1967

Linear graphs, edge sets, and Boolean functions

William Lee Reuter

Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Electrical and Electronics Commons

Recommended Citation
Reuter, William Lee, "Linear graphs, edge sets, and Boolean functions " (1967). Retrospective Theses and Dissertations. 3209.
https://lib.dr.iastate.edu/rtd/3209

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
LINEAR GRAPHS, EDGE SETS, AND BOOLEAN FUNCTIONS

by

William Lee Reuter

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of
The Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major Subject: Electrical Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

Head of Major Department

Signature was redacted for privacy.

Dean of Graduate College

Iowa State University
Ames, Iowa

1967
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II. FUNDAMENTALS</td>
<td>4</td>
</tr>
<tr>
<td>A. Linear Graphs</td>
<td>4</td>
</tr>
<tr>
<td>B. Edge Removal</td>
<td>7</td>
</tr>
<tr>
<td>C. Generated Graphs</td>
<td>9</td>
</tr>
<tr>
<td>D. Notation</td>
<td>10</td>
</tr>
<tr>
<td>E. Connectivity</td>
<td>12</td>
</tr>
<tr>
<td>F. Degrees of Independence</td>
<td>13</td>
</tr>
<tr>
<td>G. Extrema</td>
<td>15</td>
</tr>
<tr>
<td>H. Edge Sets</td>
<td>17</td>
</tr>
<tr>
<td>I. Equivalent Set Definitions</td>
<td>20</td>
</tr>
<tr>
<td>J. Edge Classifications</td>
<td>26</td>
</tr>
<tr>
<td>K. Graph Classifications</td>
<td>29</td>
</tr>
<tr>
<td>L. Interrelationships</td>
<td>31</td>
</tr>
<tr>
<td>M. Boolean Functions</td>
<td>36</td>
</tr>
<tr>
<td>III. HIERARCHY</td>
<td>46</td>
</tr>
<tr>
<td>A. K-sets</td>
<td>46</td>
</tr>
<tr>
<td>B. Equivalent K-set Definitions</td>
<td>55</td>
</tr>
<tr>
<td>C. Interrelationships</td>
<td>66</td>
</tr>
<tr>
<td>D. Boolean Functions</td>
<td>76</td>
</tr>
<tr>
<td>IV. RELATED ASPECTS</td>
<td>84</td>
</tr>
<tr>
<td>A. Abelian Groups</td>
<td>84</td>
</tr>
<tr>
<td>B. Additional Constraints</td>
<td>87</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

Linear graph theory is applicable to a broad spectrum of problems. Consequently numerous individuals from many professions have occasion to utilize linear graphs in a profusion of widely different ways. An inevitable result is that a certain amount of inconsistency and a high amount of redundancy occur in regard to the terminology, definitions, symbolism, and theorems used in the seemingly myriad of publications that evolve. Even if the scope of publications is narrowed to those concerned with applying linear graph theory to electrical engineering problems, the inconsistency and redundancy can still be troublesome, especially to the tyro.

If the major inconsistencies and the excess redundancies are eliminated, yet other difficulties exist. For example linear graph theory contains a multitude of terms, definitions, and theorems. Furthermore the application of linear graph theory to specific problems often results in the generation of special symbolism and the employment of a broad collection of mathematical operations. Many times the symbolism and the mathematics are so specialized that it is difficult to apply them to other problems.

It is obvious that there is a need for standardization. It is also obvious that versatile and simple standardization which covers even a reasonable portion of linear graph theory is difficult to generate, let alone agree on. Such standardization is a slowly evolving process. Considerable effort must first be expended both by the individuals who develop the theory and by the individuals who apply the theory.
This presentation is written in an effort to explore in depth one major facet of linear graph theory, namely edge sets. It is anticipated that this material will contribute to the standardization process. It is also anticipated that this material will serve as an introduction and as a reference to the various edge sets of a linear graph and to the various interrelationships that exist between these edge sets. Of course the approach is slanted, because of the author's interest, towards electrical engineering. Hence the predominate coverage concerns those edge sets that are applicable to electrical engineering problems.

Every effort is made to present the material in a logical and consistent manner. A minimum amount of terminology is used, and the major definitions are concise and worded so as to emphasize both the similarities and the differences. In a number of instances, other commonly used terms and definitions are included to stress the fact that there are numerous alternatives and viewpoints and to serve as a bridge to some of the current literature. Figures are used to illustrate the many interrelationships that exist between groups of edge sets and to illustrate the numerous ways in which one group of edge sets can be generated from another group. For the most part the symbolism and the mathematical operations are restricted entirely to those used in Boolean algebra. When possible, Boolean functions are employed to represent edge-set interrelationships.

A conscientious effort is made to reference all significant material that is expanded upon in readily available publications. However from an investigation of the literature, it becomes apparent that a detailed
bibliography which would credit all previous publications having some connection to any of the facets included herein might possibly contain as many pages as there are in this entire presentation. This conclusion is based partially on the fact that one recent bibliography concerning graph theory publications contains 1617 entries (3). Consequently it is difficult and most presumptuous to state precisely what portion of the following material is indeed unique or distinct from what is already available. However to the best of the author's knowledge, the resulting Boolean functions and associated viewpoint are a new contribution to the state of the art. In fact the Boolean functions were the initial impetus for this presentation, and all of the material is developed towards verifying the Boolean functions that describe edge-set interrelationships.
This chapter provides the fundamental terminology and the associated definitions which serve as the foundation for all concepts developed in this presentation. Linear graphs are defined in general, and four special edge sets are defined and then investigated in detail. Also an edge-set notation is presented and used to obtain and describe generated graphs.

Necessary definitions are given in an order of logical development accompanied by explanatory material and a continuing example. The four special edge sets are investigated in depth by listing alternative definitions which illustrate how these four sets are interrelated. Finally a symbolic formulation of the interrelationships is presented by means of Boolean functions.

A. Linear Graphs

Because of the number of necessary definitions, all terminology in this presentation is defined as it occurs in the order of development. In an effort to facilitate referring back to major definitions, all such definitions are closely preceded by major subheadings or by minor subdivisions as in the immediately following manner.

1. Abstract

Linear graphs, or simply graphs, are defined abstractly by Busacker and Saaty (2) and by Tutte (15) essentially as follows:

Definition 1. An abstract graph consists of:

(a) A set of elements, \( V \).
(b) A second set of elements, \( E \).

(c) A relation of incidence, \( \mathfrak{r} \), which associates each element of \( E \) with two elements of \( V \).

The abstract graph is denoted by \( G \) or by \( (V,E) \) or \( (V,E,\mathfrak{r}) \). The elements of \( V \) are called vertices, and the elements of \( E \) are called edges. These two sets of elements are considered as disjoint sets in the material to follow. Consequently the vertices are denoted by integers, and the edges are denoted by lower-case letters from the beginning of the alphabet. An example of this notation is given in Table 1 which is used to illustrate the incidence relation for a particular graph composed of the two sets

\[
V = \{1,2,3,4,5\} \quad (1)
\]

and

\[
E = \{a,b,c,d,e,f,g,h\} \quad (2)
\]

It should be noticed that the two vertices associated with each edge are not necessarily distinct, as is the case with edge \( h \).

Table 1. Abstract graph

<table>
<thead>
<tr>
<th>Edges</th>
<th>Corresponding Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1, 4</td>
</tr>
<tr>
<td>b</td>
<td>1, 2</td>
</tr>
<tr>
<td>c</td>
<td>2, 3</td>
</tr>
</tbody>
</table>
Table 1. (Continued)

<table>
<thead>
<tr>
<th>Edges</th>
<th>Corresponding Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>d</td>
<td>3,4</td>
</tr>
<tr>
<td>e</td>
<td>3,4</td>
</tr>
<tr>
<td>f</td>
<td>2,4</td>
</tr>
<tr>
<td>g</td>
<td>4,5</td>
</tr>
<tr>
<td>h</td>
<td>2,2</td>
</tr>
</tbody>
</table>

The number of edges in V and E are denoted by $n_v$ and $n_e$ respectively. In this presentation both $n_v$ and $n_e$ are always finite which leads to the following:

Definition 2. A graph is a finite graph if and only if both $n_v$ and $n_e$ are finite.

2. Geometric

While the abstract graph is mathematically sufficient, it is not conceptually satisfying. Hence we define a geometric graph in a manner similar to Busacker and Saaty (2) as follows:

Definition 3. A geometric graph is a set of points, $V$, in $n$-dimensional Euclidean space and a set of simple curves, $E$, such that:

(a) The end points and only the end points of each curve coincide with points of $V$. 
(b) The curves have no common points, except for points of \( V \).

Since every finite (abstract) graph has a geometric realization in 3-dimensional Euclidean space (2), we use the geometric graph in all examples for conceptual purposes. Thus we employ Figure 1 to convey the information contained in Table 1.

B. Edge Removal

A set of \( n \) elements can be used to form \( 2^n \) distinct subsets, ranging from the null set, \( \emptyset \), to the entire set. For example the eight edges in Equation 2 provide \( 2^8 \), or 256, distinct edge sets, some of which are illustrated in Table 2.

Table 2. Edge sets from Equation 2

<table>
<thead>
<tr>
<th>( \emptyset )</th>
<th>( a )</th>
<th>( a,b )</th>
<th>( a,b,c )</th>
<th>( \ldots )</th>
<th>( a,b,c,d,e,f,g )</th>
<th>( a,b,c,d,e,f,g,h )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b )</td>
<td>( a,c )</td>
<td>( a,b,d )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( c )</td>
<td>( a,d )</td>
<td>( a,b,e )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td>( \cdot )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

By choosing some distinguishing criterion, these edge sets can be classified into two groups, namely those sets which meet the criterion and those which do not. Of course we are assuming that some meaningful criterion either exists or can be formulated.

Numerous edge-set criteria are readily available in the literature
Figure 1. Geometric realization of the abstract graph given in Table 1
on linear graph theory. These criteria either are or can be specified in terms of two edge operations.

1. **Opening**
   
   Both edge operations involve edge removal. One operation does not affect the vertices and is defined as follows:
   
   **Definition 4.** Opening, or removal by opening, means to simply delete an edge from the reference graph (6).

2. **Shorting**
   
   The other edge operation can affect the vertices and is defined in the following manner:
   
   **Definition 5.** Shorting, or removal by shorting, means to delete an edge from the reference graph and to coalesce, or identify, the associated vertices if they are distinct (5, 6).

C. **Generated Graphs**

When the previously defined edge removal operations are used in part to classify edge sets, a given edge set can be conveniently tested in order to ascertain whether or not it meets the criterion. This is done by operating on the given graph, which is hereafter referred to as the reference graph, to produce another graph that is defined as follows:

**Definition 6.** A generated graph is the graph that results from either opening or shorting those edges of a reference graph that belong to a given edge set.
Perhaps this definition seems so obvious as to be unnecessary, but we point out that the term *generated graph* is used to imply two things. First there is an associated reference graph. Second there is an associated edge set that was either opened or shorted.

The informed reader may wonder why subgraph, partial graph, and partial subgraph were deliberately avoided in Definition 6. Admittedly these terms are commonly used but are too restrictive for the material that follows. For example the partial graph as defined by Berge (1) is a generated graph obtained only by opening edges. Furthermore a subgraph as defined by Seshu and Reed (14) is a generated graph obtained only by opening edges and by deleting any resulting isolated vertices.

The geometric realizations of six generated graphs are shown in Figure 2. Three of these generated graphs evolve from opening three different edge sets of the reference graph shown in Figure 1. The other three generated graphs evolve from shorting three different edge sets of the same reference graph. The notation used in Figure 2 to describe the six generated graphs is explained in the following section.

D. Notation

In order to symbolically distinguish between the two methods of edge removal and in order to conveniently label generated graphs, we utilize the notation of Boolean algebra. For example to symbolically represent the generated graph shown in Figure 2(a), which resulted from opening edges d, e, f, and h of the reference graph in Figure 1, we write $d \cdot e \cdot f \cdot h$. The presence of the bar signifies opening, and the dot (•) signifies the word and. Hence the notation is read as open edge d and open edge e and open
Figure 2. Six generated graphs obtained from the reference graph in Figure 1.
edge f and open edge h or, more simply, open d, e, f, and h.

As another example we symbolically represent the generated graph shown in Figure 2(b), which resulted from shorting edges a, b, c, and g, by writing a\cdot b\cdot c\cdot g. The absence of the bar signifies shorting, and the dot (•) signifies the word and as before. Again this symbolic portrayal can be worded as a rather long and repetitive statement. It should be noted in this second example that only one vertex integer was retained.

As a matter of consistency, whenever two distinct vertices are coalesced, the smaller integer is used to represent the combined vertex.

Another example is shown in Figure 2(c). This generated graph has the distinguishing characteristic of being disconnected into two parts. A rigorous definition of these terms is presented in the next section, and the other three examples are referred to in later sections.

E. Connectivity

It is of interest to be able to designate when a table or a figure representing a graph cannot be partitioned into two or more disjoint tables or figures. To this end we define a part, or a component, as follows:

Definition 7. A graph for which V cannot be partitioned into two nonempty subsets \( V_1 \) and \( V_2 \) in such a way that both vertices associated with every edge are in the same subset is called a part.

The reference graph in Figure 1 and the generated graphs in Figure 2(a), (b), (d), and (f) each have one part. The generated graphs in
Figure 2(c) and (e) each have two parts. This characteristic of a graph is used in definitions to follow, and the number of parts is symbolized by $n_p$, which of course is always equal to or less than $n_v$. A special case is when $n_p$ is unity which leads to the following related definition:

**Definition 8.** A graph is said to be connected if and only if it is composed of precisely one part.

Hereafter all reference graphs are connected, which is of no major consequence but is indeed quite convenient. Furthermore any generated graphs that are obtained by shorting edges are necessarily also connected. Only edge removal by opening can produce generated graphs that are disconnected.

**F. Degrees of Independence**

When analyzing an electrical network, it is usually beneficial to determine the number of independent voltages and the number of independent currents. If the network is represented by a graph, these two numbers are well defined terms that are associated directly with the graph.

1. **Rank**

The number of independent voltages is identical to the rank of the graph, which is defined as follows:

**Definition 9.** The rank of a graph, $n_r$, is defined as

$$n_r = n_v - n_p$$

\[ (3) \]
2. **Nullity**

The number of independent currents is identical to the nullity of the graph, which is defined as follows:

**Definition 10.** The nullity of a graph, $n_n$, is defined as

\[
n_n = n_e - n_v + n_p
\]

\[= n_e - n_r\] 

(4)

One important result of these definitions is that

\[
n_r + n_n = n_e\] 

(5)

Therefore, removal of an edge from a graph, which decreases $n_e$ by one, must decrease the sum of $n_r$ and $n_n$. To be more specific, from Definitions 5 and 7 and from Equations 3 and 4, removal of an edge by shorting results in the following changes:

(a) Decrease $n_e$ by one.
(b) Possible decrease in $n_v$.
(c) No change in $n_p$.
(d) Decrease in $n_r$ only if there is a decrease in $n_v$.
(e) Decrease in $n_n$ only if there is no decrease in $n_v$.

Consequently each edge removed by shorting decreases the sum of $n_r$ and $n_n$ by either decreasing $n_r$ while $n_n$ remains constant or by decreasing $n_n$ while $n_r$ remains constant.
Reference to Definitions 4 and 7 and to Equations 3 and 4 shows that removal of an edge by opening results in the following changes:

(a) Decrease in $n_e$ by one.
(b) No change in $n_v$.
(c) Possible increase in $n_p$.
(d) Decrease in $n_r$ only if there is an increase in $n_p$.
(e) Decrease in $n_n$ only if there is no increase in $n_p$.

Again each edge removed by opening decreases either $n_r$ while $n_n$ remains constant or decreases $n_n$ while $n_r$ remains constant.

Table 3 is given as a summary of how the removal of the edge sets listed in Figure 2 affect the rank and nullity of the reference graph. The removed edge sets in Table 3 are later named, based in part on how the removal process affects the rank and nullity.

G. Extrema

Two important qualifying words are employed in many definitions to follow. These two words are used to save appending the following constraining statement to the end of many definitions:

... provided that no proper subset (or superset) precisely fulfills the same criterion.

1. Minimal

The qualifying word that constrains proper subsets is defined as follows:
Definition 11. The word \textit{minimal} when applied to a set of elements fulfilling a certain criterion means that no proper subset (obtained by deleting one or more elements of the set) still precisely fulfills the criterion.

2. \textbf{Maximal}

The qualifying word that constrains proper supersets is defined as follows:

Definition 12. The word \textit{maximal} when applied to a set of elements fulfilling a certain criterion means that no proper superset (obtained by adding one or more new elements to the set) still precisely fulfills the criterion.

Table 3. Effects of edge removal shown in Figure 2

<table>
<thead>
<tr>
<th>Figure</th>
<th>Removal by</th>
<th>(n_e)</th>
<th>(n_v)</th>
<th>(n_p)</th>
<th>(n_r)</th>
<th>(n_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>2(a)</td>
<td>Opening 4 edges</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>2(b)</td>
<td>Shorting 4 edges</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2(c)</td>
<td>Opening 3 edges</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2(d)</td>
<td>Shorting 5 edges</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2(e)</td>
<td>Opening 4 edges</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2(f)</td>
<td>Shorting 4 edges</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

An example of a minimal disconnecting edge set and the generated graph is shown in Figure 2(c). The set is clearly minimal since reference to
Figure 1 illustrates that neither $\bar{a} \cdot \bar{c}$, $\bar{a} \cdot \bar{f}$, nor $\bar{c} \cdot \bar{f}$ results in a disconnected generated graph.

An example of a maximal edge set whose removal by shorting reduces the nullity of Figure 1 by zero and the generated graph is given in Figure 2(b). It is interesting to note that this same set, $a \cdot b \cdot c \cdot g$, is also a minimal set whose removal by shorting reduces the rank of Figure 1 to zero.

H. Edge Sets

This section defines four edge sets that are commonly referred to in the literature. All four sets are defined independently and are symbolized by the notation used in the previous sections. Because of this particular notation, each set can be used to obtain a generated graph from the reference graph, as was done in Figure 2. Consequently the reader is cautioned to understand that the defined terms apply directly to the sets themselves and indirectly to the generated graphs. Of course the resulting generated graphs can also be defined by use of the word complement which, for the time being, can be loosely interpreted as meaning that which remains.

1. Cut set

Definition 13. A cut set is a minimal set of edges which when opened reduces the rank of a connected graph by one (5, 14).

For example all seven cut sets of Figure 1 are listed in Table 4. Notice that the dot (*) has been deleted for the purposes of brevity.
Table 4. Cut sets of Figure 1

<table>
<thead>
<tr>
<th>e</th>
<th>b</th>
<th>c</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>g</td>
<td>b</td>
<td>c</td>
<td>a</td>
</tr>
<tr>
<td>c</td>
<td>f</td>
<td>e</td>
<td>d</td>
</tr>
</tbody>
</table>

2. Tree set

Definition 14. A tree set is a minimal set of edges which when shorted reduces the rank of a connected graph to zero.

The thirteen trees of Figure 1 are given in Table 5.

Table 5. Tree sets of Figure 1

<table>
<thead>
<tr>
<th>a bc g</th>
<th>a cd g</th>
<th>a df g</th>
<th>b cd g</th>
<th>b df g</th>
</tr>
</thead>
<tbody>
<tr>
<td>a bd g</td>
<td>a ce g</td>
<td>a ef g</td>
<td>b ce g</td>
<td>b ef g</td>
</tr>
<tr>
<td>a be g</td>
<td>a cf g</td>
<td>b cf g</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These two definitions are both associated with rank. The only distinguishing differences can be denoted by the following:

(a) Interchange the words cut and tree.
(b) Interchange the words opened and shorted.
(c) Interchange the words by and to.
(d) Interchange the words one and zero.

This same type of similarity exists in the next two definitions.
3. Circuit set

Definition 15. A circuit set is a minimal set of edges which when shorted reduces the nullity of a connected graph by one.

The graph in Figure 1 has seven circuits. These are listed in Table 6.

Table 6. Circuit sets of Figure 1

<table>
<thead>
<tr>
<th>h</th>
<th>d</th>
<th>e</th>
<th>a</th>
<th>b</th>
<th>f</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. Cotree set

Definition 16. A cotree set is a minimal set of edges which when opened reduces the nullity of a connected graph to zero.

The thirteen cotrees of Figure 1 are given in Table 7.

Table 7. Cotree sets of Figure 1

| a | c | d | h | a | d | f | h | b | c | d | h | b | d | f | h | c | d | f | h |
| a | c | e | h | a | c | e | h | b | c | e | h | b | c | e | h | a | d | e | h | b | d | e | h | c | e | f | h |

All four definitions are concerned with minimal sets of edges. Also all four sets are only defined for connected graphs. Of course each
definition can be applied to any part of a disconnected graph.

The key words which distinguish one definition from another are opened or shorted, rank or nullity, by or to, and one or zero. Thus by using all combinations of these words, there is a total of $2^4$, or 16, possible definitions. Of course if minimal is changed to maximal, there are 16 more distinct definitions. This chapter is concerned with the four given definitions. The next chapter finally introduces the other 28 definitions indirectly as part of the entire edge-set hierarchy.

I. Equivalent Set Definitions

It would be convenient to simply state that obviously the four defined sets of the previous section are equivalent to those used in the literature. Unfortunately this asserted equivalence might not be obvious. Hence this section proceeds to verify that the previous definitions do not really introduce any new edge sets but simply define the sets from a different point of view. Also the term complement is considered in more detail.

1. Cut set

The cut set as defined by Definition 13 is virtually identical to the cut-set definition given by Hakimi (5). The definition is also directly comparable to the cut-set definition used by Seshu and Reed (14). So we simply accept Definition 13 as being obviously equivalent to accepted cut-set definitions.

2. Tree set

To illustrate the equivalence of Definition 14 to other tree-set
Theorems used in the literature, the following theorem is given:

Theorem 1. An edge set is a minimal set of edges which when shorted reduces the rank of a connected graph to zero if and only if the edge set contains \( n_v - 1 \) edges, is incident to all \( n_v \) vertices, and is connected.

The second portion of the theorem is one of the many definitions that has already been proved by others to be an equivalent description of those edge sets that have been referred to as tree sets. Therefore proof of the above theorem constitutes a proof for any other equivalent definition.

To begin the proof, we demonstrate that the first portion of the theorem implies the second portion. To do this, we note from Equation 3 that reducing the rank of a graph to zero implies that an operation must be performed that results in a generated graph where the number of vertices equals the number of parts. Since shorting edges does not change \( n_p \) and since \( n_p \) for the connected reference graph is unity, the operation must produce a generated graph having only one vertex. In other words the operation must coalesce all vertices of the reference graph into one vertex as illustrated by Figure 2(b). Furthermore since the shorted set is minimal, since the rank is being reduced by \( n_v - 1 \), and since each shorted edge can only reduce the rank by one, the shorting set must contain precisely \( n_v - 1 \) edges. Also since shorting an edge only coalesces those two vertices associated with that particular edge, the set of edges must be incident to all \( n_v \) vertices for otherwise some vertices are not coalesced by the shorting process, and more than one vertex remains in the generated
graph. Finally the set must necessarily be connected because shorting unconnected sets of edges results in more than one remaining vertex in the generated graph.

Now we illustrate that the second portion of the proof implies the first portion. A set of connected edges incident to \( n_v \) vertices when removed by shorting certainly coalesces all vertices into one vertex and thereby results in a generated graph having zero rank. Since the rank has been reduced by \( n_v - 1 \) and since one shorted edge can only reduce the rank by one, a set of \( n_v - 1 \) shorting edges is definitely a minimal set.

Perhaps some comments are in order concerning the two equivalent definitions given in Theorem 1, which of course describe a tree set. The first definition points out the relationship between a tree set and the rank of the graph. The definition also is worded so as to illustrate both the similarities and the differences between tree sets, cut sets, circuit sets, and cotree sets. Finally the definition closely agrees with the chosen set notation. On the other hand the second definition is perhaps easier to visualize. Also the second definition more clearly points out that a tree set essentially connects the graph while a cut set disconnects the graph.

3. Circuit set

The equivalence of Definition 15 to another commonly used circuit-set definition is given in the following:

Theorem 2. An edge set is a minimal set of edges which when shorted reduces the nullity of a connected graph by one if and
only if the edge set is a minimal set not contained in any tree set.

To begin the proof, we again illustrate that the first portion of the theorem implies the second portion. To accomplish this, we recall that each shorted edge either reduces the rank or the nullity but not both and that no mention has ever been made regarding the particular order in which the edges of a set are shorted. Since the circuit set is minimal and since the nullity is only reduced by one, it must always be the removal of the last edge in any chosen shorting sequence that finally reduces the nullity. The shorting of all previous edges of the sequence simply reduces the rank. Since the order of the sequence is immaterial, any edge of a circuit set can be the last edge to be shorted. Thus every proper subset of a circuit set reduces the rank when removed by shorting, and by comparing this to Definition 14 for a tree set, we can state the following:

Lemma 1. Every proper subset of a circuit set is a subset of some tree set (14).

Consequently since every proper subset belongs to a tree set and since a circuit set cannot belong to a tree set because it affects the nullity, a circuit set must be a minimal set not contained in any tree set.

To prove that the second portion of the theorem implies the first portion, we note that every proper subset of a minimal set which is not contained in any tree set is a subset of some tree set. If such were not the case, the original set would not be minimal. If all but one edge of
the minimal set is shorted, only the rank of the graph is affected. Shorting the last remaining edge must reduce the nullity for otherwise the original set would indeed be a subset of some tree. Of course the nullity is only reduced by one because only one edge is involved. So far we have shown that the second definition implies an edge set which when shorted reduces the nullity by one. The reason that the minimal qualifier can be added is apparent from the fact that every proper subset of the second definition only affects the rank of the graph.

It is convenient at this point to list a lemma which corresponds to that given in the foregoing proof. If the following changes are made in the paragraph preceding Lemma 1:

(a) change shorted to opened,
(b) interchange rank and nullity,
(c) change circuit to cut,
(d) change Definition 14 to Definition 16,
(e) and change tree to cotree,

then we have:

Lemma 2. Every proper subset of a cut set is a subset of some cotree set (14).

4. Cotree set

To show the equivalence of Definition 16 to other cotree-set definitions the next theorem is given:
Theorem 3. An edge set is a minimal set of edges which when opened reduces the nullity of a connected graph to zero if and only if the edge set is a tree-set complement.

Since complement has been heretofore used rather loosely, we tender the following specific definition:

Definition 17. An edge-set complement is a second set composed of all edges of a graph not in the first set.

It is convenient to denote an edge-set complement symbolically by listing all edges not in the given edge set and by changing opening to shorting or shorting to opening. This has nothing to do with the complement definition but is instead a matter of notation. Examples of this notation are given in Figure 2 where the generating sets of (a) and (b), (c) and (d), and (e) and (f) are complements of each other.

Again we begin the theorem proof by illustrating that the first portion implies the second portion. To do this, we first restate the problem as follows:

If a minimal set of edges which when opened reduces the nullity of a connected graph to zero, then the remaining edge set is a tree set.

Since the set is minimal, only \( n_e - (n_v - 1) \) edges are opened; hence \( n_v - 1 \) edges remain unaffected. Since the opened set only affects the nullity and since opening does not affect \( n_v \), we see from Equation 4 that the number of parts remains unity which implies that the remaining edges
are connected. To summarize, the number of vertices remains unchanged at \( n_v \), and the remaining edges are \( n_v - 1 \) in number and connected. Therefore the remaining edge set is indeed a tree set as defined by Theorem 1.

Now we need to prove that the second portion implies the first portion. By definition the opening of all edges of a tree-set complement results in a generated graph the edges of which are a tree set, as is illustrated in Figure 2(a). Since such a generated graph has the same rank as the reference graph, opening the tree-set complement must only have reduced the nullity, and since the nullity of such a generated graph is zero, the tree-set complement must have reduced the nullity to zero. Also since the tree-set complement contains precisely \( n_e - (n_v - 1) \) edges, which is the amount by which the nullity was reduced, the edge set must indeed be minimal.

J. Edge Classifications

As is often the case, an edge can also be an edge set. In fact there are special edges which are not only an edge set but are such that they cannot appear in certain other edge sets. Even though defining these special edges adds more definitions to a formidable list, we will do so because of the benefits derived and because of the relative simplicity of the definitions.

1. Cut edge

Definition 18. A cut edge is an edge that is a cut set of that part to which the edge is incident.
Since a cut edge is a cut set, its removal by opening reduces the rank of the part by one. This reduction is accomplished by disconnecting the part. Thus such an edge must also be in every tree set of the part for otherwise the tree set could not possibly connect all vertices of the part as required by Theorem 1. Hence a cut edge also reduces the rank when it is shorted which leads to the following theorem:

Theorem 4. A cut edge reduces the rank if opened or shorted.

This special characteristic in turn implies that the cut edge cannot be an edge of any circuit set or any cotree set. Thus such an edge could also be called a circuitless edge or a cotreeless edge. These names in turn indicate that the cut edge occurs in every circuit complement and every cotree complement, or tree. An example of a cut edge is edge g in Figure 1. Other examples also occur in Figure 2(a), (c), and (f).

2. Tree edge

Definition 19. A tree edge is an edge that is a tree set of that part to which the edge is incident.

A tree edge is of limited usefulness and is defined for completeness. For an example refer to Figure 2(d) where each edge is a tree edge.

3. Circuit edge

Definition 20. A circuit edge is an edge that is a circuit set of that part to which the edge is incident.
Since a circuit edge is a circuit set, its removal by shorting reduces the nullity of the part by one. Furthermore from Theorems 2 and 3 a circuit edge must be in every tree-set complement, or cotree set, of the part. Consequently a circuit edge must also reduce the nullity when opened which gives the following theorem:

**Theorem 5.** A circuit edge reduces the nullity if shorted or opened.

This special characteristic implies that the circuit edge cannot be an edge of any cut set or any tree set. Hence such an edge could also be referred to as a cutless edge or a treeless edge. These names indicate that the circuit edge occurs in every cut complement and every tree complement, or cotree. The circuit edge is also commonly referred to as a self loop or loop, but this introduces new and unnecessary terminology.

An example of a circuit edge is edge h in Figure 1. Other examples also occur in Figure 2(b), (c), and (f).

4. **Cotree edge**

Definition 21. A cotree edge is an edge that is a cotree set of that part to which the edge is incident.

As in the case of the tree edge, this edge has limited usefulness other than completeness. An example is given in Figure 2(e) where each edge is indeed a cotree edge of the connected portion of the generated graph shown.
K. Graph Classifications

The terminology used to denote four specific sets and four specific edges can also be used to denote four specific graphs. This section defines these four graphs and provides a list of properties in Table 8.

1. Cut graph

Definition 22. A cut graph is a connected graph whose edge set is a cut set.

Since a cut set is minimal, a cut graph has one and only one cut-set. Also because the entire edge set comprises a cut set, removal by opening means the generated graph is composed of isolated vertices, each of which is a part. Since we know that a cut set can only reduce the rank by one and since Equation 3 illustrates that the generated graph has a rank of zero, the original cut graph must have unity rank or only two vertices. Furthermore Equation 4 indicates that the number of edges in a cut graph equals the nullity plus one. From Definition 22 and the discussion following Theorem 5, we can also state that a cut graph has no circuit edges, or cutless edges. In fact since there are only two vertices and since each edge must be incident to both of these vertices, each edge is a tree edge. An illustration of these properties is given in Figure 2(d).

2. Tree graph

Definition 23. A tree graph is a connected graph whose edge set is a tree set.
The pertinent properties of a tree graph are given in Table 8, and
the most distinguishing property is that the nullity is zero. The ver-
ification of these properties closely follows the foregoing format used
for the cut graph and is based on the previous definitions and equations.
Thus the development is omitted and left to the interested reader.

3. Circuit graph

Definition 24. A circuit graph is a connected graph whose edge
set is a circuit set.

The most distinguishing property of a circuit graph is that the nul-
lity is always one. The verification of this property as well as the
other properties listed in Table 8 is again left to the interested reader.

4. Cotree graph

Definition 25. A cotree graph is a connected graph whose edge set
is a cotree set.

The most distinguishing property of a cotree graph is that it is com-
posed of one vertex and a set of circuit edges. Nevertheless a cotree
graph can assume many geometric forms by arranging the circuit edges in
various ways. For example any given circuit edge can be drawn so as to
encompass numerous combinations of other circuit edges belonging to the
cotree graph. Again verification is omitted.

Comparison of the geometric graphs in Figure 2(a), (b), (d), and (e)
to their respective generating sets illustrates that the geometric graphs
can be generated from the reference graph by removing the respective set complement. For example to generate a tree graph, remove the tree-set complement, or cotree set, by opening as indicated by the notation.

<table>
<thead>
<tr>
<th>Category</th>
<th>Cut Graph</th>
<th>Tree Graph</th>
<th>Circuit Graph</th>
<th>Cotree Graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge set is a</td>
<td>cut set</td>
<td>tree set</td>
<td>circuit set</td>
<td>cotree set</td>
</tr>
<tr>
<td>Rank</td>
<td>1</td>
<td>$n_e$</td>
<td>$n_v-1$</td>
<td>0</td>
</tr>
<tr>
<td>Nullity</td>
<td>$n_e-1$</td>
<td>0</td>
<td>1</td>
<td>$n_e$</td>
</tr>
<tr>
<td>No. of edges</td>
<td>$n_n+1$</td>
<td>$n_r$</td>
<td>$n_r+1$</td>
<td>$n_e$</td>
</tr>
<tr>
<td>No. of vertices</td>
<td>$n_r+1=2$</td>
<td>$n_r+1$</td>
<td>$n_r+1$</td>
<td>$n_r+1=1$</td>
</tr>
<tr>
<td>Contains only</td>
<td>tree edges</td>
<td>cut edges</td>
<td>cotree edges</td>
<td>circuit edges</td>
</tr>
<tr>
<td>Example</td>
<td>Figure 2(d)</td>
<td>Figure 2(a)</td>
<td>Figure 2(e)</td>
<td>Figure 2(b)</td>
</tr>
</tbody>
</table>

L. Interrelationships

This section is a compendium of the previous set definitions and equivalent set definitions as found in the literature. This compendium serves to directly point out the variations encountered in describing sets and to thus indirectly point out variations encountered in describing edges and graphs. Also this section illustrates the interrelationships between the four sets that have been defined and discussed at length in the previous sections.
In an effort to condense and to eliminate tedium, the three appendages set, edge, and graph are often deleted. These appendages are hereafter used only when necessary to avoid confusion. Also as far as possible, those definitions that are in some way related occur at the same location in the following lists. This is not meant to imply that the third definition, for example, in each list has to be related.

1. Cut

Any one of the following definitions is an equivalent and complete description of a cut:

Definition 13(a). A minimal set of edges which when opened reduces the rank of a connected graph by one (5, 14).

Definition 13(b). A minimal set of edges with at least one edge of every tree (14).

Definition 13(c). A minimal set of edges not contained in any cotree (14).

Definition 13(d). A minimal set of edges with an even number of edges from each circuit (14).

Definition 13(e). A minimal set of edges that disconnects a connected graph (2).

Since all of the above definitions are readily available in the literature as indicated and since the equivalence of the definitions has been considered by others, no further comment is deemed necessary.
2. Tree

Any one of the following definitions is an equivalent and complete description of a tree:

Definition 14(a). A minimal set of edges which when shorted reduces the rank of a connected graph to zero.

Definition 14(b). A minimal set of edges with at least one edge of every cut.

Definition 14(c). A maximal set of edges that contains no circuit.

Definition 14(d). A maximal set of edges which when shorted reduces the nullity of a connected graph by zero.

Definition 14(e). A maximal set of edges that does not contain at least one edge of every cotree.

Definition 14(f). A cotree complement (13).

Definition 14(g). A set of $n_v-1$ edges that contains no circuit (14).

Definition 14(h). A set of $n_v-1$ edges that connects $n_v$ vertices (1, 14).

Definition 14(i). A set of edges that connects $n_v$ vertices and contains no circuits (1, 2, 14).
Definition 14(j). A set of edges that connects $n_v$ vertices but loses this property if any edge is opened (1).

Definition 14(k). A minimal set of edges that connects the graph.

The validity of the first definition is established by Theorem 1. The approach used in the next section shows that Definition 14(b) is a direct consequence of Definition 13(b). Later material also establishes the validities of Definitions 14(c) and (d). The wording of Definition 14(e) is a direct consequence of Definition 14(f), as becomes apparent in the next section. The last definition is merely a rewording of Definition 14(h).

The last four definitions have no counterpart in the cotree list. This is essentially because of the qualifying word connected occurring in each definition.

3. Circuit

Any one of the following definitions is an equivalent and complete description of a circuit:

Definition 15(a). A minimal set of edges which when shorted reduces the nullity of a connected graph by one.

Definition 15(b). A minimal set of edges with at least one edge of every cotree (14).
Definition 15(c). A minimal set of edges not contained in any tree (14).

Definition 15(d). Minimal set of edges with an even number of edges from each cut (14).

Definition 15(e). A set of $n$ distinct edges and a set of $n$ distinct vertices which can be ordered in two sequences $e_1, e_2, \ldots, e_n$ and $v_0, v_1, \ldots, v_n$ where $v_0$ is identical to $v_n$ such that $e_i$ is incident to $v_{i-1}$ and $v_i$ for $i = 1, 2, \ldots, n$ (2).

The equivalence of the first definition is covered in Theorem 2. All remaining definitions are referenced to the literature.

4. Cotree

Any one of the following definitions is an equivalent and complete description of a cotree:

Definition 16(a). A minimal set of edges which when opened reduces the nullity of a connected graph to zero.

Definition 16(b). A minimal set of edges with at least one edge of every circuit.

Definition 16(c). A maximal set of edges that contains no cut.

Definition 16(d). A maximal set of edges which when opened reduces the rank of a connected graph by zero.
Definition 16(e). A maximal set of edges that does not contain at least one edge of every tree.

Definition 16(f). A tree complement (13).

Definition 16(g). A set of \( n_v - 1 \) edges that contains no cut.

The validity of the first definition is covered in Theorem 3. Definition 16(b) is a consequence of Definition 15(b). Later material provides Definitions 16(c) and (d), and Definition 16(e) evolves directly from Definition 16(f). The last definition is a result of combining Definition 16(f), the fact that a tree contains \( n_v - 1 \) edges, and Definition 16(c).

The previous lists clearly point out that the four sets are interrelated. The key definitions illustrating these facts are given in Figure 3. Here we finally have a compact illustration showing how any edge set can be obtained from a complete listing of any other group of edge sets.

Perhaps a brief explanation is in order regarding the wording of the interrelationship shown at the top of Figure 3. The format agrees with the format of the other interrelationships in that it describes how to generate an edge set from an entire group of other edge sets. Actually the wording is equivalent to the word complement, which instead describes how to generate an edge set from another edge set.

M. Boolean Functions

We are now in a position to exploit more fully the set notation that has been adopted. In doing this, we also utilize a few more concepts...
Figure 3. Interrelationships between groups of edge sets
from Boolean algebra. The result is a compact symbolic representation of some of the interrelationships in the previous section. In particular the resulting Boolean functions illustrate an orderly method for obtaining one group of edge sets from another group of edge sets along the lines indicated in Figure 3.

l. Not-tree

Definition 26. A not-tree is the result of applying DeMorgan's theorem to the negation of a tree set.

In applying the above definition to generate a not-tree from a tree set, the overbar is used to represent negation, and DeMorgan's theorem is then applied in the same manner as in Boolean algebra. An example clarifies the definition and the resulting notation. To this end let us use the first tree in Table 5, which we denote by

\[ T_1 = a \cdot b \cdot c \cdot g \quad (6) \]

The corresponding not-tree is

\[ \overline{T_1} = \overline{a \cdot b \cdot c \cdot g} \]

\[ = a + b + c + g \quad (7) \]

where the plus sign, or addition, is interpreted in the Boolean sense as signifying the word or. Thus DeMorgan's theorem changes the shorting operation symbolized in the tree set to the negation of shorting, or to the opening operation. The theorem also changes the and symbolism (multiplication) to the or symbolism (addition). The end result of these two
changes is a not-tree which symbolically indicates how to disconnect the corresponding tree. For example Equation 7 illustrates how to disconnect $T_1^*$; that is, open edge $a$ or open edge $b$ or open edge $c$ or open edge $g$ or open any combination of these edges in Figure 2(a).

Since $T_1$ connects all vertices in Figure 1 and since every cut set disconnects the reference graph, every cut set must disconnect $T_1$. Consequently $\overline{T_1}$ can be interpreted as a necessary condition, or constraint, that must be satisfied by every cut set. For example every cut set of Figure 1 must open $a$, $b$, $c$, or $g$. In fact every cut set must simultaneously disconnect every tree in Table 5 in the manner indicated by the not-trees.

Thus far we have established that all of the not-trees of a graph are necessary cut-set constraints. For the present let us assume that the not-trees are also sufficient cut-set constraints. Let us also introduce the graph in Figure 4 as a simple example having three cut sets, denoted by $S_1$, $S_2$, and $S_3$, and three not-trees. Based on the previous discussion and the above sufficiency assumption, it seems logical to conjecture that $S_1$ or $S_2$ or $S_3$ results from satisfying $\overline{T_1}$ and $\overline{T_2}$ and $\overline{T_3}$ simultaneously with a minimal set of edges. Symbolically this conjecture has the Boolean algebra representation

$$S_1 + S_2 + S_3 = \overline{T_1} \cdot \overline{T_2} \cdot \overline{T_3} \quad (8)$$

which is subject to the simplifying properties

$$x + x = x \quad (9)$$
$$x + 1 = 1 \quad (10)$$
$$x \cdot x = x \quad (11)$$
Figure 4. Graph having three trees and three cuts.
and
\[ x \cdot 1 = x \]  
(12)

In fact it is these simplifying properties of Boolean algebra that enable Equation 8 to be methodically reduced to a sum of minimal products. To be more specific, substitution of the trees into Equation 8 yields
\[ S_1 + S_2 + S_3 = (a \cdot b) \cdot (a \cdot c) \cdot (b \cdot c) \]  
(13)

which by applying DeMorgan's theorem and the simplifying properties becomes
\[ S_1 + S_2 + S_3 = \overline{(a + b)} \cdot \overline{(a + c)} \cdot (b + c) \]

\[ = \overline{(a + b)} \cdot \overline{(a \cdot b + c)} \]  
(14)
\[ = \overline{a} \cdot b + \overline{a} \cdot c + b \cdot \overline{c} \]

The simplifying properties were applied as follows:
\[ \overline{c} \cdot \overline{c} = \overline{c} \]  
(15)
\[ \overline{a} \cdot \overline{c} + b \cdot \overline{c} + \overline{c} = (\overline{a + b + 1}) \cdot \overline{c} \]
\[ = 1 \cdot \overline{c} \]  
(16)
\[ = \overline{c} \]
\[ \overline{a} \cdot \overline{a} \cdot \overline{b} = \overline{a} \cdot \overline{b} \]  
(17)
\[ \overline{a} \cdot \overline{b} \cdot \overline{b} = \overline{a} \cdot \overline{b} \]  
(18)
\[ \overline{a} \cdot \overline{b} + \overline{a} \cdot b = \overline{a} \cdot \overline{b} \]  
(19)
Upon comparing Equation 14 to Figure 4, we see that each product term is indeed one of the cut sets, and the conjecture is correct, at least for this simple example. The clue that the conjecture applies in general to every finite connected graph lies in the fact that each product term in the sum of minimal products is actually a minimal set of edges that contains at least one edge of every not-tree, which is actually equivalent to containing at least one edge of every tree. In retrospect this approach is no more than a symbolic application of Definition 13(b), hence the following theorem:

Theorem 6. The Boolean product of all not-trees of a graph produces all cut sets of the graph when converted to the sum of minimal products.

Symbolically Theorem 6 can be written as

\[ \sum S = \bigcap \bar{T} \]

where the equation is understood to be a Boolean function and the product of sums (not-trees) is to be converted to the sum of minimal products.

2. Not-cut

Definition 27. A not-cut is the result of applying DeMorgan's theorem to the negation of a cut set.

An argument could be developed along the lines given above to show that the not-cuts are necessary and sufficient tree-set constraints. An alternate approach, which yields the same end result, is to apply DeMorgan's
theorem to the negation of Equation 20

\[ \Pi T = \Sigma S \] (21)

which yields

\[ \Sigma T = \Pi S \] (22)

where it must be understood that the product of sums is to be converted
to the simplest sum of products using the simplifying properties of
Boolean algebra. Thus applying DeMorgan's theorem to the negation of
the symbolic representation of Theorem 6 produces a symbolic representa-
tion of the following:

Theorem 7. The Boolean product of all not-cuts of a graph
produces all tree sets of the graph when converted to the
sum of minimal products.

The above conversion to a sum of minimal products results in minimal
sets that contain at least one edge of every not-cut, which is equivalent
to containing one edge of every cut. Consequently Theorem 7 leads directly
to the wording of Definition 14(b) and thereby constitutes proof that this
definition is an equivalent and complete description of a tree.

3. Not-cotree

Definition 28. A not-cotree is the result of applying DeMorgan's
theorem to the negation of a cotree set.

The previous development and Definition 15(b) lead to the following:
Theorem 8. The Boolean product of all not-cotrees of a graph produces all circuit sets of the graph when converted to the sum of minimal products.

If we use $C$ to denote a circuit set and $K$ to denote a cotree set, the theorem yields the Boolean function

$$\Sigma C = \Pi \overline{K}$$

(23)

where the product of sums (not-cotrees) is to be converted to the sum of minimal products.

The graph of Figure 4 provides an exceedingly simple example. Substitution of the three cotrees into Equation 23 gives

$$C = \overline{a} \cdot \overline{b} \cdot \overline{c}$$

(24)

$$= a \cdot b \cdot c$$

which is of course the only circuit set because the graph is indeed a circuit graph.

4. Not-circuit

Definition 29. A not-circuit is the result of applying DeMorgan's theorem to the negation of a circuit set.

Applying DeMorgan's theorem to the negation of Equation 23 gives the Boolean function

$$\Sigma K = \Pi \overline{C}$$

(25)
which is subject to usual minimization constraint. This symbolic representation produces the following:

Theorem 9. The Boolean product of all not-circuits of a graph produces all cotrees of the graph when converted to the sum of minimal products.

The theorem in turn provides a proof for the equivalency of Definition 16(b).

The four preceding theorems along with the complement operation provide an orderly symbolic method for obtaining all of the cuts, trees, circuits, and cotrees from a complete listing of any one of the set categories. For example if all of the cuts of a graph are known, we can use Equation 22 to obtain the trees. By complementing each tree, we can obtain the complete list of cotrees. Using Equation 23, we can then obtain the circuits in a straightforward manner. Of course Figure 3 illustrates other alternatives for obtaining the trees, circuits, and cotrees from the cuts. No attempt is made in this presentation to symbolically illustrate these alternate methods.
III. HIERARCHY

Now that a basic foundation has been laid, the four edge-set definitions can be extended. This extension results in a hierarchy of groups of edge sets and in a generalization of the previously developed Boolean functions. This extension also utilizes more fully the maximal qualifier, which was all but ignored in the previous chapter, and illustrates some of the many alternatives available in proving that certain edge sets can be generated directly from other groups of edge sets in the hierarchy.

A. K-sets

Since the same format is initially used for the four edge-set definitions in the previous chapter and since the set definitions in this chapter are extensions of Definitions 13, 14, 15, and 16, we simply list the new definitions in tabular form in Tables 9 and 10. Besides saving space, such a listing more readily serves to illustrate both the similarities and the differences. Of course the tables also act as a summary for the edge-set definitions of the previous chapter. In fact Tables 9 and 10 also include the 32 different edge-set definitions that result from using all combinations of the key words previously discussed. The interested reader may verify that all of these combinations exist by rewriting each definition in Tables 9 and 10 using the word by instead of to or to instead of by and changing k-1 accordingly. For example a k-cut is a minimal set of edges which when removed by opening reduces the rank to $n_x-k+1$. To complete the verification, let k range from 1 to $n_x+1$ for sets pertaining to rank or from 1 to $n_y+1$ for sets pertaining to nullity.
Table 9. Minimal edge sets

<table>
<thead>
<tr>
<th>Definition Number</th>
<th>Name</th>
<th>Symbol</th>
<th>Removal Process</th>
<th>Reduces</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>k-cut</td>
<td>$S^k$</td>
<td>Opening</td>
<td>Rank by $k-1$</td>
</tr>
<tr>
<td>31</td>
<td>k-tree</td>
<td>$T^k$</td>
<td>Shorting</td>
<td>Rank to $k-1$</td>
</tr>
<tr>
<td>32</td>
<td>k-circuit</td>
<td>$C^k$</td>
<td>Shorting</td>
<td>Nullity by $k-1$</td>
</tr>
<tr>
<td>33</td>
<td>k-cotree</td>
<td>$K^k$</td>
<td>Opening</td>
<td>Nullity to $k-1$</td>
</tr>
</tbody>
</table>

Table 10. Maximal edge sets

<table>
<thead>
<tr>
<th>Definition Number</th>
<th>Name</th>
<th>Symbol</th>
<th>Removal Process</th>
<th>Reduces</th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>k-cut</td>
<td>C($S^k$)</td>
<td>Shorting</td>
<td>Rank to $k-1$</td>
</tr>
<tr>
<td>35</td>
<td>k-tree</td>
<td>C($T^k$)</td>
<td>Opening</td>
<td>Rank by $k-1$</td>
</tr>
<tr>
<td>36</td>
<td>k-circuit</td>
<td>C($C^k$)</td>
<td>Opening</td>
<td>Nullity to $k-1$</td>
</tr>
<tr>
<td>37</td>
<td>k-cotree</td>
<td>C($K^k$)</td>
<td>Shorting</td>
<td>Nullity by $k-1$</td>
</tr>
</tbody>
</table>


This procedure serves to provide all combinations of by or to and one or zero along with all combinations of the other key words.

Perhaps we should explain where the four edge-sets of the previous chapter are located in this new hierarchy of edge sets. The tree and co-tree have simply become the 1-tree and 1-cotree respectively. On the other hand the cut and circuit have become the 2-cut and 2-circuit respectively. This change in terminology is used throughout the remainder of the presentation.

By using the negation operation of Boolean algebra, we can define eight not-k-sets. Instead of listing these not-k-sets, we simply state that a not-k-set is the result of applying DeMorgan's theorem to the negation of the respective k-set. Again this agrees with the approach previously used and saves the explicit listing of eight more definitions.

Before proceeding with detailed discussions about equivalent definitions, interrelationships, and Boolean functions, the graph of Figure 1 is used as an example of the various k-sets. These sets are listed in the following five tables using the adopted set notation, and many aspects of the k-sets are apparent in this set of tables. For example the k-tree complements are identical to the k-cotrees only if k is unity. Also the null set is a part of each minimal-set hierarchy but not necessarily a part of any maximal-set hierarchy. On the other hand the entire edge set is a part of each maximal-set hierarchy but not necessarily a part of any minimal-set hierarchy.
Table 11. The 1-sets of Figure 1

<table>
<thead>
<tr>
<th>1-cut complement</th>
<th>l-cut</th>
<th>l-circuit</th>
<th>l-circuit complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b c d e f g h</td>
<td>ø</td>
<td>ø</td>
<td>a b c d e f h g</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1-tree complement</th>
<th>l-tree</th>
<th>l-cotree</th>
<th>l-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a b c g</td>
<td>ã ã ã ã</td>
<td>b e f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a b d g</td>
<td>ã ã ã ã</td>
<td>b d f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a b e g</td>
<td>ã ã ã ã</td>
<td>b c f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a c d g</td>
<td>ã ã ã ã</td>
<td>b c e g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a c e g</td>
<td>ã ã ã ã</td>
<td>b c d g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a c f g</td>
<td>ã ã ã ã</td>
<td>a e f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a d f g</td>
<td>ã ã ã ã</td>
<td>a d f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>a e f g</td>
<td>ã ã ã ã</td>
<td>a c f g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>b c d g</td>
<td>ã ã ã ã</td>
<td>a c e g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>b c e g</td>
<td>ã ã ã ã</td>
<td>a c d g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>b c f g</td>
<td>ã ã ã ã</td>
<td>a b e g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>b d f g</td>
<td>ã ã ã ã</td>
<td>a b d g</td>
</tr>
<tr>
<td>ã ã ã ã ã ã ã ã</td>
<td>b e f g</td>
<td>ã ã ã ã</td>
<td>a b c g</td>
</tr>
</tbody>
</table>
Table 12. The 2-sets of Figure 1

<table>
<thead>
<tr>
<th>2-cut complement</th>
<th>2-cut</th>
<th>2-circuit</th>
<th>2-circuit complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b c d e f h</td>
<td>\overline{g}</td>
<td>h</td>
<td>\overline{a b c d e f g}</td>
</tr>
<tr>
<td>c d e f g h</td>
<td>\overline{a b}</td>
<td>d e</td>
<td>\overline{a b c f h g}</td>
</tr>
<tr>
<td>b d e g h</td>
<td>\overline{a c f}</td>
<td>a b f</td>
<td>\overline{c d e h g}</td>
</tr>
<tr>
<td>a d e g h</td>
<td>\overline{b c f}</td>
<td>c d f</td>
<td>\overline{a b e h g}</td>
</tr>
<tr>
<td>a b f g h</td>
<td>\overline{c d e}</td>
<td>c e f</td>
<td>\overline{a b d h g}</td>
</tr>
<tr>
<td>b c g h</td>
<td>\overline{a d e f}</td>
<td>a b c d</td>
<td>\overline{e f h g}</td>
</tr>
<tr>
<td>a c g h</td>
<td>\overline{b d e f}</td>
<td>a b c e</td>
<td>\overline{d f h g}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2-tree complement</th>
<th>2-tree</th>
<th>2-cotree</th>
<th>2-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>\overline{a b c f g h}</td>
<td>a b c</td>
<td>\overline{a c d}</td>
<td>b e f h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a b d</td>
<td>\overline{a c e}</td>
<td>b d f h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a b e</td>
<td>\overline{a c h}</td>
<td>b d e f g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a b g</td>
<td>\overline{a d e}</td>
<td>b c f h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a c d</td>
<td>\overline{a d f}</td>
<td>b c e h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a c e</td>
<td>\overline{a d h}</td>
<td>b c e f g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a c f</td>
<td>\overline{a e f}</td>
<td>b c d h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a c g</td>
<td>\overline{a e h}</td>
<td>b c d f g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a d f</td>
<td>\overline{a f h}</td>
<td>b c d e g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a d g</td>
<td>\overline{b c d}</td>
<td>a e f h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a e f</td>
<td>\overline{b c e}</td>
<td>a d f h g</td>
</tr>
<tr>
<td>\overline{a b c f g h}</td>
<td>a e g</td>
<td>\overline{b c h}</td>
<td>a d e f g</td>
</tr>
</tbody>
</table>
Table 12. (Continued)

<table>
<thead>
<tr>
<th>2-tree complement</th>
<th>2-tree</th>
<th>2-cotree</th>
<th>2-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\overline{b} \overline{c} \overline{d} \overline{e} \overline{f} \overline{g} \overline{h}$</td>
<td>$a f g$</td>
<td>$\overline{b} \overline{d} \overline{e}$</td>
<td>$a c f h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{e} \overline{f} \overline{g} \overline{h}$</td>
<td>$b c d$</td>
<td>$\overline{b} \overline{d} \overline{f}$</td>
<td>$a c e h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{d} \overline{f} \overline{g} \overline{h}$</td>
<td>$b c e$</td>
<td>$\overline{b} \overline{d} \overline{h}$</td>
<td>$a c e f g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{d} \overline{e} \overline{g} \overline{h}$</td>
<td>$b c f$</td>
<td>$\overline{b} \overline{e} \overline{f}$</td>
<td>$a c d h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{d} \overline{e} \overline{f} \overline{h}$</td>
<td>$b c g$</td>
<td>$\overline{b} \overline{e} \overline{h}$</td>
<td>$a c d f g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{c} \overline{e} \overline{g} \overline{h}$</td>
<td>$b d f$</td>
<td>$\overline{c} \overline{f} \overline{h}$</td>
<td>$a c d e g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{c} \overline{e} \overline{f} \overline{h}$</td>
<td>$b d g$</td>
<td>$\overline{c} \overline{d} \overline{f}$</td>
<td>$a b e h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{c} \overline{d} \overline{g} \overline{h}$</td>
<td>$b e f$</td>
<td>$\overline{c} \overline{d} \overline{h}$</td>
<td>$a b e f g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{c} \overline{d} \overline{f} \overline{h}$</td>
<td>$b e g$</td>
<td>$\overline{c} \overline{e} \overline{f}$</td>
<td>$a b d h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{c} \overline{d} \overline{e} \overline{h}$</td>
<td>$b f g$</td>
<td>$\overline{c} \overline{e} \overline{h}$</td>
<td>$a b d f g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{b} \overline{e} \overline{f} \overline{h}$</td>
<td>$c d g$</td>
<td>$\overline{c} \overline{f} \overline{h}$</td>
<td>$a b d e g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{b} \overline{d} \overline{f} \overline{h}$</td>
<td>$c e g$</td>
<td>$\overline{d} \overline{e} \overline{f}$</td>
<td>$a b c h g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{b} \overline{d} \overline{e} \overline{h}$</td>
<td>$c f g$</td>
<td>$\overline{d} \overline{e} \overline{h}$</td>
<td>$a b c f g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{b} \overline{c} \overline{e} \overline{h}$</td>
<td>$d f g$</td>
<td>$\overline{d} \overline{f} \overline{h}$</td>
<td>$a b c e g$</td>
</tr>
<tr>
<td>$\overline{a} \overline{b} \overline{c} \overline{d} \overline{h}$</td>
<td>$e f g$</td>
<td>$\overline{e} \overline{f} \overline{h}$</td>
<td>$a b c d g$</td>
</tr>
</tbody>
</table>
Table 13. The 3-sets of Figure 1

<table>
<thead>
<tr>
<th>3-cut complement</th>
<th>3-cut</th>
<th>3-circuit</th>
<th>3-circuit complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>c d e f h</td>
<td>(\overline{a}\overline{b}\overline{g})</td>
<td>d e h</td>
<td>(\overline{a}\overline{b}\overline{c}\overline{f}\overline{g})</td>
</tr>
<tr>
<td>d e g h</td>
<td>(\overline{a}\overline{b}\overline{c}\overline{f})</td>
<td>a b f h</td>
<td>(\overline{c}\overline{d}\overline{e}\overline{g})</td>
</tr>
<tr>
<td>b d e h</td>
<td>(\overline{a}\overline{c}\overline{f}\overline{g})</td>
<td>c d f h</td>
<td>(\overline{a}\overline{b}\overline{e}\overline{g})</td>
</tr>
<tr>
<td>a d e h</td>
<td>(\overline{b}\overline{c}\overline{f}\overline{g})</td>
<td>c d e f</td>
<td>(\overline{a}\overline{b}\overline{h}\overline{g})</td>
</tr>
<tr>
<td>a b f h</td>
<td>(\overline{c}\overline{d}\overline{e}\overline{g})</td>
<td>c e f h</td>
<td>(\overline{a}\overline{b}\overline{d}\overline{g})</td>
</tr>
<tr>
<td>f g h</td>
<td>(\overline{a}\overline{b}\overline{c}\overline{d}\overline{e})</td>
<td>a b c d e</td>
<td>(\overline{f}\overline{h}\overline{g})</td>
</tr>
<tr>
<td>c g h</td>
<td>(\overline{a}\overline{b}\overline{d}\overline{e}\overline{f})</td>
<td>a b c d f</td>
<td>(\overline{e}\overline{h}\overline{g})</td>
</tr>
<tr>
<td>b g h</td>
<td>(\overline{a}\overline{c}\overline{d}\overline{e}\overline{f})</td>
<td>a b c d h</td>
<td>(\overline{e}\overline{f}\overline{g})</td>
</tr>
<tr>
<td>b c h</td>
<td>(\overline{a}\overline{d}\overline{e}\overline{f}\overline{g})</td>
<td>a b c e f</td>
<td>(\overline{d}\overline{h}\overline{g})</td>
</tr>
<tr>
<td>a g h</td>
<td>(\overline{b}\overline{c}\overline{d}\overline{e}\overline{f})</td>
<td>a b c e h</td>
<td>(\overline{d}\overline{f}\overline{g})</td>
</tr>
<tr>
<td>a c h</td>
<td>(\overline{b}\overline{d}\overline{e}\overline{f}\overline{g})</td>
<td>a b d e f</td>
<td>(\overline{c}\overline{h}\overline{g})</td>
</tr>
</tbody>
</table>
Table 13. (Continued)

<table>
<thead>
<tr>
<th>3-tree complement</th>
<th>3-tree</th>
<th>3-cotree</th>
<th>3-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{d} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( a \ b )</td>
<td>( \overline{a} \ \overline{c} )</td>
<td>( b \ d e f h g )</td>
</tr>
<tr>
<td>( \overline{b} \overline{d} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( a \ c )</td>
<td>( \overline{a} \ \overline{d} )</td>
<td>( b \ c e f h g )</td>
</tr>
<tr>
<td>( \overline{b} \overline{c} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( a \ d )</td>
<td>( \overline{a} \ \overline{e} )</td>
<td>( b \ c d f h g )</td>
</tr>
<tr>
<td>( \overline{b} \overline{c} \overline{d} \overline{f} \overline{g} \overline{h} )</td>
<td>( a \ e )</td>
<td>( \overline{a} \ \overline{f} )</td>
<td>( b \ c d e h g )</td>
</tr>
<tr>
<td>( \overline{b} \overline{c} \overline{d} \overline{e} \overline{g} \overline{h} )</td>
<td>( a \ f )</td>
<td>( \overline{a} \ \overline{h} )</td>
<td>( b \ c d e f g )</td>
</tr>
<tr>
<td>( \overline{b} \overline{c} \overline{d} \overline{e} \overline{f} \overline{h} )</td>
<td>( a \ g )</td>
<td>( \overline{b} \ \overline{c} )</td>
<td>( a \ d e f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{d} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( b \ c )</td>
<td>( \overline{b} \ \overline{d} )</td>
<td>( a \ c e f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{c} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( b \ d )</td>
<td>( \overline{b} \ \overline{e} )</td>
<td>( a \ c d f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{c} \overline{d} \overline{f} \overline{g} \overline{h} )</td>
<td>( b \ e )</td>
<td>( \overline{b} \ \overline{f} )</td>
<td>( a \ c d e h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{c} \overline{d} \overline{e} \overline{g} \overline{h} )</td>
<td>( b \ f )</td>
<td>( \overline{b} \ \overline{h} )</td>
<td>( a \ c d e f g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{c} \overline{d} \overline{e} \overline{f} \overline{h} )</td>
<td>( b \ g )</td>
<td>( \overline{c} \ \overline{d} )</td>
<td>( a \ b e f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{e} \overline{f} \overline{g} \overline{h} )</td>
<td>( c \ d )</td>
<td>( \overline{c} \ \overline{e} )</td>
<td>( a \ b d f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{d} \overline{f} \overline{g} \overline{h} )</td>
<td>( c \ e )</td>
<td>( \overline{c} \ \overline{f} )</td>
<td>( a \ b d e h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{d} \overline{e} \overline{g} \overline{h} )</td>
<td>( c \ f )</td>
<td>( \overline{c} \ \overline{h} )</td>
<td>( a \ b d e f g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{d} \overline{e} \overline{f} \overline{h} )</td>
<td>( c \ g )</td>
<td>( \overline{d} \ \overline{e} )</td>
<td>( a \ b c f h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{e} \overline{g} \overline{h} )</td>
<td>( d \ f )</td>
<td>( \overline{d} \ \overline{f} )</td>
<td>( a \ b c e h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{e} \overline{f} \overline{h} )</td>
<td>( d \ g )</td>
<td>( \overline{d} \ \overline{h} )</td>
<td>( a \ b c e f g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{d} \overline{g} \overline{h} )</td>
<td>( e \ f )</td>
<td>( \overline{e} \ \overline{f} )</td>
<td>( a \ b c d h g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{d} \overline{f} \overline{h} )</td>
<td>( e \ g )</td>
<td>( \overline{e} \ \overline{h} )</td>
<td>( a \ b c d f g )</td>
</tr>
<tr>
<td>( \overline{a} \overline{b} \overline{c} \overline{d} \overline{e} \overline{h} )</td>
<td>( f \ g )</td>
<td>( \overline{f} \ \overline{h} )</td>
<td>( a \ b c d e g )</td>
</tr>
</tbody>
</table>
Table 14. The 4-sets of Figure 1

<table>
<thead>
<tr>
<th>4-cut complement</th>
<th>4-cut</th>
<th>4-circuit</th>
<th>4-circuit complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>deh</td>
<td>abcdfg</td>
<td>cdefh</td>
<td>abg</td>
</tr>
<tr>
<td>gfh</td>
<td>abcdef</td>
<td>abcdef</td>
<td>fg</td>
</tr>
<tr>
<td>chf</td>
<td>abcdeg</td>
<td>abcdeh</td>
<td>fg</td>
</tr>
<tr>
<td>cbg</td>
<td>abcdef</td>
<td>abcdefh</td>
<td>cgb</td>
</tr>
<tr>
<td>bhg</td>
<td>abcdef</td>
<td>abcdefh</td>
<td>bgh</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4-tree complement</th>
<th>4-tree</th>
<th>4-cotree</th>
<th>4-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcdefgh</td>
<td>a</td>
<td>a</td>
<td>bcddefhgf</td>
</tr>
<tr>
<td>acdefgh</td>
<td>b</td>
<td>b</td>
<td>acdefhgf</td>
</tr>
<tr>
<td>abdefgh</td>
<td>c</td>
<td>c</td>
<td>abdefhgf</td>
</tr>
<tr>
<td>abcdefgh</td>
<td>d</td>
<td>d</td>
<td>abcdefhgf</td>
</tr>
<tr>
<td>abcdefgh</td>
<td>e</td>
<td>e</td>
<td>abcdefhgf</td>
</tr>
<tr>
<td>abcdefgh</td>
<td>f</td>
<td>f</td>
<td>abcdefhgf</td>
</tr>
<tr>
<td>abcdefgh</td>
<td>g</td>
<td>h</td>
<td>abcdefg</td>
</tr>
</tbody>
</table>
Table 15. The 5-sets of Figure 1

<table>
<thead>
<tr>
<th>5-cut complement</th>
<th>5-cut</th>
<th>5-circuit complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>h</td>
<td>(\overline{a\ b\ c\ d\ e\ f\ g})</td>
<td>a b c d e f h</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(\overline{g})</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5-tree complement</th>
<th>5-tree</th>
<th>5-cotree complement</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\overline{a\ b\ c\ d\ e\ f\ g\ h})</td>
<td>(\emptyset)</td>
<td>a b c d e f h g</td>
</tr>
</tbody>
</table>

B. Equivalent K-set Definitions

The edge-set definitions given in Table 9 can each be worded in many different ways. Sometimes a change in wording provides added insight or offers new alternatives in later applications. Thus one purpose of this section is to provide some useful equivalent definitions.

The edge-set names used in Table 10 describe sets that are already defined indirectly by Table 9 and the word complement. Consequently another purpose of this section is to prove that these edge-set names actually describe sets that fulfill the definitions as given in Table 10.

1. K-cut

Theorem 10. An edge set is a minimal set of edges which when opened reduces the rank of a connected graph by \(k-1\) if and only if the edge set is a minimal set of a connected graph which when opened creates \(k\) parts.

To show that each portion of the theorem implies the other portion,
we need to illustrate that reducing the rank by \( k-1 \) is equivalent to generating \( k \) parts. To this end we note that the opening process can only decrease the rank by \( k-1 \) by increasing the number of parts by \( k-1 \) because the number of vertices in Equation 3 remains unchanged. Since the number of parts is originally one, the number of parts in the generated graph must be \( k \).

2. K-tree

**Theorem 11.** An edge set is a minimal set of edges which when shorted reduces the rank of a connected graph to \( k-1 \) if and only if the edge set contains \( n_v-k \) edges connecting the \( n_v \) vertices into \( k \) parts.

We begin the proof in the usual manner of showing that the first portion of the theorem implies the second portion. To do this, we recognize that the rank is being reduced from \( n_v-1 \) to \( k-1 \) which amounts to a reduction of \( n_v-k \). Thus a minimal set contains \( n_v-k \) edges.

Let us now assume that these \( n_v-k \) edges connect the \( n_v \) vertices into \( x \) parts. Clearly \( x \) cannot exceed \( n_v \), and induction quickly illustrates that \( n_v-k \) edges cannot connect \( n_v \) vertices into less than \( k \) parts. When a subset of the \( n_v-k \) edges connecting a subset of the \( n_v \) vertices into one part is shorted, all vertices of that part are coalesced into one composite vertex. Consequently shorting all \( n_v-k \) edges results in \( x \) distinct composite vertices, or in a generated graph having a rank of \( x-1 \). Since the theorem states that the rank of the generated graph is to be \( k-1 \), \( x \) must equal \( k \); that is, the \( n_v-k \) edges must connect the \( n_v \) vertices into the
fewest number of parts.

The proof demonstrating that the second portion of the theorem implies the first portion rests on the fact that the $n_v$ vertices are subdivided into $k$ connected subsets. Shorting the edges coalesces each connected vertex subset into one composite vertex. The rank of the generated graph, according to Equation 3, is $k-1$.

While we are on the subject of $k$-trees, it is convenient to introduce the following:

**Corollary 1.** A set of edges is a $(k+1)$-tree set if and only if the set can be obtained by deleting one edge of some $k$-tree set.

The validity of this corollary is apparent from Definition 31, Theorem 11, and the fact that every edge of a $k$-tree reduces the rank when shorted, regardless of the shorting sequence.

### 3. K-circuit

**Theorem 12.** An edge set is a minimal set of edges which when shorted reduces the nullity of a connected graph by $k-1$ if and only if the edge set can be partitioned into two mutually exclusive, all inclusive subsets such that the first subset contains $k-1$ edges and the second subset is a minimal set which does not contain a circuit set but which when appended by any edge of the first subset contains a circuit set.

The second portion of the above theorem provides some conceptual insight for the $k$-circuit. The theorem indicates that an edge from the first
subset is a sufficient addition to the second subset to form a circuit set. Furthermore each of the \( k-1 \) edges forms a circuit set having a distinct edge not in the other circuit sets. Hence the \( k \)-circuit contains \( k-1 \) independent circuit sets which leads to the following:

**Corollary 2.** A \((k+1)\)-circuit set can be constructed from a \( k \)-circuit set by adding a minimal set of edges necessary to define another independent circuit set.

To prove the theorem, using the usual approach, we first note that the shorting process can only reduce the nullity by eventually removing a circuit edge because such a removal does not change either the number of vertices or the number of parts in Equation 4. Thus a set that reduces the nullity by \( k-1 \) via the process of shorting must generate and then remove \( k-1 \) circuit edges. In order for such a set to be minimal, the subset that converts \( k-1 \) edges into \( k-1 \) circuit edges must be minimal in the sense that no proper subset converts the \( k-1 \) edges into circuit edges. In other words the minimal qualifier in the theorem actually applies to the subset that initially generates the \( k-1 \) circuit edges.

To continue with the proof, we investigate how to convert edges to circuit edges in a minimal manner; that is, given a particular edge we need to ascertain how to make it a circuit edge. Actually the procedure is quite simple. One merely needs to select any circuit set containing the edge in question and proceed to short all other edges of the set. Obviously cut, or circuitless, edges are ruled out.

From this discussion it becomes apparent that the \( k \)-circuit set must
contain \( k-1 \) edges that become circuit edges if and only if the remaining edges of the set are shorted. For this to occur, each of the \( k-1 \) edges that is not a circuit edge in the reference graph must form a circuit set when appended by the minimal subset. Finally the minimal subset cannot contain a circuit set because shorting such a subset would reduce the nullity prematurely.

To prove the remainder of the theorem, we recognize that shorting the second subset as described in the second portion of the theorem must indeed create \( k-1 \) circuit edges out of the first subset without affecting the nullity. Shorting the remaining \( k-1 \) edges then reduces the nullity by \( k-1 \). Of course the entire set is minimal because the second subset is minimal.

By changing circuit to cut, nullity to rank, and shorting to opening in the foregoing, we obtain a closely related theorem and corollary for \( k \)-cuts.

**Theorem 13.** An edge set is a minimal set of edges which when opened reduces the rank of a connected graph by \( k-1 \) if and only if the edge set can be partitioned into two mutually exclusive, all inclusive subsets such that the first subset contains \( k-1 \) edges and the second subset is a minimal set which does not contain a cut set but which when appended by an edge of the first subset contains a cut set.

**Corollary 3.** A \( (k+1) \)-cut set can be constructed from a \( k \)-cut set by adding a minimal set of edges necessary to define another independent cut set.
4. K-cotree

Theorem 14. An edge set is a minimal set of edges which when opened reduces the nullity of a connected graph to \( k-l \) if and only if the edge set contains \( n_n-(k-l) \) edges and each edge belongs to some circuit set containing none of the other edges of the set.

Starting the proof in the usual manner, we quickly recognize that the minimal set must have \( n_n-(k-l) \) edges in order to reduce the nullity from \( n_n \) to \( k-l \). Then we recognize that the opening process only reduces the rank by generating and opening a cut edge. In other words opening any edge of a cotree set must never disconnect the graph regardless of the opening sequence. From the previous material, especially Definition 15(e), it should be apparent that only by opening an edge of some previously unopened circuit set can the opening process keep from disconnecting a graph. Thus each of the \( n_n-(k-l) \) edges of a cotree set must belong to some circuit set containing none of the other edges of the cotree set.

For the second portion of the proof, it is clear that opening an edge of a circuit set does not increase the number of parts and therefore reduces the nullity. Opening \( n_n-(k-l) \) such edges each of which belongs to some circuit set containing none of the other edges must not disconnect the graph. Hence the nullity would be reduced to \( k-l \) without affecting the rank, and the set is minimal.

In light of the foregoing discussion and by referring to Definition 33 and Corollary 1, we state the following without proof:
Corollary 4. A set of edges is a \((k+1)\)-cotree set if and only if the set can be obtained by deleting one edge of some \(k\)-cotree set.

By changing cotree to tree, opening to shorting, and \(n_n\) to \(n_r\) and by interchanging circuit and cut and rank and nullity in the theorem proof, the following becomes apparent:

Theorem 15. An edge set is a minimal set of edges which when shorted reduces the rank of a connected graph to \(k-1\) if and only if the edge set contains \(n_r-(k-1)\) edges and each edge belongs to some cut set containing none of the other edges of the set.

5. \(K\)-cut complement

Theorem 16. An edge set is a maximal set of edges which when shorted reduces the rank of a connected graph to \(k-1\) if and only if the edge set is a \(k\)-cut complement.

To start the proof, we note that reducing the rank from \(n_v-1\) to \(k-1\) amounts to a change of \(n_v-k\). Since shorting cannot increase the number of parts, the process must reduce the number of vertices by \(n_v-k\). Hence the generated graph has a total of \(k\) vertices connected into one part by the remaining edges. Since the removed set is maximal, none of the remaining edges can be circuit edges; consequently each remaining edge connects a pair of distinct vertices. To completely disconnect the generated graph, all edges of the generated graph would have to be opened. Such a
process would clearly reduce the rank from k-1 to zero, or by k-1. Thus according to Definition 30, the entire edge set of the generated graph is a k-cut with respect to the generated graph.

Returning to the reference graph, it should be apparent that the edge set of the generated graph is also a k-cut of the original graph because the shorting process merely lumped k vertex subsets into k composite vertices. In other words by opening all edges in the reference graph that exist in the generated graph, the reference graph would be separated into k parts where each part would contain all vertices represented by a composite vertex in the generated graph. Therefore it may be concluded that the maximal shorted set is indeed a k-cut complement.

To finish the proof, we refer to Theorem 10 and recognize that opening a k-cut results in a generated graph of k parts, the edges of which are the k-cut complement. If the k-cut complement is then shorted, each part is reduced to one composite vertex having no incident edges. If the k-cut set is then reinserted by connecting each edge to that pair of composite vertices which in essence contains the original incident vertices, the result is a connected graph having k vertices, a rank of k-1, and no circuit, or cutless, edges. Consequently shorting a k-cut complement reduces the rank to k-1, and the set is maximal because shorting any additional edge reduces the rank to k-2 by coalescing two distinct vertices.

6. **K-tree complement**

**Theorem 17.** An edge set is a maximal set of edges which when opened reduces the rank of a connected graph by k-1 if and only
if the edge set is a $k$-tree complement.

To prove the first implication, we note that the set reduces the rank to $n^{-k}$. Since opening does not affect the number of vertices, the process must increase the number of parts to $k$. Since the opened set is maximal, the unopened edges must be the minimal number necessary to connect $n_v$ vertices into $k$ parts; that is, the unopened edges are $n_v^{-k}$ in number. Comparison to Theorem 11 shows that this unopened set constitutes a $k$-tree. Therefore the opened set is a $k$-tree complement.

To prove the second implication, we recognize that opening a $k$-tree complement results in a generated graph composed of $k$ 1-trees and having a rank of $n^{-k}$. Thus the generation process reduces the rank of the reference graph by $k-1$. This opened set is surely maximal because all remaining edges are edges of 1-trees, which are cut edges, and opening any of these edges further reduces the rank.

7. **K-circuit complement**

**Theorem 18.** An edge set is a maximal set of edges which when opened reduces the nullity of a connected graph to $k^{-1}$ if and only if the edge set is a $k$-circuit complement.

To initiate the proof, we notice that the generated graph has no cut, or circuitless, edges because the maximal set certainly must include any edges that can affect only the rank when removed. Hence every edge in the generated graph must either be a circuit or else an edge that belongs to at least one circuit set of the generated graph. By referring to Theorem
12 and to the associated proof, it should be reasonably apparent that the entire edge set of the generated graph is a minimal set which when shorted reduces the nullity of the generated graph to zero. Since the generated graph only has a nullity of \( k-1 \) by definition, the entire edge set is a minimal set that reduces the nullity by \( k-1 \).

Let us now return our attention to the original graph. Since the opening process employed to obtain a generated graph does not disturb the remaining edges in any way, it should be clear that shorting all edges of the generated graph within the reference graph should also reduce the nullity by \( k-1 \). Therefore the edge set of the generated graph constitutes a \( k \)-circuit set, and the opened edges must then be a \( k \)-circuit complement.

In proving the remainder of the theorem, it should be pointed out that opening a \( k \)-circuit complement results in a generated graph whose edges constitute a minimal set which when removed by shorting reduces the nullity by \( k-1 \). On the other hand when all edges of a generated graph are shorted, the nullity is also reduced to zero. Consequently the generated graph has a nullity of \( k-1 \) which means that the opened set reduced the nullity to \( k-1 \). Furthermore since all remaining edges are a \( k \)-circuit by definition, we can conclude that each edge is a circuit or an edge that belongs to one or more circuit sets of the generated graph. Opening any of these edges does not change the number of parts or the number of vertices. Thus from Equation 4 we see that opening any additional edge other than those in the \( k \)-circuit complement further reduces the nullity. Thus the \( k \)-circuit complement is indeed a maximal set under the stipulated criterion.
8. **K-cotree complement**

Theorem 19. An edge set is a maximal set of edges which when shorted reduces the nullity of a connected graph by $k-1$ if and only if the edge set is a $k$-cotree complement.

Before starting the proof, we list the following steps as a means of obtaining a maximal set of edges that reduces the nullity by $k-1$:

(a) If any edge exists that connects two distinct vertices, remove this edge by shorting. (The rank is reduced, and the nullity remains unchanged.)

(b) Repeat the above process in any order until the generated graph consists of one vertex and a set of circuit edges. (This generated graph still has the same nullity as the reference graph.)

(c) Remove $k-1$ of the remaining circuit edges to finally reduce the nullity by $k-1$.

This procedure is strikingly similar to that discussed in the proof of Theorem 12 concerning $k$-circuits. In fact the major difference is that the above process reduces the rank by a maximum amount and the $k$-circuit reduces the rank by a minimum amount. More will be said of this in a later section.

To begin the proof, we notice that the generated graph consists of $n_n-(k-1)$ circuit edges. Each of these edges is either a circuit in the original graph or is an edge of some circuit set that contained none of
the other edges in the generated graph. By referring to Theorem 14, we see that the edges of the generated graph constitute a k-cotree set; therefore the opened set is a k-cotree complement.

To complete the proof, we recall from Theorem 14 that each edge of a k-cotree is associated with some circuit set such that all other edges of the circuit set belong to the k-cotree complement. From this we see that shorting all edges of a k-cotree complement serves to make each of the remaining $n_h^-(k-l)$ edges a circuit edge, all of which are incident to the one remaining vertex. Thus the generated graph has a nullity of $n_h^-(k-l)$, and shorting the k-cotree complement reduces the nullity by $k-1$. Since any additional edge shorting must further reduce the nullity, the set is a maximal set under the stated criterion.

C. Interrelationships

Before proceeding with various interrelationships between specific groups of k-sets in the edge-set hierarchy, we refer to Figure 3 and conjecture that the interrelationships which link the tree and cotree to the cut and circuit are themselves interrelated. To be more specific, we refer to Figure 5 and conjecture that if the group of $\alpha$-sets or $\alpha$-set complements are related to the $\beta$-sets by any one of the four linkages shown and if the $\alpha$-sets and $\beta$-sets are minimal sets then the other three linkages are automatically valid.

In an effort to establish that the existence of any one of the four interrelationships, or linkages, between the $\alpha$-sets or $\alpha$-set complements and the $\beta$-sets implies the other three linkages, we start with the following:
Figure 5. Interrelationships between groups of sets and set complements
Theorem 20. A $\beta$-set is a minimal set with at least one edge of every $\alpha$-set if and only if the $\beta$-set is a minimal set not contained in any $\alpha$-set complement.

To prove that the first portion of the theorem implies the second portion, we recognize some pertinent facts regarding the $\beta$-set as defined in the beginning of the theorem. Since the $\beta$-set contains at least one edge of every $\alpha$-set, the entire $\beta$-set cannot possibly be a subset of any of the $\alpha$-set complements. Since the $\beta$-set is a minimal set, no edge can be deleted from the $\beta$-set without violating the constraint that the $\beta$-set contains at least one edge of every $\alpha$-set. Hence for every edge in the $\beta$-set, there must be at least one $\alpha$-set that contains this edge and no other edges of the $\beta$-set. Thus every proper subset of the $\beta$-set is a subset of some $\alpha$-set complement. In other words the $\beta$-set is not contained in any $\alpha$-set complements and is minimal because every proper subset is contained in some $\alpha$-set complement.

To prove that the second portion of the theorem implies the first portion, we recognize some pertinent facts regarding the $\beta$-set as defined in the end of the theorem. Since the entire $\beta$-set is not contained in any $\alpha$-set complement, the $\beta$-set must contain one or more edges of every $\alpha$-set. Since the $\beta$-set is a minimal set not contained in any $\alpha$-set complement, every proper subset is contained in some $\alpha$-set complement. Therefore every individual edge of the $\beta$-set appears in at least one $\alpha$-set that contains no other edges of the $\beta$-set. In other words the $\beta$-set is a minimal set with at least one edge of every $\alpha$-set.

In continuing to verify the generalizations indicated in Figure 5,
we submit the following:

Theorem 21. An \( \alpha \)-set is a minimal set with at least one edge of every \( \beta \)-set if and only if the \( \alpha \)-set complement is a maximal set that contains no \( \beta \)-set.

To prove the first implication, we recall from the foregoing proof that for any edge in the \( \alpha \)-set there must be at least one \( \beta \)-set that contains this edge and no other edges of the \( \alpha \)-set for otherwise the \( \alpha \)-set would not be a minimal set with at least one edge of every \( \beta \)-set. This being the case, the \( \alpha \)-set complement must contain all but one edge of some \( \beta \)-set. Since this is true for every edge of the \( \alpha \)-set, the \( \alpha \)-set complement appended by any missing edge contains at least one \( \beta \)-set. Furthermore since the \( \alpha \)-set contains at least one edge of every \( \beta \)-set by definition, the \( \alpha \)-set complement cannot possibly contain any \( \beta \)-set. In other words the \( \alpha \)-set complement is a maximal set that contains no \( \beta \)-set.

To prove the second implication, we need to verify the following statement:

If a set (\( \alpha \)-set complement) is a maximal set that contains no \( \beta \)-set, then the set complement (\( \alpha \)-set) is a minimal set with at least one edge of every \( \beta \)-set.

To start the verification, we append any missing edge (\( \alpha \)-set edge) to the \( \alpha \)-set complement. This superset must contain at least one \( \beta \)-set which in turn contains the appended edge. Furthermore all other edges of the
β-set are a subset of the α-set complement. Hence at least one β-set contains only one edge (the appended edge) of the α-set, and this is true for every edge in the α-set. All β-sets that are not subsets of the superset (obtained by appending one edge to the α-set complement) must contain at least one additional edge that was not in the original α-set complement but instead in the α-set. Thus the α-set contains at least one edge of every β-set. In other words the α-set is indeed a minimal set containing one edge of every β-set.

One final theorem serves to tie together the four interrelationships, and the theorem can be stated as follows:

Theorem 22. If all possible β-sets (α-sets) are generated from the criterion that each set is a minimal set containing at least one edge of every α-set (β-set) and if no α-set (β-set) is an edge disjoint union of other α-sets, then the group of α-sets (β-sets) contains all and only those sets which are minimal sets that contain at least one edge of every β-set (α-set).

It should be noticed that the validity of this theorem requires that the α-sets (β-sets) also be restricted by the minimal qualifier. For example if the α-sets are any of the sets given in Table 9, then no α-set can be an edge disjoint union of other α-sets.

The proof of this theorem may or may not be apparent from the material already covered. In any event we simply state at this point that the adopted set notation and DeMorgan's theorem as employed in the previous chapter demonstrate the validity of the theorem when the α-sets (β-sets)
are indeed minimal sets for one reason or another. The following chapter illustrates two cases where the \( \alpha \)-sets (\( \beta \)-sets) are not restricted by the minimal qualifier.

By now the reader may think, and justifiably so, that Figure 5 is much ado about nothing. Thus we hasten to point out, as will be demonstrated later, that the figure offers many alternate ways to prove that some linkage exists between two groups of sets. Also the figure and the theorems illustrate that if one linkage is known to apply to two groups of sets then the other interrelationships also apply. Finally we point out that a set which is a minimal set that contains at least one edge of every \( \alpha \)-set complement is not necessarily a minimal set with an even number of edges of every \( \beta \)-set as might also be conjectured from Figure 3. For example, as is shown later, Figure 5 applies when the \( \alpha \)-sets are the 2-trees in Table 12 and the \( \beta \)-sets are the 3-cuts in Table 13. By referring to these two tables, it can be readily ascertained that \( ab\cdot c\cdot d \) is a minimal set of edges containing at least one edge of every 2-tree complement but not containing an even number of edges of every 3-cut. Thus, whereas the interrelationships that link tree and cotree to cut and circuit in Figure 3 are themselves interrelated, these linkages do not imply that the linkage at the bottom of the figure is a necessary consequence.

1. \((k+1)-cut\)

To demonstrate that the \((k+1)\)-cuts are interrelated with the \(k\)-trees, we state the following:
Theorem 23. An edge set is a \((k+1)\)-cut if and only if it is a minimal set with at least one edge of every \(k\)-tree.

To initiate the proof, we recall from Theorem 10 that opening a \((k+1)\)-cut disconnects the graph into \(k+1\) parts. According to Theorem 11 a \(k\)-tree is a minimal subset of edges that connects the \(n\) vertices into precisely \(k\) parts. Therefore a \((k+1)\)-cut necessarily contains at least one edge of every \(k\)-tree. Furthermore a \((k+1)\)-cut cannot contain at least two edges of every \(k\)-tree because opening such a set would disconnect the graph in at least \(k+2\) parts. Consequently a \((k+1)\)-cut is a minimal set of edges containing at least one edge of every \(k\)-tree.

To complete the proof, let us open the minimal set one edge at a time in any sequence. As long as at least one edge of the minimal set remains unopened, there is at least one \(k\)-tree remaining within the generated graph. As long as at least one \(k\)-tree exists, the generated graph cannot have more than \(k\) parts. Opening the final edge of the minimal set must finally disconnect the last of the \(k\)-trees and increase the number of parts by one from \(k\) to \(k+1\). Thus opening the minimal set disconnects the reference graph into \(k+1\) parts, and according to Theorem 10 such a set is a \((k+1)\)-cut.

An equivalent to Theorem 23 is the following:

Theorem 24. An edge set is a \((k+1)\)-cut if and only if it is a minimal set not contained in any \(k\)-tree complement.

The proof of this theorem results from combining Theorems 23 and 20.

Another equivalent theorem is the following:
Theorem 25. An edge set is a \((k+1)\)-cut if and only if the set complement is a maximal set that contains no \(k\)-trees.

The proof of this theorem results from combining Theorems 23 and 21.

2. \(k\)-tree

Combining Theorems 20, 21, 22, and 23 leads to the following three theorems:

Theorem 26. An edge set is a \(k\)-tree if and only if it is a minimal set with at least one edge of every \((k+1)\)-cut.

Theorem 27. An edge set is a \(k\)-tree if and only if it is a minimal set not contained in any \((k+1)\)-cut complement.

Theorem 28. An edge set is a \(k\)-tree if and only if the set complement is a maximal set that contains no \((k+1)\)-cuts.

A summary of these three theorems and the related three theorems of the previous section is given in Figure 6.

3. \((k+1)\)-circuit

To illustrate that the \((k+1)\)-circuits are linked directly to the \(k\)-cotrees, we stipulate the following:

Theorem 29. An edge set is a \((k+1)\)-circuit if and only if it is a minimal set not contained in any \(k\)-cotree complement.

The first portion of the proof necessitates referring back to Definitions 32 and 37 and Theorem 19, which show that \(k\)-circuits and \(k\)-cotree
Figure 6. Interrelationships between groups of k-trees and (k+1)-cuts
complements both reduce the nullity by k-1 via the process of shorting. Since the k-circuit is minimal while the k-cotree complement is maximal, the following must be true:

Corollary 5. Every k-circuit is a subset of at least one k-cotree complement.

On the other hand a (k+1)-circuit certainly cannot be a subset of any k-cotree complement because shorting the (k+1)-circuit reduces the nullity by k. Of course every proper subset of a (k+1)-circuit obtained by deleting one edge can only reduce the nullity by k-1 for otherwise the (k+1)-circuit would not be a minimal set. Therefore each such proper subset must be a subset of some k-cotree complement. In other words each (k+1)-circuit is a minimal set not contained in any k-cotree complement.

For the final portion of the proof, we show that every minimal set not contained in any k-cotree complement is indeed a (k+1)-circuit. Since the k-cotree complements encompass all maximal sets that reduce the nullity by k-1 through the process of shorting, the above minimal set by definition must reduce the nullity by more than k-1, in fact by precisely k. Furthermore every proper subset obtained by deleting any one edge of the minimal set cannot reduce the nullity by more than k-1 for otherwise it could not be a subset of some k-cotree complement. Therefore the defined minimal set is also a minimal set which when removed by shorting reduces the nullity by k, or a (k+1)-circuit.

By combining Theorems 20 and 29, we obtain the following:
Theorem 30. An edge set is a \((k+1)\)-circuit if and only if it is a minimal set with at least one edge of every \(k\)-cotree.

By combining Theorems 21 and 30, we get the following:

Theorem 31. An edge set is a \((k+1)\)-circuit if and only if the set complement is a maximal set that contains no \(k\)-cotrees.

4. \(k\)-cotree

Combining Theorems 20, 21, 22, and 30 leads to the following three theorems:

Theorem 32. An edge set is a \(k\)-cotree if and only if it is a minimal set with at least one edge of every \((k+1)\)-circuit.

Theorem 33. An edge set is a \(k\)-cotree if and only if it is a minimal set not contained in any \((k+1)\)-circuit complement.

Theorem 34. An edge set is a \(k\)-cotree if and only if the set complement is a maximal set that contains no \((k+1)\)-circuits.

A summary of these three theorems and the related three theorems of the previous section is given in Figure 7.

D. Boolean Functions

As in the first chapter, we are now in a position to further exploit the adopted set notation. Again the result is a compact symbolic representation of some of the interrelationships of the previous section. The ensuing Boolean functions illustrate an orderly method for obtaining one
Figure 7. Interrelationships between groups of k-cotrees and (k+1)-circuits
group of edge sets from a related group of edge sets. The Boolean func-
tions can also be interpreted as a symbolic listing of a sufficient set
of constraints that must simultaneously be fulfilled in the minimal sense
by a particular type of edge set.

Since all equations in this section are Boolean functions and pertain
only to the minimal sets in Table 9, we simply remind the reader that
products of sums are to be converted to sums of products and that sums
of products are to be simplified to sums of minimal products. Comparable
equations will not be developed for the maximal sets in Table 10. Of
course the maximal sets can be obtained by complementing appropriate mini-
mal sets according to the procedure given following Definition 17.

1. (K+l)-cut constraints

Combining Theorem 23 with Boolean algebra concepts results in the
following generalization of Theorem 6:

Theorem 35. The Boolean product of all not-k-trees of a graph
produces all (k+l)-cuts of the graph when converted to the sum
of minimal products.

Symbolically this theorem takes the form

\[ \bigoplus s^{k+1} = \Pi \overline{t^k} \]  \hspace{1cm} (26)

which is a generalization of Equation 20.

The preceding theorem and equation gives a set of (k+l)-cut con-
straints in terms of not-k-trees. The (k+l)-cuts can also be constrained
in terms of k-cuts. To illustrate, we start with the following:
Lemma 3. The union of any two distinct k-cuts, denoted by $S_i^k \cdot S_j^k$ where $i \neq j$, contains at least one $(k+1)$-cut.

In proving the lemma, we recall from Theorem 10 that opening a k-cut results in a generated graph of k parts. If we then proceed to open a second and different k-cut, at least one of the k parts must in turn be disconnected into at least two parts. Hence the generated graph finally contains at least $k+1$ parts, and the entire edge set that was opened must include a $(k+1)$-cut.

Of course the lemma does not apply when k equals one since there is only one distinct 1-cut, namely the null set. The lemma also does not apply for k equal to $n^+1$ because again there is only one $(n^+1)$-cut. Thus the lemma is applicable only if $n^+ \geq k \geq 2$.

Now we introduce another lemma as follows:

Lemma 4. Every $(k+1)$-cut is the union of some pair of distinct k-cuts.

As before the wording of the lemma automatically restricts the range of k to $n^+ \geq k \geq 2$.

To prove the lemma, we obtain a generated graph containing three or more parts by opening the $(k+1)$-cut. Then we select any two of the parts and delete all edges from the $(k+1)$-cut that connect these two parts. The remaining edge set must still be a k-cut. Now we proceed to select a different pair of parts and again delete all edges from the $(k+1)$-cut that connect these two parts. The remaining edge set is a second k-cut. Of course the process can be continued until there are no more distinct
pairs of parts. Such an approach is unnecessary for our purposes because the first and second $k$-cuts together contain all of the edges and only the edges of the $(k+1)$-cut.

We now combine the two lemmas into a theorem which is most easily represented by

$$\sum s^{k+1} = \sum_{i \neq j} s^k_i \cdot s^k_j, \ n \geq k > 2$$

(27)

The theorem itself can be worded as follows:

Theorem 36. An edge set is a $(k+1)$-cut if and only if it is a minimal set containing all edges of two distinct $k$-cuts.

2. **$k$-tree constraints**

From Theorem 26 we obtain the following generalization of Theorem 7:

Theorem 37. The Boolean product of all not-$(k+1)$-cuts of a graph produces all $k$-trees of the graph when converted to the sum of minimal products.

Symbolically this theorem is written as

$$\sum T^k = \Pi s^{k+1}$$

(28)

which is a generalization of Equation 22 and the result of applying DeMorgan's theorem to the negation of Equation 26.

The $(k+1)$-trees can also be constrained in terms of $k$-trees. Instead of trying to formulate a simple equation, we refer the reader to Theorem 11 and the resulting corollary, Corollary 1.
3. (K+1)-circuit constraints

From Theorem 30 we get the following generalization of Theorem 8:

Theorem 38. The Boolean product of all not-k-cotrees of a graph produces all (k+1)-circuits of the graph when converted to the sum of minimal products.

The Boolean form of this theorem is

\[ \sum c_{k+1} = \Pi k^k \]  

(29)

which is a generalization of Equation 23.

The (k+1)-circuits can also be constrained in terms of k-circuits, but first we need two lemmas comparable to those just developed for k-cuts. Thus we start with the following:

Lemma 5. The union of any two distinct k-circuits, denoted by \( c^k_i \cdot c^k_j \) where \( i \neq j \), contains at least one (k+1)-circuit.

To prove the lemma, we recall from the proof of Theorem 12 that shorting a k-circuit reduces the nullity by systematically shorting out k-1 independent circuit sets. If a second and different k-circuit is also removed by shorting, at least one additional independent circuit set must be affected. Hence the combined removal by shorting of two distinct k-circuits reduces the nullity by at least k, and the entire edge set that was shorted must include a (k+1)-circuit set.

The second lemma of interest is the following:
Lemma 6. Every \((k+1)\)-circuit is the union of some pair of distinct \(k\)-circuits.

Perhaps it should be pointed out again that the word distinct in both lemmas automatically serves to restrict the range of \(k\). In these two cases the restriction is \(n \geq k \geq 2\).

In proving the lemma, we first partition the \((k+1)\)-circuit as stipulated in Theorem 12 into two mutually exclusive, all inclusive subsets such that the first subset contains \(k\) edges and the second subset is a minimal set which does not contain a circuit set but which when appended by any edge of the first subset contains a circuit set. It should be apparent that \(k-1\) edges of the first subset along with some portion of the second subset comprise a \(k\)-circuit. All remaining edges of the \((k+1)\)-circuit not in the chosen \(k\)-circuit must be a minimal set which reduces the nullity by one if shorted after first shorting the \(k\)-circuit. Consequently all of these remaining edges must be included in some circuit set of the graph. In fact there is a circuit set that consists of only the remaining edges and edges in the chosen \(k\)-circuit. Using this particular circuit set as a nucleus, we can construct another \(k\)-circuit that includes only edges belonging to the \((k+1)\)-circuit and that includes all edges of the \((k+1)\)-circuit that are not in the first \(k\)-circuit. The result is two \(k\)-circuits that together contain all of the edges of and only the edges of the \((k+1)\)-circuit.

We now combine the two lemmas into a theorem comparable to Theorem 36 which is most easily represented by
\[ \sum c^{k+1} = \sum_{i \neq j} c_i^k \cdot c_j^k, \ n_n > k > 2 \quad (30) \]

The theorem itself can be written as follows:

Theorem 39. An edge set is a \((k+1)\)-circuit if and only if it is a minimal set containing all edges of two distinct \(k\)-circuits.

4. \( k\)-cotree constraints

From Theorem 32 we get the following generalization of Theorem 9:

Theorem 40. The Boolean product of all not- \((k+1)\)-circuits of a graph produces all \(k\)-cotrees of the graph when converted to the sum of minimal products.

The symbolic representation is

\[ \sum k^k = \Pi \overline{c}^{k+1} \quad (31) \]

which is a generalization of Equation 25 and the result of applying DeMorgan's theorem to the negation of Equation 29.

As was the case with the \((k+1)\)-trees, the \((k+1)\)-cotrees can also be constrained in terms of \(k\)-cotrees. For such constraints we refer the reader to Theorem 14 and the resulting corollary, Corollary 4.
IV. RELATED ASPECTS

The material to follow introduces additional terminology, edge sets, and interrelationships. Boolean functions are also formulated to describe certain edge set constraints and linkages between edge sets. The coverage given is in no sense exhaustive but instead serves to open the door to many of the special edge sets often considered in linear graph theory, particularly when used in the study of network analysis.

A. Abelian Groups

Two commutative groups of edge sets can be generated using the binary operation commonly referred to as the ring sum. Before delving into these groups, we define the binary operation as follows:

Definition 38. If $E_1$ and $E_2$ are any two edge sets of a graph, the ring sum of $E_1$ and $E_2$, $E_1 \oplus E_2$, is the edge set composed of all edges in $E_1$ or $E_2$ but not in both.

This binary operation is also referred to as the symmetric difference or as the module two sum, depending upon the notation being used to represent the sets.

1. Sews

From a number of theorems proved by Paul (10), we obtain the following:

Theorem 41. Any $n_r$ independent 2-cuts or edge disjoint unions of 2-cuts and all ring-sum combinations of these edge sets result
in an Abelian group of edge sets that contains the l-cut (null set), all possible 2-cuts, and all other k-cuts that are edge disjoint unions of 2-cuts.

This Abelian group contains \(2^n\) distinct edge sets \((10, 11)\), counting the null set, and each edge set is called a seg. This term originated with Reed \((11)\) and evolved from the word *segregation*, which was used in Reed's original definition.

If we ignore the null seg for the moment and if we refer to Definition 14(b), it becomes apparent that the not-segs comprise a sufficient set of 1-tree constraints. Symbolically these constraints can be written in the form

\[
\sum T^j = T \overline{\text{Segs}}
\]  

(32)

where again it is understood that the product of sums (not-segs) is to be converted to a sum of minimal products. Actually Equation 32 can also include the null seg if we write

\[
\overline{\text{Null seg}} = \overline{\emptyset} = 1
\]

(33)

where 1 represents the negation of the null seg which can be interpreted as the entire edge universe of the graph being considered.

It should be pointed out that the not-segs are not in general necessary 1-tree constraints. Therefore applying DeMorgan's theorem to the negation of Equation 32 does not lead to a summation of minimal products that necessarily includes all possible segs. Of course such an approach leads to a summation of minimal products that includes every set that is
also a 2-cut. In other words the not-2-cuts are necessary and sufficient 1-tree constraints while the not-segs are sufficient 1-tree constraints.

Since \( n_r \) independent 2-cuts can be used to generate all of the segs and since the segs can be used to generate all of the 1-trees, the reader might surmise that it is possible to generate all of the 1-trees directly from the \( n_r \) independent 2-cuts bypassing the Abelian group of segs. Such is the case. In fact there are two well known ways to accomplish the 1-tree generation directly. The most common method given in textbooks on linear graph theory is to denote the 2-cuts by using vectors, to use these vectors to form a cut-set matrix, and to then test for nonzero determinants. Another method, somewhat along the lines of Boolean algebra, is to use Wang algebra. For such an approach we refer the reader to work by Duffin (4) and by Maxwell and Cline (9).

2. Cirks

As mentioned by Paul (10), all of his development concerning cut sets could also be done using circuit sets. Such an approach leads to the following:

**Theorem 42.** Any \( n \) independent 2-circuits or edge disjoint unions of 2-circuits and all ring-sum combinations of these edge sets result in an Abelian group of edge sets that contains the 1-circuit (null set), all possible 2-circuits, and all other \( k \)-circuits that are edge disjoint unions of 2-circuits.

This Abelian group contains \( 2^n \) distinct edge sets, counting the null set, and each edge set is called a cirk. This term originated with Veerkamp
and Brown (16) and was originally defined in a different manner.

As in the previous section, the not-cirks comprise a sufficient set of constraints which can be used to generate the 1-cotrees by employing the Boolean function

\[ \sum K^1 = \text{Not Cirks} \] (34)

Of course the not-2-circuits are contained within the group of not-cirks and represent the necessary portion of the 1-cotree constraints.

All 1-cotrees can also be obtained directly from a set of \( n \) independent 2-circuits. One approach is to form a circuit matrix and test for nonzero determinants. Another approach is to employ Wang algebra.

B. Additional Constraints

The material in this section will deal with minimal edge sets that are further restricted by some additional constraint. For example we might be interested in all edge sets that are 2-cuts and that also separate two distinct vertices into different parts. Another example might be all edge sets that are 1-trees and that do or do not contain some specified edge. A list of examples could go on and on, and the following material only scratches the surface, so to speak, in an attempt to illustrate the myriad possibilities.

1. Basic 2-cuts

As we have seen, all 2-cuts can be obtained from a complete listing of all 1-trees or from any \( n \) independent 2-cuts or edge disjoint unions of 2-cuts. Suppose our interest is now constrained to basic 2-cuts which can
be defined as follows:

Definition 39. A basic 2-cut is a 2-cut which when opened places two distinct reference vertices in different parts.

This definition and the choice of terminology agrees with recent publications (20, 21). To denote the basic 2-cut, we employ the subscripted symbol $S_{I,J}^2$ where $I$ and $J$ represent the two reference vertices.

Before proceeding, it is convenient to introduce another term which can be defined as follows:

Definition 40. A chain is a minimal set of edges which when shorted coalesces two distinct reference vertices.

This particular edge set is also commonly referred to as a path between the two vertices because of the conceptual implications. From this alternate designation we adopt the symbol $P_{I,J}$ for the chain where $I$ and $J$ again denote the two vertices of interest.

It is evident that chains are related to 1-trees. To be more specific, each edge in a chain reduces the rank when shorted as does each edge in a 1-tree. Hence each chain is either a 1-tree or a proper subset of a least one 1-tree. Furthermore every 1-tree must include one and only one chain between a given pair of distinct vertices for otherwise the 1-tree would contain a 2-circuit contrary to Definitions 14(c), 14(g), and 14(i).

To show that the chains are also related to the 2-cuts, we submit the following, which is related to theorems and statements given by Hakimi (?):
Theorem 43. An edge set is a basic 2-cut with respect to two
distinct vertices if and only if it is a minimal set not con­
tained in the complement of any chain connecting the two refer­
ence vertices.

This theorem indicates that a basic 2-cut is related to a chain in the
same manner that a 2-cut is related to a 1-tree.

To begin the proof, we note that Definition 40 implies that chain
complements are maximal sets which when opened do not disconnect the two
vertices of interest. On the other hand basic 2-cuts are minimal sets
which when opened do disconnect the two vertices. Consequently basic
2-cuts cannot be subsets of any chain complement. Furthermore opening
all but one edge of a basic 2-cut does not disconnect the graph and there­
fore does not disconnect the two vertices. Hence all proper subsets of
a basic 2-cut must be subsets of at least one chain complement. In other
words a basic 2-cut is a minimal set not contained in the complement of
any chain that connects the two vertices in question.

To complete the proof, we recognize that an edge set not contained
in the complement of any chain connecting two distinct vertices must con­
tain at least one edge of every such chain. Thus opening such an edge set
must in turn open every chain connecting the vertices and thereby dis­
connects the vertices and the graph. From the minimal criterion we further
realize that each edge of the set in question is contained in at least one
chain which contains no other edges of the set. Therefore if all but one
edge of the set are opened, some path still exists between the two vertices,
and the vertices are not disconnected. Hence the defined edge set is a
minimal set which when opened disconnects the two vertices. Such an edge set is a basic 2-cut with respect to the vertices in question.

Combining this theorem with Theorems 20, 21, and 22 leads to five additional theorems linking chains and basic 2-cuts. Instead of listing the theorems, we refer to Figure 6 where $k$-tree can be changed to chain and $(k+1)$-cut can be changed to basic 2-cut.

Changing Figure 6 as mentioned above leads to two theorems regarding chains, basic 2-cuts, and Boolean functions. Instead of adding these theorems to an already formidable list, we simply give the symbolic representations which are

\[ \sum_{I,J}^{2} S_{I,J} = \Pi_{I,J}^{2} \tag{35} \]

and

\[ \sum_{I,J}^{2} P_{I,J} = \Pi_{I,J}^{2} \tag{36} \]

These two equations are related to some of the work done in switching theory. In fact the adopted set notation closely corresponds to that used by Seshu (12) and by Wing and Kim (20) and to what is commonly called the hindrance function.

To illustrate another facet of the chains, we use Equation 36 to write

\[ \prod_{j \neq I} (\sum_{I,J}^{2} P_{I,J}) = \prod_{j \neq I} (\Pi_{I,J}^{2} S_{I,J}) \]

\[ = \prod_{j \neq I} S_{I,J} \tag{37} \]
where the first subscript represents some fixed vertex and the second subscript ranges over all remaining vertices. Thus the above equation includes all 2-cuts that separate the chosen vertex from any other vertex of the graph. Since each and every 2-cut separates any chosen vertex from some of the other vertices, Equation 37 actually includes all possible 2-cuts and can be simplified to a product of all not-2-cuts. From Equation 28 we see that the product of not-2-cuts produces the 1-trees. Therefore we can write

\[ \sum T^1 = \prod_{j \notin I} (\sum P_{i,j}) \]  

(38)

where again it is understood that the product of sums is to be converted to the sum of minimal products. Thus we have the 1-trees defined in terms of another sufficient set of constraints.

We note in passing that the first subscript in Equation 38 could also be a free subscript; that is, we could utilize all sums of chains between every possible pair of vertices instead of between \( n_r \) pairs. The result would still simplify to the sum of trees, but the equation would contain a high degree of redundancy.

Let us momentarily return to the basic 2-cut, which has meaning only after two distinct reference vertices have been selected. If an edge connects the two vertices of interest, it is apparent that all basic 2-cuts referenced to these two vertices include the connecting edge. Also all 2-cuts which do not contain the connecting edge cannot be basic 2-cuts referenced to the selected vertices. Thus in many graphs we can reference basic 2-cuts to an edge, which in effect references the basic
2-cuts to the vertices incident to that edge. This change in viewpoint provides an alternate use for Equation 35; that is, Equation 35 can be employed to generate all 2-cuts that contain some specified edge. In a similar vein Equation 36 can be used to generate all chains that connect the same two vertices that are connected by some specified edge. By appending the reference edge to each of these chains, we also have all 2-circuits that contain the reference edge.

2. **Basic 2-trees**

Just as there are special 2-cuts which may be of interest in certain problems there are special 2-trees (14). These 2-trees are also referenced to a pair of distinct vertices and can be defined as follows:

Definition 41(a). A basic 2-tree is a 2-tree which when shortened does not coalesce the two distinct reference vertices.

Perhaps the following graph definition provides more insight:

Definition 41(b). A basic 2-tree graph is a generated graph obtained by opening any 2-tree complement which results in the two distinct reference vertices being in different parts.

To represent the basic 2-trees, we use $T_{I,J}^2$ where $I$ and $J$ again denote the two vertices of interest. This symbolism closely agrees to the current literature with the exception of the capitalized subscripts.

A widely used method for obtaining a complete set of basic 2-trees is to coalesce the reference vertices and then determine the 1-trees of the resulting graph. It is convenient to recognize that if the complete
listing of 1-trees of the original graph is available and if an edge connects the two reference vertices, the basic 2-trees can be obtained by deleting the connecting edge from each 1-tree which contains that edge. Those 1-trees that do not contain the connecting edge are not considered in this approach. For example all \( T_{1,4}^2 \) of Figure 1 are readily obtained by deleting edge a from the first eight 1-trees in Table 5.

As is the case with basic 2-cuts, we can also reference basic 2-trees to an edge. In effect this references the basic 2-trees to the associated vertices. This alternate manner of referencing leads to the following:

Theorem 44. An edge set is a basic 2-tree referenced to a specific edge if and only if it is a minimal set with at least one edge of every 2-cut that does not contain the reference edge.

To start the proof, we note that the basic 2-trees referenced to an edge are actually the 1-trees of the generated graph that is obtained by shorting the reference edge. We further recognize that shorting an edge in no way affects any 2-cut that does not contain that edge; that is, such a 2-cut is still a 2-cut of the generated graph. Finally we point out that each 2-cut of the generated graph is a 2-cut of the original graph and is a 2-cut that certainly does not contain the shorted reference edge. In other words an edge set is a 2-cut of the generated graph if and only if it is a 2-cut of the original graph and it does not contain the shorted edge. To complete the proof, we merely note that the 1-trees of the generated graph are minimal sets with at least one edge
of every 2-cut of the generated graph.

As before this theorem immediately leads to five other theorems that link the basic 2-trees to what we call the nonbasic 2-cuts where the referencing is with respect to a specific edge, or a pair of distinct vertices connected by an edge. We hasten to point out that Theorem 44 and the five associated theorems do not hold in general if the referencing is with respect to a pair of distinct vertices which are not connected by an edge as can readily be shown by counterexample. So that as it may, we neither list the additional theorems nor illustrate them with a figure. Instead we refer the reader to Figure 6 which is apropos if k-tree is replaced by basic 2-tree, (k+1)-cut is replaced by nonbasic 2-cut, and the referencing is understood to be with respect to an edge.

Two additional theorems involving Boolean functions evolve from Theorem 44. For sake of brevity we list the theorems symbolically as

\[ \sum T_{I,J}^2 = \Pi S_{e_{I,J}}^2 \]  \hspace{1cm} (39)  

and

\[ \sum S_{e_{I,J}}^2 = \Pi T_{I,J}^2 \]  \hspace{1cm} (40)  

The new subscripting represents the edge referencing and also denotes that the edge sets do not contain the reference edge. Of course Equation 40 does not apply if no edge connects the two reference vertices.

3. Basic 2-circuits
In order to define a basic 2-circuit in a reasonably simple and useful manner, some preliminary definitions are beneficial. First we define a special subset of graphs as follows:

Definition 42. A planar graph is a graph that can be geometrically realized in 2-dimensional Euclidean space with curves having no common points, except for points of $V$.

Second we define the areas of the plane marked off by the curves as follows:

Definition 43. A region is any open area of the plane which contains no points belonging to any of the curves.

These regions are also called windows and sometimes meshes. Since the number of regions is one greater than the nullity of the planar graph, the regions serve as a convenient means for quickly determining the nullity of a planar graph.

We are now in a position to tender the following:

Definition 43. A basic 2-circuit of a planar graph is a 2-circuit that encompasses one and only one of two distinct reference regions.

This particular description of a basic 2-circuit provides some conceptual insight but does not demonstrate the close relationship to the basic 2-cut. To distinguish these constrained 2-circuits, we use the symbol $c_{1,j}^2$ where the subscripts denote the two reference regions.
Before continuing, we introduce another term patterned after Definition 40 as follows:

Definition 44. A circuit chain of a planar graph is a minimal set of edges which when opened identifies, or coalesces, two distinct reference regions.

This particular edge set could also be referred to as a region chain. Because of the connection to regions, the circuit chain is denoted by $R_{i,j}$ in the material to follow with the subscripts representing the two regions of interest.

Circuit chains are related to 1-cotrees in the same manner that chains are related to 1-trees. In fact a circuit chain is either a 1-cotree or a proper subset of at least one 1-cotree. Furthermore every 1-cotree of a planar graph must include one and only one circuit chain between a given pair of distinct regions for otherwise the 1-cotree would contain a 2-cut contrary to Definitions 16(c) and (g).

To demonstrate that the circuit chains are also related to the 2-circuits, we give the following:

Theorem 45. An edge set is a circuit chain with respect to two distinct regions of a planar graph if and only if it is a minimal set with at least one edge of every basic 2-circuit referenced to the same two regions.

This theorem indicates that circuit chains and basic 2-circuits are linked in a manner comparable to the linkage between chains and basic 2-cuts.
To prove the first implication, we recognize from Definition 43 that the basic 2-circuits in effect isolate the two reference regions from each other. Hence a circuit chain must contain at least one edge of every basic 2-circuit referenced to the two regions in question. Furthermore since a circuit chain is a minimal set that coalesces two regions, opening all edges but one does not coalesce the regions. In other words the regions are still separated by some basic 2-circuit. Thus each edge in a circuit chain is contained in at least one basic 2-circuit that contains no other edges of the circuit chain. Consequently a circuit chain is a minimal set with at least one edge of every basic 2-circuit.

To prove the second implication, we note that opening an edge set that contains at least one edge of every basic 2-circuit certainly coalesces the two regions that are separated by the basic 2-circuits. Since such a set is also defined as being minimal, opening all but one edge leaves at least one basic 2-circuit intact. Thus such a defined set is a minimal set that coalesces two distinct reference regions, which is of course the definition of a circuit chain.

Combining this theorem with Theorems 20, 21, and 22 again leads to five additional theorems describing the linkage that exists between circuit chains and basic 2-circuits. Instead of listing the theorems, we refer to Figure 7 where k-cotree can be changed to circuit chain and (k+1)-circuit can be changed to basic 2-circuit.

Changing Figure 7 as mentioned in the preceding paragraph and applying the concepts of Boolean algebra yields

\[
\sum C_{i,j}^2 = \prod F_{i,j}
\] (41)
and
\[ \sum r_{I,J} = \prod c_{I,J} \]  \hspace{1cm} (42)

which are comparable to Equations 35 and 36. Using an approach comparable
to that which led to Equation 38 gives the Boolean function
\[ \sum R = \prod_{j \neq I} (\sum r_{I,J}) \]  \hspace{1cm} (43)

which describes the 1-cotrees of a planar graph in terms of circuit chains.

As a matter of interest, if an edge is incident to both reference
regions, it should be apparent that all basic 2-circuits referenced to
these two regions include this incident edge. Furthermore all 2-circuits
which do not contain the incidence edge cannot be basic 2-circuits ref­
erenced to the chosen regions. Hence in many planar graphs we can ref­
ERENCE basic 2-circuits to an edge which in effect references the basic
2-circuits to the regions incident to that edge. Using this viewpoint,
Equation 41 can be employed to generate all 2-circuits that contain some
specified edge. In a like manner Equation 42 can be used to generate all
circuit chains that link the same two regions that are incident to some
specified edge of a planar graph. By appending each circuit chain with
the reference edge, we obtain all 2-cuts that contain the reference edge.

4. **Basic 2-cotrees**

We define a special set of 2-cotrees in a manner comparable to
Definition 41(a) as follows:

**Definition 45.** A basic 2-cotree of a planar graph is a 2-cotree
which when opened does not coalesce the two distinct reference regions.

This special 2-cotree can be symbolized by $k^2_{I,J}$, where $I$ and $J$ denote the two regions of interest.

To obtain a complete set of basic 2-cotrees, it is often possible to first open an edge that is incident to the two regions in question thereby coalescing the regions and then obtain all 1-cotrees of this generated subgraph. If no edge is incident to both regions, it might still be possible to coalesce the two regions by splitting some vertex into two vertices and then obtain all 1-cotrees of the resulting graph. A final alternative is available if the complete listing of 1-cotrees of the original graph is available and if an edge is incident to the two reference regions. The approach is to simply delete the incident edge from each 1-cotree which contains that edge. For example all basic 2-cotrees of Figure 1 referenced to the two regions incident to edge $a$ are readily obtained by deleting edge $a$ from the first five 1-cotrees in Table 7.

As in the previous three sections, we can also reference the basic 2-cotrees to a specific edge. Such an approach leads to the elimination of the planar constraint which heretofore applied to all of the discussion concerning basic 2-cotrees. This particular approach is essentially what was used in the example mentioned in the previous paragraph.

Referencing the basic 2-cotrees to a specific edge leads to the following:

**Theorem 46.** An edge set is a basic 2-cotree referenced to a specific edge if and only if it is a minimal set with at least
one edge of every 2-circuit that does not contain the reference edge.

The proof of this theorem can be obtained by using the proof of Theorem 44. Simply change tree to cotree, shorting to opening, and cut to circuit in the proof.

We do not list the five related theorems nor illustrate them with a figure but instead refer the reader to Figure 7 which can be made relevant by changing the edge set terminology. We do however give two Boolean functions which are

\[ \sum_{2} x_{I,J} = \prod_{e_{I,J}} 2 \]  \hspace{1cm} (44)

and

\[ \sum_{2} c_{e_{I,J}} = \prod_{x_{I,J}} 2 \]  \hspace{1cm} (45)

Again the \[ e_{I,J} \] subscript represents the edge referencing and the fact that none of the 2-circuits contain the reference edge. For this reason the 2-circuits in Equations 44 and 45 could be referred to as nonbasic circuits.
V. PROJECTIONS

The preceding material is certainly not an exhaustive treatment of linear graphs, edge sets, and Boolean functions. In fact the presentation thus far has probably created more questions than it has answered, particularly regarding the many facets of linear graph theory that are in one way or another related to the quantities that have been herein defined. Thus we have reached that stage of development which is appropriately described by the following question: Where can we go from here?

The avenues open for investigation are numerous, and their directions vary. Nevertheless we can separate the possibilities into two broad categories. We can either extend the basic foundation that has been laid or else build on the foundation by using it as a stepping stone towards the solution of related problems.

A. Extensions

The following material briefly lists and discusses some extensions of the basic foundation. Additional terminology is used, and references are listed which provide explicit definitions and which may yield additional information pertinent to the extension being discussed.

1. Isomorphism

A graph which contains no circuit edges is defined to within a 2-isomorphism by either the set of 2-cuts or the set of 1-trees (14, 18). A graph which contains no cut edges is defined to within a 2-isomorphism by either the set of 2-circuits or the set of 1-cotrees. An extension, that may prove to be informative, is to ascertain to what degree a graph
is defined by other groups of k-sets in the hierarchy. For example the single 5-cut in Table 15 defines a connected graph having the following properties:

(a) Seven edges which are not circuit edges.
(b) Rank of four.
(c) Five vertices.
(d) Nullity of three plus the number of circuit edges.

2. Matrix rank

A complete 2-cut matrix composed of row vectors each of which represents a 2-cut of a reference graph has a rank equal to the rank of the graph. A complete 2-circuit matrix of similar composition has a rank equal to the nullity of the graph. A possible extension is to determine the rank of other matrices that can be formed from other groups of edge sets in the hierarchy. For example Hakimi (6) has developed some theorems on the rank of a modified l-tree matrix.

Additional knowledge about the rank of various matrices could provide necessary constraints that must be fulfilled in order for corresponding groups of sets to be realizable. Such knowledge might also yield some clues regarding the generation of complete groups of k-sets from independent groups of k-sets or edge disjoint unions of k-sets.

3. Basic k-sets

The definition of basic 2-circuits was deliberately restricted to planar graphs. Basic 2-circuits could be extended to nonplanar graphs by generalizing the definition. Another possibility is to generalize the def-
inition of circuit chains and thereby extend the basic 2-circuits.

Other basic k-sets can also be defined. One such possibility is the basic 3-tree, which proves useful in the determination of the short-circuit admittance functions of certain two-port networks (14).

4. **Special linkage**

The 2-cuts and 2-circuits are always interrelated in the manner shown at the bottom of Figure 3. Since this linkage does not hold in general, it may be instructive to ascertain just what factors determine the existence of this particular interrelationship. It is surmised that the linkage is directly related to the fact that all 2-cuts and 2-circuits can be generated from independent sets using ring-sum combinations.

5. **Complete graphs**

When linear graphs are restricted to certain categories, it appears that many special interrelationships occur. As an example the following conjectures are given for complete graphs:

**Conjecture 1.** The Boolean product of the negation of all vertex 2-cuts of a complete graph produces $n_V$ 1-trees and all 2-trees with an isolated vertex that are not proper subsets of these 1-trees when converted to a sum of minimal products.

**Conjecture 2.** The Boolean product of the negation of all nonvertex 2-cuts of a complete graph produces all 2-trees without an isolated vertex when converted to the sum of minimal products.
The first conjecture does not apply to the trivial case of a complete graph having only two vertices. As a matter of fact, the formidable wording of the conjecture is a result of including the complete graph having three vertices. For complete graphs having more than three vertices, it appears that the vertex 2-cuts produce all 2-trees that have an isolated vertex.

B. Applications

Perhaps the most important projections are the applications of the fundamentals that have been presented. The following material illustrates a few possibilities.

1. Realizability

A great deal of work has been done concerning the realizability of matrices that represent various groups of edge sets, particularly 1-trees, 2-cuts, and 2-circuits. If two such matrices exist such that each matrix can be generated uniquely from the other matrix and if the necessary and sufficient conditions of realizability are known for one of the matrices, then the other matrix can also be tested for realizability. Such is the case for the 1-tree matrix and the 2-cut matrix (14, 17) where the realizability of a 1-tree matrix can be checked by generating the associated 2-cut matrix. Of course this approach is often long and tedious, and it behooves us to try to develop necessary and sufficient conditions that directly describe the realizability of a 1-tree matrix. For example Hakimi (6) has developed some necessary realizability conditions for the 1-tree matrix and a modified 1-tree matrix. Hopefully increased know-
ledge of the interrelationships between k-trees and k-cuts coupled with the known realizability conditions for a 2-cut matrix can provide further assistance in this direction.

2. Computer programming

In illustrating applications of the basic material to computer programming, we limit the following discussion to obtaining the 1-trees of a linear graph. The simplest approach is to program an indexing routine that provides all combinations of \( n_e \) edges taken \( n_e \) at a time. This can be easily accomplished in Fortran by using nested DO statements. Each of these edge sets is then tested to ascertain if it fulfills all constraints described by the not-2-cuts. This can be done in Fortran by using a sequence of IF statements.

In order to decrease the needed computer time, the most restrictive constraints should be tested first; that is, the not-2-cuts with the fewest number of edges should be utilized in the beginning of the routine that tests each individual edge set. The running efficiency can usually be further increased if the indexing routine that generates all combinations to be tested also utilizes the most restrictive constraints so that many edge sets that obviously cannot be 1-trees are not even generated. For example if a graph contains ten edges, if each tree contains four edges, and if the edges are numbered such that the first and second edge comprise a 2-cut, then the indexing routine only needs to generate all combinations of ten edges taken four at a time that contain the first or second edge. Clearly no other edge sets can possibly be 1-trees.

Thus far the discussion has assumed the availability of all 2-cuts.
Since determining the 2-cuts could be a sizeable problem in itself, it would probably be advantageous to write a subroutine that would generate the segs from an independent set of 2-cuts or edge disjoint unions of 2-cuts. These segs could then be used as 1-tree constraints.

Another approach is to use chains for a sufficient set of 1-tree constraints as indicated in Equation 38. From some conjectures to follow, it might even be feasible to generate all 1-trees and a basic set of 2-trees from a set of chains connecting two reference vertices.

An alternate approach to obtaining the 1-trees is to program the conversion of the product of sums into a sum of minimal products. Such a process might be carried out using the designation numbers described by Ledley (8).

3. Synthesis

To illustrate the application of a few of the basic concepts to network synthesis, we employ an example by Seshu (13) which involves using linear graph theory to synthesize the driving-point admittance

\[ Y(s) = \frac{(3 + 10/s + 4/s^2)}{(2 + 4/s)} \]  

(46)

Since it is known that the numerator of a driving-point admittance is the sum of all 1-tree admittance products and that the denominator is the sum of all basic 2-tree admittance products referenced to the two input vertices, Seshu's approach was to synthesize the admittance after obtaining a realizable set of 1-trees and a corresponding set of basic 2-trees. In order to bypass the realizability problems of k-trees, we make use of the concepts illustrated in Equations 22 and 39 and tackle the synthesis prob-
lem using realizable 2-cuts and associated nonbasic 2-cuts. To be more specific, we recognize that the denominator of Equation 46 can either result from the negation of one nonbasic 2-cut or from a Boolean product of the negation of a set of nonbasic 2-cuts where the edge notation is understood to represent the admittance function of the edge. For example the simplest possibility is that the denominator resulted from the negation of one nonbasic 2-cut

$$S_1 = (\overline{a \cdot b}) = a + b \quad (47)$$

where edge $a$ of the linear graph represents a 2-mho conductor, and edge $b$ represents a $1/4$-henry inductor.

The linear graph representing the synthesized network must have at least one other edge that appears in all remaining 2-cuts. Again the simplest possibility that comes to mind is that the remaining 2-cuts are

$$S_2 = \overline{a} \cdot \overline{c} \quad (48)$$

and

$$S_3 = \overline{b} \cdot \overline{c} \quad (49)$$

which are realized by the complete graph in Figure 4.

It remains to be determined as to whether or not a network represented by the linear graph in Figure 4 is capable of synthesizing the stipulated admittance function between the vertices incident to the reference edge, edge $c$, when edges $a$ and $b$ are a conductor and inductor respectively. We can quickly check this possibility by formulating the
admittance across edge c as

$$Y_1(s) = \frac{(ab + ac + bc)}{(a + b)}$$  \hspace{1cm} (50)

and substituting in the admittance functions of edges a and b to give

$$Y_1(s) = \frac{(8/s + 2c + 4c/s)}{(2 + 4/s)}$$  \hspace{1cm} (51)

Comparison to Equation 46 readily shows that replacing edge c with a simple admittance function will not produce the desired numerator. Hence we can conclude that the denominator of Equation 46 might result from two or more nonbasic 2-cuts or from one nonbasic 2-cut that has more than two edges.

In fact, as Seshu’s answer shows, one possibility is to use a nonbasic 2-cut of three edges representing two 1-mho conductors and one 1/4-henry inductor respectively.

As a matter of interest, Equation 51 is still of some value. If edge c represents a 1-henry inductor, we have realized the admittance

$$Y_1(s) = \frac{(10/s + 4/s^2)}{(2 + 4/s)}$$  \hspace{1cm} (52)

The remainder of the problem then boils down to synthesizing

$$Y_2(s) = \frac{3}{(2 + 4/s)}$$  \hspace{1cm} (53)

which is simply a conductor and capacitor series combination in parallel with edge c, the 1-henry inductor.

4. **Switching functions**

Before beginning this discussion concerning switching functions, we caution the reader that all graphs referred to are connected and contain
no path-isolated subgraphs as defined by Wing and Kim (20). The necessity of these two restrictions and the fact that they are more a matter of convenience than of consequence should become apparent as the discussion progresses.

In an article by Seshu (12), it is stated that an unsolved problem is the determination of the 1-trees and a set of basic 2-trees of the linear graph which synthesizes a given switching function without first determining the linear graph. In other words the problem is to be able to generate all of the 1-trees and the set of basic 2-trees referenced to the same two vertices as the given switching function without the necessity of actually generating the associated linear graph.

The first clue towards a solution of the above problem is that the switching function is a Boolean sum of all chains connecting two reference vertices. Thus the negation of the switching function becomes a Boolean product of not-chains, which, according to Equation 35, can be used to generate all basic 2-cuts referenced to the same pair of vertices. One way to continue with the solution is to generate all of the segs from these basic 2-cuts. To this end we offer the following conjecture:

Conjecture 3. The basic 2-cuts and all ring-sum combinations of basic 2-cuts include all segs of a connected graph that contains no path-isolated subgraphs.

Assuming the validity of this conjecture, the next step is to generate the segs which in turn can be used to generate the 1-trees. Of course some conservation of effort can be obtained by first selecting an independent
set of basic 2-cuts before listing the ring-sum combinations. When generating the trees, another reduction in effort can be gained by using only those segments which are 2-cuts.

The remainder of the solution is decidedly simple if the switching function contains at least one chain composed of only one edge. When such is the case, all 2-cuts that do not contain the chain edge are used in Equation 39 to generate the basic 2-trees that are referenced to the same two vertices as the switching function.

As is often the case, the switching function does not contain a chain edge. Since Equation 39 is then no longer applicable, we tender another conjecture as follows:

Conjecture 4. The Boolean product of the negation of all segments that are not basic 2-cuts produces the basic 2-trees, referenced to the same vertices as the switching function, when converted to a sum of minimal products.

Assuming the validity of this conjecture, the basic 2-trees can be generated without generating the associated graph.

To assuage some doubts that may have arisen concerning the conjectures and the proposed solution, it should be pointed out that both examples given by Seshu can be worked in the manner prescribed. Furthermore neither example contains a chain edge, and the first example

\[ F = ab + de + ace + bcd \]  \hspace{1cm} (54)

is neither difficult nor very time consuming.
A side aspect that evolves from the proposed solution is that the approach for generating 1-trees indicated in Equation 38 is exceedingly redundant, at least for graphs that contain no path-isolated subgraphs. Supposedly one only needs a set of chains referenced to one pair of vertices in order to generate the 1-trees. In fact the entire hierarchy can be generated if the edges in the set of chains comprise the entire edge set of interest.
VI. SUMMARY

The four dominate terms in this dissertation are cut, tree, circuit, and cotree. Each of these terms is used in defining special edges, sets, and graphs. Many of the resulting definitions can be worded in a variety of ways, depending on the envisioned application and on the desired viewpoint. As has been demonstrated, one consistent approach to the wording is based on using the key words minimal or maximal, rank or nullity, and opening or shorting. The result is a set of definitions that are concise and consistent, that are similar in form but clearly different in scope, that are easily generalized, and that are readily symbolized.

The conciseness leads to a simple tabular description of the major edge sets. The consistency helps in symbolizing and generating the edge sets.

The similarities lead to pictorial illustrations showing the numerous linkages that exist between groups of edge sets and to a better realization and understanding of the redundancy that exists in linear graph theory. The differences lead to alternatives in viewpoint and application.

The ability to generalize leads to the hierarchy of edge sets. This hierarchy is composed of two parallel divisions based on rank and nullity. The only direct coupling of these two divisions appears to be between the 2-cuts, 1-trees, 2-circuits, and 1-cotrees.

The adaptability to symbolism leads to the use of Boolean algebra. This in turn yields a simple and compact means for representing many theorems by Boolean functions. These Boolean functions then provide an
orderly means for solving certain problems.

The overall result is a reasonably consistent terminology that is applicable to many problems and examples discussed in the literature on graph theory. The terminology is also readily extended to other edge sets such as chains and to other combinations of edge sets such as switching functions. Of course much more work needs to be done in extending and applying the fundamentals. Eventually a useful standardization should evolve.
VII. BIBLIOGRAPHY


VIII. ACKNOWLEDGMENTS

The author would like to thank his major professor, Dr. H. W. Hale, for the able assistance that was rendered during the preparation of this dissertation. The author also gratefully acknowledges the support given by the South Dakota Board of Regents in the form of a sabbatical leave and by the National Science Foundation in the form of a Science Faculty Fellowship. Special thanks are also extended to the author's wife, Loreli, and to Mrs. Kathy Lucas for their combined efforts in the preparation of this manuscript.