The Electronic Journal of Linear Algebra.
A publication of the International Linear Algebra Society.
Volume 1, pp. 34-43, October 1996.
ISSN 1081-3810. http://gauss.technion.ac.il/iic/ela

ELA

REDUCING THE ADJACENCY MATRIX OF A TREE*

GERD H. FRICKE†, STEPHEN T. HEDETNIEMI‡, DAVID P. JACOBS‡¶ AND VILMAR TREVISAN§¶

Abstract. Let T be a tree, A its adjacency matrix, and α a scalar. We describe a linear-time algorithm for reducing the matrix $\alpha I_n + A$. Applications include computing the rank of A, finding a maximum matching in T, computing the rank and determinant of the associated neighborhood matrix, and computing the characteristic polynomial of A.

Key words. tree, graph, adjacency matrix, determinant, rank, eigenvalue

AMS(MOS) subject classification. Primary 05C05, 15A15, 15A18, 68R10

1. Introduction. Let $G = (V, E)$ be an undirected graph with vertices $V = (v_1, \ldots, v_n)$ and edge set E. The *adjacency matrix* $A = [a_{ij}]$ of G is the $n \times n$ $(0-1)$ symmetric matrix in which $a_{ij} = 1$ if and only if $i \neq j$ and v_i is adjacent to v_j (that is, there is an edge between v_i and v_j). The *neighborhood matrix* of G, which we denote with N, is obtained by placing 1's along the diagonal of the adjacency matrix (i.e. $N = A + I_n$). Note that the rank and determinant of these matrices are independent of the vertex ordering, since interchanging two rows and then interchanging two columns leaves the rank and determinant unchanged.

In this paper, we are concerned only with trees T (i.e., connected, acyclic graphs), their adjacency matrices A, and neighborhood matrices N. In [3], it was shown that the rank of A is exactly twice the matching number of T (i.e., the maximum number of edges in a set, no two of which have a common vertex). Recently, it was shown how the determinant of N can be computed in linear time [9]. In this paper we present a single linear-time algorithm for computing the rank and determinant, over a field F, of the matrix $\alpha I_n + A$, for arbitrary $\alpha \in F$. As special cases, we obtain linear-time algorithms for the rank and determinant of the adjacency and neighborhood matrices, by taking $\alpha = 0$ and $\alpha = 1$, respectively. By treating α symbolically, we can also compute the characteristic polynomial of A.

The primary contribution is that our algorithm offers a general way of

* Received by the editors on 21 February 1996. Final manuscript accepted on 20 October 1996. Handling editor: Daniel Hershkowitz.

† Department of Mathematics, Wright State University, Dayton, OH 45435, USA (gfricke@desire.wright.edu)

‡ Department of Computer Science, Clemson University, Clemson, SC 29634. USA ({hedet,dpj}@cs.clemson.edu)

§ UFRGS, Instituto de Matemática, 91509–900 Porto Alegre, RS, Brazil (trevisan@mat.ufrgs.br)

¶ The third and fourth authors thank CNPq and Clemson University for their generous support.

looking at several seemingly unrelated topics including rank, determinant, matching number, independence, and the characteristic polynomial. Several known results are looked at in this new light, in addition to some new applications.

2. The reduction algorithm. Let F be any field, T a tree having adjacency matrix A, and $M = \alpha I_n + A$, for some $\alpha \in F$. We wish to compute $\mathrm{rank}(M)$ and $\det(M)$ over F. Instead of computing with the matrix M, however, we compute directly on T in the following way.

Our algorithm associates with each vertex v, a scalar $a(v) \in F$. Initially, $a(v) = \alpha$, for all $v \in V$. A variable d (for deletions) is initialized to 0. The tree is rooted at an arbitrary vertex. The algorithm then processes the vertices bottom-up, beginning with the leaves, which are initially declared to be processed. (We do *not* regard a degree-1 root as a leaf.) In general, we choose any unprocessed vertex v of maximum depth, and mark it processed. If there exist undeleted children w of v for which $a(w) = 0$ then *one* such child w is selected. Both v and w are deleted, and d is incremented by one. (All other children of v are unaffected, including those that might also have zero values.) On the other hand, if all children of v have nonzero values, and C is the set of (undeleted) children of v, then $a(v)$ becomes

$$a(v) - \sum_{c \in C} \frac{1}{a(c)}. \tag{1}$$

After all vertices have been processed, we compute

$$\det = (-1)^d \cdot \prod_{u \in U} a(u) \tag{2}$$

$$\mathrm{rank} = 2d + k \tag{3}$$

where U is the set of undeleted vertices, and $k = |\{u \in U \mid a(u) \neq 0\}|$, the number of undeleted vertices having nonzero values.* This algorithm, called TreeReduction, is summarized in Figure 1. In Figure 2 we illustrate the effect of the algorithm on the tree shown, assuming $\alpha = 1$, and $F = \mathbb{Q}$. The number appearing on each vertex v is the value $a(v)$ at the algorithm's termination.

THEOREM 2.1. *Let F be a field, $\alpha \in F$, T a tree with adjacency matrix A, and $M = \alpha I_n + A$. Then TreeReduction(T, α) computes the determinant and rank of M over F, assuming arithmetic is done in this field.*

Consider applying TreeReduction to the tree in Figure 2, where $\alpha = 1$. This tree has 14 vertices, but after the computation there is exactly one undeleted vertex having value zero. By Theorem 2.1 we conclude that $\mathrm{rank}(N) = 13$, where $N = I_n + A$ is the tree's neighborhood matrix.

* If $U = \emptyset$ then the right side of (2) is $(-1)^n$.

```
Algorithm TreeReduction(T, α)
    Initialize a(v) := α, for all vertices v
    Initialize U := V (all vertices are Undeleted)
    Initialize all leaves "processed" and non-leaves "unprocessed"
    Initialize d = 0
    while Unprocessed ≠ ∅ do
        choose any v ∈ Unprocessed of greatest depth
        Unprocessed := Unprocessed − {v}
        if v has an undeleted child with value 0 then
            select one such child w
            U := U − {v, w}
            d := d + 1
        else  a(v) := a(v) − Σ 1/a(c_i),
                where the sum ranges over the undeleted children c_i of v
    end loop
    det = (−1)^d · ∏_{u∈U} a(u)
    rank = 2d + |{u ∈ U | a(u) ≠ 0}|
    return rank, det
```

FIG. 1. *Tree reduction algorithm.*

Rather than give a formal proof to Theorem 2.1, we will make some general comments that will help convince the motivated reader. Our algorithm, in disguise, is really operating on M and is transforming it into a diagonal matrix (of possibly smaller dimension). The values $a(v)$ in the (undeleted) vertices of the algorithm represent main diagonal values of the matrix. During the transformation, certain rows and columns may be deleted. In the submatrix of undeleted rows and columns, main diagonal entries are modified, and all other entries become zero. A precise description of this matrix transformation appears in [9] for the case when $\alpha = 1$. However, an inspection of the transformation in [9] shows that it applies for any α.

In general, the fastest known algorithms for computing the determinant and rank of an $n \times n$ matrix require time in $O(n^{2+\epsilon})$ (for example, see [1]). *The significance of Theorem 2.1 is that for the matrices M of trees we have a linear-time algorithm.* Taking $\alpha = 1$, M is the neighborhood matrix of T. In [9] it was reported that $\det(M)$ can be computed in linear time, but we see that this applies to computing rank(M) as well.

COROLLARY 2.2. *The determinant and rank of the neighborhood matrix of a tree with n vertices can be computed in $O(n)$ arithmetic operations.*

Another benefit of algorithm TreeReduction is that it provides conceptual insight into various issues such as matching and eigenvalues. These topics will be discussed in the remainder of this paper.

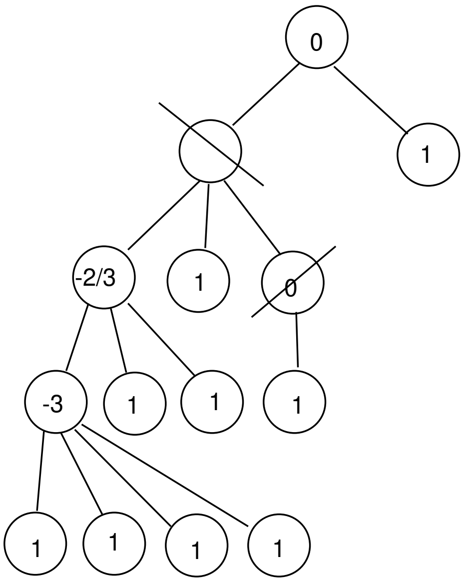

FIG. 2. *Computing the determinant of N.*

3. Matching number of trees. Recall that a *matching* in a graph $G = (V, E)$ is a set $S \subseteq E$ no two edges in which share a common vertex. A matching S is *perfect* if $|V| = 2|S|$. The *matching number*, denoted $\beta_1(G)$, is the largest cardinality of a matching in G. It is known that for some classes of graphs, the matching number and the rank of the adjacency matrix are related [2]. The following elegant theorem for trees is due to Bevis, Domke and Miller [3].

THEOREM 3.1 (BEVIS, DOMKE, MILLER). *For T a tree with adjacency matrix A,*

$$\mathit{rank}(A) = 2 \cdot \beta_1(T).$$

A striking corollary of this theorem is that the rank of a tree's adjacency matrix must be even. Theorem 3.1 is not a direct corollary of Theorem 2.1, but there is a relationship which we now explain. Consider what happens when algorithm TreeReduction is applied to the adjacency matrix A (i.e. $\alpha = 0$). Because all vertices are initialized with 0, no undeleted vertex will ever have nonzero value. Hence $k = 0$ in (3), and so $\text{rank}(A) = 2d$, the number of deleted vertices. But note that vertices are always deleted as parent-child *pairs*, no vertex can ever be deleted twice, and so the edges between these pairs must be disjoint. Thus Theorem 2.1 implies that $\text{rank}(A)$ is even, and that T has a matching of size $\frac{\text{rank}(A)}{2}$, i.e.,

$$\frac{\text{rank}(A)}{2} \leq \beta_1(T).$$

To obtain Theorem 3.1, it suffices to show that

$$\text{rank}(A) \geq 2 \cdot \beta_1(T). \tag{4}$$

Let $S \subseteq E$ be any set of $\beta_1(T)$ disjoint edges, and let X be the set of vertices incident with the edges in S. By the disjointness of S, $|X| = 2\beta_1$. Let $A' = A[X|X]$ be the principal submatrix of A whose rows and columns correspond to X. To establish (4), it suffices to show that A' has full rank. We leave the proof as an exercise. We mention that a linear-time algorithm for finding a maximum matching in a tree is also given in [3]. This method is very similar to the method generated by TreeReduction.

We now seek alternate characterizations of rank(A). Define a *perfect tree* as follows: The tree K_2 is perfect, and if T_1 and T_2 are disjoint perfect trees, so is the tree formed by adding an edge between any vertex of T_1 and any vertex of T_2.

LEMMA 3.2. *A tree T has a perfect matching if and only if it is perfect.*

Proof. It is trivial to show, by induction, that every perfect tree has a perfect matching. Conversely, assume that a tree T has a perfect matching, and assume, by induction that smaller trees with perfect matchings are perfect. Let v be a leaf of T, w its neighbor, and let $T_1, \ldots, T_k$ be the subtrees formed by removing v and w. Since any perfect matching must include $\{v, w\}$, the edges between the T_i and w cannot be used in a perfect matching. Therefore, the perfect matching of T induces a perfect matching in each T_i. By the induction assumption, each T_i is perfect. We can now use the definition of perfect to reconstruct T from the T_i and the $K_2 = \{v, w\}$. □

There is an equivalent way to define perfect trees. Let us define an *even tree* as follows: The graph K_2 is even. If T is an even tree, so is the tree obtained by appending a length-2 path to any vertex of T. In particular, we see that all paths of even length are even trees. It is possible to show that T is even if and only if T is perfect. Since this is not crucial to our discussion, we leave it as an exercise.

Gunther, Hartnell and Rall defined a graph to be β^+-*stable* if the addition of a single edge does not affect the graph's vertex independence number. They showed (Corollary 4.7, [7]) that for trees T with at least two vertices, T is β^+-stable if and only if T has a perfect matching. They also characterized (Theorem 4.6, [7]) these (i.e. nontrivial β^+-stable) trees as (what we call) even.

LEMMA 3.3. *A subgraph of a tree has a perfect matching if and only if it is a forest of disjoint perfect trees.*

Proof. This is a direct consequence of Lemma 3.2. □

THEOREM 3.4. *Let T be a tree with adjacency matrix A. The following are equal:*
(a) rank(A)
(b) $2\beta_1(T)$
(c) the number of vertices of a largest induced forest of disjoint perfect trees
(d) $2\alpha_0(T)$, where $\alpha_0(T)$ denotes the minimum number of vertices required to cover all edges of T.

Proof. By Theorem 3.1, (a) and (b) are equal. It is well-known (see [12], p. 346) that (b) and (d) are equal in any bipartite graph, and hence in any tree. It suffices to now show $(a) = (c)$. A complex matrix M is said to be *normal* if it commutes with its conjugate-transpose M^*, a trivial property of any real-symmetric matrix since $M = M^*$. By [10] (Theorem 3.8, p. 171), because A is normal, rank(A) equals the size of the largest nonsingular principal submatrix. That is, rank(A) equals the size of the largest induced subgraph (of T) with full rank. By Theorem 3.1, this is the size of the largest induced subgraph having a perfect matching. By Lemma 3.3, it is the largest induced forest of disjoint perfect trees. □

It appears to be well-known that the determinant of a tree's adjacency matrix is 0, −1, or 1. It is interesting to seek a precise formulation for the determinant. †

THEOREM 3.5. *For a tree T with adjacency matrix A,*

$$\det(A) = \begin{cases} 1 & \text{if } T \text{ is a perfect tree of size } 4k \\ -1 & \text{if } T \text{ is a perfect tree of size } 4k+2 \\ 0 & \text{otherwise.} \end{cases}$$

Proof. By Theorem 3.1 and Lemma 3.2, A has full rank, or nonzero determinant, if and only if T is perfect. It suffices, therefore, to show that the first two equations hold as stated. Thus, assume T is a perfect tree having n vertices, and consider applying algorithm TreeReduction to T with $\alpha = 0$. If $n \equiv 0 \bmod 4$ there will be an even number of sign changes (i.e. d in equation (2) will be even) and so $\det(A) = 1$. On the other hand, if $n \equiv 2 \bmod 4$ there will be an odd number of sign changes and $\det(A) = -1$. □

4. Eigenvalues of trees. Recall that the *characteristic polynomial* of a square matrix M is

$$q(\lambda) = \det(\lambda I_n - M), \tag{5}$$

and its roots are called the *eigenvalues* of M. It will be helpful to also define

$$p(\lambda) = \det(M - \lambda I_n). \tag{6}$$

† Actually a stronger property holds for trees, namely that every square submatrix has determinant 0, −1, or 1. Such matrices are called *totally unimodular* and are important in guaranteeing an integer solution to a linear programming problem [11].

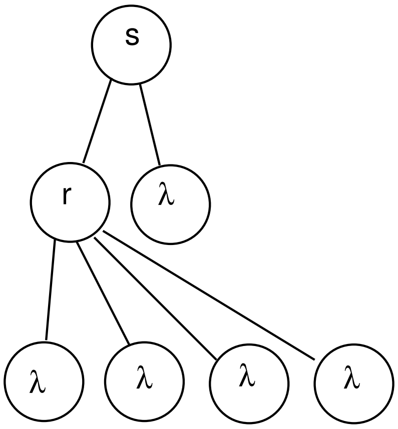

FIG. 3. *Computing the characteristic polynomial.*

Since multiplying a row by -1 reverses the sign of a matrix, $q(\lambda) = (-1)^n p(\lambda)$ and so p and q have the same roots.

From here on, p and q have the meanings given above.

There has been considerable interest in the eigenvalues of adjacency matrices [4, 5, 6, 13]. One elementary property is that for any graph, these eigenvalues are all real [10]. We are interested in the eigenvalues of A, the adjacency matrix of a tree. Note that according to Theorem 2.1, $\alpha \in \mathbb{R}$ is an eigenvalue of A if and only if the call to TreeReduction$(T, -\alpha)$ produces a zero. In our algorithm, there are two ways that this can happen. One way is that at some stage in the algorithm, two vertices w_1 and w_2 occur having a common parent v and for which $a(w_1) = a(w_2) = 0$. Secondly, it might occur that the undeleted children of the root r all have nonzero values, but expression (1) becomes zero at the last stage (as happened in Figure 2). For purposes of discussion, let us call these two kinds of eigenvalues *deep* roots and *top* roots, respectively. From these observations one obtains the following theorem.

THEOREM 4.1. *Let T be a rooted tree, and let T' be formed by taking a new root and adjoining it to the roots of two or more copies of T. Then the set of eigenvalues of T is properly contained in the set of eigenvalues of T'.*

Proof. Clearly every deep root of T must also be a deep root of T'. And every top root of T also becomes a root because there are at least two copies of T. This shows containment. Proper containment comes from the fact that since T is a proper subgraph of T', the largest eigenvalue of T' is greater than the largest eigenvalue of T (see Lemma 1 (3), [4]). $\square$

Let T be a tree with adjacency matrix A, and let p be the polynomial in (6). We can compute $p(\lambda)$ in the following way. Let F represent the field of quotients for the polynomial ring $\mathbb{Q}[\lambda]$. We then make a call to

TreeReduction$(T, -\lambda)$, performing all arithmetic over F. The values that are placed in the vertices are rational functions, that is, quotients $\frac{r(\lambda)}{s(\lambda)}$ where $r(\lambda), s(\lambda) \in \mathbb{Q}[\lambda]$. Clearly the zero function will never appear at a vertex v, for this would imply that the adjacency matrix of the subtree rooted at v has a 0 characteristic function. Algorithm TreeReduction, therefore, deletes no vertices, and $p(\lambda)$ is the product of all resulting rational functions.

Now consider computing $p(-\lambda)$. We may do this in the same way, but by calling TreeReduction(T, λ). One observes that at each stage of the algorithm the corresponding functions on the vertices are *opposite* to the functions in the previous computation. Hence their corresponding products will be the same, up to a sign. In particular,

$$p(-\lambda) = (-1)^n p(\lambda). \tag{7}$$

Thus we have shown that one of the following must hold:

$$p(-\lambda) = p(\lambda) \tag{8}$$

$$p(-\lambda) = -p(\lambda). \tag{9}$$

The following theorem is known to hold even for bipartite graphs (see Lemma 1 (2), [4]), but we stop to give a simple proof in the case of trees.

THEOREM 4.2. *For the adjacency matrix of a tree, α is an eigenvalue if and only if $-\alpha$ is an eigenvalue having the same multiplicity.*

Proof. We know that p satisfies (8) or (9). One can prove (by induction on $\deg(p)$) that any polynomial satisfying (8) or (9) must have roots that satisfy the condition of the theorem. $\square$

Let us now make a simplifying observation. We remarked earlier that $q(\lambda) = (-1)^n p(\lambda)$. But by (7), $p(\lambda) = (-1)^n p(-\lambda)$. It follows that the characteristic function $q(\lambda) = (-1)^{2n} p(-\lambda) = p(-\lambda)$, which can be computed directly by simply calling TreeReduction(T, λ). We illustrate the computation on the tree shown in Figure 3. We assign the variable λ to each vertex, and apply our algorithm. In Figure 3,

$$\begin{aligned} r &= \lambda - \frac{4}{\lambda} \\ s &= \lambda - \frac{1}{\lambda - \frac{4}{\lambda}} - \frac{1}{\lambda}. \end{aligned}$$

Taking the product of all seven rational functions in Figure 3 gives us the characteristic polynomial

$$\lambda^3(\lambda^4 - 6\lambda^2 + 4),$$

and from this it is seen that there are exactly four nonzero eigenvalues.

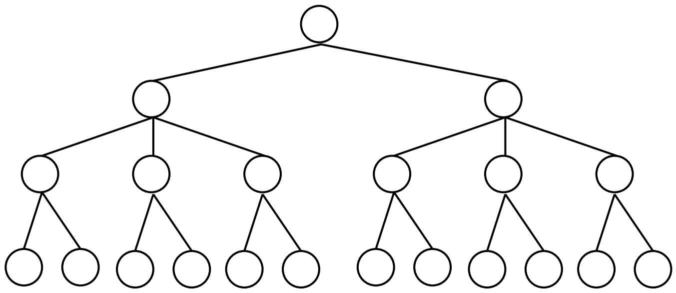

FIG. 4. *Vertices at the same level having same degree.*

Let us examine one more interesting aspect of eigenvalues of trees. Let T be a rooted tree such that all vertices at the same level have the same number of children. Such a tree is shown in Figure 4. In computing the characteristic function as above, we note that the rational functions appearing at each level have the form of partial fractions:

$$\lambda, \quad \lambda - \frac{2}{\lambda}, \quad \lambda - \frac{3}{\lambda - \frac{2}{\lambda}}, \quad \lambda - \frac{2}{\lambda - \frac{3}{\lambda - \frac{2}{\lambda}}}$$

Note that the numerators which appear are the number of children at each level.

5. Eigenvalues of paths. We finish this paper by making some simple observations about paths. Let d_n be the characteristic polynomial for the adjacency matrix of the n-vertex path P_n. The following theorem appears in [8], but we give an alternate proof based on our method.

THEOREM 5.1 (HARARY, KING, MOWSHOWITZ). *The d_n satisfy $d_0 = 1$, $d_1 = \lambda$, and $d_n = d_{n-1}\lambda - d_{n-2}$ for $n \geq 2$.*

Proof. Consider applying our algorithm to P_n by using the indeterminate λ, and let r_n denote the rational function appearing on the n-th vertex. Then we have that

$$\begin{aligned} d_n &= d_{n-1} r_n \\ &= d_{n-1}(\lambda - \frac{1}{r_{n-1}}) \\ &= d_{n-1}\lambda - \frac{d_{n-1}}{r_{n-1}} \\ &= d_{n-1}\lambda - d_{n-2}. \qquad \square \end{aligned} \tag{10}$$

Thus we have a recursive definition for the d_n; computing the next few polynomials we get $d_2 = \lambda^2 - 1$, $d_3 = \lambda^3 - 2\lambda$, and $d_4 = \lambda^4 - 3\lambda^2 + 1$. Now let E_n denote the set of distinct eigenvalues of P_n.

THEOREM 5.2. *For any even integer n, $E_n \cap E_{n+2} = \emptyset$.*

Proof. Suppose $\lambda \in E_n$. Since n is even, we must have $\lambda \neq 0$, using Theorem 3.5 and the fact that the determinant of a matrix is the product of its eigenvalues. Now consider applying TreeReduction to $T = P_{n+2}$ and $\alpha = \lambda$. Since $\lambda \in E_n$, the algorithm will produce a zero on the n-th vertex, causing both it and vertex $n+1$ to be deleted. Hence vertex n+2 will remain with its initial value $\lambda \neq 0$. □

THEOREM 5.3. *Let n and k be integers such that $0 < k \leq n$ and $n \equiv k \bmod (k+1)$. Then $E_k \subseteq E_n$.*

Proof. Let $\lambda \in E_k$. Imagine applying our algorithm to P_n. The value appearing on the k-th vertex must be zero causing it and vertex $k+1$ to be eliminated. Within every block of $k+1$ vertices, the last two vertices will be deleted. However by the assumption on n, the last vertex will remain undeleted with a zero value. □

THEOREM 5.4. *For all n, $E_n \cap E_{n+1} = \emptyset$.*

Proof. If $\lambda \in E_n \cap E_{n+1}$, then by (10) we would have $\lambda \in E_{n+2}$, and $\lambda \in E_{n+3}$. Hence there would be an even k (n or $n+1$) for which $\lambda \in E_k \cap E_{k+2}$, contradicting Theorem 5.2. □

REFERENCES

[1] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, 1974.

[2] W. N. ANDERSON, Maximum matching and the rank of a matrix, SIAM J. of Math. Appl., 28 (1975), pp. 114–123.

[3] J. H. BEVIS, G. S. DOMKE AND V. A. MILLER, Ranks of trees and grid graphs, J. of Combinatorial Math. and Combinatorial Computing 18 (1995), pp. 109-119.

[4] D. CAO AND H. YUAN, The distribution of eigenvalues in graphs, Linear Algebra and Its Applications, 216 (1995), pp. 211–224.

[5] D. CVETKOVIĆ, M. DOOB, AND H. SACHS, *Spectra in Graphs*, Academic Press, New York, 1980.

[6] M. DOOB AND D. CVETKOVIĆ, On the spectral characterizations and embeddings of graphs, Linear Algebra and Its Applications, 27 (1979), pp. 17–26.

[7] G. GUNTHER, B. HARTNELL, AND D. RALL, Graphs whose vertex independence number is unaffected by single edge addition or deletion, Discrete Applied Math. 46 (1993), pp. 167–172.

[8] F. HARARY, C. KING, AND A. MOWSHOWITZ, Cospectral graphs and digraphs, Bulletin of London Math. Soc., 3 (1971), pp. 321–328.

[9] D. P. JACOBS AND V. TREVISAN, The determinant of a tree's neighborhood matrix, Linear Algebra and Its Applications, to appear.

[10] M. MARCUS AND H. MINC, *Introduction to Linear Algebra*, Macmillan, New York, 1965.

[11] C. H. PAPADIMITRIOU AND K. STEIGLITZ, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, 1982.

[12] A. TUCKER, *Applied Combinatorics*, John Wiley & Sons, New York, 1980.

[13] E. R. VAN DAM, Regular graphs with four eigenvalues, Linear Algebra and Its Applications, 226–288 (1995), pp. 139–162.