Linear Algorithms for Isomorphism of Maximal Outerplanar Graphs

T. BEYER

University of Oregon, Eugene, Oregon

W. JONES AND S. MITCHELL

University of Louisville, Louisville, Kentucky

ABSTRACT Two linear algorithms are presented for solving the isomorphism problem for maximal outerplanar graphs (mops) These algorithms present improvements over corresponding linear algorithms for planar graph isomorphism when applied to mops The algorithms are based on a code for a mop G which is obtained from a unique Hamiltonian cycle in G The first involves a string-matching automaton and the second involves the removal of vertices of degree two in layers until either an edge or triangular face remains

KEY WORDS AND PHRASES isomorphism, analysis of algorithms, graph theory, maximal outerplanar graphs

CR CATEGORIES 5 32

1. *Introduction*

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there exists a 1-1 function f $V_1 \rightarrow V_2$ from V_1 onto V_2 such that two vertices u_1, v_1 are adjacent in G_1 if and only if $f(u_1), f(v_1)$ are adjacent in G_2. The graph isomorphism problem (given two graphs G_1 and G_2, determine whether G_1 and G_2 are isomorphic) has been much studied in the literature [17]. Although it is not known whether the general graph isomorphism problem is NP-complete, the related subgraph isomorphism problem is NP-complete [1]. One of the better isomorphism algorithms, due to Corneil and Gotlieb [4], has an exponential time complexity in the worst case

Linear-time solutions to the isomorphism problem have been obtained by Hopcroft and Tarjan [9] for trees, and by Hopcroft and Wong [10] for planar graphs. This planar graph algorithm presents an improvement over an $O(V^2)$ algorithm for triply connected planar graphs by Weinberg [19] and an $O(V \log V)$ algorithm by Hopcroft and Tarjan [9].

In this paper the isomorphism problem is solved for a subclass of planar, chordal graphs called maximal outerplanar graphs (mops). Since it was shown in [8] that trees, outerplanar graphs, and planar graphs form a natural hierarchy, this solution, together with those for trees and planar graphs, completes the sequence It is also worth noting that the general graph isomorphism problem is polynomially reducible to the isomorphism problem for chordal graphs [3]. Consequently it is of interest to know which other classes of chordal graphs, in addition to trees and mops, have polynomial-time isomorphism solutions.

The algorithms presented here, by taking advantage of specific properties of mops, are more efficient than Hopcroft and Wong's algorithm when applied to mops.

It will be shown that each mop has associated with it a unique Hamiltonian cycle, which

Authors' present addresses T Beyer, The Computer Center, University of Oregon, Eugene, OR 97403; W Jones, Department of Applied Mathematics and Computer Science, University of Louisville, Louisville, KY 40208, S Mitchell, Department of Computer Science, University of Oregon, Eugene, OR 97403

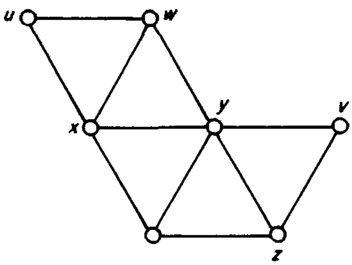


FIG 1 A maximal outerplanar graph (mop)

can be determined in linear time [14]. This algorithm is briefly stated in the appendix. This cycle can be used, in a manner similar to Weinberg's use of Eulerian cycles in triply connected graphs [19], to provide a unique code for a mop. The first algorithm relies on a string matching algorithm presented by Morris and Pratt [16], appearing in [1]. The second algorithm is further characterized by the use of a restricted geometric dual of a mop.

2. *Definitions*

An *outerplanar graph* is a graph which can be embedded in the plane in such a way that all vertices lie on the exterior face. A *maximal outerplanar graph* (*mop*) is an outerplanar graph such that the addition of an edge between any two nonadjacent vertices results in a nonouterplanar graph. The class of mops is equivalent to the class of triangulations of polygons, and forms a subclass of the class of planar 2-trees [18] and a subclass of the class of chordal graphs [5] (see also [3] and [14]).

A mop can be defined recursively as follows [2]:

(1) The graph K_3, consisting of three mutually adjacent vertices, is a mop.

(2) If G is a mop which is embedded in the plane so that every vertex lies on the exterior face, and H is obtained by joining a new vertex to the two vertices of an edge on the exterior face of G, then H is a mop.

(3) Nothing is a mop unless its being so follows from a finite number of applications of statements (1) and (2).

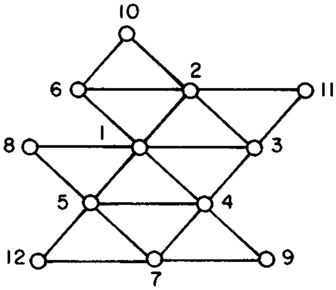
An immediate consequence of the recursive definition of a mop is that every mop contains at least two vertices of degree 2 (cf. [7]); we call such a vertex a *2-vertex*. The graph in Figure 1 is a mop in which u and v are 2-vertices.

A mop G having m vertices labeled $1, 2, \ldots, m$ is *recursively labeled* if vertices 1, 2, and 3 form a triangle in G, and every vertex with label k is adjacent to exactly two vertices with label k for $3 \leq k \leq m$. Mops which are recursively labeled reflect the manner in which they can be constructed according to the recursive definition given above. Recursive mops can also be represented canonically by two linear arrays $HIGH(1), HIGH(2), \ldots, HIGH(m)$ and $LOW(1), LOW(2), \ldots, LOW(m)$, where $HIGH(k)$ and $LOW(k)$ are the labels of the two vertices adjacent to vertex k whose labels are less than k. We assume that $HIGH(k) > LOW(k)$, that $HIGH(1)$, $LOW(1)$, and $LOW(2)$ are undefined, and that $HIGH(2) = 1$. Figure 2 illustrates a recursive mop and its canonical representation.

An algorithm, linear in the number of vertices, which has been previously presented [14] to find the unique Hamiltonian cycle in G, is restated in the appendix.

3. *Hamiltonian Degree Sequence of Mops*

Since every mop G can be embedded in the plane in such a way that every vertex lies on the exterior face, the collection of exterior edges defines a Hamiltonian cycle in G (a cycle which contains every vertex of G exactly once). Let $u_1, u_2, \ldots, u_m, u_1$ define a Hamiltonian cycle in a mop G, and let $D = d_1, d_2, \ldots, d_m$ be the corresponding sequence of degrees of these vertices—i.e., the degree of vertex u_i is d_i; we call D a Hamiltonian degree sequence of G.



	1	2	3	4	5	6	7	8	9	10	11	12
HIGH		1	2	3	4	2	5	5	7	6	3	7
LOW			1	1	1	1	4	1	4	2	2	5

FIG 2 A linear representation of a mop

Since each vertex u_1 can initiate a Hamiltonian degree sequence in each of two ways (clockwise and counterclockwise), every mop G with m vertices has $2m$ (not necessarily distinct) Hamiltonian degree sequences We next show that any one of these sequences uniquely determines the mop G up to isomorphism.

THEOREM 1. *Let $D = d_1, d_2, \ldots, d_m$ be a Hamiltonian degree sequence of some mop G Then G is unique up to isomorphism.*

PROOF. We proceed by induction on the number m of vertices. The result is trivially true for $m \le 5$ since there is only one mop on 3, 4, and 5 vertices, respectively

Assume the result is true for all values of $m \le k$. Let $D = d_1, d_2, \ldots, d_k, d_{k+1}$ be the Hamiltonian degree sequence of some mop G and let $u_1, u_2, \ldots, u_k, u_{k+1}$ be the corresponding sequence of vertices. Since this is the degree sequence of some mop G, at least one vertex u_i in G must have degree 2, i.e, for some i, $d_i = 2$ We assume without loss of generality that $1 < i < k + 1$. It follows, therefore, that vertex u_i must be adjacent to vertices u_{i-1} and u_{i+1}, and furthermore that vertices u_{i-1} and u_{i+1} are themselves adjacent. Thus $D' = d_1, d_2, \ldots, d_{i-1}, d_{i-1} - 1, \ldots, d_k, d_{k+1}$ is a Hamiltonian degree sequence of a mop G' with k vertices. But by our inductive hypothesis, this mop is unique. Consequently it follows that the mop described by D is unique. □

COROLLARY 1. *A sequence $d_1, d_2, \ldots, d_{i-1}, 2, d_{i+1}, \ldots, d_m$ is a Hamiltonian degree sequence of a mop if and only if the sequence*

$$d_1, d_2, \ldots, d_{i-1} - 1, d_{i+1} - 1, \ldots, d_m$$

is a Hamiltonian degree sequence

We next present Algorithm MOP, which in linear time takes a Hamiltonian degree sequence and produces the unique, corresponding mop. The Hamiltonian degree sequence is reduced by the process described in Corollary 1 until the sequence 222 remains. As each vertex is removed from the sequence, the highest unused label from the set of integers 1, 2, ..., M is associated with it. This results in a recursive labeling of the mop, from which a "canonical" representation is easily obtained. The algorithm requires that the Hamiltonian degree sequence be rotated so as not to begin with a vertex of degree 2.

ALGORITHM MOP

Given a Hamiltonian degree sequence $DEG(1)$, $DEG(2)$, $\ldots$, $DEG(M)$ of a mop with M vertices, to produce a recursive labeling $LABEL(1)$, $LABEL(2)$, $\ldots$, $LABEL(M)$ of G, where $LABEL(I)$ is the new label of the vertex with $DEG(I)$, arrays *LEFT* and *RIGHT* form a doubly linked list of the items in the array *DEG*, vertices of degree 2 are labeled as they are removed from the doubly linked list, throughout L is the highest integer between 1 and M which has not been used to label a vertex

```
Step 0  [Initialize]
        Set LEFT(1) ← M
        For I ← 2 to M do set LEFT(I) ← I − 1 od
        For I ← 1 to M − 1 do set RIGHT(I) ← I + 1 od
        Set RIGHT(M) ← 1
        Set NEXT ← 2
        Set L ← M
Step 1  [Remove and label all but three vertices]
        While L > 3 do
          [find a 2-vertex and label it]
          while DEG(NEXT) ≠ 2 do set NEXT ← RIGHT(NEXT) od
          set LABEL(NEXT) ← L
          set L ← L − 1
          [update adjacencies]
          set L ← LEFT(NEXT)
          set R ← RIGHT(NEXT)
          set DEG(L) ← DEG(L) − 1
          set DEG(R) ← DEG(R) − 1
          set RIGHT(L) ← R
          set LEFT(R) ← L
          [get NEXT vertex]
          if DEG(L) = 2
          then set NEXT ← L
          else set NEXT ← R
          fi od
Step 2  [Label last three vertices]
        Set LABEL(NEXT) ← 3
        Set LABEL(RIGHT(NEXT)) ← 2
        Set LABEL(LEFT(NEXT)) ← 1
        Stop
```

The example in Figure 3 illustrates the application of Algorithm MOP to a Hamiltonian degree sequence for the mop G. The new labels of the vertices in the degree sequence are used to generate a new representation of a mop H which is isomorphic to G.

4. *Isomorphism of Mops*

We have by now established that a Hamiltonian degree sequence uniquely defines a mop. Therefore, two mops are isomorphic if and only if they have an identical Hamiltonian degree sequence. However, two recognition problems arise when we attempt to decide whether two degree sequences define the same mop. First, two sequences for the same mop may not start at the same vertex. Second, even if they both begin at the same vertex, one may represent a clockwise traversal of the mop whereas the other reflects a counter-clockwise traversal.

We present two algorithms for solving this problem in time linear in the number of vertices. The first is based on a string-matching algorithm of Morris and Pratt [16], which appears in [1]. The second algorithm determines unique vertices at which to initiate Hamiltonian degree sequences.

STRING-MATCHING SOLUTION. This isomorphism problem can be formally characterized by the following result, which follows immediately from Theorem 1

THEOREM 2. *Let G and G' be mops with Hamiltonian degree sequences $D = d_1, d_2, \ldots, d_m$ and $D' = d'_1, d'_2, \ldots, d'_m$, respectively. G and G' are isomorphic if and only if D' is a cyclic shift and/or inversion of D.*

If \$ denotes an end-of-string marker, then the following result is immediate from Theorem 2

COROLLARY 2 *G and G' are isomorphic if and only if D' is a substring of $DD\$D^RD^R$.*

Morris and Pratt have presented an algorithm which will decide if D' is a substring of $DD\$D^RD^R$ in $O(m)$ steps independent of the size of the alphabet. Their algorithm constructs a deterministic finite automaton from sequence D' so that if the automaton is

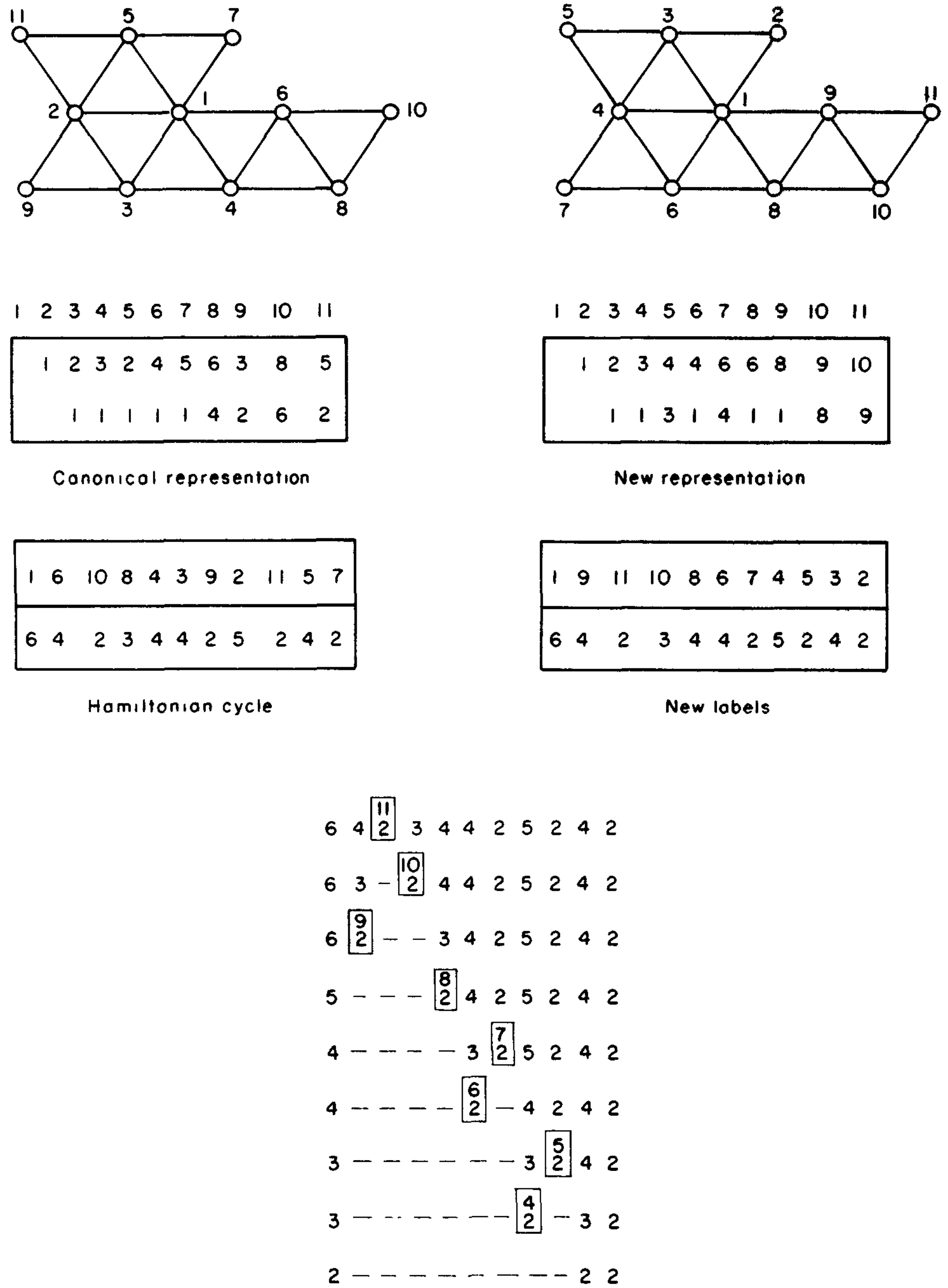
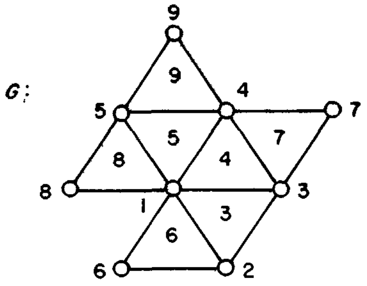


FIG 3 An example of Algorithm MOP

in state i after having read symbol d'_i and the next symbol read is d'_{i+1}, then the machine will move to state $i + 1$. The string $DD\$D^RD^R$ will serve as the input string to the automaton, and the subsequent acceptance or rejection of the input string is performed in $O(m)$ steps.

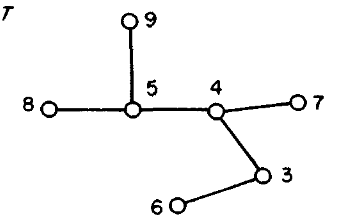
LAYERED REMOVAL OF 2-VERTICES SOLUTION. We wish to compare the Hamiltonian degree sequences of two mops after first ensuring that they begin at the same vertex. In



(a)

	1	2	3	4	5	6	7	8	9
HIGH			2	3	4	2	4	5	5
LOW			1	1	1	1	3	1	4

(b)



(c)

	1	2	3	4	5	6	7	8	9
DUAL				3	4	3	4	5	5

(d)

FIG 4 A recursive mop G and its dual tree T (a) recursive mop G with labeled triangular faces, (b) recursive representation of mop G, (c) dual tree T, (d) recursive representation of dual tree T

actuality we determine a set of either two or three vertices, which we can uniquely identify in a mop. We then use only the set of Hamiltonian degree sequences which begin with these vertices for comparative purposes.

We obtain this special set of vertices by methodically removing 2-vertices in layers, until either an edge (two vertices) or a triangle face (three vertices) remains. A first isomorphic check can be made at this time since the two mops are isomorphic only if what remains by this process is the same for both. Each of these two (or three) "central" vertices v in G_1 will be taken as the initial vertex of the two Hamiltonian degree sequences (clockwise and counterclockwise) which start at v. Only one such sequence S need be taken from the second mop G_2. The sequence is compared to each of the four (or six) sequences from G_1 symbol by symbol; G_1 and G_2 are isomorphic if and only if S matches one of these sequences.

We briefly describe a process by which the "central" vertices can be determined. Note that removing a 2-vertex from a mop is equivalent to removing a triangular face. The geometric dual of a mop G is a tree T if the exterior region is ignored. Hence, the removal of a 2-vertex from G is equivalent to removing an endvertex from the dual tree T.

Numerous examples or recursive representations of trees have appeared in the literature (cf. [6, 12–15]). We can easily obtain a recursive representation for the dual tree T from the recursive representation of the mop G. We associate with each triangular face (vertex of T) the highest label of the vertices which define the face. The dual tree T, therefore, will have labels in the range 3 to m (cf. Figure 4). The recursive representation $DUAL$ of T (shown in Figure 4(d)) is the same as the array $HIGH(I)$, $I = 4, \ldots, M$, of the recursive representation of G (with all 2's changed to 3's). We can then determine the Jordan center of T (cf. [11]) using an algorithm presented in [14]. The Jordan center of T will either be one tree vertex, corresponding to three "central" vertices in G, or two tree vertices, corresponding to two "central" vertices in G.

Two nonisomorphic mops which have isomorphic unlabeled dual trees are presented in Figure 5. If the dual trees are to be used directly in the isomorphic process, they must be

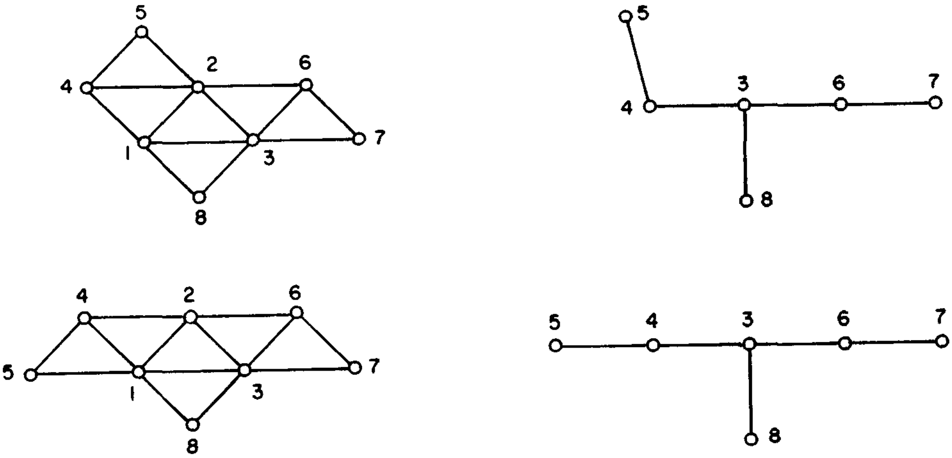


FIG 5

appropriately labeled to reflect their orientation in the plane and a labeled tree isomorphism algorithm must be used.

Appendix

We have seen that the exterior edges of a mop G define the Hamiltonian cycle; hence, to find the Hamiltonian cycle it is sufficient to be able to distinguish between exterior and interior edges. This is simple to do when G is described using the canonical recursive representation *HIGH* and *LOW*. The interior edges of G are precisely the edges $(HIGH(I), LOW(I))$ for $I = 3, \ldots, M$.

The algorithm iteratively removes 2-vertices by making one right-to-left pass over the canonical recursive representation, exposing interior edges to the outer face. Let us denote the two edges joining a 2-vertex to its adjacent vertices as *pendant edges* and the two adjacent vertices as *remote vertices*. (w and x are remote vertices of u in Figure 1.) Each of these interior edges can be marked so as to prohibit its addition to the set through the use of an array *CANT*. A value of $CANT(I) = 0$ indicates that both of the pendant edges incident to vertex I are in the Hamiltonian cycle. A value of $CANT(I) = -1$ indicates that neither of these pendant edges are in the cycle. Otherwise, only one pendant edge incident to vertex I is on the Hamiltonian cycle and $CANT(I)$ identifies the remote vertices of the forbidden edge.

ACKNOWLEDGMENTS. The authors wish to thank Professor Kellogg Booth for his suggested inclusion of the string matching solution and for his restatement of the isomorphism problem in Theorem 2. They also thank Professor Stephen Hedetniemi for his many helpful comments during the development of this paper.

REFERENCES

1 AHO, A, HOPCROFT, J, AND ULLMAN, J *The Design and Analysis of Computer Algorithms* Addison-Wesley, Reading, Mass, 1974

2 BEINEKE, L, AND PIPPERT, R A census of ball and disk dissections In Proc of the Graph Theory Conf. at Western Michigan University, 1972, Y. Alavi, D. Lick, and A White, Eds., Springer Lecture Notes in Math #303, Springer-Verlag, Heidelberg, 1974, pp 25–40

3 BOOTH, K, AND LUEKER, G Linear algorithms to recognize interval graphs and test for the consecutive ones property. *J Comptr. Syst Sci 13* (1976), 335–379

4 CORNEIL, D G, AND GOTLIEB, C C An efficient algorithm for graph isomorphism. *J ACM 17*, 1 (Jan. 1970), 51–64

5 GAVRIL, F Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph *SIAM J. Comptng 1* (1972), 180–187.

6 GOODMAN, S, HEDETNIEMI, S., AND TARJAN, R. *B*-matchings in trees *SIAM J. Comptng. 5* (1976), 104–108.

7 HARARY, F. *Graph Theory* Addison-Wesley, Reading, Mass., 1969.

8 HEDETNIEMI, S , CHARTRAND, G , AND GELLER, D Graphs with forbidden subgraphs. *J. Combinatorial Theory 10* (1971) 12-41.

9 HOPCROFT, J , AND TARJAN, R Isomorphism of planar graphs In *Complexity of Computations*, R. Miller and J. Thatcher, Eds., Plenum Press, New York, 1972, pp. 143-150.

10 HOPCROFT, J , AND WONG, J Linear time algorithm for isomorphism of planar graphs Proc. of Sixth Annual ACM Symp. on Theory of Computing, Seattle, Wash., 1974, pp. 172-184.

11 JORDAN, C. Sur les assemblages de lignes. *J Reine Angew. Math. 70* (1869), 185-190.

12 KERSHENBAUM, A., AND VAN SLYKE, R Recursive analysis of network reliability *Networks 3* (1973), 81-94

13 MISHRA, J , AND TARJAN, R Optimal chain partitions of trees *Information Processing Letters 4* (1975), 24-26

14 MITCHELL, S Algorithms on trees and maximal outerplanar graphs Design, complexity analysis, and data structures study Ph D Th , U of Virginia, Charlottesville, Va., 1977

15 MITCHELL, S , COCKAYNE, E , AND HEDETNIEMI, S Linear algorithms on recursive representations of trees. *J Comptr Syst Sci 18* (1979), 76-85

16 MORRIS, J. H JR., AND PRATT, V R A linear pattern matching algorithm Tech Rep No 40, Computing Center, U. of California, Berkeley, Calif., 1970.

17 READ, R , AND CORNEIL, D The graph isomorphism disease *J Graph Theory 1* (1977), 239-363

18 ROSE, D On single characterizations of *K*-trees *Discrete Math 7* (1974), 317-322

19 WEINBERG, L A simple and efficient algorithm for determining isomorphism of planar triply connected graphs. *IEEE Trans Circuit Theory CT-13* (1966), 142-148

RECEIVED JANUARY 1978, REVISED AUGUST 1978