SIAM J. MATRIX ANAL. APPL.
Vol. 14, No. 3, pp. 766–775, July 1993

NESTED DISSECTION FOR SPARSE NULLSPACE BASES*

JULIO M. STERN† AND STEPHEN A. VAVASIS‡

Abstract. The authors propose a nested dissection approach to finding a fundamental cycle basis in a planar graph. The cycle basis corresponds to a fundamental nullspace basis of the adjacency matrix. This problem is meant to model sparse nullspace basis computations occurring in a variety of settings. An $O(n^{3/2})$ bound is achieved on the nullspace basis size (i.e., the number of nonzero entries in the basis), and an $O(n \log n)$ bound on the size in the special case of grid graphs.

Key words. sparse, nullspace, basis, force method, nested dissection

AMS subject classifications. 65F05, 65F50

1. Fundamental nullspace bases and graphs.

Let A be an $n \times m$ matrix with $m \geq n$, and let its rank be r. An important problem in scientific computation is to produce a basis for the nullspace of A. In other words, we want to compute an $m \times (m-r)$ matrix V of rank $m-r$ such that $AV = 0$. The columns of V satisfy the equation $Av = 0$; such a vector v is usually called a *null vector.*

The nullspace basis problem arises in the following context, as explained further by Strang [15]. Solving the square nonsingular linear system

$$\begin{pmatrix} H & -A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} \qquad (1)$$

is the key step for structural analysis, the finite element method, optimization problems, and electrical network analysis. In these applications, H is diagonal or block diagonal and symmetric positive definite, and A has full row rank. Usually, H encodes element stiffnesses or resistances, and A encodes geometry and connectivity among elements. Two common methods to solve this include the "displacement" method and the "force" method (see Kaneko, Lawo, and Thierauf [8]), terminology arising in the structural analysis application. The displacement method involves substituting the first block of equations $x = H^{-1}(A^T y + f)$ into the second formula $Ax = g$, arriving at a symmetric positive definite linear system with y as the only unknown. The resulting coefficient matrix $AH^{-1}A^T$ is called the "assembled stiffness matrix" in the context of finite elements.

The force method instead attempts to eliminate y and to solve for x. Let V be a nullspace basis for A. Let x_0 be any solution for $Ax_0 = g$ (typically there is a simple application-dependent method for finding some solution to the underdetermined system $Ax = g$). Then we know that the vector x, which is the first solution component to (1), satisfies $A(x - x_0) = 0$, i.e., $x - x_0 = Vq$ for some vector

* Received by the editors May 26, 1991; accepted for publication (in revised form) December 27, 1991. This work was supported in part by the Applied Mathematical Sciences Program (KC-04-02) of the Office of Energy Research of the U.S. Department of Energy under grant DE-FG02-86ER25013.A000, the U.S. Army Research Office through the Mathematical Sciences Institute of Cornell, and by a National Science Foundation Presidential Young Investigator award. The second author's work was supported in part by grant 87/0540-9 from FAPESP, Fundação de Amparo à Pesquisa do Estado de São Paulo, Brasil.

† School of Operations Research, Cornell University, Ithaca, New York, 14853. Present address, Departamento de Ciência da Computação do Instituto de Matemática e Estatística da Universidade de São Paulo, 05508 São Paulo, Brasil (jstern@ime.usp.br).

‡ Department of Computer Science, Cornell University, Ithaca, New York 14853 (vavasis@cs.cornell.edu).

q. Then we can substitute $x = Vq + x_0$ in the equation $Hx - A^Ty = f$ to obtain $H(Vq+x_0)-A^Ty = f$. Now multiply on the left by V^T, using the fact that $V^TA^T = 0$, obtaining $V^TH(Vq - x_0) = V^Tf$. This is a linear system that may be solved for q, hence obtaining x. The force method can be advantageous in circumstances involving fixed geometry but changing structural properties.

If A is sparse (as usually happens in applications), then we might hope to compute a nullspace basis V that is itself sparse. This has been a topic of recent interest in the literature; it has been addressed by Coleman and Pothen [3], Gilbert and Heath [6], and Plemmons and White [11]. In general there is no reason to believe that sparsity in A implies sparsity in V: it is necessary that A have some additional structure.

None of these earlier works are able to establish bounds on the number of nonzero entries in V. In this report we propose an algorithm that computes a nullspace basis with an asymptotic bound on the number of nonzero entries in V. In order to establish bounds, we focus attention on a simple model problem that captures the features of many real problems.

If V contains an embedded $(m-r) \times (m-r)$ identity matrix, the basis is said to be *fundamental.* In other words, there exist permutation matrices P and Q such that

$$PVQ = \begin{pmatrix} I \\ C \end{pmatrix}.$$

Fundamental nullspace bases are especially useful in certain applications because they lead to simplified algorithms. In particular, the rows of V not in the identity matrix correspond to columns of A forming a basis of the span of A, and the null vectors give equations that express each nonbasic column of A in terms of the basis columns. In this report we focus on the fundamental nullspace basis problem.

D. Rose, in a private communication, has observed that the problem of finding a fundamental nullspace basis of A corresponds to a certain symmetric elimination order applied to (1). This implies that both the force method (in the context of a fundamental basis) and displacement method can be put into a more general framework of elimination orderings. We do not pursue this generalization here.

The particular model problem we have selected is the case that A is the node-edge incidence matrix of an undirected graph. This is a matrix with n rows, one for each vertex of the graph, and m columns, one for each edge. Each column has two nonzero entries in the positions corresponding to the endpoints of the edge. These two entries are one $+1$ and one -1 in each column in arbitrary order. (In some applications, the graph is actually directed, in which case the signs of the entries indicate edge orientation. However, the choice of orientation has no effect on the construction of the nullspace basis since multiplying a column by -1 does not change its span.)

A nullspace basis for such a matrix has a natural combinatorial interpretation. In particular, the nonzero entries of a particular column of V corresponds to a closed walk in the graph. A fundamental nullspace basis for a connected graph corresponds to the identification of a spanning tree T of the graph. A null vector corresponds to the unique cycle formed by edges of T in conjunction with a single edge of $G-T$. See Welsh [17] for more information.

Thus, the problem of finding a fundamental nullspace basis with suitable properties is reduced to the problem of finding the right spanning tree T of G. We propose an algorithm reminiscent of nested dissection for finding this tree T. Nested dissection, a technique due to George [5], finds a *separator* of the graph and then recursively works with subgraphs. Here, a *separator* refers to a small set of nodes whose removal disconnects the graph into pieces of size at most $2n/3$ nodes. The exact definition is contained in the next section. Unlike traditional nested dissection, which requires no further properties of the separator, we will also need the separator nodes to form a cycle.

Such separators are found in planar graphs that have a bound on the maximum face size, a result due to Miller [9]. See the next section for the theorem.

The existence of any kind of separator automatically means that A can be partitioned in *row block angular form* (RBAF) if the rows corresponding to separator vertices are numbered last. An example of block angular form is

$$(2)\qquad \begin{pmatrix} B_1 & & & & & \\ & B_2 & & & & \\ & & B_3 & & & \\ & & & \ddots & & \\ & & & & B_k & \\ S_1 & S_2 & S_3 & \cdots & S_k & S_{k+1} \end{pmatrix}.$$

In this regard, our approach is most reminiscent of Plemmons and White. They work with *column block angular form* (CBAF) and do not attempt to derive bounds on the number of nonzero entries in the basis.

It may seem that we are speaking of a very restricted class of matrices by limiting attention to node-edge adjacency matrices of planar graphs, but this is a model for the kind of matrices that occur in practice. For example, optimization problems with flow constraints have constraint equations in the form of a graph, and the graph is typically planar or nearly planar. As another example, Pothen [12] shows that for certain kinds of structural problems, the nullspace basis of the structural equilibrium matrix A can be entirely deduced from a cycle basis for the underlying graph.

The remainder of this paper is organized as follows. In §2 we give an algorithm that achieves an $O(n^{3/2})$ bound on the size of the nullspace. An added bonus of our algorithm is that the resulting basis has structure useful for parallelism. In §3 we specialize our algorithm to the case of a grid graph, achieving an $O(n \log n)$ algorithm. Finally, in §4 we discuss how to generalize our ideas to other kinds of matrices.

In many applications where A is not explicitly given in RBAF, we can permute rows and columns to put PAQ in RBAF. Finding the permutation matrices P and Q that minimize the number of residual rows or columns, while producing diagonal blocks of approximately the same size, is in general a very difficult combinatorial problem to be solved exactly. See work by Stern [14].

2. A tree from nested simple cycle separators.

In a graph $G = (V, E)$ with $|V| = n$, a set $S \subset V$ is called an (α, β)-*separator* if and only if

1. Set S contains at most $\beta\sqrt{n}$ vertices, and
2. Graph $G - S$ has no connected component with more than αn vertices.

For this section we need the following theorem from Miller [9].

THEOREM 2.1. *Let G be a biconnected planar graph with n vertices, m edges, and maximal face size φ. Then there is an (α, β)-SCS (simple cycle separator) S, with $\alpha = 2/3$ and $\beta = 2\sqrt{2\lfloor \varphi/2 \rfloor}$. Moreover, S can be found in linear time.*

Recall that *biconnected* means that the graph is connected, and remains connected even after the deletion of any single vertex. If a graph is connected but not biconnected, then it has an *articulation point*, that is, a vertex whose removal disconnects the graph into more than one component. The *maximal face size* of a planar graph

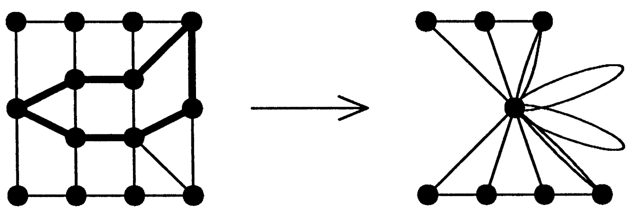

FIG. 1. *Contracting a cycle to a single node.*

refers to the maximum number of edges around any face. Finally, the term *simple cycle separator* refers to a set of vertices S that form a separator in the sense of the previous definition, and that are the vertex set of a simple cycle in G.

For the remainder of this section we restrict attention to biconnected planar graphs with face size bounded by φ. We remark that a strengthening of the foregoing theorem appears in Gazit and Miller [4], which allows G to have a constant number of faces with, say, $\sqrt{n}$ vertices without disturbing the bound.

The assumption that G is biconnected is not actually a restriction. First, if G is disconnected, then each component can be treated separately. Second, if G is connected but not biconnected, then the biconnected components can be identified in linear time (see Aho, Hopcroft, and Ullman [1]). It suffices to work with biconnected components, since any particular cycle (null vector) is contained in a unique biconnected component of G.

Our main goal for this section is to analyze a procedure **T** that finds a spanning tree. A basic building block for **T** is Algorithm **P** that decomposes the graph using the previous theorem. The first two steps of Algorithm **P** are as follows:

P-1. Find in G an (α, β)-SCS, say S.

P-2. Transform G into G' by contracting S into a single vertex σ.

These steps are illustrated in Fig. 1.

LEMMA 2.2. *Vertex σ is the only articulation point of G'.*

Proof. Suppose $\tau \neq \sigma$ is an articulation point (a.p.) in G'. Let $G'_1, G'_2, \ldots, G'_k$ be the biconnected components of G' articulated by σ. Since σ is an a.p. in G', τ belongs to exactly one G'_i, say G'_1. Let $G'_{1,1}$, $G'_{1,2}$ be two of the biconnected components of G'_1 articulated by τ. Since τ is an a.p., σ belongs to exactly one $G'_{1,i}$, say $G'_{1,1}$.

Now $G'_{1,2}$ is a subgraph of G, because it has not been affected by the contraction of S. But then τ is an articulation point in G between $G'_{1,2}$ and S, which contradicts the hypothesis that G is biconnected. □

By virtue of Lemma 2.2 we can define the final step of procedure **P**.

P-3. Separate the biconnected components of G' into disconnected subgraphs,

$$G^1_1, G^1_2, \ldots, G^1_k,$$

with every subgraph receiving its own copy of σ, the a.p. in G'. The final graph, G^1, is the disjoint union of the subgraphs G^1_i (see Fig. 2).

LEMMA 2.3. *The maximal face size of G^1_i, a connected subgraph of G^1, does not exceed the maximal face size of the original graph G.*

Proof. Observe that, given an embedding of G, there is an embedding of G^1_i that corresponds to topologically contracting the embedded edges of S. Focus on this embedding of G^1_i.

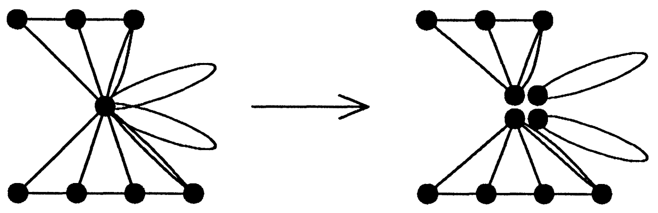

FIG. 2. *Graph G_1 based on Fig. 1.*

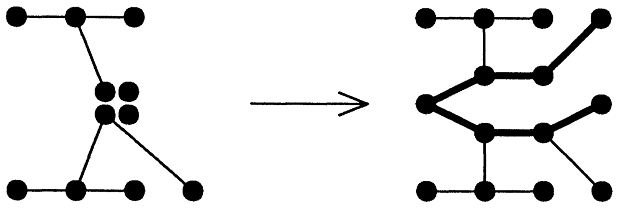

FIG. 3. *Finding a tree recursively.*

Let $v_1, v_2, \ldots, v_l$ be the vertices in a face of G_i^1. If $v_k \neq \sigma$ for $k = 1, \ldots, l$, then the face was not changed by the contraction of S, and the lemma follows. Now say $v_l = \sigma$. The list of vertices $v_1, \ldots, v_l$ in the embedding of G_i^1 have no edges adjacent to them in the interior of the face. Since no edge adjacent to these vertices was deleted by the contraction process, this means that vertices $v_1, \ldots, v_{l-1}$ form a path in G such that there are no edges adjacent to these vertices that are embedded on one "side" (either left or right) of the path. Thus this path is part of a face in G of size at least l. □

By virtue of Lemma 2.3 we can recursively apply procedure **P** to each of the connected subgraphs in G^1. That will be the basis for procedure **T** to construct the spanning tree. The parameter κ below is a constant depending on φ that we will specify later.

```
If $|V(G)| \leq \kappa$
    T-1: Return an arbitrary spanning tree $T$ in $G$.
Else
    T-2: Apply procedure P to $G$ (using the SCS $S$).
         In each subgraph $G_i^1$, recursively use procedure T to get a tree $T_i$.
    T-3: Construct in $G$ the spanning tree $T$, joining:
         (a) Forests $\hat{T}_i$, the uncontracted versions of $T_i$, and
         (b) The SCS $S$ with an arbitrary edge deleted.
```

This procedure is illustrated in Fig. 3 for the graph used in Figs. 1 and 2.

LEMMA 2.4. *The graph returned by procedure* **T** *is a spanning tree.*

Proof. We will prove the theorem by induction on the number of levels we recursively called procedure **T**. At the last recursive call, **T-1** returns a spanning tree. This is the basis for our inductive argument. Now assume that the recursive calls to **T** return spanning trees T_i at each subgraph G_i^1. To complete our induction we must prove that T is a spanning tree for G.

Let C be cycle S with an arbitrary edge deleted, so that C is a simple path spanning the vertices of S. We argue by contradiction that T cannot have cycles. Suppose T has a cycle. Since each $\hat{T}_i$ is a forest and the various $\hat{T}_i$'s have no common vertices, the cycle must be formed by edges of C and a particular $\hat{T}_i$. But this is not possible because then the same cycle could be contracted to form a cycle in T_i.

Also, we claim that T is connected. This is because the forests $\hat{T}_i$ and C span all the vertices, and each tree in the forest made up of $\hat{T}_i$'s contains a vertex of C. □

We have now demonstrated that procedure **T** constructs a spanning tree. We now put an upper bound on the length of the cycle formed by adding any edge of $G-T$ to T. We use the term *co-tree edge* to refer to such edges, and the term *co-tree cycle* to refer to the unique cycle induced by a co-tree edge. The co-tree cycles correspond to the null vectors in the basis, and the number of edges in the co-tree cycles correspond to the number of nonzero entries of the nullspace basis.

THEOREM 2.5. *A co-tree cycle of T has length at most $\delta\sqrt{n}+\kappa$, where*

$$\delta = \beta/\left(1-\sqrt{\alpha+1/\kappa}\right).$$

Proof. Let $f(n)$ be the maximum number of edges in a co-tree cycle. We will prove the theorem by induction on the number of levels we recursively called procedure **T**.

If $n \le \kappa$, the theorem is trivial.

Otherwise, let S denote the SCS constructed at the top level of **T**, and observe that a co-tree cycle must lie interior to one of the subgraphs G_i^1, plus S. Now we can use Theorem 2.1 to get the recursion

$$f(n) \le f(\alpha n+1) + \beta\sqrt{n}.$$

The first term accounts for the edges in G_i^1, a graph that has at most αn vertices plus the copy of σ, and the second term accounts for the edges in S. Our induction hypothesis states that

$$f(\alpha n+1) \le \delta\sqrt{\alpha n+1}+\kappa.$$

So to prove our theorem it suffices to establish, for all $\kappa \le m \le n$, that

$$\delta\sqrt{\alpha m+1}+\beta\sqrt{m} \le \delta\sqrt{m},$$

i.e.,

$$\delta \ge \beta/\left(1-\sqrt{\alpha+1/m}\right).$$

Since $\alpha = \frac{2}{3} < 1$, we can choose $\kappa = 30$ to ensure that the last denominator is positive, and given κ and β we can choose δ to satisfy the inequality for $m = \kappa$. For example, if the graph is triangulated, then Theorem 2.1 gives us $\beta = \sqrt{8}$, and taking $\kappa = 30$ we can take $\delta = 20$. □

As a corollary to Theorem 2.5, we can easily put a bound on $g(n)$, the total length of the all co-tree cycles, i.e., the number of nonzeros in the nullspace basis. Theorem 2.5 gives us a bound for $f(n)$, the length of the longest co-tree cycle. A planar graph with n vertices has at most $3n-6$ edges [7]. Thus there are at most $(3n-6)-(n-1)$ co-tree cycles, so the total co-tree length $g(n)$ is bounded by $(2n-5)f(n)$, i.e., $O(n\sqrt{n})$.

Finally, let T be the tree in G constructed by algorithm **T**. Now we can see that the incidence matrix of the cycles in the cycle basis can be written in RBAF. In

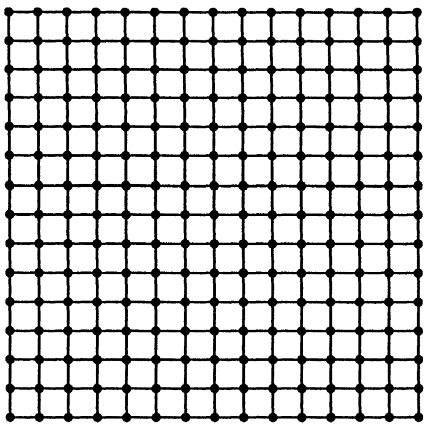
FIG. 4. *The grid graph in the $k = 16$ case.*

particular, we number the edges of G according to the biconnected components of G', with the edges of S numbered last. Then each cycle in the nullspace basis has nonzeros in rows of V corresponding to edges of one particular biconnected component, plus edges from S. We can also nest the RBAF structure in V according to the recursive levels of procedure $\mathbf{T}$.

3. The grid graph. For the $(k-1) \times (k-1)$ grid graph $G(k)$, with $n = (k-1)^2$ vertices and $m = (k-1)(k-2)$ edges, we can define a spanning tree $T(k)$ that gives us a better bound on the total length of the co-tree cycle basis.

For this section we consider only the case that $k = 2^j$, although our results can be generalized. The $k = 16$ graph is illustrated in Fig. 4.

Tree $T(k)$ is constructed recursively. In the case that $k = 2$, i.e., a 1×1 grid graph, the spanning tree is the unique node of the graph. This is the basis of the recursion. In order to construct $T(k)$ for $k > 2$, we need the following basic building blocks:

$\alpha(k)$: The central vertex of $G(k)$, i.e., the vertex at position $(k/2, k/2)$.

$\beta(k)$: The vertex of $G(k)$ at position $(k/2, k/4)$.

$\gamma(k)$: The vertex of $G(k)$ at position $(k/2, 3k/4)$.

$X(k)$: The edges of $G(k)$ of the central row and column.

$H(k)$: The edges in $X(k)$ plus the edges incident to $\beta(k)$ or $\gamma(k)$.

Graph $H(k)$ is illustrated in Fig. 5; vertices $\beta(k)$ and $\gamma(k)$ are shown as squares, and vertex $\alpha(k)$ is shown as a bullseye.

Notice that $G(k) - X(k)$ has four connected components, each of which is isomorphic to $G(k/2)$. We can recursively define $T(k)$ as $H(k)$, plus a copy of $T(k/2)$ for each component. We omit the proof that $T(k)$ is indeed a spanning tree of $G(k)$. An example of $T(k)$ in the case $k = 16$ is illustrated in Fig. 6. This tree is similar to a graph that has occurred in the very large scale integration (VLSI) literature known as the H-tree [16].

We consider the tree $T(k)$ to be rooted at $\alpha(k)$, and define $d^k(\omega)$ as the distance

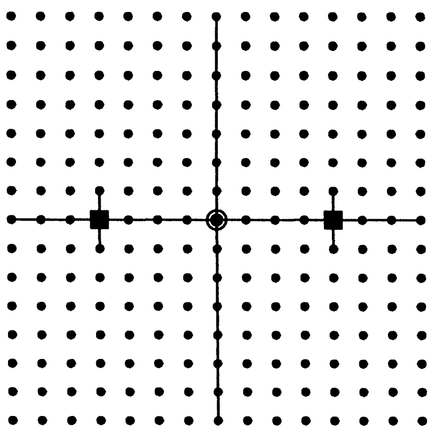

FIG. 5. *The graph $H(k)$ with $\alpha(k), \beta(k), \gamma(k)$ illustrated.*

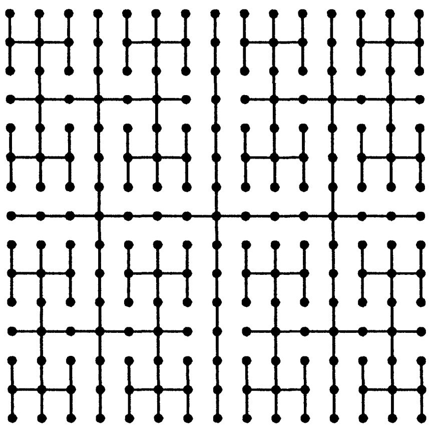

FIG. 6. *The graph $T(k)$ in the case $k = 16$.*

of a vertex ω in $T(k)$ to the root, i.e., $d(\omega, \alpha(k))$. We denote by $r(k)$ the radius of $T(k)$, i.e., the maximal distance $d^k(\omega)$.

LEMMA 3.1. *The radius $r(k)$ of $T(k)$ is bounded by k.*

Proof. Recall $k = 2^j$. We will prove the lemma by induction on j. If $j = 1$, then $T(2) = H(2) = \alpha$ and $r(2) = 0$. This is the basis for our induction.

For $k > 1$ let us consider $d^k(\omega)$ for an arbitrary vertex $\omega \in T(k)$. If $\omega \in X(k)$ then $d^k(\omega) \leq k/2$, because each of the four branches of $X(k)$ has diameter $k/2$. If $\omega \notin X(k)$, then ω belongs to one of the four copies of $T(k/2)$ rooted at $\alpha(k/2)$, and connected to $H(k)$ through $\beta(k)$ or $\gamma(k)$. Let us say that it is in the upper left copy of $T(k/2)$, so that it is rooted through $\beta(k)$. Let $\alpha(k/2)$ denote the root of the particular copy of $T(k/2)$ under consideration. Then we have the chain of equations

$$\begin{aligned} d^k(\omega) &= d(\omega, \alpha(k)) \\ &= d(\omega, \alpha(k/2)) + d(\alpha(k/2), \alpha(k)) \\ &= d^{k/2}(\omega) + d(\alpha(k/2), \beta(k)) + d(\beta(k), \alpha(k)) \\ &= d^{k/2}(\omega) + k/4 + k/4 \\ &\le r(k/2) + k/2. \end{aligned}$$

Hence $r(k) \le r(k/2) + k/2$, and that gives the induction step for our hypothesis, $r(k) \le k$. □

THEOREM 3.2. *The total length $g(k)$ of the co-tree cycle basis for the tree $T(k)$ in the $(k-1) \times (k-1)$ square grid $G(k)$ is asymptotically bounded by $6k^2 \log k$.*

Proof. Let us first look at the co-tree cycles that have edges in $X(k)$. We observe that the only co-tree cycles with edges in $X(k)$ are those generated by co-tree edges incident to vertices of $X(k)$. Also, there are only $4k$ of those edges. Let $f(k)$ be length of a given co-tree cycle, S, intersecting $X(k)$. Observe that the edges of S not in $X(k)$ must lie in one of the four copies of $T(k/2)$. The number of such edges, as in the previous lemma, is at most $r(k/2) + k/4$. The number of edges of S in $X(k)$ is at most $3k/4$, because they must be in the border of one of the quadrant regions, and cannot cross $\beta(k)$. Therefore, the total number of edges in S is bounded as follows:

$$f(k) \le (r(k/2) + k/4) + 3k/4 \le 3k/2.$$

Now, all the remaining co-tree cycles lie in one of the copies of $T(k/2)$, and we can write

$$g(k) \le 4g(k/2) + (4k)f(k) \le 4g(k/2) + 6k^2.$$

But a recursion in the form $g(k) \le 4g(k/2) + ck^2$ is known to be bounded by $ck^2 \log(k) + O(k^2)$. Indeed, George's original [5] derivation of nested dissection on grids came up with the same expression for the fill during Gaussian elimination. So we can limit the total length of the $T(k)$ co-tree cycle basis for $G(k)$ by

$$g(k) \le 6k^2 \log k + O(k^2) = 6n \log n + O(n). \qquad \square$$

4. Generalizing the construction.

In this section we discuss how to generalize the results of §2 to matrices other than adjacency matrices of planar graphs. We first note that the planarity assumption is necessary only to obtain simple cycle separators. We see that the necessary property of the separator is not that it be a simple cycle, but rather that the subgraph induced by the separator nodes be connected. Indeed, the separators in §3 are not cycles. With this weaker hypothesis all of the results of §2 go through. Recent results by Miller, Teng, and Vavasis [10] suggest that very broad classes of graphs could have connected separators.

If we want to generalize beyond adjacency matrices of graphs, we see that the first crucial property in our analysis was that matrix A can be written in the form of (2). This is not enough to carry out our algorithm; this form is analogous to a graph G having a node separator but not necessarily a connected node separator. The generalization of the "connected" property is that matrix S_{k+1} have full row rank.

If this happens, then it can be shown using linear algebra that it suffices to find a nullspace basis for S_{k+1} and for $\hat{B}_1,\ldots,\hat{B}_k$, where $\hat{B}_i$ denotes the matrix

$$\begin{pmatrix} B_i \\ w_i^T \end{pmatrix}.$$

In this formulation w_i^T is a row vector with a "don't care" entry for each column in which S_i has at least one nonzero entry. From fundamental nullspace bases for $\hat{B}_i$ and for S_i we can assemble a fundamental nullspace basis for A. This construction works only under the usual noncancellation assumption that the nullspace basis of A can be predicted entirely from the positions of the nonzero entries in A. See, for example, Pothen [13].

Note added in proof. After this paper was written and accepted, we learned of recent work [2] that improves on some of the results in this paper.

REFERENCES

[1] A. Aho, J. Hopcroft, and J. Ullman, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.

[2] N. Alon, R. M. Karp, D. Peleg, and D. West, *A graph-theoretic game and its application to the k-server problem,* Tech. Rep. 91-066, International Computer Science Institute, Berkeley, CA, December 1991.

[3] T. Coleman and A. Pothen, *The null space problem* II. *Algorithms*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 544–563.

[4] H. Gazit and G. Miller, *Planar separators and the Euclidean norm*, preprint, 1988.

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

[6] J. Gilbert and M. Heath, *Computing a sparse basis for the null space*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 446–459.

[7] F. Harary, *Graph Theory*, Addison-Wesley, Reading, MA, 1972.

[8] I. Kaneko, M. Lawo, and G. Thierauf, *On computational procedures for the force method,* Internat. J. Numer. Methods Engrg., 18 (1982), pp. 1469–1495.

[9] G. Miller, *Finding small simple cycle separators for* 2*-connected planar graphs*, J. Comput. System Sci., 32 (1986), pp. 265–279.

[10] G. Miller, S.-H. Teng, and S. Vavasis, *A unified geometric approach to graph separators*, preprint.

[11] R. Plemmons and R. White, *Substructuring methods for computing the nullspace of equilibrium matrices*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 1–22.

[12] A. Pothen, *Sparse Null Basis Computations in Structural Optimization*, Tech. Rep. CS-88-27, Dept. of Computer Science, the Pennsylvania State Univ., University Park, PA, 1988.

[13] ———, *Sparse Null Bases and Marriage Theorems*, Ph.D. thesis, Cornell University, Ithaca, NY, 1984; Tech. Rep. 85-676, Computer Science Dept., Cornell University, Ithaca, NY, 1985.

[14] J. Stern, *Simulated Annealing with a Temperature Dependent Cost Function*, ORSA J. Comput., 4 (1992), pp. 311–319.

[15] G. Strang, *A framework for equilibrium equations*, SIAM Rev., 30 (1988), pp. 283–297.

[16] J. Ullman, *Computational Aspects of* VLSI, Computer Science Press, Rockville, MD, 1984.

[17] D. Welsh, *Matroid Theory*, London Math. Soc. Monograph 8, Academic Press, New York, 1976.