branches. However, the graph in Fig. 4 contains more branch disjoint paths between a vertex in $\{v_1, v_2, v_3\}$ and a vertex in $\{v_4, v_5, v_6\}$ than the graph in Fig. 3(c).

Example 3

Given $\{d_i\} = \{6, 6, 5, 4, 3, 2\}$, we want to construct a pseudosymmetric graph by using Algorithm C. The resulting graph is in Fig. 5.

Example 4

Given $\{r_{12}, \cdots, r_{56}\} = \{6, 4, 2, 5, 4\}$, we want to construct a pseudosymmetric graph by using Algorithm D. The resulting graph is shown in Fig. 6.

References

[1] F. T. Boesch and R. T. Thomas, "Optimal damage resistant nets," *Proc. IEEE Internatl. Conf. on Communications*, vol. 4, pp. 688–693, June 1968.

[2] F. T. Boesch and I. T. Frisch, "On the smallest disconnecting set in a graph," *IEEE Trans. Circuit Theory*, vol. CT-15, pp. 286–288, September 1968.

[3] I. T. Frisch, "An algorithm for vertex pair connectivity," *Internatl. J. Control*, vol. 6, pp. 579–593, 1967.

[4] S. L. Hakimi, "An algorithm for the construction of the least vulnerable communication network," *IEEE Trans. Circuit Theory*, vol. CT-16, pp. 229–230, May 1969.

[5] D. Rosenbaum and J. B. Friedman, "Redundant networks," Mitre Corp., Bedford, Mass., Rept. TM-3195, October 1961.

[6] K. Steiglitz, P. Weiner, and D. Kleitman, "The design of minimum cost survivable networks," *IEEE Trans. Circuit Theory*, vol. CT-16, pp. 455–460, November 1969.

[7] T. G. Williams, "On the survivability of communications systems," *Proc. 5th Natl. Conv. on Military Electronics* (Washington, D. C.), June 27–28, 1961, pp. 251–255.

[8] H. Frank and W. Chou, "Connectivity considerations in the design of survivable networks," *IEEE Trans. Circuit Theory* (to be published).

[9] I. T. Frisch, "Flow variation in multiple min-cut calculations," *J. Franklin Inst.*, vol. 287, pp. 61–72, January 1969.

[10] ——, "Analysis of the vulnerability of communication nets," *Proc. 1st Ann. Princeton Conf. on Systems Science* (Princeton, N. J.), March 1967.

[11] H. Frank, "Vulnerability of communication networks," *IEEE Trans. Communications Technology*, vol. COM-15, pp. 778–789, December 1967.

[12] F. Harary, "The maximum connectivity of a graph," *Proc. Natl. Acad. Sci.*, vol. 48, pp. 1142–1146, July 1962.

[13] T. G. Williams, "The design of survivable communications networks," *IEEE Trans. Communication Systems*, vol. CS-11, pp. 230–241, June 1963.

[14] L. R. Ford, Jr. and D. R. Fulkerson, *Flows in Networks*. Princeton, N. J.: Princeton University Press, 1962.

[15] W. H. Kim and R. T. Chien, *Topological Analysis and Synthesis of Communication Networks*. New York: Columbia University Press, 1962.

[16] R. E. Gomory and T. G. Hu, "Synthesis of a communication network," *J. Soc. Indust. Appl. Math.*, June 1964.

A New Planarity Test Based on 3-Connectivity

JOHN BRUNO, MEMBER, IEEE, KENNETH STEIGLITZ, MEMBER, IEEE, AND LOUIS WEINBERG, FELLOW, IEEE

***Abstract*—In this paper we give a new algorithm for determining if a graph is planar. The algorithm is based on Tutte's theory of 3-connected graphs, and provides a structural decomposition of the graph. Results are presented in the algorithmic form, and a computer program is described.**

I. Introduction

IN THIS paper we present a new algorithm for determining whether or not a given graph can be drawn in the plane without crossed edges [5], [8]–[11]. The method used here reveals certain structural properties of the graph in the course of testing for planarity. More specifically, the structural characterization of planar graphs, first given by Mac Lane [4], is used to decompose the original graph into smaller pieces that are simple and triply connected. Thus the underlying triply connected structure of the given graph (planar or nonplanar) is obtained.

Our Algorithm Planar is based on Tutte's theory [1] of triply connected graphs and relies on the fact that a graph is planar if and only if all of its triply connected components are planar [4]. We use Tutte's theory to obtain an efficient algorithm for determining if a simple, triply connected graph is planar. It will become clear that the uniqueness of the map of a triply connected graph is the key to the efficiency of this algorithm.

In Section III we present Tutte's theory in preparation for Section V, wherein we present Algorithm P (the determination of the planarity of a simple, triply connected graph). The case of arbitrary graphs is presented in Section VI. Lastly, we give the salient features of a computer program based on Algorithm Planar.

There are two notational conventions that we use. If S is a finite set, then we use the notation $\alpha(S)$ for the cardinality of the set S. In order to avoid a disproportionate amount of subscripts (or superscripts), we

Manuscript received April 25, 1969; revised September 12, 1969. This research was supported in part by the U. S. Army, Durham, N. C., under Contract DA-C04-69-0019 and by the National Science Foundation under Grant GK 1093.

J. Bruno and K. Steiglitz are with the Department of Electrical Engineering, Princeton University, Princeton, N. J.

L. Weinberg is with the Department of Electrical Engineering, City University of New York, New York, N. Y.

adopt the following notation, which is quite conventional in computer programming. Let $f(S, T, U, \cdots)$ be some expression. Then by

$$K \leftarrow f(S, T, U, \cdots)$$

we mean that one calculates the quantity determined by the expression on the right-hand side of the arrow and then uses the letter K to denote the calculated quantity. This operation allows one to have the symbol K on both sides of the arrow. For example, let S, T, and K be three finite sets, and write

$$K \leftarrow (S \cup T) \cap K.$$

We first calculate the expression on the right-hand side of the arrow; that is, we determine the set $(S \cup T) \cap K$. Then this set is subsequently denoted by the letter K. We can view this operation as one in which the elements in the set K are "changed" or "updated." It is, of course, not necessary for the symbol on the left-hand side of the arrow to appear on the right-hand side also.

II. Terminology

A *graph* G is defined by the following:

1) $E(G)$, a finite set of *edges*;
2) $V(G)$, a finite set of *vertices*; and
3) a relation of incidence that associates with each edge a pair of vertices, not necessarily distinct, called its *ends*. An edge with coincident ends is called a *loop*.

Let $v \in V(G)$. The *valence* of v is equal to the number of edges incident to v, where loops are counted twice.

A graph G is called *simple* if it has no loops and no two edges have the same pair of ends.

Two vertices are said to be *adjacent* if they are the ends of the same edge.

A sequence $v_1, \cdots, v_k$ of vertices of a graph G is said to form a *path* from v_1 to v_k if v_i is adjacent to v_{i+1} for $i = 1, \cdots, k-1$.

A graph G is said to be *connected* if there exists a path between every pair of vertices in $V(G)$.

A graph H is called a *subgraph* of G if $E(H) \subseteq E(G)$, $V(H) \subseteq V(G)$, and the ends of the edges of H are the same as in G.

There are two operations whereby we can obtain new graphs from a given graph. These operations play a central role in the following sections. Let G be a graph and $e \in E(G)$.

1) By G op e we mean the subgraph of G with edge set $(E(G) - \{e\})$ and vertex set $V(G)$; that is,

$$E(G \text{ op } e) = (E(G) - \{e\}),$$

$$V(G \text{ op } e) = V(G),$$

and G op e is a subgraph of G.

2) By G cl e we mean the graph obtained from G by the following steps:

a) form G op e;
b) if v_1 and v_2 are the ends of e in G, then we coalesce these vertices in the graph G op e and denote the resulting graph by G cl e.

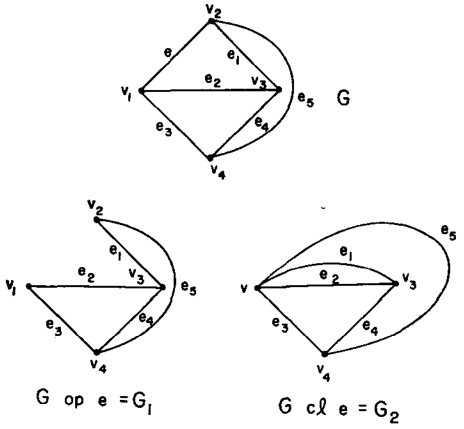

Fig. 1. Graphs of Example 1.

Example 1

Let G be the graph in Fig. 1. G op e and G cl e are also shown in Fig. 1.

We define the *union* of two subgraphs H and K of G as the subgraph $H \cup K$ of G, which satisfies:

1) $E(H \cup K) = E(H) \cup E(K)$ and
2) $V(H \cup K) = V(H) \cup V(K)$.

The *intersection* of two subgraphs H and K of G is defined as the subgraph $H \cap K$ of G, which satisfies:

1) $E(H \cap K) = E(H) \cap E(K)$ and
2) $V(H \cap K) = V(H) \cap V(K)$.

We say G is *2-separable* if it is the union of two subgraphs J and K with the following properties:

1) $E(J) \cap E(K) = \phi$,
2) $\alpha(V(J) \cap V(K)) \leq 2$, and
3) each of the subgraphs J and K has a vertex not belonging to the other.

A pair $\{J, K\}$ satisfying the above conditions is called a *2-separator* of G.

A graph with at least 4 vertices which is not 2-separable, is called a *triply connected* graph. We will be very much concerned with the properties of simple, triply connected graphs.

III. Tutte's Theory of Simple, Triple Connected Graphs

A wheel is a special kind of simple, triply connected graph and it plays a central role in Tutte's theory [1]. A *wheel of order n*, where n is an integer ≥ 3, is a graph W_n defined as follows:

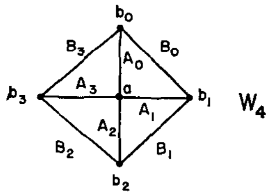

Fig. 2. The wheel of order 4.

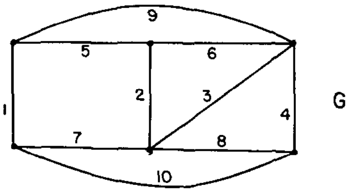

Fig. 3. Graph of Example 2.

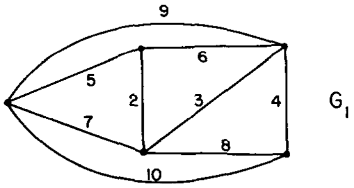

Fig. 4. Graph G_1 of Example 2.

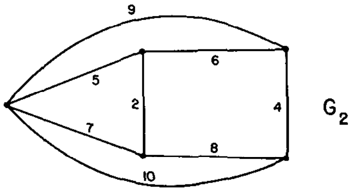

Fig. 5. Graph G_2 of Example 2.

1) $V(W_n) = \{a, b_0, b_1, \cdots, b_{n-1}\}$,
2) $E(W_n) = \{A_0, A_1, \cdots, A_{n-1}, B_0, B_1, \cdots, B_{n-1}\}$, and
3) the ends of A_i are a and b_i and the ends of B_i are b_i and b_{i+1}.

The suffices are residues mod n. The wheel W_4 is shown in Fig. 2.

It is easy to prove that every wheel is a simple, triply connected graph. There is, however, an additional property of a wheel that is extremely important.

Let G be a simple, triply connected graph. We call an edge $e \in V(G)$ an *essential* edge if neither G op e nor G cl e is both simple and triply connected. It is easy to see that *in a wheel every edge is essential*. Tutte [1] has proved the converse statement: if G is a simple, triply connected graph in which every edge is essential, then G is a wheel. We state this important result explicitly.

Theorem 1 [Tutte]

Let G be a simple, triply connected graph in which every edge is essential. Then G is a wheel.

IV. Reduction Sequences

Let G be a simple, triply connected graph. Then the sequence

$$G_0, G_1, G_2, \cdots, G_r$$

of graphs is called a *reduction sequence of G* if the following conditions are satisfied:

1) G_k is a simple, triply connected graph for $k = 1, \cdots, r$,
2) $G_0 = G$,
3) G_r is a wheel, and
4) either $G_{k+1} = G_k$ cl e or $G_{k+1} = G_k$ op e, where $e \in E(G_k)$ for $k = 0, \cdots, r-1$.

Example 2

Let G be the graph in Fig. 3. G is a simple, triply connected graph. According to Theorem 1, G is a wheel if every edge of G is essential. Since G cl 1 is a simple, triply connected graph, G is not a wheel. Set

$$G_1 = G \text{ cl } 1.$$

G_1 is shown in Fig. 4.

Again by Theorem 1, G_1 is not a wheel since G_1 op 3 is a simple, triply connected graph. Set

$$G_2 = G_1 \text{ op } 3.$$

G_2 is shown in Fig. 5.

One can check and see that every edge of G_2 is essential and therefore G_2 is a wheel. In fact G_2 is W_4, the wheel of order 4.

The sequence

$$G, G_1, G_2$$

is a reduction sequence of G.

Reduction sequences of G are easily constructed as in Example 2. There are, in general, many reduction sequences of a simple, triply connected graph G. The following algorithm can be used to generate a reduction sequence for G. The validity of the algorithm is a simple consequence of Theorem 1.

Algorithm R: Generation of a Reduction Sequence

Let G be a simple, triply connected graph. Set $k = 0$, $G_0 = G$, and go to step 1.

Step 1 [Test for wheel]: If every edge of G_k is an essential edge, then the sequence

$$G_0, \cdots, G_k$$

is a reduction sequence of G. Otherwise go to step 2.

Step 2 [Remove a nonessential edge]: Pick some nonessential edge $e \in E(G_k)$. The edge e is nonessential either because G_k op e or G_k cl e is a simple, triply connected graph. If G_k op e is a simple, triply connected graph set

$$G_{k+1} = G_k \text{ op } e$$

and go to step 3. Otherwise, set

$$G_{k+1} = G_k \text{ cl } e$$

and go to step 3.

Step 3 [Index k]: Set

$$k \leftarrow k + 1$$

and go to step 1.

V. Testing the Planarity of Simple, Triply Connected Graphs

Let G be a graph. Informally, we say that G is a planar graph if it can be drawn on the sphere without crossed edges. Mac Lane [4] gives the following, more precise, definition.

> A *map* σ of a graph G is a correspondence which assigns to each vertex $v \in V(G)$ a point σv on the sphere and to each edge $e \in E(G)$ a Jordan arc $\sigma(e)$, on the sphere such that the ends of $\sigma(e)$ are the maps of the ends of e, while $\sigma v_1 \neq \sigma v_2$ if $v_1 \neq v_2$ and two arcs $\sigma(e_1)$ and $\sigma(e_2)$ with $e_1 \neq e_2$ do not intersect, except perhaps at their end points.

A graph G is called a *planar graph* if there exists a map σ of G on the sphere. The image $\sigma(G)$ of G on the sphere is called a *plane representation* of G.

Let G be a planar, simple, triply connected graph and $\sigma(G)$ a plane representation of G. The sphere is divided into connected regions by $\sigma(G)$ and these regions are bounded by the maps of certain of the members of $E(G) \cup V(G)$. The set of vertices that are on the boundary of a connected region is called a *mesh*. We denote by $\mathfrak{M}(G)$ the set of meshes of G.

Example 3

Let G be the graph in Fig. 1. This representation of G is a plane representation of G since G is finite and, consequently, the maps of G on the plane and on the sphere are equivalent. The set $\mathfrak{M}(G)$ is given by

$$\mathfrak{M}(G) = \{\{v_1, v_2, v_3\}, \{v_1, v_3, v_4\}, \{v_2, v_3, v_4\}, \{v_1, v_2, v_4\}\}.$$

It would seem that the set of meshes of G depends on the plane representation of G, but the fact is that if G is a planar, simple, triply connected graph, then $\mathfrak{M}(G)$ is independent of the plane representation chosen to calculate the set of meshes. This property of planar, simple, triply connected graphs follows from the work of Whitney [2], [3] and is also enunciated in a paper by Mac Lane [4].

Theorem 2

Let G be a planar, simple, triply connected graph. Then the set $\mathfrak{M}(G)$ is unique.

Theorem 2 is a key result that enables us to obtain an efficient algorithm for testing whether a simple, triply connected graph is planar. Before presenting the algorithm we will consider two operations on a graph that correspond to the inverses, in a certain sense, of G op e and G cl e, respectively.

Let G be a simple, triply connected graph.

1) Operation $(op)^{-1}$: Let e be some element satisfying $e \notin E(G)$ and v_1 and v_2 two vertices of G. The vertices v_1 and v_2 are to be the ends of e in the graph $G(\text{op})^{-1}e$, which is defined as

a) $E(G(\text{op})^{-1}e) = E(G) \cup \{e\}$,
b) $V(G(\text{op})^{-1}e) = V(G)$, and
c) the ends of the members of $E(G)$ in $G(\text{op})^{-1}e$ are the same as in G, while the ends of e in $G(\text{op})^{-1}e$ are v_1 and v_2.

The graph $G(\text{op})^{-1}e$ is defined only if $G(\text{op})^{-1}e$ is a simple, triply connected graph.

2) Operation $(cl)^{-1}$: Let e be some element satisfying $e \notin E(G)$ and v a vertex of G. Let the edges of $E(G)$ that have v as an end be partitioned into two sets L and R. Let v_L and v_R be two elements not in $V(G)$. The element e is to have v_L and v_R as its ends and is to be inserted into G by "splitting" the vertex v. This new graph is denoted by $G(\text{cl})^{-1}e$ and is defined as

a) $E(G(\text{cl})^{-1}e) = E(G) \cup \{e\}$,
b) $V(G(\text{cl})^{-1}e) = (V(G) - \{v\}) \cup \{v_L, v_R\}$, and
c) the ends of the members of $E(G)$ in $G(\text{cl})^{-1}e$ are the same as in G with the exception of the edges in L and R. The ends of e in $G(\text{cl})^{-1}e$ are v_L and v_R. Each edge in L is to have v_L as one of its ends and each edge in R is to have v_R as one of its ends. The other ends of the edges in $L \cup R$ in $G(\text{cl})^{-1}e$ are the same as in G.

The graph $G(\text{cl})^{-1}e$ is defined only if $G(\text{cl})^{-1}e$ is a simple, triply connected graph.

It is important to note that the notation $G(\text{op})^{-1}e$ and $G(\text{cl})^{-1}e$ does not give enough information to construct the corresponding graphs. Thus for $G(\text{op})^{-1}e$, it is necessary to know the ends of e in $V(G)$, and for $G(\text{cl})^{-1}e$, it is necessary to know the vertex v in $V(G)$ as well as the sets L and R. Accordingly, whenever we use this notation the appropriate information will always be supplied.

Example 4

Consider the graph G in Fig. 1. The operation $G(\text{op})^{-1}e$ is not defined for G since the addition of an edge between any pair of vertices of G causes G to become nonsimple.

Consider the graph G_1 in Fig. 1. The operation $G_1(\text{op})^{-1}e$ is defined, where the ends of e are v_1 and v_2. $G_1(\text{op})^{-1}e$ is precisely the graph G.

Next consider the graph G_2 in Fig. 1. The operation $G_2(\text{cl})^{-1}e$ is defined, where the vertex to be split is v, $L = \{e_2, e_3\}$, and $R = \{e_1, e_5\}$. $G_2(\text{cl})^{-1}e$ is precisely the graph G in Fig. 1.

We are now in a position to develop an algorithm for testing to see if G, a simple, triply connected graph, is planar. The first step is to apply Algorithm R to the graph G and thereby determine a reduction sequence

$$G_0, G_1, \cdots, G_r \tag{1}$$

of G. Corresponding to (1), there is a sequence

$$e_1, e_2, \cdots, e_r$$

of distinct edges in $E(G)$ such that either

$$G_{k+1} = G_k \text{ op } e_{k+1} \tag{2}$$

or

$$G_{k+1} = G_k \text{ cl } e_{k+1} \tag{3}$$

for $k = 1, \cdots, r - 1$.

Since G_r is a wheel, it should be clear that G_r is a planar graph and, consequently, there exists a map σ_r of G_r. The basic idea of the proposed method is to work backwards through the reduction sequence and at each stage determine whether G_p is a planar graph for $p = r - 1, \cdots, 0$.

Since we already know that G_r is planar and we have a map σ_r of G_r, it suffices to show how one determines whether or not G_p is a planar graph, given that G_{p+1} is a planar graph. If G_p is planar, then we also determine a map σ_p of G_p. There are two cases to consider.

A. Case 1: $G_{p+1} = G_p \text{ op } e_{p+1}$

Let v_1, v_2 designate the ends of e_{p+1} in G_p and let us consider the operation $G_{p+1}(\text{op})^{-1}e_{p+1}$ to be carried out with respect to v_1 and v_2. It is clear from the definition of a reduction sequence that $G_{p+1}(\text{op})^{-1}e_{p+1}$ is defined and in fact

$$G_p = G_{p+1}(\text{op})^{-1}e_{p+1}.$$

We must determine if $G_{p+1}(\text{op})^{-1}e_{p+1}$ is a planar graph. Suppose there exists a mesh $M \in \mathfrak{M}(G_{p+1})$ such that $v_1 \in M$ and $v_2 \in M$, then clearly the edge e_{p+1} can be added to G_{p+1} "in a planar fashion" and, accordingly, G_p is a planar graph. Thus the condition that v_1 and v_2 are in the same mesh M is sufficient to insure that G_p is a planar graph. The remarkable fact is that by Theorem 2 we can assert that this condition is also necessary.

Theorem 3

The graph G_p is planar if and only if there exists a mesh $M \in \mathfrak{M}(G_{p+1})$ satisfying $v_1 \in M$ and $v_2 \in M$, where v_1 and v_2 are the ends of e_{p+1} in G_p.

If G_p is planar, then the map σ_p of G_p is a straightforward extension of the map σ_{p+1}.

B. Case 2: $G_{p+1} = G_p \text{ cl } e_{p+1}$

Let v_L and v_R designate the ends of e_{p+1} in G_p and v designate the vertex in G_{p+1} that corresponds to the coalesced vertices v_L and v_R. The set L contains those edges in $(E(G_p) - \{e_{p+1}\})$ that have v_L as an end in G_p, and the set R contains those edges in $(E(G_p) - \{e_{p+1}\})$ which have v_R as an end in G_p. Let us consider the operation $G_{p+1}(\text{cl})^{-1}e_{p+1}$ to be carried out with respect to these quantities. It is clear from the definition of a reduction sequence that $G_{p+1}(\text{cl})^{-1}e_{p+1}$ is defined and, in fact,

$$G_p = G_{p+1}(\text{cl})^{-1}e_{p+1}.$$

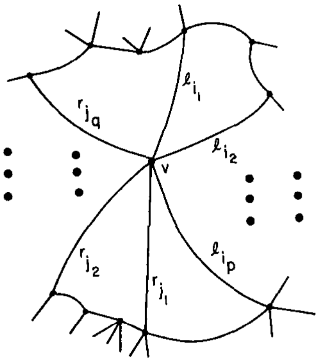

Fig. 6. Operation $(\text{cl})^{-1}$.

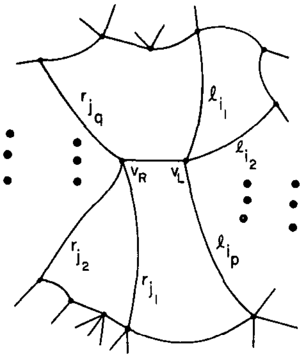

Fig. 7. Operation $(\text{cl})^{-1}$.

We must next determine if $G_{p+1}(\text{cl})^{-1}e_{p+1}$ is a planar graph. Since G_p is a simple, triply connected graph, $\alpha(L) \geq 2 \leq \alpha(R)$. Let

$$L = \{l_1, \cdots, l_p\}$$

and

$$R = \{r_1, \cdots, r_q\},$$

where $p \geq 2 \leq q$. Suppose that the map σ_{p+1} of G_{p+1} is such that the images of the edges in L are located consecutively around the point $\sigma_{p+1}v$; that is, there exists an element $l_{i_1} \in L$ such that if one moves in a clockwise direction around the point $\sigma_{p+1}v$ beginning with the image of l_{i_1}, then the images of all the edges of L are encountered before coming to the image of a member of R. This situation is depicted in Fig. 6. It should be clear that the edge e_{p+1} can be added to G_{p+1}, "in a planar fashion." Therefore G_p is a planar graph. The addition of e_{p+1} to G_{p+1} is shown in Fig. 7. Thus the condition that the images under σ_{p+1} of the members of L be located consecutively around the point $\sigma_{p+1}v$ is sufficient to insure that G_p is a planar graph. Again, the fact is that by Theorem 2 we can assert that the condition is also necessary.

Theorem 4

The graph G_p is planar if and only if the images under σ_{p+1} of the members of L are located consecutively around the point $\sigma_{p+1}v$.

If G_p is planar, then the map σ_p is easily obtained from the map σ_{p+1} by the construction shown in Figs. 6 and 7.

As we have seen in the previous considerations, the test for determining if G_p is a planar graph, given that G_{p+1} is a planar graph, is extremely simple and, clearly, the reason for this is that $\mathfrak{M}(G_{p+1})$ is unique. Loosely speaking, it means that no redrawing of G_{p+1} can alter the meshes and, accordingly, if the conditions of Theorem 3 [4] are not satisfied, the addition of the edge e_{p+1} must necessarily cause $G_{p+1}(\text{op})^{-1}e_{p+1}$ $[G_{p+1}(\text{cl})^{-1}e_{p+1}]$ to be a nonplanar graph. It follows easily that if G_p is nonplanar, then G must also be nonplanar.

We will now give an explicit statement of the algorithm which can be used to test to see if a given simple, triply connected graph G is a planar graph.

Algorithm P: Planarity Test for a Simple, Triply Connected Graph

Let G be a simple, triply connected graph.

Step 1 [*Obtain a reduction sequence of G*]: Apply Algorithm R to G and obtain a reduction sequence

$$G_0, \cdots, G_r \tag{4}$$

of G. Corresponding to (4), there is a sequence

$$e_1, \cdots, e_r$$

of distinct edges in $E(G)$ such that either

$$G_{k+1} = G_k \text{ op } e_{k+1} \tag{5}$$

or

$$G_{k+1} = G_k \text{ cl } e_{k+1} \tag{6}$$

for $k = 0, \cdots, r - 1$. Let $p = r - 1$ and go to step 2.

Step 2 [*Test for $G_{p+1} = G$*]: If $p = -1$, then $G = G_0$ is a *planar graph,* and σ_0 is a map of G. Otherwise go to step 3.

Step 3 [*Test G_p for planarity*]: There are two possible cases.

Case 1: $G_{p+1} = G_p$ op e_{p+1}. This case is described above under Section V-A and consequently, G_p is planar if and only if the conditions of Theorem 3 are satisfied.

If G_p is planar, then σ_p, the map of G_p, is obtained from σ_{p+1} by the addition of a Jordan arc, corresponding to e_{p+1}, with ends $\sigma_{p+1}\, v_1$ and $\sigma_{p+1}\, v_2$. Go to step 4.

If the conditions of Theorem 3 are not satisfied, then *G is nonplanar.*

Case 2: $G_{p+1} = G_p$ cl e_{p+1}. This case is described above under Section V-B and consequently, G_p is planar if and only if the conditions of Theorem 4 are satisfied.

If G_p is planar, then σ_p, the map of G_p, is obtained from σ_{p+1} by the construction shown in Figs. 6 and 7. Go to step 4.

If the conditions of Theorem 4 are not satisfied, then *G is nonplanar.*

Step 4 [*Index p*]: Set $p \leftarrow p - 1$ and go to step 2.

In Fig. 8 we present a flowchart of Algorithm P. In the following sections we shall discuss how one makes use of Algorithm P in testing any graph for planarity.

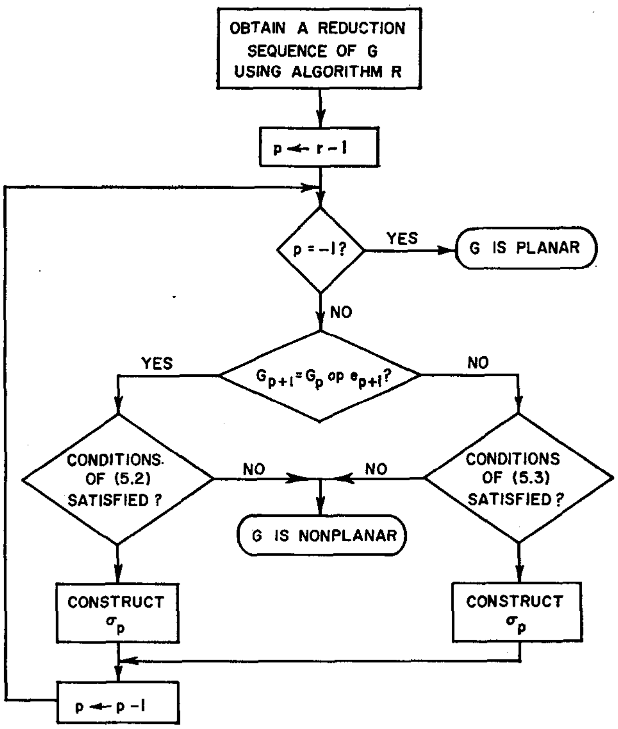
OBTAIN A REDUCTION SEQUENCE OF G USING ALGORITHM R
p ← r - 1
p = -1?
YES
G IS PLANAR
NO
YES
$G_{p+1} = G_p$ op e_{p+1}?
NO
CONDITIONS OF (5.2) SATISFIED ?
NO
NO
CONDITIONS OF (5.3) SATISFIED ?
G IS NONPLANAR
CONSTRUCT σ_p
CONSTRUCT σ_p
p ← p - 1

Fig. 8. Flowchart of Algorithm P.

Specifically, we will use a structural characterization of planar graphs, developed by Mac Lane [4], to show how one can reduce the problem of testing a graph for planarity to the problem of testing a set of simple, triply connected graphs for planarity. Weinberg [5] has previously discussed this possibility.

We will also discuss Algorithm R in more detail. To obtain a reduction sequence it is clearly necessary to have some efficient way to test to see if a graph is triply connected. For this part of our work we use some recent results of Frisch [6] and Steiglitz *et al.* [7].

Lastly we will describe a computer program based on the algorithms and an example using this program.

VI. Testing the Planarity of Any Graph

In the preceding section we developed an algorithm that can be used to determine whether or not a simple, triply connected graph is a planar graph. Suppose, however, that G is an arbitrary graph and the problem is to determine if G is a planar graph. In this section we will show how Algorithm P forms the nucleus of a general algorithm for determining if an arbitrary graph G is a planar graph. There are some obvious preliminary steps one can take before getting into a general algorithm. These steps are designed to reduce the overall work in testing the planarity of G. We give two preliminary steps.

1) Let $v_1, \cdots, v_k$ be a path in G, then an edge e is said to be *on P* if the ends of e are some pair of consecutive members of P.

A path is called *simple* if none of its members are repeated and it is called *closed* if its first and last members are identical.

A simple, closed path is called a *polygon*.

We define the equivalence relation $\sim$ on the edges of a graph G as follows.

a) For all $e \in E(G)$, $e \sim e$.
b) $e \sim e'$ if there is a polygon B of G such that e and e' are both on B.

It is easy to verify that $\sim$ is an equivalence relation and, consequently, it induces a partition Π on the edges $E(G)$. Each block in this partition defines a subgraph of G, which consists of the edges in the block and all those vertices that are the ends of members in the block. Let the set

$$\mathcal{G} = \{G_1, G_2, \cdots, G_m\}$$

consist of the subgraphs of G induced by the partition Π. The members of $\mathcal{G}$ are called the *nonseparable* components of G. Whitney [2] has shown that G is planar if and only if each of its nonseparable components are planar. Accordingly, we could restrict our attention to graphs G that are nonseparable, since there are efficient algorithms that determine the nonseparable components of an arbitrary graph.

2) We say that two edges e_i, $e_j \in E(G)$ are *parallel edges* if they have the same ends.

Two edges e_k, $e_l \in E(G)$ are called *series edges* if there is a vertex $v \in V(G)$ that is an end of both e_l and e_k and the valence of v is 2.

Series and parallel edges are unimportant since they do not affect, in a certain sense, the planarity of a given graph.

Theorem 5

Let e_i and e_j be parallel edges of G. Then G op e_i and G op e_j are both planar graphs if and only if G is a planar graph.

Theorem 6

Let e_k and e_l be series edges of G. The G cl e_k and G cl e_l are both planar graphs if and only if G is a planar graph.

In this preliminary step we "remove" all series and parallel edges from G. Suppose the edge $e \in E(G)$ is a parallel edge, that is, there exists another edge e' such that e and e' have the same ends. To remove the parallel edge we form G op e and set

$$G \leftarrow G \text{ op } e.$$

We continue to look for parallel edges in G and remove them in the same manner as e was removed. If we can no longer find any parallel edges we then look for series edges. Suppose the edge e'' is a series edge of G, that is, there exists an edge e''' such that e'' and e''' are series edges. To remove the series edge e'' from G we form G cl e'' and set

$$G \leftarrow G \text{ cl } e''.$$

We continue to look for series edges in G and remove them in the same manner as e'' was removed. When we can no longer find any series edges, the process of removing series and parallel edges is *not* finished. We must go back and look for parallel edges once more, since the removal of series edges can cause some edges to become parallel edges. Also, when we can no longer find any parallel edges we must again look for series edges since the removal of parallel edges can cause some edges to become series edges. In this manner we continue to remove series and parallel edges until one can no longer find a series or a parallel edge of G.

We will state the process of removing series and parallel edges from G as an algorithm since we will have occasion to use it again.

Algorithm S–P: Series–Parallel Reduction

Step 1 [*Locate a series or a parallel edge*]: If G has no series or parallel edges, terminate the algorithm. If G has a series edge go to step 2, otherwise go to step 3.

Step 2 [*Remove a series edge from G*]: Pick some series edge $e \in E(G)$, set

$$G \leftarrow G \text{ cl } e$$

and go to step 1.

Step 3 [*Remove a parallel edge from G*]: Pick some parallel edge $e \in E(G)$, set

$$G \leftarrow G \text{ op } e$$

and go to step 1.

The two preliminary steps given above form the initialization stage of our general algorithm. We now state this initialization stage explicitly.

Initialization of Algorithm Planar

Let G be an arbitrary graph.

Step 1 [*Find nonseparable components*]: Determine the set

$$\mathcal{G} = \{G_1, G_2, \cdots, G_m\},$$

where G_i corresponds to the unique nonseparable components of G.

Step 2 [*Apply Algorithm S–P*]: Let H_i be the graph obtained by applying Algorithm S–P to G_i for $i = 1, \cdots, m$. Set

$$\mathcal{H} = \{H_1, \cdots, H_m\}.$$

We have the following result, which is a consequence of the preceding remarks.

Theorem 7

Let G be an arbitrary graph. Then G is a planar graph if and only if each of the members of $\mathcal{H}$, obtained by applying the initialization of Algorithm Planar, is a planar graph.

The initialization part of Algorithm Planar results in a set of graphs that can be individually tested for planarity and by Theorem 6; these tests are sufficient to determine whether G is a planar graph. Let us first examine the kinds of graphs that can belong to $\mathcal{H}$. It should be clear that if G has any isolated vertices that they do not affect the planarity of G nor are they in any member of $\mathcal{H}$. Moreover, no member of $\mathcal{H}$ consists of a single vertex. Also the loops of G become singletons in the partition Π and they survive Algorithm S–P to become members of $\mathcal{H}$ that are graphs consisting of one edge and one vertex. Conversely, any member of $\mathcal{H}$ that is a graph consisting of one edge and one vertex corresponds to a unique loop of G. Clearly, the *loop graphs* (a loop graph is a graph consisting of one edge and one vertex) in $\mathcal{H}$ are planar and, consequently, need not be examined further.

Some of the members of $\mathcal{H}$ may consist of a single edge with two distinct ends. We call such a graph a *link graph*. A nonseparable component of G that is a link graph is not affected by Algorithm S–P and so becomes a link graph in the set $\mathcal{H}$. However, the converse is not necessarily true. The members of $\mathcal{H}$ that are link graphs do not necessarily correspond to nonseparable components of G that are also link graphs. Each member of $\mathcal{H}$ does correspond to a unique nonseparable component of G (i.e., a member of $\mathcal{G}$) and it is easy to construct nonseparable graphs that become link graphs when Algorithm S–P is applied to them. Such a nonseparable graph is planar by Theorem 7.

Example 5

Consider the graph G shown in Fig. 9. G is a nonseparable graph and application of Algorithm S–P to G results in a link graph. Therefore, G is a planar graph. To obtain a map of G it is necessary to save enough information so that one may retrace the steps of Algorithm S–P.

Link graphs and loop graphs consist of single edges and it might well be asked if there are graphs in $\mathcal{H}$ consisting of two edges, three edges, etc. It is not too difficult to show that every member of $\mathcal{H}$ is a nonseparable graph and it follows from this that, besides the link graphs and loop graphs, every graph in $\mathcal{H}$ must contain at least six edges. To see this one can merely enumerate all graphs on five edges or less and check that, except for the link graph and the loop graph, none of these can be a member of $\mathcal{H}$ on the grounds that either the graph is separable or the graph contains a series or a parallel edge. The wheel of order 3 is a nonseparable graph on six edges that can qualify as a member of $\mathcal{H}$. It is also not difficult to see that the wheel of order 3 is the only graph on six edges that can qualify as a member of $\mathcal{H}$. Therefore we have the following result.

Theorem 8

Let H_i be a member of $\mathcal{H}$ and suppose that H_i is not a link graph or a loop graph. Then H_i is a graph on at least six edges. Moreover, if H_i is on exactly six edges, then H_i is the wheel of order 3.

The main thrust of Algorithm Planar will be to determine if the members of $\mathcal{H}$, which are on at least six edges, are planar. Let H be a member of $\mathcal{H}$ satisfying $\alpha(E(H)) \geq 6$. If H is not 2-separable, then H is a simple, triply connected graph and Algorithm P can be applied to H to determine whether H is a planar graph. Therefore let us assume that H is 2-separable and that $\{J, K\}$ is a 2-separator of H. We now describe a process of "splitting" H into two smaller graphs such that H is planar if and only if both of the two new graphs are planar.

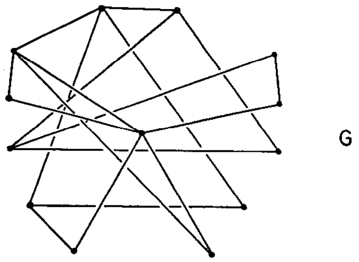

Fig. 9. Graph G of Example 5.

Since H is nonseparable it follows that

$$\alpha(V(J) \cap V(K)) = 2.$$

Let

$$V(J) \cap V(K) = \{u, v\}.$$

We form two graphs J^+ and K^+ by adding a new edge between the vertices u and v in J and K, respectively. Let e be some element not belonging to $E(J) \cup E(K)$. Then J^+ is defined as

1) $V(J^+) = V(J)$,
2) $E(J^+) = E(J) \cup \{e\}$, and
3) the ends of the members of $E(J)$ in J^+ are the same as in J and v and u are the ends of e in J^+.

The element e is added to K in the same manner to form K^+. It has been shown by Mac Lane [4] that the graph H is planar if and only if J^+ and K^+ are planar graphs. This splitting process is a basic step in Algorithm Planar.

Theorem 9

Let H be a nonseparable graph. Suppose H is split into two graphs J^+ and K^+ as described above. Then H is planar if and only if J^+ and K^+ are planar graphs.

Since H is nonseparable, both J^+ and K^+ are nonseparable graphs. However, it is possible that J^+ and K^+ contain series or parallel edges as a result of the splitting process. Therefore, the next step is the application of Algorithm S–P to J^+ and K^+. The resultant graphs are nonseparable and without series or parallel edges. It should be clear now that the splitting process can again be applied to these new graphs in order to reduce the size of our test still further.

The process of alternately splitting graphs and applying Algorithm S–P beginning with the graph H results, in general, in a set of graphs that are either link graphs or simple, triply connected graphs. The graph H is a planar graph if and only if the resulting simple, triply

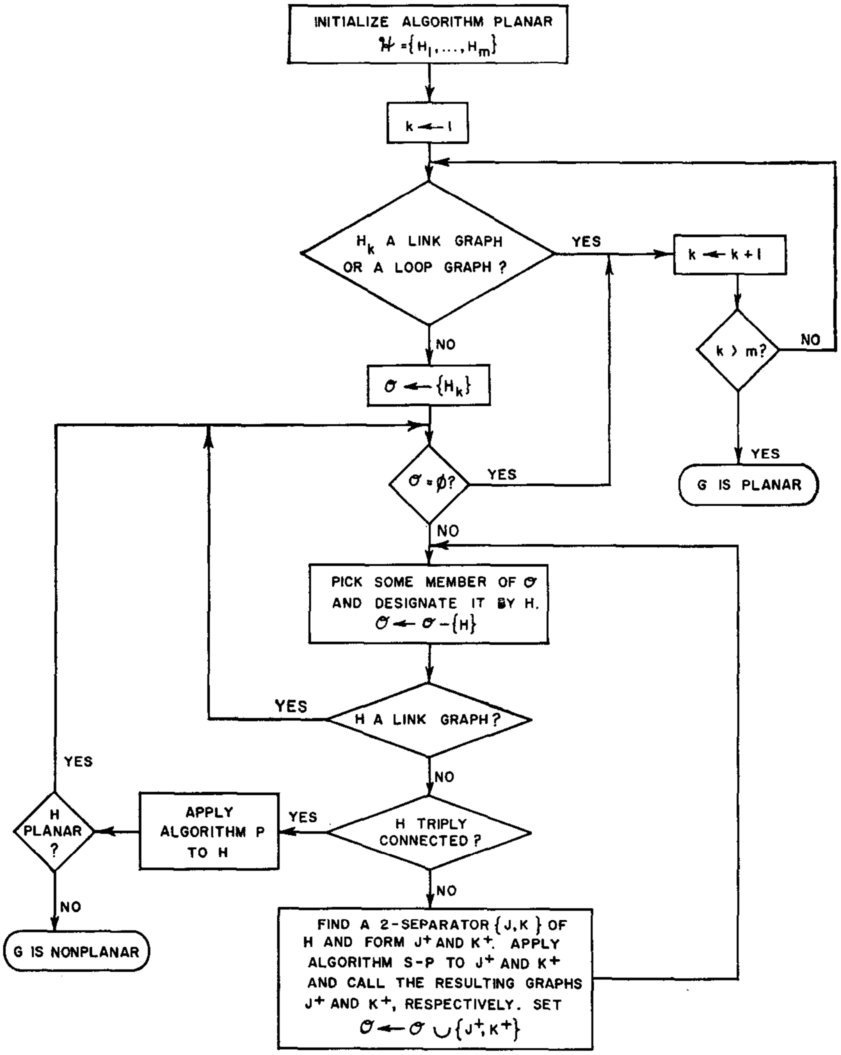
INITIALIZE ALGORITHM PLANAR
$\mathcal{H}=\{H_1,\ldots,H_m\}$
$k \leftarrow 1$
H_k A LINK GRAPH OR A LOOP GRAPH ?
YES
$k \leftarrow k+1$
$k > m?$
NO
YES
G IS PLANAR
NO
$\mathcal{O} \leftarrow \{H_k\}$
$\mathcal{O}=\phi?$
YES
NO
PICK SOME MEMBER OF $\mathcal{O}$ AND DESIGNATE IT BY H. $\mathcal{O} \leftarrow \mathcal{O}-\{H\}$
YES
H A LINK GRAPH?
NO
H TRIPLY CONNECTED ?
YES
APPLY ALGORITHM P TO H
H PLANAR ?
YES
NO
G IS NONPLANAR
NO
FIND A 2-SEPARATOR $\{J,K\}$ OF H AND FORM J^+ AND K^+. APPLY ALGORITHM S-P TO J^+ AND K^+ AND CALL THE RESULTING GRAPHS J^+ AND K^+, RESPECTIVELY. SET $\mathcal{O} \leftarrow \mathcal{O} \cup \{J^+, K^+\}$

Fig. 10. Flowchart of Algorithm Planar.

connected graphs are planar. Thus we have the essential steps in Algorithm Planar. The idea is simply to keep splitting the graph H and applying Algorithm S-P in order to reduce the problem of determining whether or not H is planar to the problem of determining whether or not a derived set of simple, triply connected graphs is planar. Mac Lane [4] has described the above splitting process in some detail and has employed this technique in counting the number of maps of a nonseparable graph. We conclude this section with a formal presentation of the Algorithm Planar. (See Fig. 10.)

Algorithm Planar: Planarity Test for any Graph G

Let G be an arbitrary graph. Apply the initialization of Algorithm Planar to graph G obtaining the set

$$\mathcal{H} = \{H_1, \cdots, H_m\}.$$

Set $k = 1$ and go to step 1.

Step 1 [*Check for a link graph or a loop graph*]: If H_k is a link graph or a loop graph set $k \leftarrow k + 1$ and go to step 5. Otherwise set $\mathcal{O} \leftarrow \{H_k\}$ and go to step 2.

Step 2 [*Pick a new graph from* $\mathcal{O}$]: If $\mathcal{O} = \phi$ set $k \leftarrow k + 1$ and go to step 5. Otherwise, pick some member of $\mathcal{O}$ and designate this member by H. Set $\mathcal{O} \leftarrow \mathcal{O} - \{H\}$. If H is a link graph, go to step 2. Otherwise go to step 3.

Step 3 [*Apply Algorithm Planar*]: If H is not triply connected go to step 4. Otherwise apply Algorithm P and if H is planar go to step 2. If H is nonplanar, then G is nonplanar and therefore, terminate the algorithm.

Step 4 [*Find a 2 separator*]: Let $\{J, K\}$ be a 2 separator for H and form J^+ and K^+ as described above. Set

$$\mathcal{O} \leftarrow \mathcal{O} \cup \{J^+, K^+\}$$

and go to step 2.

Step 5 [*Terminate algorithm if* $k > m$]: If $k > m$, G is a planar graph and therefore terminate the algorithm. Otherwise go to step 1.

VII. Description of a Computer Program

Algorithm Planar was programmed in Fortran IV and run on the IBM 360/65 computer. There is no room here to present the complete program, and we shall discuss only the important features of the implementation.

Most crucial is the algorithm for finding the connectivity

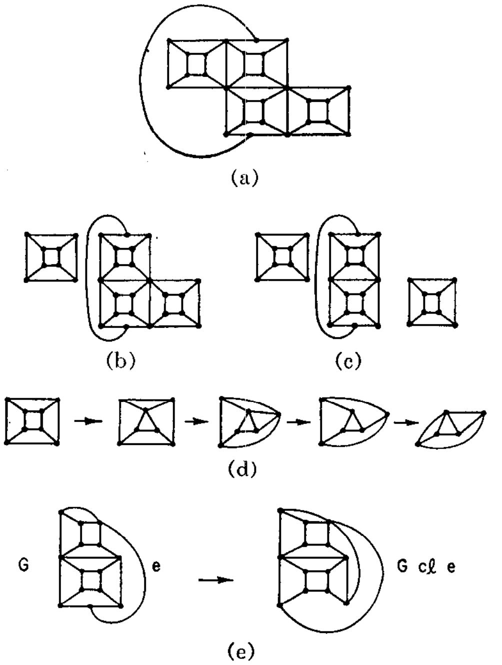

Fig. 11. A 28-vertex planar example. (a) Original graph. (b) Result of first split. (c) Result of second split. (d) A reduction sequence for each of the 8-vertex graphs in (c). (e) A typical step in the reduction sequence for the 16-vertex graph in (c), showing the operation *G* cl *e*.

between a given pair of vertices; that is, the number of node disjoint paths that exist between them. This is used to find a reduction sequence in Algorithm *R*, to test triple connectivity in step 3, and to find the 2 separator in step 4. The labeling algorithm of Frisch [6] was used for this purpose, together with the economies described in [7]. A complete test for triple connectivity requires only $3n$-6 applications of the labeling algorithm, where n is the number of vertices, and is used after each (cl) operation in obtaining a reduction sequence in Algorithm *R*. To test triple connectivity after each (op) operation, it is only necessary to apply the labeling algorithm once, between the two vertices that were previously adjacent before the (op) operation. The labeling algorithm is also used to identify 2 separators in the splitting process. The fact that only $3n$-6 connectivity tests are required to determine triple connectivity, as opposed to the naive application of $\binom{n}{2}$ tests, greatly speeds up Algorithm *R*.

In order to perform the $(\mathrm{cl})^{-1}$ operation, it is necessary to know which edges were connected to vertex v_L and which to v_R. Saving such information during the reduction process would entail too much storage space. Instead, the following method was used. A list that recorded the sequence of (cl) and (op) operations used during the reduction process was kept. When a $(\mathrm{cl})^{-1}$ is to be performed, the preceding graph in the reduction sequence is reconstructed by performing all the (cl) and (op) operations leading to the desired graph. This takes much less time than the original reduction process, since no connectivity tests are required. This device is called a "quick collapse," and trades a relatively small amount of computer time for a large amount of storage.

Fig. 11 shows a 28-node planar graph that was run as an example. Execution time was about one-half minute on the IBM 360/65 computer, including time for printing intermediate steps. Note that as a result of the splitting process, the testing of a large graph is quite naturally broken down into a series of tests on a set of much smaller graphs, these graphs being a set of simple, triply connected graphs.

References

[1] W. T. Tutte, "A theory of 3-connected graphs," *Indag. Math.*, vol. 23, pp. 441–455, 1961.

[2] H. Whitney, "Non-separable and planar graphs," *Trans. Am. Math. Soc.*, vol. 34, pp. 339–362, 1932.

[3] ——, "2-isomorphic graphs," *Am. J. Math.*, vol. 55, pp. 245–254, 1933.

[4] S. Mac Lane, "A structural characterization of planar combinatorial graphs," *Duke Math. J.*, vol. 3, pp. 460–472, September 1937.

[5] L. Weinberg, "Two new characterizations of planar graphs," presented at the 5th Ann. Allerton Conf. on Circuit and Systems Theory (University of Illinois, Urbana), October 4–6, 1967.

[6] I. T. Frisch, "An algorithm for vertex pair connectivity," *Internatl. J. Control*, vol. 6, pp. 579–593, 1967.

[7] K. Steiglitz, P. Weiner, and D. J. Kleitman, "The design of minimum-cost survivable networks," presented at 1968 Internatl. Symp. on Circuit Theory (Miami, Fla.); also *IEEE Trans. Circuit Theory*, vol. CT-16, pp. 455–460, November 1969.

[8] A. J. Goldstein, "An efficient and constructive algorithm for testing whether a graph can be embedded in the plane," Bell Telephone Labs., Murray Hill, N. J., 1963 (unpublished).

[9] G. J. Fisher and O. Wing, "Computer recognition and extraction of planar graphs from an incidence matrix," *IEEE Trans. Circuit Theory*, vol. CT-13, pp. 154–163, June 1966.

[10] L. Auslander and L. V. Parter, "On embedding graphs in the sphere," *J. Math. Mech.*, vol. 10, pp. 517–523, May 1961.

[11] A. Lempel, S. Even, and I. Cederbaum, "An algorithm for planarity testing of graphs," in *Theory of Graphs: International Symposium*, (Rome, July, 1966). New York: Gordon and Breach, 1967, pp. 215–232.