA 23665. The research of the first author was also supported in part by the National Research Council of Canada under Grant A8111. The research of the second author was also supported in part by the Office of Naval Research Contract N00014-75-C-0879.

A VARIANT OF NESTED DISSECTION FOR SOLVING n BY n GRID PROBLEMS

1. INTRODUCTION

Recently there has been considerable effort ([1], [2]) to demonstrate the efficiency of an ordering strategy known as nested dissection for solving the sparse symmetric positive definite systems of linear equations associated with an n by n grid consisting of $(n-1)^2$ small squares, called elements. Any numbering of the grid points from 1 to n^2 yields the n^2 by n^2 matrix equation

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

where $A_{ij} = 0$ unless grid points i and j belong to the same element.

Although the analysis in this paper is for a square region, that is not as restrictive as it might seem. For example, a general region may be "subdivided" or "substructured" into a union of smaller regions, many of which will be squares if the subdivision is chosen properly.

The system (1.1) is solved by first factoring A into LL^T where L is lower triangular and then by solving the systems $Ly = b$ and $L^Tx = y$. When the factorization of A is carried out, it usually suffers from "fill"; that is, some locations of A that were zero become nonzero in the triangular factors. Thus we are led to consider the equivalent problem,

$$(PAP^T)Px = Pb,$$

where P is a permutation matrix chosen so as to reduce arithmetic requirements, to reduce the fill, or to achieve other objectives.

It is well-known ([1], [2]) that the nested dissection ordering reduces the factorization stage operation count from $O(n^4)$ for the row by row or band-oriented ordering to $O(n^3)$; similarly the fill occurring during the factorization is reduced from $O(n^3)$ to $O(n^2\log_2 n)$. Furthermore, nested dissection has been shown to be optimal in the asymptotic sense ([1], [4]).

In essence, the nested dissection algorithm repeatedly applies a basic step to square submeshes of the original mesh. The basic step consists of choosing a separating "+" (see Figure 1.1) which, as nearly as possible, equally divides the submesh into quadrants. The nodes in the quadrants are numbered before those on the cross. This basic step is applied successively until it is no longer possible to subdivide the mesh.

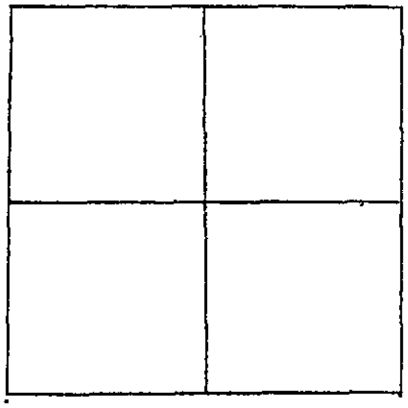

Figure 1.1. A basic step in nested dissection, indicating the separating cross.

The purpose of this paper is to study the consequences of terminating the mesh subdivision before completion. That is, at some stage, do not subdivide the quadrants further, but simply use a row by row, band-oriented ordering for the nodes in each quadrant. This idea is not completely new; the related idea of substructuring has been used for many years in structural engineering applications [5]. However, a careful analysis has never appeared in the literature.

There are at least three reasons for studying this ordering: the arithmetic and storage requirements have never been analysed; the overhead for handling the sparse matrix components is simpler than for nested dissection; and vector computers like the Control Data Corporation STAR-100 and the Texas Instruments, Inc. ASC can be better utilized because the lengths of vector operands can be greater than for nested dissection [3].

In section 2 a detailed description of the variant of nested dissection is given. The operation counts developed are for multiplicative operations. Section 3 contains information regarding the arithmetic and storage requirements of the new algorithm. It is shown that this variant is quite competitive with nested dissection.

2. INCOMPLETE NESTED DISSECTION ORDERINGS

Following [2], let V be the set of nodes of the n by n mesh and let C_1 consist of 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 $R_1^1 \cup R_1^2 = R_1 = V \backslash C_1$. Numbering nodes in R_1^1, followed by those in R_1^2, followed finally by 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 vertices in $R_1^\ell \backslash S_1^\ell$ before those associated with S_1^ℓ, $\ell = 1,2$, and then the remaining nodes as before, we induce the 7 by 7 partitioning in 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 node sets of the mesh corresponding to the i-th diagonal block are depicted schematically in Figure 2.1. So far we have not specified how to number the nodes indicated by 1-4 in Figure 2.1.

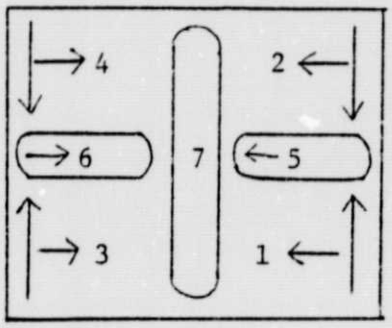

Figure 2.1 A one-level dissection of the mesh.

We could repeat the dissection process just described, numbering the nodes in mesh regions 1-4 in the same manner (but on a smaller scale) as for the original n by n mesh. This would involve choosing three <u>dissectors</u> for each of the four regions, and would create sixteen mesh regions of size about n/16 by n/16 whose numbering must still be specified. If we repeat this procedure until the choice of dissecrors finally leaves no nodes left to number, we obtain a <u>nested dissection</u> ordering [1]. These orderings yield $0(n^3)$ arithmetic counts and $0(n^2 \log_2 n)$ fill for the factorization. Of course when $n \neq 2^j - 1$, j a positive integer, this dissection process will not be completely uniform, and at the ℓ-th dissection stage the node sub-arrays whose numbering is yet to be prescribed may not all be exactly the same size. We will refer to these sub-arrays as <u>ℓ-level node arrays</u>.

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 node arrays grid row by grid row (or grid column by grid column). We will refer to such an ordering of the n by n grid as an <u>incomplete nested dissection ordering</u>. For example, if the nodes specified by regions 1-4 in Figure 2.1 were numbered column by column, this would be a one-level incomplete nested dissection ordering. The standard band ordering is a zero-level dissection ordering.

For a given ℓ-level incomplete nested dissection ordering, the dissectors together with the ℓ-level mesh subarrays induce a partitioning of the nodes into μ subsets, where

$$\mu = 2^{2\ell} + 3\sum_{k=0}^{\ell-1} 2^{2k} = 2^{2\ell} + (2^{2\ell} - 1).$$

The matrix A has $2^{2\ell}$ leading diagonal submatrices whose sizes are about $(n/2^{\ell})^2$ and whose bandwidths are about $n/2^{\ell}$; these correspond to the last-level mesh sub-arrays. The remaining diagonal submatrices correspond to the dissectors. Although they are initially sparse, it can be shown that by the time they are factored, they have become full [2].

In order to show how we arrived at our timing formulas, consider Figure 2.2 which displays the approximate structure of the one-level dissection ordering and partitioning specified by (2.2). The manner in which the last-level mesh subarrays were numbered is indicated by the arrows in Figure 2.1.

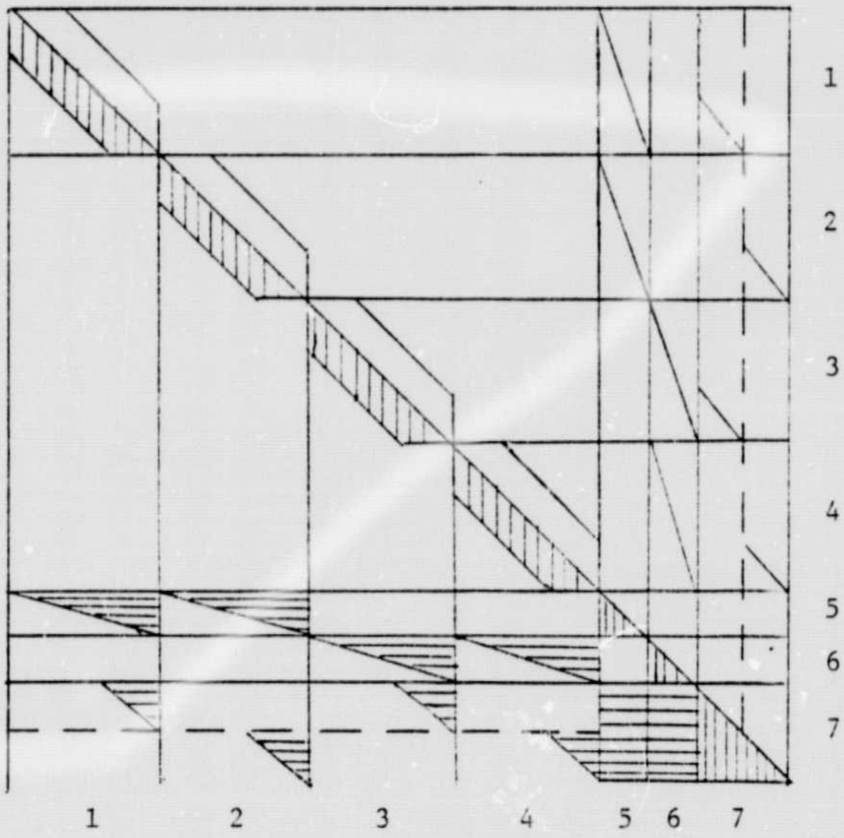

Figure 2.2 The approximate structure of A is shown in the upper triangle, and the structure and the storage format of the corresponding factor L are indicated by the hashing in the lower triangle.

Observe that the "block columns" corresponding to the μ diagonal blocks of A have sets of adjacent non-null rows creating non-null rectangular subarrays. Furthermore, these non-null subarrays are of two general types: those lying in block columns corresponding to dissectors are full, but those lying in block columns corresponding to last-level node arrays have some zeros and vary in structure as seen in Figure 2.2.

Now consider the depiction of a three level dissection ordering contained in Figure 2.3. Notice that there are three different kinds of last-level node subarrays, which for obvious reasons we will refer to as corner, side, and interior node arrays. These correspond to leading diagonal blocks of A, all with bandwidth about $n/2^{\ell}$ and size about $(n/2^{\ell})^2$. However, the number of non-null submatrices in their respective block columns of L differ. Similarly, the "+" shaped dissector triples with labels r, $1 \leq r \leq \ell$ are of three corresponding types, with corresponding similarities and differences. The character of the block columns corresponding to the last level blocks is displayed in Figure 2.4. The row dimension of the subarrays in each such block column is about $n/2^{\ell}$. Note that many of the components of the off-diagonal subarrays in Figure 2.4 are zero. When these zeros are in leading positions in their rows, the number of operations required for the factorization is significantly reduced. For example, if $\ell = 1$ or 2, the total time for the factorization can be reduced by as much as 50%. The locations of the zeros are determined by the manner in which the nodes of the last level subarrays are ordered. This observation was made by Noor, et al. in [5]. Assuming that a band solver is to be used for factoring the diagonal blocks, the optimum row by row ordering is attained by numbering toward the dissectors. For example, for a corner array, this leads to the numbering indicated by the arrows in Figure 2.1 and the off-diagonal blocks shown in Figure 2.4. The block columns corresponding to corner, side, and interior dissectors are similar but the matrix subarrays in the block columns are full.

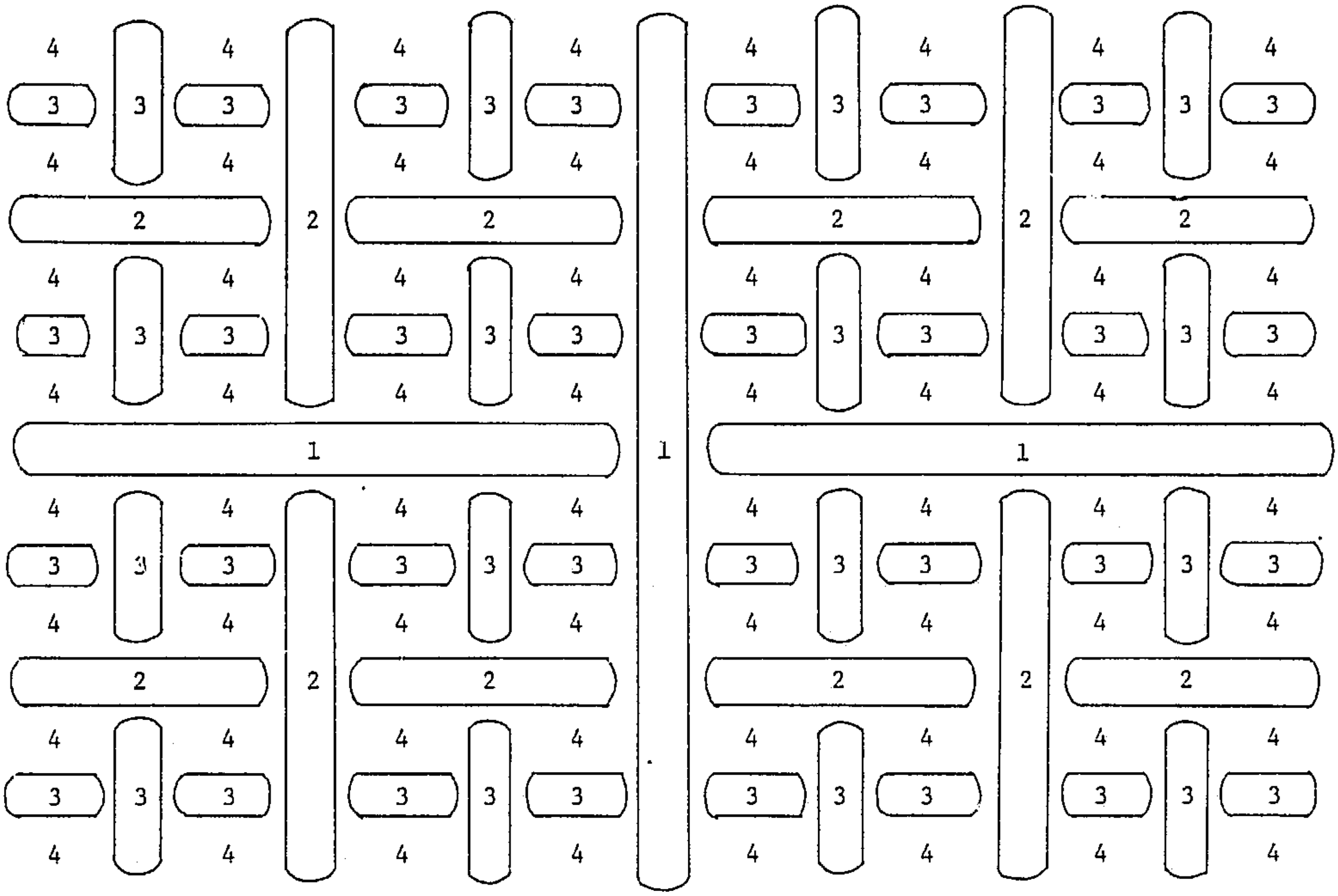

Figure 2.3 Depiction of a three level dissection ordering, where nodes with number k are numbered before nodes numbered k-1, and vertical node sets are numbered after horizontal node sets with the same label.

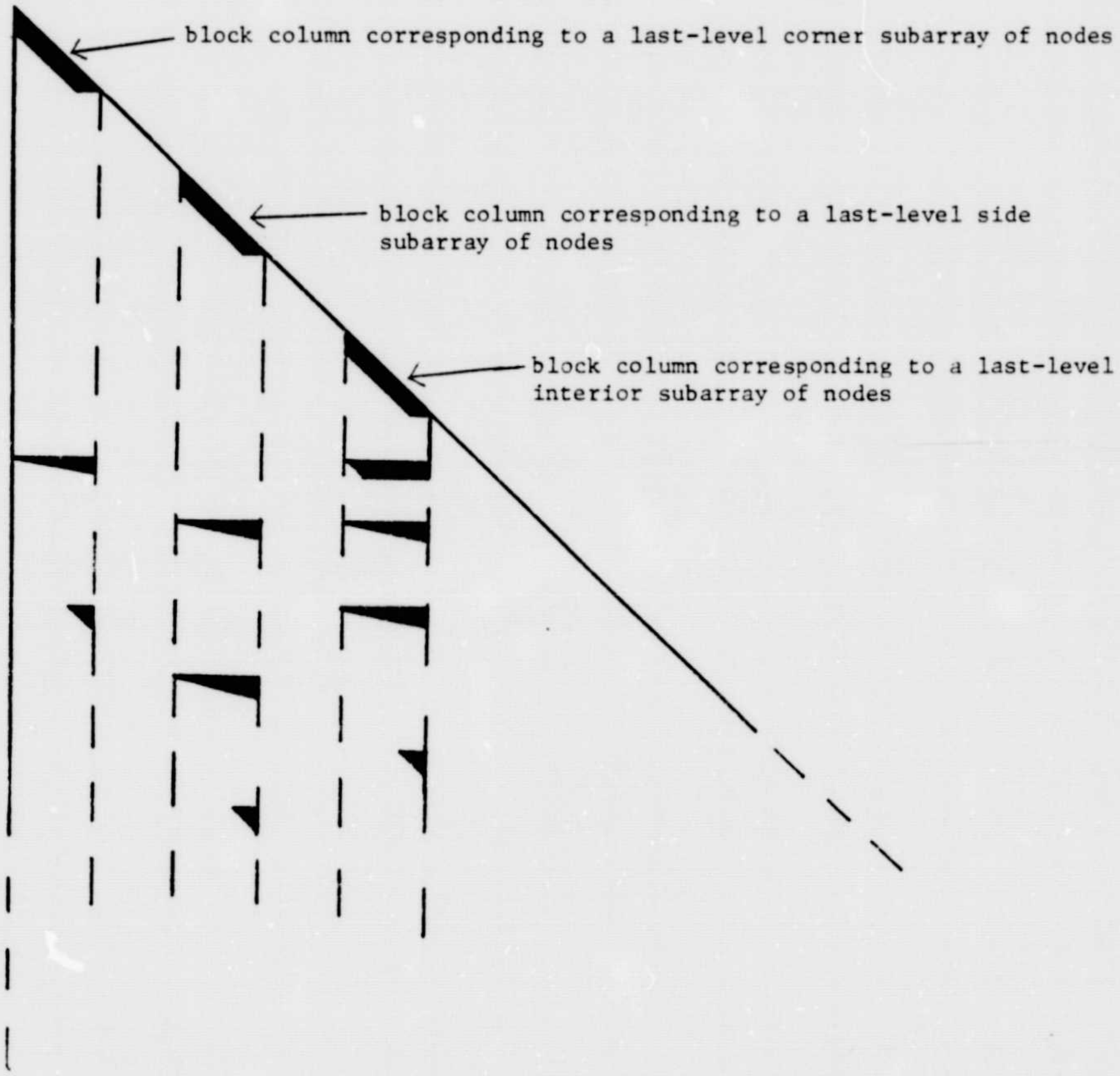
block column corresponding to a last-level corner subarray of nodes
block column corresponding to a last-level side subarray of nodes
block column corresponding to a last-level interior subarray of nodes

Figure 2.4 Diagram showing the character of the block columns of L corresponding to last-level corner, side, and interior node subarrays.

Our operation counts are obtained under the assumption that the banded character of the leading $2^{2\ell}$ diagonal blocks is exploited and the leading zeros of the off-diagonal blocks are exploited. Our basic strategy in obtaining operation counts is to determine the number of multiplications and divisions it takes to execute the factorization steps corresponding to each individual diagonal block. In order to do this for the leading diagonal blocks we need to establish the *structure* as well as the size of the submatrices in the corresponding block columns. However, since the submatrices in the block columns corresponding to dissectors are full, we need only be concerned with the total *number* of non-null rows in these block columns. Let S be a dissector which was chosen to subdivide the node subarray R. Then the number of non-null rows in the block column of L corresponding to S is $|\partial S|$, where ∂S is the set of nodes which are not in R but share at least one grid square with some node in R (a full explanation of this recipe can be found in [2]). Thus, the computation required to perform the $|S|$ factorization steps corresponding to the set S is equivalent to carrying out the first $|S|$ steps of the factorization of the $|S| + |\partial S|$ by $|S| + |\partial S|$ block two by two matrix shown below, which becomes what is shown at the right if we compute the partial factorization "in place".

$$\begin{pmatrix} A_S & B \\ B^T & A_{\partial S} \end{pmatrix} \longrightarrow \begin{pmatrix} L_S & W \\ W^T & A'_{\partial S} \end{pmatrix}$$

Here $L_S L_S^T = A_S$, $W = L_S^{-1}B$, and $A'_{\partial S} = A_{\partial S} - W^T W$. We denote the number of operations required to carry out this computation by $\tau_F(|S|, |\partial S|, k)$ where k denotes the number of dissectois connected to S. Similarly, the number of operations for the lower solve attributable to the block column of L corresponding to S is denoted by $\tau_L(|S|, |\partial S|, k)$, and for the upper solve, $\tau_U(|S|, |\partial|, k)$. Details can be found in Appendix C of [3].

Now consider the computation involved in carrying our the factorization steps corresponding to a last-level interior node array having p^2 factorization steps of the $p^2 + 4(p+1)$ by $p^2 + 4(p+1)$ matrix below, where A_{11} is p^2 by p^2 with bandwidth $p+1$, and the other diagonal blocks are either p by p or $p+2$ by $p+2$. For convenience, we assume they are all $p+1$ by $p+1$, since this will not change our estimates by very much.

$$(2.3)\qquad \begin{pmatrix} A_{11} & & & \text{symmetric} & \\ A_{12}^T & A_{22} & & & \\ A_{13}^T & A_{23}^T & A_{33} & & \\ A_{14}^T & A_{24}^T & A_{34}^T & A_{44} & \\ A_{15}^T & A_{25}^T & A_{35}^T & A_{45}^T & A_{55} \end{pmatrix}$$

The partial factorization, with the parts of L left "in place" in the array becomes

(2.4) $$\begin{pmatrix} L_{11} & & & & \\ W_{12}^T & A_{22}' & & \text{symmetric} & \\ W_{13}^T & A_{23}'^T & A_{33}' & & \\ W_{14}^T & A_{24}'^T & A_{34}'^T & A_{44}' & \\ W_{15}^T & A_{25}'^T & A_{35}'^T & A_{45}'^T & A_{55}' \end{pmatrix}$$

where $L_{11}L_{11}^T = A_{11}$,

(2.5) $$W_{1i} = L_{11}^{-1}A_{1i} \quad 2 \le i \le 5,$$

and

(2.6) $$A_{ij}' = A_{ij} - W_{1i}^T W_{1j}, \; 2 \le i \le 5, \; i \le j \le 5.$$

Now in carrying out the computation (2.5), we exploit the fact that A_{1i} has leading zeros as indicated in Figure 2.4, and we also exploit the structure in the W_{ij}'s when performing the matrix multiplication in (2.6). In addition, we of course exploit the banded character of A_{11} when we factor it and in (2.5), and we also exploit symmetry wherever possible. We denote the number of operations required to carry out the above computation by $\tau_F^I(p)$, where the superscript implies that the computation is associated with a last level interior p by p node subarray. In a similar way, we can derive functions $\tau_F^C(p)$ and $\tau_F^S(p)$ for last level corner and side node subarrays. The subscript F denotes factorization; similar functions with subscripts L and U can be defined corresponding to the work attributable to the block column during the lower and upper solve.

Having made these observations, using Figure 2.3 as a guide, it is now possible to derive our operation counts. They are only estimates because we assume

$n = 2^j - 1$ so that the dissection is uniform. However, some independent work has shown that estimates obtained using this strategy are extremely good for general n (see [2], Table 3.2). The details of these formulas may be found in Appendix C of [3].

The formula $\tau_F(q,p,k)$ represents the number of operations required to eliminate the nodes corresponding to a particular dissector. The first variable, q, represents the number of nodes in the dissector; p denotes the number of other nodes which are connected to the dissector; and k is the number of other dissectors connected to the dissector (see C.6.13 of Appendix C in [3]). Obviously, the number of operations required to execute the last $2n - 1$ steps of the factorization is

$$\tau_F(n,0,0) + 2\tau_F\left(\frac{n-1}{2},n,1\right), \tag{2.7}$$

where the first term is due to the vertical line of nodes labelled 1, and the second is due to the two horizontal lines of nodes labelled 1. The number of operations required for the factorization steps corresponding to the nodes labelled 2 is given approximately by

$$4\tau_F\left(\left(\frac{n-1}{2}\right),n,2\right) + 4\tau_F\left(\left(\frac{n-3}{4}\right), \frac{3n-1}{4},2\right) + 4\tau_F\left(\left(\frac{n-3}{4}\right), \frac{5n+1}{4},3\right). \tag{2.8}$$

In general, the number of operations required to eliminate nodes labelled r, $2 \leq r \leq \ell$ is given approximately by

$$\begin{aligned} &\left(2^{r-1} - 2\right)^2 \left\{\tau_F\left(\frac{n-2^{r-1}+1}{2^{r-1}}, \frac{4(n+1)}{2^{r-1}}, 4\right) + 2\tau_F\left(\frac{n-2^r+1}{2^r}, \frac{3(n+1)}{2^{r-1}}, 4\right)\right\} \\ &+ 4\left(2^{r-1} - 2\right)\left\{\tau_F\left(\frac{n-2^{r-1}+1}{2^{r-1}}, \frac{3(n+1)}{2^{r-1}} - 1, 3\right) + 2\tau_F\left(\frac{n-2^r+1}{2^r}, \frac{5(n+1)}{2^r} - 1, 3\right)\right\} \\ &+ 4\left\{\tau_F\left(\frac{n-2^{r-1}+1}{2^{r-1}}, \frac{2(n+1)}{2^{r-1}} - 1, 2\right) + \tau_F\left(\frac{n-2^r+1}{2^r}, \frac{3(n+1)}{2^r} - 1, 2\right) + \tau_F\left(\frac{n-2^r+1}{2^r}, \frac{5(n+1)}{2^r} - 1, 3\right)\right\} \end{aligned} \tag{2.9}$$

Here the first, second and third terms are due to interior, side, and corner dissector triples respectively.

The number of operations required for the factorization steps corresponding to the last-level blocks is given approximately by the following, where $\xi = (n-2^{\ell}+1)/2^{\ell}$.

$$(2.10) \qquad (2^{\ell} - 2)^2 \tau_F^I(\xi) + 4(2^{\ell} - 2)\tau_F^S(\xi) + 4\tau_F^C(\xi).$$

Using the symbolic computation system MACSYMA* [6] to sum (2.9) from $r = 2$ to ℓ and adding in (2.7) and (2.10) yields the estimate (see Appendix B of [3] for details):

if $\ell = 0$,

$$(2.11) \qquad M_F(n,0) \simeq \frac{1}{2} n^4 + \frac{13}{6} n^3 + 2n^2 - \frac{11}{3} n$$

if $\ell > 0$,

$$\begin{aligned}(2.12) \quad M_F(n,\ell) \simeq\ & n^4\left(\frac{14}{3} \times 2^{-2\ell} - 10 \times 2^{-3\ell} + 6 \times 2^{-4\ell}\right) \\ & + n^3\left(\frac{829}{84} - \frac{353}{12} \times 2^{-\ell} + \frac{148}{3} \times 2^{-2\ell} - \frac{1084}{21} \times 2^{-3\ell} + 24 \times 2^{-4\ell}\right) \\ & + n^2\left(-17 \times \ell + \frac{2003}{42} - \frac{963}{7} \times 2^{-\ell} + 86 \times 2^{-2\ell} - \frac{664}{7} \times 2^{-3\ell} + 36 \times 2^{-4\ell}\right) \\ & + n\left(- \frac{17}{6} \times 2^{\ell} + 6 \times \ell - \frac{1657}{84} + \frac{325}{12} \times 2^{-\ell} + \frac{128}{3} \times 2^{-2\ell} - \frac{524}{7} \times 2^{-3\ell} \right. \\ & \qquad \left. + 24 \times 2^{-4\ell}\right) \\ & + \left(2 \times 2^{2\ell} - \frac{17}{6} \times 2^{\ell} + 23 \times \ell - \frac{2459}{41} + \frac{935}{12} \times 2^{-\ell} + \frac{4}{3} \times 2^{-2\ell} \right. \\ & \qquad \left. - \frac{454}{21} \times 2^{-3\ell} + 6 \times 2^{-3\ell}\right)\end{aligned}$$

*MACSYMA is supported by the Defense Advanced Research Projects Agency work order 2095, under Office of Naval Research Contract #N00014-75-C-0561.

Similarly, an estimate for the lower solve, $M_L(n,\ell)$, is obtained by summing exactly the same terms with τ_F replaced by τ_L. We obtain, for $\ell = 0$,

$$M_L(n,\ell) \simeq n^3 + \frac{3}{2}n^2 - \frac{3}{2}n \tag{2.13}$$

For $\ell > 0$,

$$\begin{aligned} M_L(n,\ell) \simeq\ & n^3\left(3 \times 2^{-\ell} - 4 \times 2^{-2\ell} + 2 \times 2^{-3\ell}\right) \\ & + n^2\left(\frac{31}{4} \times \ell - \frac{35}{2} + 31 \times 2^{-\ell} - 18 \times 2^{-2\ell} + 6 \times 2^{-3\ell}\right) \\ & + n\left(-\frac{21}{2} \times 2^{\ell} + \frac{31}{2} \times \ell - 13 + 39 \times 2^{-\ell} - 24 \times 2^{-2\ell} + 6 \times 2^{-3\ell}\right) \\ & + \left(2^{2\ell} + \frac{3}{2} \times 2^{\ell} - \frac{17}{4} \times \ell - \frac{11}{2} + 11 \times 2^{-\ell} - 10 \times 2^{-2\ell} + 2 \times 2^{-3\ell}\right) \end{aligned} \tag{2.14}$$

The number of multiplications for the upper solve, M_U, as well as the storage requirements (see section 3) are also given by (2.14).

3. OPERATION AND STORAGE REQUIREMENTS

It is not obvious which are the dominant terms of (2.12). For example,

$$M_F(n,1) \simeq 0.292n^4 + 2.54n^3 + O(n^2) \tag{3.1}$$

and the n^4 term is dominant for $n > 8$. On the other hand,

$$M_F(n,3) \simeq 0.055n^4 + 6.87n^3 + O(n^2) \tag{3.2}$$

and n must be greater than 125 before the n^4 term is larger than the n^3 term. Finally, for the largest value of ℓ, $\log_2(n+1) - 1$,* we get

*In this section we assume that $n = 2^j - 1$, for j an integer, in order to simplify the analysis. The general observations hold for an arbitrary value of n.

$$(3.3) \quad M_F\Big(n,\ \log_2(n+1) - 1\Big) \simeq 9.87n^3 - 17n^2\log_2(n+1) + 22.1n^2 + O\Big(n \log_2(n+1)\Big)$$

The estimate (3.3) for the multiplication count of nested dissection differs only slightly in the third term from a similar formula given in [2]. The difference can be attributed to a slight improvement in the estimate of the multiplication count.

Table 3.1 and Figure 3.1 display the multiplication counts for the factorization stage of the incomplete nested dissection ordering. The number of operations decreases quite rapidly for small ℓ as the level of dissection increases. However, as ℓ approaches its largest possible value, very little is gained. For all of the cases listed in Table 3.1, the final step of dissection decreases the multiplication count by less than 5%. On the other hand, the last step increases the complexity of data management, as will be described later.

ℓ \ n	15	31	63	127	255
banded	31.1 (3)	512 (3)	8.30 (6)	133.0 (6)	2142 (6)
1	23.9 (3)	347 (3)	5.24 (6)	81.1 (6)	1280 (6)
2	23.9 (3)	290 (3)	3.71 (6)	51.3 (6)	753 (6)
3	23.0 (3)	243 (3)	2.53 (6)	28.1 (6)	345 (6)
4		235 (3)	2.20 (6)	20.5 (6)	202 (6)
5			2.16 (6)	18.9 (6)	165 (6)
6				18.7 (6)	157 (6)
7					156 (6)

Table 3.1 Multiplication count for factorization stage of incomplete nested dissection and banded orderings for several values of n and $\ell = 1\ (1)\ \lfloor \log_2(n+1) - 1 \rfloor$.

Now consider multiplication counts for the lower solve, as displayed in Table 3.2 and Figure 3.2 (the counts for the upper solve are the same). Since each component of the matrix L is handled exactly once in the lower solve, it is obvious that the storage requirements for the components of the matrix are identical to the

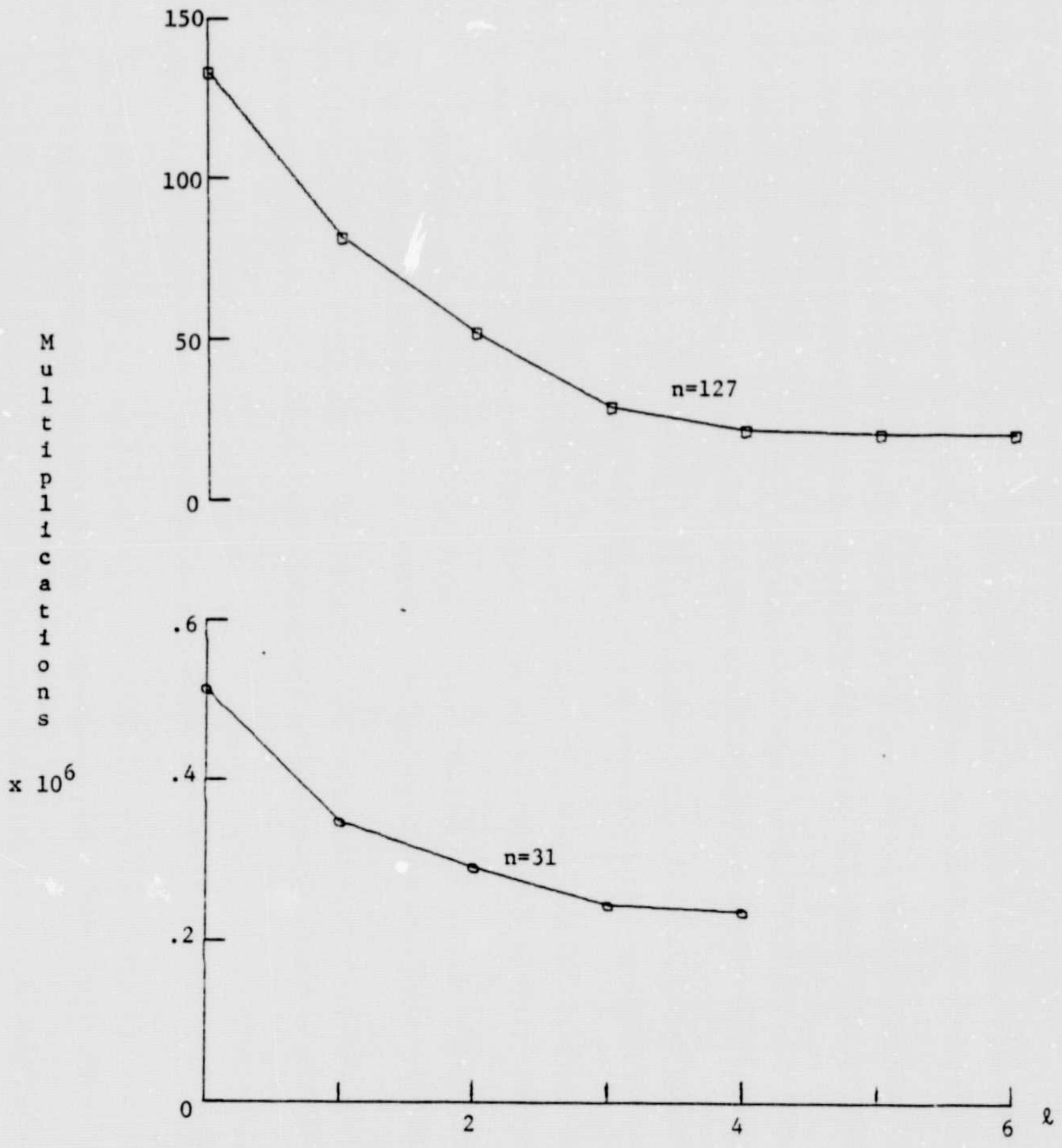

Figure 3.1 Multiplication counts for factorization stage of incomplete nested dissection ordering for $n = 31$ and $n = 127$.

multiplication count of the lower solve. The remaining comments in this section will be concerned with storage although analogous comments can be made about the lower and upper solves.

n ℓ	15	31	63	127	255
banded	3.58 (3)	30.7 (3)	254 (3)	2.064 (6)	16.65 (6)
1	2.92 (3)	24.1 (3)	195 (3)	1.568 (6)	12.56 (6)
2	2.63 (3)	19.9 (3)	151 (3)	1.162 (6)	9.11 (6)
3	2.54 (3)	17.1 (3)	113 (3)	.793 (6)	5.84 (6)
4		16.5 (3)	97 (3)	.586 (6)	3.83 (6)
5			95 (3)	.509 (6)	2.87 (6)
6				.499 (6)	2.53 (6)
7					2.49 (6)

Table 3.2 Multiplication counts for lower solve (same for upper solve and storage) stage of incomplete nested dissection and banded orderings for several values of n and $\ell = 1\ (1)\ \lfloor \log_2(n+1) - 1 \rfloor$.

As was the case for the factorization stage, very little is gained by the final step of dissection. The storage requirement drops less than 5% from the penultimate step to the last step for all values of n shown in Table 3.2. Moreover, the following analysis of the storage requirements of an implementation of incomplete nested dissection shows no decrease at all.

Consider an implementation scheme for incomplete nested dissection analogous to that described by George [2] for nested dissection. George's scheme requires about $14b + 2n^2$ additional memory locations for information describing the blocks of L, where b is the number of diagonal blocks. This "data management" information includes dimensions of blocks, location of the first element in each row,

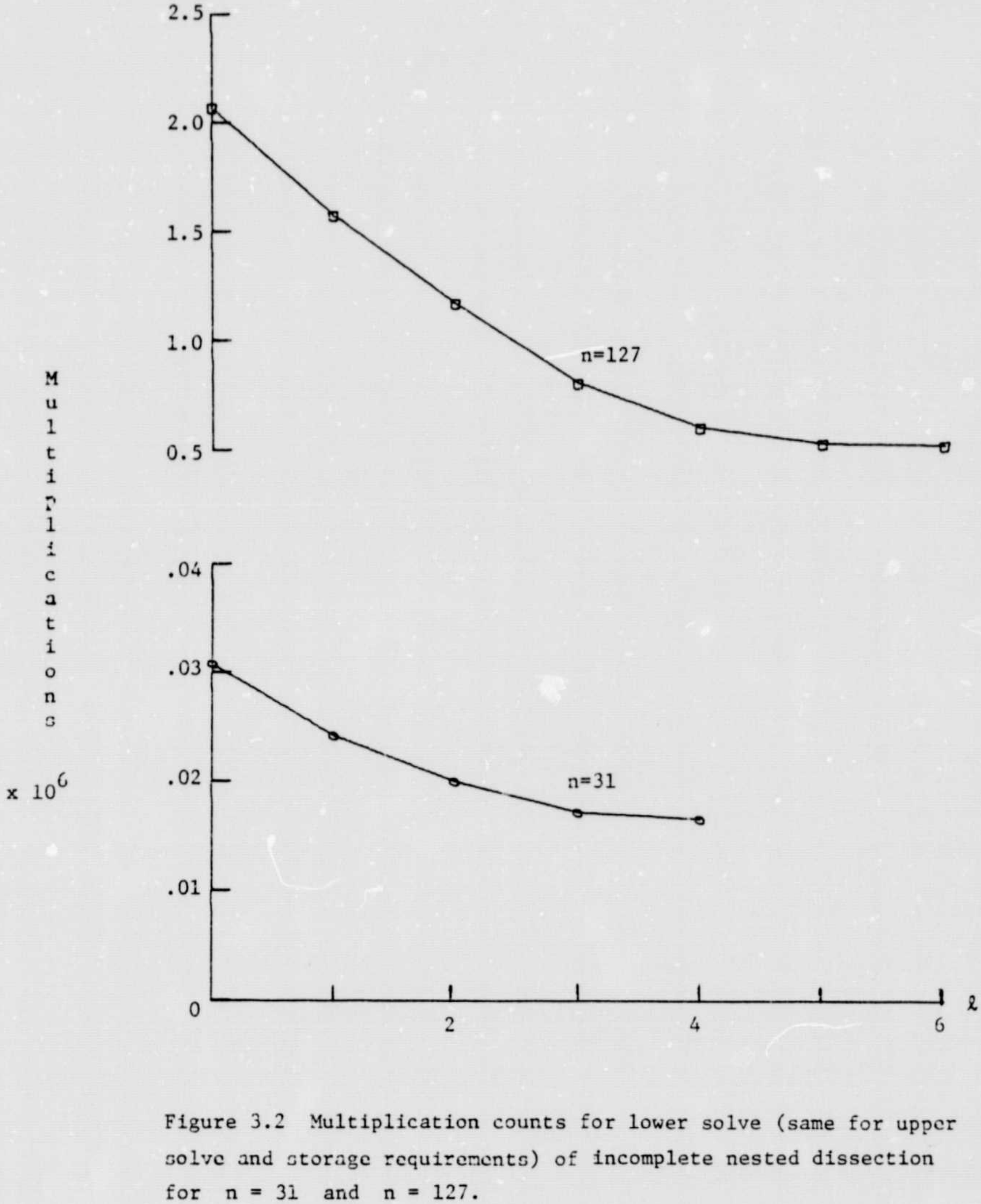

Figure 3.2 Multiplication counts for lower solve (same for upper solve and storage requirements) of incomplete nested dissection for n = 31 and n = 127.

pointers, etc. For nested dissection b is approximately $\frac{n^2}{2}$ and this additional storage is

$$9n^2 + O(n).$$

Now consider incomplete dissection where the dissection is stopped 1 short of completion -- $\ell = \log_2(n+1) - 2$. As mentioned in section 2, the number of diagonal blocks is

$$\mu = 2 \times 2^{2\ell} - 1 = \frac{n^2}{8} + O(n). \tag{3.4}$$

It follows that George's scheme requires oniy

$$\frac{15}{4} n^2 + O(n)$$

locations. By stopping short 1 step, one may save as much as $\frac{21}{4} n^2$ locations for data management storage. Taking this into account in Table 3.2, one sees that, assuming George's implementation scheme, the final step of dissection (i.e., nested dissection) is considerably more costly in terms of total storage than stopping short 1 step. Moreover, if one takes into account the storage overhead, it can be shown that stopping short by 2 steps is competitive with nested dissection.

In summary, it follows that the penultimate level of the variant of nested dissection may reduce storage requirements of nested dissection with very little increase of computer time.

REFERENCES

[1] George, A., "Nested Dissection of a Regular Finite Element Mesh," SIAM J. Numer. Anal., 10 (1973), pp. 345-363.

[2] George, A., "Numerical Experiments Using Dissection Methods to Solve n by n Grid Problems," SIAM J. Numer. Anal., to appear.

[3] George, A., W. G. Poole, Jr., and R. G. Voigt, "Analysis of Dissection Algorithms for Vector Computers," ICASE Report No. 76-17, June 1976.

[4] Hoffman, Alan J., 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.

[5] Noor, A. K., H. A. Kamal, and R. E. Fulton, "Substructuring Techniques - Status and Projections," Journal of Computers and Structures, to appear.

[6] Project MAC-M.I.T., "MACSYMA Reference Manual," Massachusetts Institute of Technology, Cambridge, Massachusetts, 1975.