LAPACK WORKING NOTE 63 (UT CS-93-202)
LINE AND PLANE SEPARATORS *

PADMA RAGHAVAN †

Abstract. We consider sparse matrices arising from finite-element or finite-difference methods. The graphs of such matrices are embedded in two or three dimensional Euclidean space, and the coordinates of their vertices are readily available. Such coordinate information was used earlier to develop a parallel Cartesian nested dissection heuristic that computes a fill reducing ordering for a matrix with an embedding in two dimensions. We extend Cartesian nested dissection to graphs embedded in three dimensions and to compute an ordering for a non-symmetric matrix A without explicitly forming the graph of $A^T A$. We show that for an *r-local* graph with N vertices embedded in d dimensions (d=2, 3), a single step of Cartesian nested dissection computes a separator of size $O(N^{1-1/d})$. The separator also divides the graph into two subgraphs, each with at least $N/5$ vertices when $d = 2$ and at least $N/7$ vertices when $d = 3$. Computational results indicate that the algorithm performs rather well for a wide variety of graphs.

Key words. parallel algorithms, sparse linear systems, ordering, Cartesian coordinates, nested dissection, Cholesky factorization

AMS(MOS) subject classifications. 65F, 65W

1. Introduction. The solution of a sparse linear system $Ax = b$ associated with a finite-element or finite-difference mesh is often required as part of many scientific and engineering applications. Exploiting sparsity in the solution process and computing the solution in parallel are of significant interest. Factorization methods for solving the linear system involve a symbolic phase followed by a numeric phase. In the symbolic phase, a symmetric permutation $\tilde{A} = PAP^T$ is computed such that factors of the matrix $\tilde{A}$ suffer low fill-in, i.e., remain sparse. When the matrix A is symmetric and positive definite, the factorization process is numerically stable regardless of the ordering, so the permutation can be computed independent of numeric values. In the subsequent numeric phase, the reordered matrix is factored; the task parallelism in the factorization process stems from sparsity. A nested dissection ordering partitions the sparse matrix into a set of submatrices that can be processed independently and in parallel, and is thus suitable for parallel computation. For the solution process to be completely parallel, the nested dissection ordering itself must be computed in parallel.

In the next section, we describe nested dissection and provide a brief survey of nested dissection methods. To enable easy exposition of our main results, an overview of our earlier Cartesian nested dissection algorithm is provided in Section 3. Algorithmic extensions are developed in Section 4. In Section 5 we provide bounds on the sizes of the separator and resulting subgraphs when our algorithm is applied to an *r-local* graph. Section 6 contains computational results, and conclusions are presented in Section 7. Throughout this paper, the symbol $\mathcal{R}^{\lceil}$ is used to denote $d-$dimensional Euclidean space, for $d = 2$ or 3.

2. Nested Dissection. Nested dissection was first proposed by Alan George [1] as an optimal method for ordering sparse systems associated with 2-dimensional

* This research was supported by the Defense Advanced Research Projects Agency through the Army Research Office under contract number DAAL03-91-C-0047.

† National Center for Supercomputing Applications, University of Illinois, 405 N. Mathews Ave., Urbana, IL 61801.

regular grids. Nested dissection recursively restructures a sparse matrix by computing separators and renumbering vertices of an associated graph. The graph of an $N \times N$ symmetric matrix A is an undirected graph with N vertices and an edge between vertices i and j iff the matrix entry a_{ij} is nonzero. A separator V_s of a graph G partitions the set of vertices into three sets V_1, V_2 and V_s such that no edge joins a vertex in V_1 with a vertex in V_2. We now illustrate one step of nested dissection. Let V_1 and V_2 and V_s be a partition of the vertices of the graph of A such that V_s is a vertex separator. By numbering contiguously columns corresponding to V_1, V_2 and V_s, with those corresponding to V_s numbered last, the matrix will be reordered into bordered block diagonal form:

$$\begin{bmatrix} A_1 & 0 & S_1 \\ 0 & A_2 & S_2 \\ {S_1}^T & {S_2}^T & A_s \end{bmatrix}.$$

With the above form, sparsity is preserved as the zero blocks remain zero; secondly, numeric operations can be applied to A_1 and A_2 independently and in parallel. Observe that if the size of V_s is small, the blocks S_1, S_2 and A_s are also small, resulting in a smaller number of nonzeroes. Therefore, it is desired that the separator V_s contain no more than $\beta f(N)$ vertices, where $f(N)$ is $O(N^\gamma)$, $\gamma < 1$, and β is a small constant. The relative sizes of A_1 and A_2 would determine the load balance for a natural parallel implementation. As a consequence, both V_1 and V_2 should contain at least αN vertices, where $0 < \alpha < 1$. A recursive application of the step described above would give rise to a decomposition tree whose nodes are associated with operations on submatrices. Such a decomposition tree would be suitable for parallel processing of associated submatrices.

Sequential algorithms for nested dissection have traditionally used a combinatorial approach to computing separators. One such algorithm which is used extensively in sparse matrix computations is the "general nested dissection" heuristic of George and Liu [3]. Nested dissection was generalized to the class of planar graphs in [8] where it was shown that all N-vertex planar graphs can be split into subsets whose sizes are bounded by $(2/3)N$ with a separator size of no more than $2\sqrt{2N}$. However, the $O(N)$ serial algorithm for computing a single separator is quite complicated; an interesting feature is that this algorithm relies on a representation of the graph in terms of a planar embedding. Computing a nested dissection ordering in *parallel* on MIMD machines is a very challenging task, and recent algorithms have tended to use a non-combinatorial approach.

An algebraic approach to computing separators was proposed by Pothen, Simon and Liou [12]. Their heuristic is based on the idea that lower bounds on separator sizes can be obtained in terms of the eigenvalues of the Laplacian matrix of the sparse graph. The time for computing a separator depends on the convergence properties of the algorithm used to compute the eigenvector to desired accuracy. In principle, the algorithm is parallelizable in the sense that the algorithm for computing an eigenvector is parallelizable.

Geometric methods to compute a separator use an embedding of the graph in $\mathcal{R}^{\lceil}$. The approach of Vavasis [15] obtains asymptotically optimal separators for grid graphs (lattices) and generalizes to *r-local* graphs in $\mathcal{R}^{\ni}$. Algorithms proposed by Miller, Vavasis, Thurston and Teng [9, 10], compute "sphere" separators of size $O(N^{1-1/d})$. At the heart of their method lies the problem of computing a *center point*, for which there is no efficient deterministic algorithm. However, they show that a single separator can be computed in $O(N)$ randomized time. We are interested in simple and practical parallel algorithms for computing a nested dissection ordering using an embedding of the graph. Our Cartesian nested dissection algorithm [6] is shown to be an effective heuristic for computing separators in parallel.

3. Cartesian Nested Dissection. The reader is referred to [6] for a detailed discussion of the algorithm; here we provide only a brief summary of the key ideas. We assume an embedding of the graph in $\mathcal{R}^{\in}$; for managing data structures efficiently, the natural coordinate values in each dimension are sorted and consecutive integer values are assigned to distinct real values in sequence. With respect to Cartesian nested dissection, by coordinates values we mean these new integer values. We treat edges as intervals along the x- and y- dimensions. Consider a vertex v whose coordinates are given by $x(v)$ and $y(v)$; the vertex is entered into lists at coordinate values $x(v)$ and $y(v)$. We consider two intervals associated with each edges (u, v) of the form $[x(u), x(v)]$ and $[y(u), y(v)]$. An interval $[x(u), x(v)]$ associated with an edge (u, v) is listed in a structure at coordinates $x(u)$ and $x(v)$, with the entry at the smaller value marked as *begin* and at the larger value as the *end*. We compute separators by searching along the x- and y- dimensions. Consider the search along the x- dimension. With the aid of suitable data structures, the lists of vertices and intervals along the x- dimension are traversed in ascending order of coordinate values to determine a specific coordinate s. The coordinate s corresponds to the smallest estimated size of a separator. Let $\eta(i)$ denote the estimate of the size of a separator; then $\eta(i)$ is the number of vertices at coordinate i plus the number of edges (intervals) that span i. With some initial sorting, each such search is accomplished in time $O(e)$, where e is the number of edges. The algorithm does a similar search along the y- dimension to extract a coordinate r with minimum estimate of separator size over all y- coordinates. The coordinate corresponding to $\min\{\eta(s), \eta(r)\}$ is used to induce a line separator. The actual separator is computed by using a group-tree to search efficiently for edges that straddle the separating line.

The method described attempts to compute a small separator; however, it is also important to ensure that the resulting pieces are of roughly equal size. To ensure this, the separating line is constrained to lie in a subrange based on a user specified value of α, the balance factor. In each dimension, values of η are computed only for coordinates in a range containing projections of at least $(1 - 2\alpha) \mid V \mid$ vertices. For example, along the x- dimension, the selected range $[a, \ldots, b]$ is such that subgraphs to the left of a and to the right of b each contain at least $\alpha \mid V \mid$ vertices.

¿From the preceding discussion, it can be seen that computing a separator requires identifying a separating line, which in turn involves counting vertices and edges along each dimension. Such counting is formulated in terms of parallel prefix operations. Constructing a separator requires searching for edges (intervals) that straddle the separating coordinate. This is done independently and in parallel by partitioning the edges among processors and executing group-tree searches on each processor. The distributed algorithm is shown to have time and space complexity of $O(E/P \ logN)$ and $O(E/P)$ under the assumption that the original vertices and edges of the graph have been evenly distributed among P processors.

4. Extensions to Cartesian Nested Dissection. In this section, we present extensions to our Cartesian nested dissection algorithm for graphs embedded in $\mathcal{R}^{\ni}$ and for solving general sparse non-symmetric systems associated with an embedding.

4.1. Three Dimensional Problems. The Cartesian nested dissection heuristic can be naturally extended to compute separators for a graph $G = (V, E)$ whose vertices are embedded in $\mathcal{R}^{\ni}$. In the case of $\mathcal{R}^{\in}$, the separating coordinate resulted in a separating line. For an embedding in $\mathcal{R}^{\ni}$, a separating coordinate results in a separating plane. The search for a coordinate with minimum η is now performed in all three dimensions. The coordinate with the smallest η is selected to induce a separating plane. Group trees are employed for fast selection of edges that straddle a given coordinate level. As before, the user is allowed to specify an α value to ensure that the resulting subgraphs are balanced in size. Without loss of generality, assume the coordinate s is chosen as the separating plane along the x- dimension. Then the separator consists of all vertices with x- coordinate s and end points of edges (u, v) such that $x(u) < i < x(v)$ or $x(v) < i < x(u)$. The algorithm has sequential time and space complexity of $O(E\ logN)$ and $O(E)$, where E is the number of edges. With P processors, the parallel time complexity is $O(E/P\ logN)$ and the space complexity $O(E/P)$.

4.2. Non-symmetric Problems. It is often of interest to solve the system $Ax = b$ when the matrix A is non-symmetric and sparse. Let A have $M \geq N$ rows and rank N. Then, direct methods for the solution require either an LU or QR factorization. For either factorization, a fill reducing permutation for the symmetric matrix A^TA can be used induce a sparsity-preserving column order of A [2, 5]. Observe that a row of A forms a clique in the graph of A^TA. When the columns correspond to vertices embedded in $\mathcal{R}^{\lceil}$, this observation can be used along with our Cartesian nested dissection algorithm to compute a fill reducing ordering without forming A^TA.

We assume that we are given the nonzero structure of the rows of a non-symmetric matrix A. We also assume that columns of A correspond to vertices embedded in $\mathcal{R}^{\lceil}$. In this setting, we use the nonzero structure of each row of A to compute implicitly separators of the graph of $B = A^TA$. A compact Cartesian representation $C(B)$ of $G(B)$ is constructed in which edges are coalesced into "macro-intervals." Cartesian nested dissection is extended appropriately and applied to $C(B)$.

A compact representation of $G(B)$ should provide a low cost mechanism for identifying edges that straddle a separating level without explicitly storing all edges. In this compact representation, the structure of A is stored, and its relationship to the structure of B is used to identify edges required in computing a separator. Let the columns of A and B correspond to vertices labeled $\{1, \ldots, N\}$, and let $(x(i), y(i), z(i))$ be the Cartesian coordinates of vertex $i, 1 \leq i \leq N$. Furthermore, let $struc(r_k)$ denote the nonzero column subscripts in row k of A; the set $struc(r_k)$ is a clique in $G(B)$. Let $struc(r_k) = \{i_1, \ldots, i_r\}$, and let $x(i_1) \leq x(i_2) \cdots \leq x(i_r)$. Rather than store edges between all members of $struc(r_k)$, we represent $struc(r_k)$ by the "macro-interval" $[x(i_1), x(i_r)]$ in $C(B)$. The macro-interval contains pointers to identify the members of the clique. Such macro-intervals are stored along each dimension for each row of A, requiring storage bounded by $d \mid A \mid$. All macro-intervals that span s are identified using a group tree search. Each macro-interval is explored in turn to identify members with the appropriate coordinate value less than s; such members are vertices at end points of edges in $E(B)$ that straddle s. If we set E to be the number of nonzeroes in A, then the bounds given earlier for the running time remain applicable.

5. Bounds on Separator Size. We now show that the Cartesian nested dissection algorithm is provably good for a class of *r-local* graphs. These graphs were defined by Vavasis in [15].

Definition. Let $||.||_\infty$ denote the L_∞ norm on $\mathcal{R}^{\lceil}$ and let the graph G be embedded in $\mathcal{R}^{\lceil}$. Let μ be the minimum distance between any two nodes of G and let σ be the maximum distance between any pair of nodes that are connected by an edge.

The graph G is *r-local* if $\lceil \sigma/\mu \rceil + 1 = r$; i.e., the length of the longest edge is at most $r-1$ times the smallest internode distance.

Graphs arising from finite-difference methods are *r-local*. For graphs associated with finite-element methods, the *r-local* property will in general be true of domains triangulated with elements of roughly uniform size. For such graphs, simple geometric searching along the coordinate axes results in the construction of separators of small size.

To prove bounds on the quality of a line separator for $d = 2$ and a plane separator for $d = 3$, we consider the set of vertices (points) to be *sparse* in $\mathcal{R}_\lceil$; i.e., we assume that any box of size μ contains at most b_d vertices, where b_d is a small constant. In general, such an assumption is justified [13]. We first show in Lemma 5.1 that for any sparse set of N vertices, a box of width $r\,\mu$, perpendicular to one of the coordinate axes, can be found such that sparse subsets on either side of this box contain $O(N)$ nodes. Furthermore, if $\mathcal{S}$ denotes the set of points in the box, then $\mathcal{S}$ contains at most $O(N^{1-1/d})$ vertices. A similar result of Bentley and Shamos is given in [13].

LEMMA 5.1. *Consider a set S of N points in $\mathcal{R}^\lceil$ such that a small constant b_d is an upper bound to the number of points contained in a box of size μ. Let r be a small constant independent of N and let $N \geq 343\, b_d$. Then, there exists a set $\mathcal{S}$ consisting of points in box of width $r\,\mu$, perpendicular to one of the coordinate axes, such that:*

(i) both subsets S_1 and S_2 on either side of $\mathcal{S}$ contain at least N/τ_d points; $\tau_2 = 5$ and $\tau_3 = 7$;

(ii) there are at most $\gamma_d N^{1-1/d}$ points in $\mathcal{S}$, where $\gamma_d = 1.71\, b_d r$.

Proof: The proof is by contradiction; we show that unless conditions *(i)* and *(ii)* are satisfied, a subvolume must have *too few* and *too many* points at the same time. Assume the set of points is sorted in increasing order of the coordinate value in each dimension. Consider each dimension i in turn; find a middle range of coordinates $[a_i, \ldots, b_i]$ containing $N(1 - 2/\tau_d)$ points. For example, along the x- dimension the value a_i is such that there are N/τ_d points to the left of a_i and the value b_i is such that there are N/τ_d points to the right of b_i. Consider projections of points onto dimension i; then condition *(ii)* is satisfied if there is a strip of width $r\,\mu$ along any of the dimensions containing $\gamma_d N^{1-1/d}$ points.

Assume that no such strip exists. This implies that projections onto dimension i of points in every set of $(\gamma_d\, N^{1-1/d})$ points lie in a strip of size *less* than $r\,\mu$. ¿From the assumption of sparsity, the width of the region bounded by a_i and b_i is given by:

$$\frac{N(1-2/\tau_d)}{\gamma_d\, N^{1-1/d}}(r\,\mu).$$

Let $\mathcal{V}$ denote the volume bounded by planes corresponding to a_i and b_i along each dimension i. Once again, by the assumption of sparsity, the number of points contained in $\mathcal{V}$, denoted by $\mid \mathcal{V} \mid$, satisfies

$$\mid \mathcal{V} \mid \;\leq\; (\frac{\infty - \in/\tau_\lceil}{\gamma_\lceil})^\lceil (\nabla^\lceil \lfloor_\lceil \mathcal{N}).$$

However, by choice of values a_i and b_i, at most $(2/\tau_d)N$ points are excluded from $\mathcal{V}$ in each dimension. So the volume $\mathcal{V}$ must satisfy

$$\mid \mathcal{V} \mid \;\geq\; (\infty - \in\lceil/\tau_\lceil)\mathcal{N}.$$

It can be easily verified that this results in a contradiction for the specified values of τ_d and γ_d when $N \geq 343\, b_d$. Hence the proof.

Observe that in an *r-local* graph, all edges are of length less than $r\,\mu$ and so the vertices in the set $\mathcal{S}$ form a separator. This fact is used in Lemma 5.2 to show that Cartesian nested dissection computes a separator of size $O(N^{1-1/d})$ that splits an *r-local* graph into two subgraphs of size $O(N)$.

LEMMA 5.2. *Let $G = (V, E)$ be an r-local graph with N vertices embedded in $\mathcal{R}^{\lceil}$. The Cartesian nested dissection algorithm, with $\alpha = 1/\tau_d$, computes a vertex separator V_s with the property that*

$$| V_s | \leq \gamma_d N^{1-1/d}.$$

The values of τ_d and γ_d are as defined in Lemma 5.1. The separator V_s splits V into sets V_1 and V_2 such that

$$| V_1 |, | V_2 | \geq (N/\tau_d) - \gamma_d N^{1-1/d} \text{ and } | V_1 |, | V_2 | \leq (1 - N/\tau_d).$$

Proof: Consider an *r-local* graph embedded in $\mathcal{R}^{\lceil}$ and the sets $\mathcal{S}$, S_1 and S_2 defined by Lemma 5.1. Vertices in the set $\mathcal{S}$ form a separator since the graph is *r-local*.

Consider Cartesian nested dissection with $\alpha = 1/\tau_d$; the separator computed by the algorithm is one induced by a coordinate s with the *smallest* value of $\eta(s)$. Let V_s be the separator computed; $| V_s | \leq \eta(s)$. By the choice of α, we know that a separator $\mathcal{S}$ exists as defined. The bound on $| V_s |$ follows by observing that $\eta(s) \leq | \mathcal{S} |$. The bounds on the sizes of V_1 and V_2 follow directly from the chosen value of α.

6. Computational Results. We compute Cartesian nested dissection orderings for several irregular meshes. We provide measures of the quality of the ordering in terms of separator sizes and the relative sizes of induced subgraphs. The observed results are rather encouraging. The test suite consists of 14 meshes embedded in $\mathcal{R}^{\in}$ and 11 embedded in $\mathcal{R}^{\ni}$, as described in Tables 1 and 2. For each mesh, a count of vertices and edges is provided along with the value of r, the ratio of the length of the longest edge to the smallest internode distance in the L_∞ norm. The meshes embedded in $\mathcal{R}^{\in}$ were obtained from triangulations of domains that are either simple polygons or polygons with holes. With respect to Table 1, the airfoils and the problem labeled "binaca" arise from practical applications. The problems, "eppstein", "parc", "venkat 1" and "venkat 2" were generated by various mesh generators. The remaining problems were generated using a commercial structural analysis package called PATRAN to resemble meshes in the test collection in [4], which are typical of meshes used in structural analysis and in the study of heat conduction. However, we constructed these problems so that they would pose a challenge to Cartesian nested dissection with respect to at least the first few separators. In short, we made these meshes have very small (and hence many) elements along lines that would result in a split into pieces of roughly equal size.

Most meshes in Table 2 were obtained from tetrahedralization of objects such as an aircraft flap, or of a domain enclosed between two spherical sections. Two of the meshes in Table 2, "sphere 5" and "sphere 6", are surface triangulations of three dimensional objects. Observe that meshes in the test suite are highly graded and irregular, with element sizes that vary by factors larger than 1000 in the L_∞ norm.

We report on the size of the separator and the size of the resulting pieces for Cartesian nested dissection of problems in the test suite. At some stage in nested dissection, let V denote the set of vertices in the subgraph to be dissected. Let S

Label	$\mid V \mid$	$\mid E \mid$	r	Comments
airfoil 1	4,253	12,289	1,538	airfoil mesh by Barth and Jesperson
airfoil 2	4,720	13,722	4,640	same as airfoil 1
airfoil 3	15,606	45,878	15,224	4-element airfoil mesh by Barth
binaca	3,572	10,444	1,665	mesh around 2 airfoils
eppstein	547	1,566	19	triangulation with angles between $36°$ and $80°$ by Eppstein
graded box	7,861	22,983	4,609	small elements at bottom right corner
graded L	6,142	18,153	1,540	small elements at a middle corner
graded +	6,043	16,866	318	small elements at a middle corner
hollow box	5,512	16,224	159	small elements around hollow
parc	1,240	3,355	195	PARC cut from a rectangle, small elements around letters
pinched hole	8,848	26,208	103	pentagon with hole, small elements around hole
six hole	9,971	29,240	53	small elements around each hole
venkat 1	10,089	29,720	7,285	concentric layers with elements of increasing size
venkat 2	460	1,303	114	polygon with hole

TABLE 1
Description of two-dimensional meshes.

Label	$\mid V \mid$	$\mid E \mid$	r	Comments
ac	2,851	15,093	1,887	mostly tetrahedral, some beam and plate elments
body	45,087	163,734	16,277	same as above
flap	51,537	479,620	2,088	same as above
hscts	2,028	20,341	525	same as above
kall0	3,000	15,950	1	same as above
shuttle	10,429	46,585	262	mostly 2-D elements, some 3-D elements
skirt	12,598	91,961	1,255	same as above
sphere 5	4,098	12,288	17	surface triangulation of a sphere
sphere 6	16,386	49,152	35	surface triangulation of a sphere
two spheres	20,374	131,764	1,968	tetrahedral elements, domain between 2 spheres
vaughan	29,681	81,795	2,526	tetrahedral elements

TABLE 2
Description of three-dimensional meshes.

denote the set of vertices in the separator and A and B the sets of vertices in the resulting subgraphs. For two dimensional problems, we compare the size of S to that of $\mid V \mid^{1/2}$ and for three dimensional problems we compare $\mid S \mid$ to $\mid V \mid^{2/3}$. The balance in the size of subgraphs is expressed as the ratio of the size of the larger piece to that of the smaller. It is naturally of interest to see how these ratios vary over *all* steps of nested dissection of the original graph until the resulting subgraphs contain relatively few vertices. It is also of interest to observe how the value of the balance factor α affects the size of separators. For each problem, we indicate the range of ratios observed over *all* steps of nested dissection until the resulting subgraphs have fewer than 150 vertices. The range of ratios is also presented for two different values of α; the difference in the ranges reflects the tradeoff between small separator size and balanced subgraphs. For many of the two dimensional meshes, visual inspection indicates that the first few steps of Cartesian nested dissection indeed cut the graph into roughly equal pieces with a separator of small size. The top level separator for problems "airfoil 1" and "venkat 2" are shown in Figures 1 and 2; the vertices in the separator are marked by the symbol "$\bullet$."

The results for two dimensional problems are summarized in Table 3. For $\alpha = 1/3$, the sizes of the subgraphs are balanced to within a factor of 2. The corresponding ratio of separator sizes is within a factor of $2\sqrt{2} \mid V \mid$ for all meshes except those that we generated using PATRAN. For the latter the ratio is at most twice the bound; we consider that quite encouraging since we constructed these problems to challenge our algorithm. Hence it seems fair to conclude that for the two dimensional problems in the test suite, Cartesian nested dissection performs quite well. For $\alpha = 1/5$, the subgraphs are balanced to within a factor of 3; however the range of separator sizes varies rather little.

The results for three dimensional problems are summarized in Table 4 for $\alpha = 1/3$ and $\alpha = 1/7$. Recall that the problems "sphere 5" and "sphere 6" are surface triangulations and the problems "shuttle" and "skirt" contain mostly two dimensional elements. For these problems, we indicate the range of the ratio $\frac{|S|}{|V|^{1/2}}$ within parentheses in a second line below the main entry for the problem. Once again, the results are very encouraging since the separator sizes are within a factor of three of $\mid V \mid^{2/3}$ with subgraphs balanced to within a factor of three. The algorithm is also effective in computing separators to within $2 \mid V \mid^{1/2}$ for the surface triangulation meshes. For the three dimensional problems, the tradeoff between small separator size and balanced subgraphs is rather unclear; relaxing the balance condition does not always lead to a smaller range of separator sizes.

7. Conclusions. Computing a nested dissection ordering in parallel is a very important step in parallel sparse matrix factorization. A parallel implementation of our Cartesian nested dissection algorithm [6] has made possible the development of a suite of parallel algorithms for factorization of symmetric sparse matrices [7]. We have extended the algorithm to order a non-symmetric matrix A associated with a mesh without explicitly forming the graph of $A^T A$. We also show that the algorithm is provably good for the class of *r-local* graphs. Our experiments indicate that the algorithm computes good separators for a wide variety of irregular two and three dimensional meshes arising from practical applications.

Further improvements in the algorithm are possible. For example, in the current implementation, a vertex separator is computed from an edge separator by arbitrarily selecting one end point of each edge. The separator size could be reduced by computing a minimum vertex cover of the bipartite graph induced by the edge separator.

Problem	$\alpha = 1/3$		$\alpha = 1/5$	
	$\frac{\lvert A\rvert}{\lvert B\rvert}$	$\frac{\lvert S\rvert}{\lvert V\rvert^{1/2}}$	$\frac{\lvert A\rvert}{\lvert B\rvert}$	$\frac{\lvert S\rvert}{\lvert V\rvert^{1/2}}$
airfoil 1	1.0, 1.33	0.33, 2.02	1.24, 2.33	0.35, 1.52
airfoil 2	1.0, 1.27	0.25, 2.05	1.09, 2.47	0.30, 1.39
airfoil 3	1.0, 1.44	0.21, 3.15	1.0, 2.41	0.07, 1.49
binaca	1.0, 1.46	0.55, 2.38	1.04, 2.28	0.36, 1.41
eppstein	1.1, 1.21	0.62, 1.34	1.0, 2.01	0.51, 0.94
graded box	1.0, 1.38	0.16, 2.11	1.0, 2.25	0.12, 1.99
graded L	1.0, 1.46	0.22, 3.28	1.0, 2.35	0.16, 3.00
graded +	1.0, 1.42	0.05, 4.50	1.12, 2.16	0.04, 4.16
hollow box	1.0, 1.53	0.46, 1.44	1.0, 2.59	0.47, 1.39
parc	1.0, 1.19	0.65, 1.95	1.11, 2.34	0.07, 0.98
pinched hole	1.0, 1.65	0.47, 3.32	1.0, 2.40	0.47, 2.34
six hole	1.0, 1.35	0.18, 2.08	1.0, 1.58	0.38, 1.42
venkat 1	1.0, 1.44	0.12, 3.16	1.02, 2.38	0.75, 1.54
venkat 2	1.1, 1.28	0.75, 1.54	1.50, 2.43	0.84, 1.16

TABLE 3
Results for two-dimensional meshes.

Problem	$\alpha = 1/3$		$\alpha = 1/7$	
	$\frac{\lvert A\rvert}{\lvert B\rvert}$	$\frac{\lvert S\rvert}{\lvert V\rvert^{2/3}}$	$\frac{\lvert A\rvert}{\lvert B\rvert}$	$\frac{\lvert S\rvert}{\lvert V\rvert^{2/3}}$
ac	1.0, 1.79	0.23, 1.15	1.2, 5.25	0.17, 1.05
body	1.0, 2.19	0.01, 1.35	1.0, 4.02	0.01, 2.01
flap	1.0, 2.68	0.01, 2.28	1.04 4.69	0.01, 1.87
hscts	1.0, 2.04	0.71, 2.93	1.11, 2.38	0.30, 1.59
kall0	1.0, 1.67	0.48, 0.93	2.5, 4.00	0.48, 0.97
shuttle	1.0, 1.75	0.09, 1.34 (0.44, 4.44)	1.0, 2.17	0.09, 1.62 (0.44, 4.83)
skirt	1.0, 2.43	0.23, 2.28 (1.13, 5.85)	1.0, 3.32	0.23, 2.81 (1.13, 6.87)
sphere 5	1.0, 1.28	0.19, 0.61 (0.47, 1.49)	1.06, 2.98	0.28, 0.50 (0.73, 2.00)
sphere 6	1.0, 1.35	0.17, 0.57 (0.52, 1.40)	1.02, 3.05	0.22, 0.50 (0.71, 1.24)
two spheres	1.1, 2.01	0.37, 1.97	1.0, 3.62	0.52, 1.13
vaughan	1.0, 2.18	0.05, 1.26	1.01, 4.55	0.07, 1.02

TABLE 4
Results for three-dimensional meshes.

In the course of our experiments, we observed that for a given subgraph, a less stringent balance constraint does indeed lead to a smaller separator. However, over all steps of nested dissection, sizes of separators are not consistently improved by a smaller value of the balance factor α. Computational results for the two values of α show that a smaller value does not lead to a substantial decrease in the range of separator sizes. Since the subsequent parallel numeric phase requires good load balance, it is possible that subgraphs of almost equal size may result in better performance despite the extra arithmetic work incurred by slightly larger separators. An alternative would be to vary the value of α at different stages during nested dissection.

The class of planar graphs have been proved to have good separators [8], but this class contains graphs for which there are no line separators satisfying the bound on size [14]. Miller, Thurston, Teng and Vavasis have established several classes of graphs (more general than *r-local* graphs) with provably good sphere separators [9, 10, 11, 14]. Interestingly enough, our computational results show that good line and plane separators can be computed for problems in the test suite which are not *r-local*. A natural question that arises is that of identifying a larger class of graphs for which such separators are provably good.

8. Acknowledgement. Thanks to Paul Chew, John Gilbert, Horst Simon and Johan Zdenek for providing test problems. Thanks to Michael T. Heath for many detailed, helpful comments about the presentation.

REFERENCES

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

[2] J. A. George and M. T. Heath, *Solution of sparse linear least squares problems using givens rotations*, Linear Alg. Appl., 88 (1980), pp. 223–238.

[3] J. A. George and J. W.-H. Liu, *An automatic nested dissection algorithm for irregular finite element problems*, SIAM J. Numer. Anal., 15 (1978), pp. 1053–1069.

[4] ———, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1981.

[5] J. A. George and E. G.-Y. Ng, *Symbolic factorization for sparse gaussian elimination with partial pivoting*, SIAM J. Sci. Stat. Comput., 8 (1987), pp. 877–898.

[6] M. T. Heath and P. Raghavan, *A Cartesian parallel nested dissection algorithm*, Tech. Rep. UIUCDCS-R-92-1772, University of Illinois at Urbana-Champaign, 1992.

[7] ———, *Distributed solution of sparse linear systems*, Tech. Rep. UIUCDCS-R-93-1793, University of Illinois at Urbana-Champaign, 1993.

[8] R. J. Lipton, D. J. Rose, and R. E. Tarjan, *Generalized nested dissection*, SIAM J. Numer. Anal., 16 (1979), pp. 346–358.

[9] G. Miller, S. Teng, W. Thurston, and S. Vavasis, *Automatic mesh partitioning*, in Workshop on Sparse Matrix Computations: Graph Theory Issues and Algorithms, Institute for Mathematics and Its Applications, Springer-Verlag, 1992.

[10] G. L. Miller, S. Teng, and S. A. Vavasis, *A unified geometric approach to graph separators*, in Proceedings of the 32nd Annual Symposium on Foundations of Computer Science, IEEE, 1991, pp. 538–547.

[11] G. L. Miller and W. Thurston, *Separators in two and three dimensions*, in Proc. 22nd Ann. ACM Symp. Theory of Comput., New York, 1990, ACM, pp. 300–309.

[12] A. Pothen, H. D. Simon, and K.-P. Liou, *Partitioning sparse matrices with eigenvectors of graphs*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 430–452.

[13] F. P. Preparata and M. I. Shamos, *Computational Geometry*, Springer-Verlag, 1985.

[14] S.Teng, *Points, Spheres and Separators: A unified approach to graph partitioning*, PhD thesis, Carnegie Mellon University, 1991.

[15] S. A. Vavasis, *Automatic domain partitioning in three dimensions*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 950–970.

FIG. 1. *A single separator for airfoil 1.*

FIG. 2. *A single separator for venkat 2.*