

A GRAPH ISOMORPHISM ALGORITHM USING PSEUDOINVERSES *

J. M. BENNETT¹ and J. J. EDWARDS²

¹*Basser Department of Computer Science, University of Sydney
NSW 2006, Australia. email: jmb@cs.su.oz.au*

²*School of Computing Sciences, University of Technology
Sydney, PO Box 123, Broadway, NSW 2007, Australia. email: jenny@socs.uts.edu.au*

Abstract.

For a graph of m nodes and n edges, an algorithm for testing the isomorphism of graphs is given. The complexity of the algorithm is a maximum of $O(mn^2)$ in almost all cases, with a considerable reduction if sparsity is exploited. If isomorphism is present, the pseudoinverses of the Laplace matrices of the graphs will be row and column permutations of each other. Advantage can be taken of certain features of the incidence matrices or of properties of the graphs to reduce computation time.

AMS subject classification: 05C60 15A09

Key words: Graph isomorphism, pseudoinverse.

1 Introduction.

An efficient algorithm for testing two graphs (planar or non-planar) for isomorphism is of practical importance (eg, for organic chemists, vide Kudo et al[6]). Two undirected connected graphs G_1 and G_2 with the same number of edges and no parallel edges are isomorphic if two vertices in G_1 are adjacent if and only if the corresponding vertices of G_2 are adjacent and vice versa. In the case of directed graphs, there is an additional requirement that the orientation of the elements should be the same. If the algorithm is used for directed graphs, it may require an additional step where the direction of the elements in the final suggested graph matching is compared with the directions of the original graphs.

Most approaches (Hoffman [3] and McKay [8]) so far to the isomorphism problem have fallen into two groups - those using generalised matrix functions such as the determinant or eigenvalues of matrices derived from the graphs, and those based on classification of the vertices, e.g., Kocay [5] and Mayeda [7]. An excellent summary of the present position appears in Skiena [12]. The algorithm suggested here, involving the pseudoinverses of the Laplace matrices, belongs in the former category. The main advantage of the proposed algorithm over one based on the eigensystem of the Laplace matrix is that generally the operations counts are lower and the proposed algorithm can be extended to matrices which have multiple eigenvalues.

*Received January 1993. Revised August 1995.

For a matrix X , the pseudoinverse, here designated as X^+ , has, as a special case, the inverse of a nonsingular square matrix. For many purposes, its most useful property is that, given a set of equations $Xx = b$, $\bar{x} = X^+b$ is the minimum length x which minimises the squared error $(b - Xx)^T(b - Xx)$.

For computational purposes, if $X = UV$, where X is an $m \times n$ matrix of rank r and $m \geq n$, U is $m \times r$ of rank r and V is $r \times n$ of rank r , then X^+ is given by (Noble [9])

$$(1.1) \quad X^+ = V^+U^+ = V^T(VV^T)^{-1}(U^TU)^{-1}U^T.$$

If Y and Z are square orthogonal matrices, a useful property of pseudoinverses (Bennett [2]) is that

$$(YXZ)^+ = Z^TX^+Y^T.$$

This identity is particularly useful where X is an incidence matrix and Y and Z are permutation matrices or diagonal matrices with diagonal elements ± 1 .

Two isomorphic graphs have pseudoinverses of their incidence, adjacency or Laplace matrices which permute to the same matrix. For undirected graphs, with edge directions chosen arbitrarily, additional sign reversals of rows may be necessary. The incidence, adjacency or Laplace matrices themselves will also have this property but the small number of different elements makes the determination of suitable permutations impracticable. In this article, emphasis is placed on Laplace matrices.

For convenience, it will be assumed that graphs being tested are connected. Connectedness can be tested for, e.g., by following the procedure described by Read [10]. If a graph has more than one component, components can be tested for isomorphism separately. It will also be assumed that any self loops have been removed, and that graphs to be tested have no parallel edges and the same numbers of edges and of vertices. A preliminary test should be made to ensure that the numbers of vertices of the same degree in the two graphs are equal. It is convenient to use methods such as those outlined by Read [10] to divide the network into blocks which can be tested separately.

2 Graphs and Pseudoinverses.

The method proposed depends in part on exploiting properties of matrices arising from the underlying sparse tree to minimise the operations count. From the n edges of a directed connected graph G of m vertices may be selected $m - 1$ edges which form a spanning tree (one of the vertices being chosen as the root of the tree), the remaining $n - (m - 1)$ edges being links. By suitable choice of the ordering of columns and rows, the corresponding oriented vertex incidence matrix, M , may be written as

$$\begin{bmatrix} A & B \\ -s^T A & -s^T B \end{bmatrix}$$

where the last row corresponds to the vertex chosen as a root vertex of the spanning tree; the columns in A , an $(m - 1) \times (m - 1)$ matrix (of rank $m - 1$), correspond to the branches of a spanning tree; those in B , an $(m - 1) \times (n - (m - 1))$

matrix, to the links; and s is the $(m-1)$ element summing vector ($s_i = 1, \forall i$). If an edge represented by the column e connects the vertex represented by the row i to the vertex represented by the row j , $M_{ie} = 1$, $M_{je} = -1$ and $M_{ke} = 0$ for $k \neq i, j$. As the spanning tree is not unique, the columns of A may be selected from M in a number of ways (Mayeda [7]). Although the use of the spanning tree to represent the matrix is not essential to the algorithm, it enables efficiencies to be made in the operations count and also the detection of zero pivots generated during the process where the matrices are highly symmetric.

As has been pointed out in the introduction, if, for two undirected graphs, each with m vertices and n edges, one graph is isomorphic to the other, then the pseudoinverses of the corresponding incidence, adjacency or Laplace matrices with arbitrarily assigned edge directions will be identical after row and column permutations, perhaps with row sign changes in the case of the incidence matrix. For directed graphs, sign changes are not permitted.

2.1 The Pseudoinverse.

To appreciate the economy of the operation used, it is necessary to show how it derives from properties of the pseudoinverse. If

$$M = \begin{bmatrix} A & B \\ -s^T A & -s^T B \end{bmatrix} = \begin{bmatrix} I \\ -s^T \end{bmatrix} \begin{bmatrix} A & B \end{bmatrix}$$

then, by (1.1), $M^+ = (YZ)^+ = Z^+Y^+$ where

$$Y = \begin{bmatrix} I \\ -s^T \end{bmatrix}, \quad Z = \begin{bmatrix} A & B \end{bmatrix},$$

i.e,

$$\begin{aligned} M^+ &= \begin{bmatrix} A^T \\ B^T \end{bmatrix} [AA^T + BB^T]^{-1} [I + ss^T]^{-1} [I - s] \\ (2.1) \quad &= \begin{bmatrix} A^T \\ B^T \end{bmatrix} [AA^T + BB^T]^{-1} \left[\left(I - \frac{1}{m} ss^T \right) \middle| -\frac{1}{m} s \right] \end{aligned}$$

as

$$[I + ss^T]^{-1} = \left[I - \frac{1}{m} ss^T \right].$$

Thus

$$M^+ = \begin{bmatrix} I \\ B^T A^{-T} \end{bmatrix} [I + A^{-1} BB^T A^{-T}]^{-1} A^{-1} \left[\left(I - \frac{1}{m} ss^T \right) \middle| -\frac{1}{m} s \right].$$

Let

$$(2.2) \quad C = (I + A^{-1} BB^T A^{-T})^{-1}.$$

Then

$$(2.3) \quad M^+ = \begin{bmatrix} I \\ B^T A^{-T} \end{bmatrix} C A^{-1} \left[\left(I - \frac{1}{m} ss^T \right) \middle| -\frac{1}{m} s \right].$$

In most cases, it is more efficient to test for isomorphism using the pseudoinverse of the Laplace matrix, MM^T than with M^+ . (For convenience, this pseudoinverse of the Laplace matrix, ie $M^{+T}M^+$, will be represented by K .) A possible remapping may then be obtained using the diagonal elements of K . The effects of edge sign allocations are neutralised because the elements K_{ii} are computed. Each diagonal element represents the sum of squares of the elements of a column of the pseudoinverse of M^+ , where each column corresponds to a vertex of the original graph. The diagonal values of K (although not their order) are invariant under permutation of rows and columns.

Another possibility would be to test the diagonal elements of M^+M^{+T} . However, as there are in general fewer vertices than edges (except with trees), there are fewer operations involved in calculating the diagonal elements of K .

A further possibility was suggested by a reviewer. Rather than using the diagonals of the pseudoinverse, it would be possible to obtain the first column of the pseudoinverse of the first Laplacian matrix and successively compare it with permuted columns of the second pseudoinverse until a match was found or there were no remaining columns in the second matrix. This method is shown in example 1 below. For some pairs of graphs, this would involve less work than calculating the diagonals. However, for highly symmetric matrices as in the example in section 2.2, which occur frequently in practice, there is considerable extra work in the sorting and checking required.

A suitable algorithm for computing K may be developed as follows:

$$\begin{aligned}
 (2.4) \quad K \text{ (ie } M^{+T}M^+ \text{)} &= \begin{bmatrix} I - \frac{1}{m}ss^T \\ -\frac{1}{m}s^T \end{bmatrix} [AA^T + BB^T]^{-1} \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} A^T \\ B^T \end{bmatrix} \\
 &= \begin{bmatrix} I - \frac{1}{m}ss^T \\ -\frac{1}{m}s \end{bmatrix} [AA^T + BB^T]^{-1} \left[(I - \frac{1}{m}ss^T) \mid -\frac{1}{m}s^T \right] \\
 (2.5) \quad &= \begin{bmatrix} I - \frac{1}{m}ss^T \\ -\frac{1}{m}s \end{bmatrix} [AA^T + BB^T]^{-1} \left[(I - \frac{1}{m}ss^T) \mid -\frac{1}{m}s \right].
 \end{aligned}$$

From (2.2),

$$(2.6) \quad K = \begin{bmatrix} I - \frac{1}{m}ss^T \\ -\frac{1}{m}s^T \end{bmatrix} A^T C A^{-1} \left[I - \frac{1}{m}ss^T \mid -\frac{1}{m}s \right]$$

If a spanning tree is generated by a suitable technique (eg, by using a breadth first scanning procedure), then the rows and columns of A may be permuted to take the form of a lower triangular matrix. Advantage may be taken of this triangularity to reduce computation, as solution of equations of the form $Ax = b$ ($x = A^{-1}b$) may be effected by backward substitution with A rather than by explicit calculation of A^{-1} (Bennett [1]). This backward substitution calls for at most $\sim m^2/2$ operations.

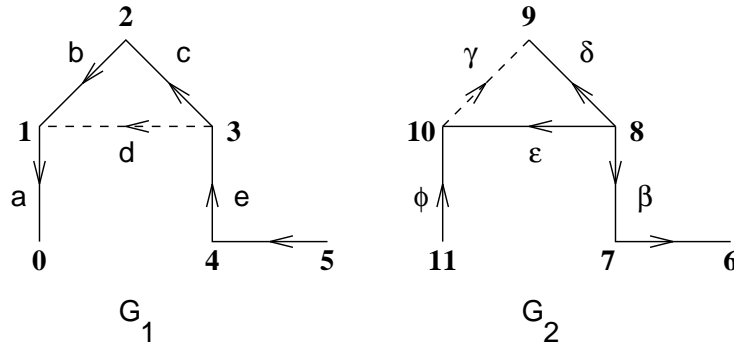


Figure 2.1: Example 1 - Graphs, G_1 and G_2

Thus in the case of the networks shown in Figure 2.1, M_1 and M_2 are

$$\begin{matrix}
 & f & e & c & b & a & d & & \phi & \epsilon & \delta & \alpha & \beta & \gamma \\
 5 & \left(\begin{matrix} 1 & . & . & . & . & . \\ -1 & 1 & . & . & . & . \\ . & -1 & 1 & . & . & 1 \\ . & . & -1 & 1 & . & . \\ . & . & . & -1 & 1 & -1 \\ . & . & . & . & -1 & . \end{matrix} \right) & & & & & & 11 & \left(\begin{matrix} 1 & . & . & . & . & . \\ -1 & -1 & . & . & . & 1 \\ . & . & -1 & . & . & -1 \\ . & . & . & -1 & . & . \\ . & . & . & 1 & -1 & . \\ . & 1 & 1 & . & 1 & . \end{matrix} \right), \\
 4 & & & & & & & & 10 & & & & & \\
 3 & & & & & & & & 9 & & & & & \\
 2 & & & & & & & & 6 & & & & & \\
 1 & & & & & & & & 7 & & & & & \\
 0 & & & & & & & & 8 & & & & &
 \end{matrix}$$

vertices 0 and 8 respectively being chosen as roots and M_1 and M_2 being partitioned (to define A and B) as

$$\left[\begin{array}{c|c} A & B \\ \hline -s^T A & -s^T B \end{array} \right].$$

As was pointed out above, for an initial test for isomorphism it is convenient to compare the diagonal elements of the corresponding K . If the diagonals differ in one or more elements then the graphs are not isomorphic. The diagonal elements of two graphs which are isomorphs can be permuted so that they correspond. If these elements are unique then a possible mapping from the vertices of one graph to those of the other is indicated and may be checked with the help of the incidence matrix by ensuring that if an edge connects a pair of vertices in one graph, then there is also an edge connecting the corresponding pair in the other. This check is necessary because the approach is essentially heuristic, and, although no example has been found, it is possible that, for an apparently unique remapping defined by the elements K_{ii} , there is no corresponding remapping of the graph elements.

In the example the diagonal elements of K_1 and K_2 permuted into descending order and arranged as elements of vectors with the corresponding vertex numbers

shown beside them are given by

$$\frac{1}{108} \begin{bmatrix} 140 \\ 116 \\ 68 \\ 56 \\ 44 \\ 32 \end{bmatrix} \begin{matrix} 5 \\ 0 \\ 4 \\ 2 \\ 1 \\ 3 \end{matrix} \quad \text{and} \quad \frac{1}{108} \begin{bmatrix} 140 \\ 116 \\ 68 \\ 56 \\ 44 \\ 32 \end{bmatrix} \begin{matrix} 6 \\ 11 \\ 7 \\ 9 \\ 10 \\ 8 \end{matrix}$$

respectively. That is, a mapping of (5 4 3 2 1 0) to (6 7 8 9 10 11) is indicated.

Alternatively, using the method suggested by the reviewer, for an initial test for isomorphism the first column of K_1 may be compared with successive permuted columns of K_2 . If there is no match then the graphs are not isomorphs. If there is a match, then the process is repeated with the second column of K_1 being matched to the remaining columns of K_2 , etc. If the columns of K_1 match the columns of K_2 then a possible mapping from the vertices of one graph to those of the other is indicated. Again it may be checked with the help of the incidence matrix by ensuring that if an edge connects a pair of vertices in one graph, then there is also an edge connecting the corresponding pair in the other.

It should be noted that the diagonal element of a column of the original K_2 must also occupy a diagonal position when the column is permuted. In the current example, K_{11} is $\frac{140}{108}$. In searching K_2 , we may map the first column of K_1 only to a column which has a diagonal element, $\frac{140}{108}$. In this case, that is the fourth column of K_2 .

In the example the first column of K_1 and successive columns of K_2 permuted into descending order and arranged as elements of vectors with the corresponding vertex numbers shown beside them are given by

$$\frac{1}{108} \begin{bmatrix} 140 \\ -70 \\ -52 \\ 50 \\ -46 \\ -22 \end{bmatrix} \begin{matrix} 5 \\ \\ \\ \\ \\ \end{matrix}$$

and

$$\frac{1}{108} \left(\begin{bmatrix} 116 \\ -70 \\ -52 \\ 26 \\ -16 \\ -4 \end{bmatrix} \begin{bmatrix} -52 \\ 44 \\ -34 \\ 26 \\ 14 \\ 2 \end{bmatrix} \begin{bmatrix} 56 \\ -46 \\ -28 \\ 14 \\ 8 \\ -4 \end{bmatrix} \begin{bmatrix} 140 \\ -70 \\ -52 \\ 50 \\ -46 \\ -22 \end{bmatrix} \right) \begin{matrix} 11 \\ 10 \\ 9 \\ 6 \end{matrix}$$

respectively. That is, a mapping of (5) to (6) is indicated. The process is repeated with successive columns of K_1 until the mapping of (5 4 3 2 1 0) to (6 7 8 9 10 11) is obtained.

2.2 An Example of Highly Symmetric Matrices.

The procedure for obtaining a possible mapping of one highly symmetric graph on to its isomorph can be demonstrated with an example having matrices of the general form represented by the adjacency matrix

$$\begin{bmatrix} 0 & 0 & E & F \\ 0 & 0 & F & E \\ E^T & F^T & 0 & 0 \\ F^T & E^T & 0 & 0. \end{bmatrix}.$$

Here $E = \frac{1}{2}(H + J)$ and $F = \frac{1}{2}(-H + J)$ where H is a Hadamard matrix (ie, a $p \times p$ matrix with elements ± 1 and such that $H^{-1} = \frac{1}{p}H^T$), and J is a matrix of the form ss^T (ie, with every element unity). The use of Hadamard matrices is, of course, not essential to the process. This example was chosen for its high degree of symmetry.

For the example shown in Figure 2.2 (Seberry Wallis[11]) the Hadamard matrices are of order four. For the Laplace matrices, $M_1M_1^T$ and $M_2M_2^T$ (represented for convenience by L_1 and L_2) of these graphs G_1 and G_2 , the diagonal elements of L_1^+ (ie K_1) and L_2^+ (ie K_2) are all identical. However, removal of, say, the j th row and column from L_1 and the k th row and column from L_2 (the reduced matrices are represented as L'_1 and L'_2 , respectively) where the j th node of G_1 is taken to be equivalent to the k th node of G_2 will lead to the pseudoinverses (represented as K'_1 and K'_2) respectively of the reduced matrices with elements on their diagonals some of which are now distinguishable. After some iterations of successive removals of pairs of rows and columns from L'_1 and L'_2 and their successors, it will be possible to generate a specific mapping using the diagonals of pseudoinverses of the reduced matrices. The pair of rows and columns to be removed from the matrices at each iteration is determined by matching diagonals of the corresponding pseudoinverses. If, say, $(K'_1)_{jj}$ is a unique match for $(K'_2)_{kk}$, then the equivalent nodes in the original graphs may be matched. If a group of diagonals in K'_1 has the same value as a group of diagonals in K'_2 , then at this stage, a match may be made between a node in G_1 selected from one of the group of diagonal elements in K'_1 and a node from G_2 in the corresponding group of K'_2 . This match will not be arbitrary. It relies on the history of previously matched groups of nodes, as shown in the example below.

If this approach is used and the initial tree roots are maintained throughout the calculation, (ie, not identified as nodes to be removed until the end of the calculation), eventually reduced graphs consisting of a string of elements connecting the two root nodes will be obtained. Further progress could be made only by breaking the string. However, the cross identification of nodes in the strings will be unambiguous. Of course, the process initially tests that the degree sequences of the graphs are identical.

Where this reduction process is used, care must be taken that the removal of nodes does not lead to two unconnected subnetworks. A zero pivot during the calculation of the pseudoinverse will indicate that such a separation of the network has occurred and that a different match should be chosen. Alternatively,

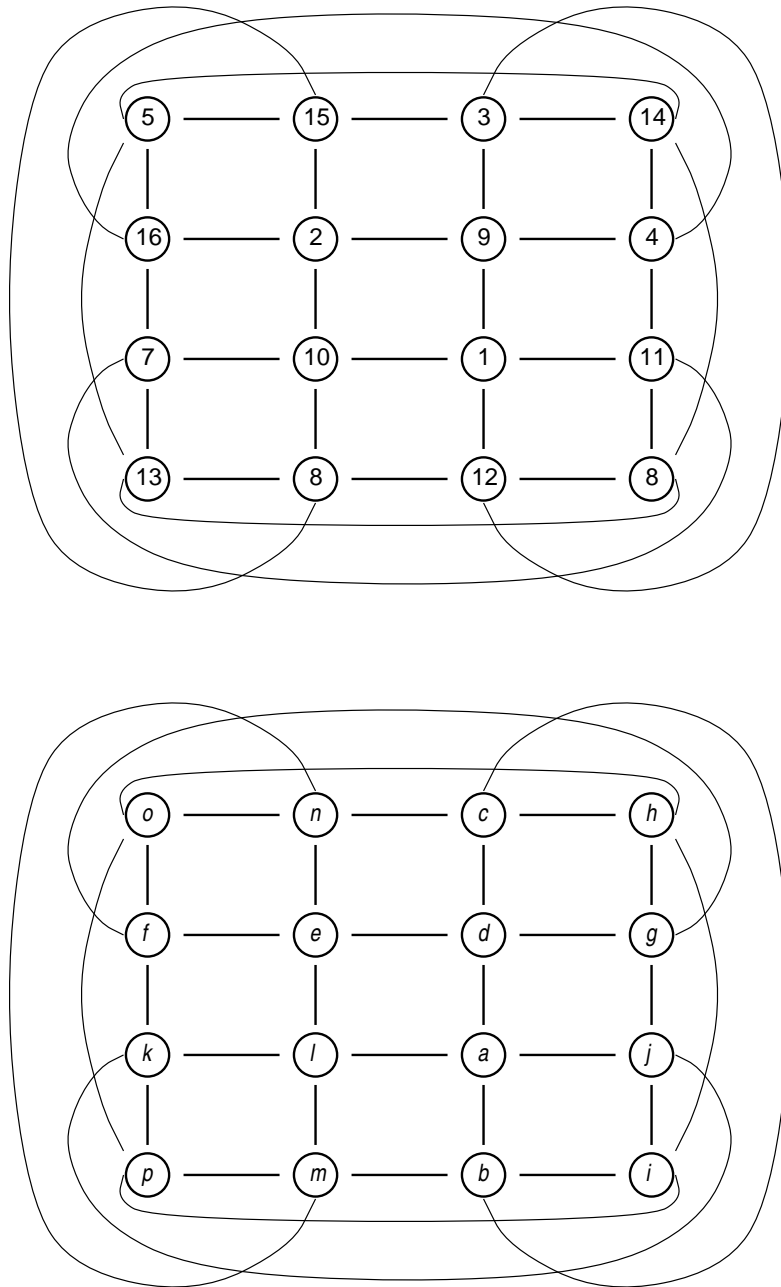


Figure 2.2: Two Highly Symmetric Graphs

zero pivots can be used to break up the two graphs into subgraphs, which can then be treated as separate problems.

For cases of highly symmetric matrices such as these, the method used in the example 1 of matching columns proved unsatisfactory, where isomorphism exists. There may be many columns of K'_2 which match the first column of K'_1 . If a match is made with the first such column of K'_2 , it may not be the correct one. In this case, subsequent removals of rows and columns can and often do lead to unconnected subnetworks. It is then necessary to traverse through a tree of different matchings until either a solution is found or it can be shown that the graphs are not isomorphic.

While it is also possible to generate unconnected subnetworks when matching diagonals, it occurs less frequently, is detected more readily and is easier to remedy. For the case of highly symmetric matrices, it is necessary to maintain a history of the matrix structures and the connecting edges which have been removed in the reduction process. For this history, a list of all possible matches to a vertex are needed at each step. As this requires the generation of each diagonal of K'_1 and K'_2 , the diagonal method proved to more efficient in overall operations counts.

The negatives of Laplace matrices for the graphs in Figure 2.2 (the negatives are shown, for convenience in presentation) are shown below.

$$\begin{matrix}
 & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \end{matrix} \\
 \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 16 \end{matrix} & \left(\begin{matrix} -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 \end{matrix} \right)
 \end{matrix}$$

All diagonals of the pseudoinverses of these matrices are identical. If the row and column corresponding to node 1 and those corresponding to node a are removed, where both 1 and a are chosen arbitrarily, and might have been any other pair of nodes, then there are a number of distinct values for the diagonals of the reduced pseudoinverses. Hence the corresponding nodes may be grouped into a number of subsets. Thus there are equivalences between:

$$\begin{matrix}
 (\begin{matrix} 2 & 3 & 4 & 6 & 7 & 8 \end{matrix})_1 & (\begin{matrix} c & e & g & i & k & m \end{matrix})_1 \\
 (\begin{matrix} 5 \end{matrix})_2 & (\begin{matrix} o \end{matrix})_2 \\
 (\begin{matrix} 9 & 10 & 11 & 12 \end{matrix})_3 & (\begin{matrix} b & d & j & l \end{matrix})_3 \\
 (\begin{matrix} 13 & 14 & 15 & 16 \end{matrix})_4 & (\begin{matrix} f & h & n & p \end{matrix})_4
 \end{matrix}$$

$$\begin{array}{c}
a \\
b \\
c \\
d \\
e \\
f \\
g \\
h \\
i \\
j \\
k \\
l \\
m \\
n \\
o \\
p
\end{array}
\begin{pmatrix}
a & b & c & d & e & f & g & h & i & j & k & l & m & n & o & p \\
-4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & -4 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & -4
\end{pmatrix}$$

For this example it is convenient to use subscripts to show nodes which correspond to particular values of a diagonal element in the pseudoinverse. If one such diagonal element has a unique value, as does that corresponding to nodes 5 and o in the above matrices, then these nodes correspond in the original graphs. Where several diagonal elements are the same, the nodes which correspond to these diagonal values are grouped into bracketed sets identified by a unique subscript. At this stage, for example, any pair of nodes may be matched from bracketed sets with the same subscript. At later stages of the algorithm, the nodes chosen for matching must be checked to ensure that their neighbours match. For each iteration illustrated below, the nodes are grouped in sets corresponding to the current values of the diagonals in the pseudoinverses of the reduced matrices. The subscripts indicate previous groupings to which each set has belonged. These in turn are a representation of the adjacency connections which each node has had. For example, in the iteration below, nodes 11 and 12 could match j , l , f or p as in this iteration all the diagonals corresponding to these nodes are the same. This is shown by the second subscript 4 . However, the first subscripts are different, being 3 for $(11\ 12)$ and $(j\ l)$ and 4 for $(f\ p)$. This shows that in the previous iteration, 11 and 12 had been connected to at least one node which had been matched to a node which had been connected to j and l . This is not the case for f and p .

An arbitrary match is made between nodes 2 and c . The rows and columns corresponding to these nodes are removed. The diagonal equivalences are now:

$$\begin{array}{l}
(\ 3 \ 4 \ 7 \ 8 \)_{11} \\
(\ 6 \)_{12} \\
(\ 9 \ 10 \)_{33} \\
(\ 11 \ 12 \)_{34}
\end{array}
\begin{array}{l}
(\ 15 \ 16 \)_{43} \\
(\ 13 \ 14 \)_{44}
\end{array}
\begin{array}{l}
(\ e \ g \ i \ m \)_{11} \\
(\ k \)_{12} \\
(\ b \ d \)_{33} \\
(\ j \ l \)_{34}
\end{array}
\begin{array}{l}
(\ h \ n \)_{43} \\
(\ f \ p \)_{44}
\end{array}$$

Removal of the rows and columns corresponding to the match of nodes 6 and k and the arbitrary pair, say, 3 and e , leads to the next set of equivalences based on the diagonals of the pseudoinverses:

$$\begin{array}{cccccccc}
 \binom{4}{7}_{112} & \binom{8}{111} & & & \binom{g}{i}_{112} & \binom{m}{111} & & \\
 \binom{9}{10}_{333} & \binom{12}{11}_{343} & \binom{14}{13}_{443} & \binom{15}{16}_{433} & \binom{d}{b}_{333} & \binom{l}{j}_{343} & \binom{f}{h}_{443} & \binom{n}{p}_{433} \\
 \binom{10}{11}_{334} & \binom{11}{12}_{344} & \binom{13}{14}_{444} & \binom{15}{16}_{434} & \binom{d}{b}_{334} & \binom{l}{j}_{344} & \binom{f}{h}_{434} & \binom{n}{p}_{444}
 \end{array}$$

Node 7 could now be matched to node i but this would lead to a disconnected graph. As will be seen nodes 7 and i are in the chain linked to the tree roots. Removal of the pair 9 and d leads to the following sets of equivalences:

$$\begin{array}{ccccccc}
 \binom{4}{7}_{1111} & & & & \binom{g}{i}_{1111} & & \\
 \binom{8}{11}_{1122} & & & & \binom{m}{1113} & & \\
 \binom{14}{10}_{4434} & & & & \binom{f}{b}_{4434} & & \\
 \binom{13}{11}_{4445} & & & & \binom{d}{j}_{3345} & \binom{p}{h}_{4445} & \\
 \binom{16}{11}_{4346} & & & & \binom{l}{j}_{3446} & \binom{h}{n}_{4346} & \\
 \binom{15}{12}_{4337} & & & & \binom{l}{n}_{3437} & \binom{n}{4337} &
 \end{array}$$

Thus there is now a matching $\{ 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 \}$ to $\{ a c e g o k i m d b j l p f n h \}$.

Using this matching to rearrange the rows and columns of the second Laplacian matrix leads to the first Laplacian matrix. The detail of the corresponding remapping of the second graph in Figure 2.2 is not given here as it is not relevant to the purpose of this paper.

At most this process would require diagonal elements of a further $2(m - 1)$ pseudoinverse evaluations of matrices of size m which is decreasing each iteration.

If the tree roots are 16 and p , and they are retained throughout the reduction process, the graphs will be reduced to the strings $\{ 16 7 13 \}$ and $\{ h i p \}$. Eliminating any pair of nodes which belong to these strings will lead to a discontinuity in the corresponding graph.

The most computationally intensive part of this process is updating the triangular factors of C^{-1} (ie, $AA^T + BB^T$), for which standard algorithms are available (Bennett[1]). Efficient algorithms for calculating the inverse of a reduced matrix from the inverse of the current matrix are available (eg, Bennett[1]). Once the triangular factors of C^{-1} (see (2.2)) have been computed, these (and those of its successors) can also be updated efficiently (eg, Bennett[1]). These algorithms can be used to ensure that the removal of a row and column from the adjacency matrix does not reduce the equivalent graph to one which is disconnected. Alternatively, if the removal of a node from each graph were to result in disconnected subgraphs, recognised by zero pivots, these could in turn be matched and the process continued with reduced subnetworks.

2.3 The Testing Process.

The process of using the pseudoinverses of a pair of matrices to test for isomorphism in the corresponding graphs, and, if they are isomorphic, to obtain a map from one onto the other may be summarised as follows:

1. Evaluate diagonal elements of K_1 and K_2 . Are both sets the same after rearrangement in (say) ascending order?

- If no, the graphs are not isomorphic. If yes,
2. Are the diagonal elements unique?
If no, go to step 5. If yes,
 3. If $(K_1)_{jj}$ is the same as $(K_2)_{kk}$ does node j in G_1 correspond to node k in G_2 , for each pair of diagonal elements, j and k , say?
If no, the graphs are not isomorphic. If yes,
 4. If the nodes in G_1 have been mapped to the nodes in G_2 and node j in G_1 , say, maps to node r in G_2 and node k maps to node s , then if there is an edge from j to k , is there a corresponding edge from r to s for all relevant matched nodes?
(This check ensures that for a mapping defined by the K_{ii} , there is a corresponding mapping of the graph elements.)
If no, the graphs are not isomorphic. (This is a check to ensure that the condition referred to in the third last paragraph in Section 2.1 does not arise.)
If yes, the graphs are isomorphic.
 5. If the diagonal elements are not unique, a procedure as described above in Section 2.2 must be undertaken. This process matches nodes one at a time, until the reduced graphs are shown not to be isomorphic or the check described in 4 above can be carried out.

2.4 Exploiting Structure.

Choosing spanning trees has numerical advantages. Extensive use may be made of triangular factors (Householder [4]) in improving computational efficiency. As noted in Section 2.1, A is assumed to have been given as a lower triangular matrix. Hence $A^{-1}G$, where G is any matrix, can always be computed by backward substitution without explicit calculation of A^{-1} . Most of the work in computing the diagonal elements of K is in evaluating (2.2) in Section 2.1, the expression for C^{-1} . This calculation can also be arranged to maximise use of triangular factors.

Trees and complete graphs (ie, graphs in which each vertex is connected to all other vertices) are special cases leading to certain computational simplifications which can be used to reduce computations count in graphs which have few links or are nearly complete. The analysis can be given in much greater detail. However, as it is very dependent on the properties of the graph, it will not be developed here.

It should be noted that the abstract notion of a pseudoinverse can be given a physical interpretation. For example, it can be used to determine currents in an electrical network.

2.5 Operations Counts.

The number of operations required to compute the diagonal elements of K is, with no allowance for sparsity, $O(mn^2)$. Once the diagonal elements of K for each

of the two incidence matrices have been computed, the number of comparisons to determine whether one pseudoinverse is a permutation of the other is, if all the diagonal elements are unique, $O(n \log n)$. In the case of a near complete graph, with r being the number of edges needed to complete the graph, or a near tree with r links, $O(rn^2)$ operations may be needed for this calculation. If there is potential ambiguity or symmetry present, to calculate the relevant elements of K at a stage with p nodes unmatched may require with no exploitation of sparsity up to $O(pn^2)$ further operations.

3 Conclusions.

An algorithm for testing isomorphism based on the pseudoinverse of the incidence matrix of a graph would seem to offer an effective way of testing for graph isomorphism. It almost always runs in polynomial time and handles special cases efficiently. It is particularly efficient for disproving isomorphism. It does not require any special properties of the graphs such as planarity and can be used for directed or undirected graphs. Considerable improvements can be made to the operations counts by exploiting the sparsity of various parts of the incidence matrices derived from the graphs.

The test problems described were run in the Mathematica system (Wolfram [13]) on a Macintosh iix. Mathematica has inbuilt functions for pseudoinverses, based on eigenvalues. These were used to verify the results.

One of the reviewers has pointed out that although the method did not invoke step 4 of Section 2.3 to demonstrate nonisomorphism on examples tried by the reviewer and the authors, more exhaustive testing might do so as the method is essentially heuristic.

REFERENCES

1. J. M. Bennett, *Triangular factors of a modified matrix*, Numerische Mathematik 7, 217-221 (1965).
2. J. M. Bennett, *The pseudo-inverse and associated topics*, Tech. Report 43, Basser Dept of Comp. Sc., Univ. of Sydney, Australia (1966).
3. C. Hoffmann, *Group Theoretic Algorithms and Graph Isomorphism*, Lecture Notes in Computer Science No. 136, Springer-Verlag, New York (1982).
4. A. S. Householder, *Principles of Numerical Analysis*, McGraw-Hill (1953).
5. W. Kocay, *Abstract data types and graph isomorphism*, Jnl Combinatorics, Information and System Sciences, 247-259 (1984).
6. Y. Kudo, T. Yamasaki and S. Sasaki, *The characteristic polynomial uniquely represents the topology of a molecule*, J. Chem. Documentation 13, 225-227 (1973).
7. W. Mayeda and S. Seshu, *Generation of trees without duplications*, IEEE Trans. Circuits, Th. 12, 181-185 (1965).
8. B. McKay, *Practical graph isomorphism*, Congressus Numerantium 30, 45-87 (1981).
9. B. Noble, *Applied Linear Algebra*, Prentice-Hall, Eaglewood Cliffs N.J. (1969).

10. R. C. Read, *Teaching graph theory to a computer* in *Recent Progress in Combinatorics*, edited. W.T. Tutte, Ac. Press, New York (1969).
11. J. Seberry Wallis, *Hadamard Matrices* in *Combinatorics: Room Squares, Sum-Free Sets, Hadamard Matrices*, Lecture Notes in Mathematics No 292, edited by W.D. Wallis, A.P. Street and J. Seberry Wallis, Springer-Verlag, Berlin (1972).
12. S. Skiena, *Implementing Discrete Mathematics*, Addison-Wesley, Chapter 5 (1990).
13. S. Wolfram, *Mathematica*, Addison Wesley, California (1991).