Formal Procedures for Connecting Terminals with a Minimum Total Wire Length*

H. LOBERMAN AND A. WEINBERGER

National Bureau of Standards, Washington, D. C.

In the construction of a digital computer in which high-frequency circuitry is used, it is desirable and often necessary when making connections between terminals to minimize the total wire length in order to reduce the capacitance and delay-line effects of long wire leads. Presented here are two methods for systematically selecting the shortest connections from a list of possible connections to obtain a minimum total wire length.

The special problem considered here is the following: Given a number of terminals, fixed in space, which must be electrically connected together, what procedure will provide the minimum wire length? A *proper pattern* of connections is one in which there exists one and only one path, either direct or through other terminals, from each terminal to every other terminal and in which there are no loops created by redundant connections. Figure 1 shows two possible proper patterns for connecting four terminals.

Problems of this type have been considered in topological areas of mathematics, more particularly in the theory of graphs [1].

In accordance with the prevailing terminology, the following definitions will be used hereafter in this paper.

A terminal either connected or unconnected will be referred to as a *node*. The direct connection between two nodes is a *branch*, the magnitude of which is the distance between the nodes. A *path* between two nodes is a connection consisting of one or more branches.

A *graph* is a structure of nodes connected pairwise by one or more branches. A *tree* is a graph having one and only one path between every two nodes. It has previously been referred to as a proper pattern. A minimum proper pattern, i.e., a proper pattern where the sum of the wire lengths is a minimum, will be called a *minimum tree*. A *subtree* is a tree comprising k of n nodes where $k < n$.

To connect electrically n nodes into a tree, exactly $(n - 1)$ branches are necessary [1]. If more than $(n - 1)$ branches are used there will be redundant connections and loops will be formed. If less than $(n - 1)$ branches are used, not all of the nodes will be interconnected. If exactly $(n - 1)$ branches are used, but incorrectly, both loops and unconnected nodes result.

To produce a minimum tree, it is conceivably possible to investigate all of the possible trees that exist for n nodes. It can be shown [2, 3] that the number of trees for n nodes is $n^{(n-2)}$, which increases rapidly as the number of nodes increases. Even with the use of a high-speed computer, a check of all these trees would be tedious for more than a moderate number of nodes. Therefore it would be more

* Received June, 1957.

advantageous to combine systematically only the shortest branches until a minimum tree is obtained [4][1].

The total number of branches between each node and every other node is $n(n - 1)/2$, which is simply the number of ways that n things can be taken two at a time. In general this number will be considerably smaller than the total number of trees.

A procedure (Procedure A) which will give a minimum tree is described by means of an example as follows:

1. Consider the six nodes as shown in figure 2. The branch between any two nodes, e.g., 1 and 2, will simply be written as 12. The number 12 is only the designation of the branch from 1 to 2; it does not represent a magnitude. Also, no direction is assigned to these

FIG. 1

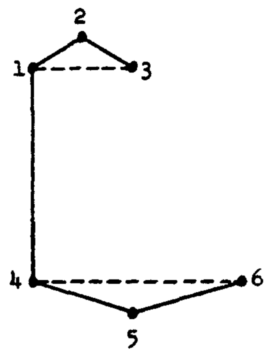

FIG. 2

branches. The fifteen branches between the nodes are: 12, 13, 14, 15, 16, 23, 24, 25, 26, 34, 35, 36, 45, 46, 56.

2. The branches are first sorted monotonically according to their length, starting with the shortest branch. Thus the sorted list might appear as: 12, 23, 13, 45, 56, 46, 14, 34, 36, 15, 16, 24, 35, 26, 25.

3. Branches are chosen from the beginning of the list. Thus branch 12 is made and a subtree consisting of nodes 1 and 2 is formed and recorded.

4. Branch 23 is now considered. A search through the subtree which has been formed above shows that only *one* of the nodes, i.e., node 2, appears in the subtree. Therefore the branch 23 can be made and node 3 is added to the subtree which now consists of nodes 1, 2, and 3.

[1] This reference was discovered by the present authors after their procedures had been formulated. It is seen that the procedures presented here and Kruskal's "constructions" are identical. However, it is felt that the more detailed implementation and general proofs of the procedures justify this paper.

5. The next shortest branch is 13. By searching through the subtree which has been formed it is found that *both* nodes 1 and 3 appear in the same subtree. Therefore this branch cannot be constructed since it is redundant and would form a loop.

6. The next branch is 45. Neither of these nodes appears in the subtree previously formed, therefore a new subtree consisting of nodes 4 and 5 is started.

7. Next, branch 56 can be made since only *one* of its nodes appears in a subtree. Node 6 is added to the subtree containing nodes 4 and 5. Branch 46 is rejected because it would form a loop since both of its nodes appear in the second subtree.

8. Branch 14 is considered. Although both nodes have been listed before, they appear in different unconnected subtrees. This branch joins the two subtrees into one, consisting of nodes 1, 2, 3, 4, 5, and 6. All of the nodes are now connected as shown in figure 2 and the minimum tree is constructed. The dotted lines indicate branches which were considered but not constructed.

In general, the rules of procedure A for selecting the necessary branches from the sorted sequence of branches are as follows:

Examinè the branches sequentially, beginning with the shortest branch. For each branch examine the two nodes which constitute its terminals. The first branch is always used, and its two nodes are recorded as belonging to a subtree. One of four possible conditions are recognized for each succeeding branch which is considered.

Condition 1. Neither of the two nodes is present in a subtree. Therefore the branch is made. The two nodes are recorded as constituting another new subtree.

Condition 2. Only one of the nodes is present in a subtree. Therefore the branch is made. The new node is added to the subtree containing the other node of this branch.

Condition 3. Each of the two nodes is present in a different subtree. Therefore the branch is made. The two different subtrees, each containing one of the nodes, are combined into a single subtree.

Condition 4. Both nodes are present in the same subtree. Therefore the branch is *not* made.

The completion of the process can be determined in either of two ways: (1) by checking to see whether all of the n nodes are in one tree, or (2) by keeping tally of the number of branches made until that number reaches $(n - 1)$.

The above procedure essentially involves connecting each node to its closest neighbor so that subtrees consisting of these nodes are formed. After this, the subtrees are joined with the shortest branches between subtrees.

An alternate procedure (Procedure B) which is a slight variation of procedure A and which involves the formation and investigation of only *one* subtree may be described as follows:

1. The branches are sorted according to their length as in procedure A and as before are: 12, 23, 13, 45, 56, 46, 14, 34, 36, 15, 16, 24, 35, 26, 25.

2. Branches 12 and 23 are made and the subtree consisting of nodes 1, 2, and 3 is formed. Branch 13 is rejected for the same reason as in procedure A.

3. Branches 45, 56, and 46 are passed over since neither of their nodes appear in the subtree formed above.

4. Branch 14 is made and node 4 is added to the subtree.

5. The branches which have been passed over are now rechecked to see whether one of

their nodes is in the subtree. Thus branches 45 and 56 can now be made and the process is complete

In general the rules of procedure B for selecting the necessary branches from the sorted sequence of branches are as follows:

The branches are examined sequentially unless otherwise specified. Beginning with the shortest branch, the two nodes which constitute its terminals are examined. The first branch is always made and its two nodes are recorded as belonging to the single subtree of nodes of Procedure B. One of three possible conditions are recognized for each succeeding branch which is considered:

Condition 1. Neither of the two nodes is present in the subtree. Therefore, the branch is *not* made at this time. The succeeding branches are then examined in sequence. After the first subsequent branch is made which fulfills the requirements of condition 2 (which follows), the examination reverts back to the first branch which has not yet been made because of condition 1.

Condition 2. Only one of the two nodes is present in the subtree. Therefore the branch is made and the other node is added to the subtree. Next it is determined whether there are any branches in previous positions in the sequence which met condition 1 (above) and have not yet been made. If so, the examination of the branches reverts back to the first of such branches. If not, the examination of the branches proceeds in normal sequence.

Condition 3. Both nodes are present in the subtree. Therefore, the branch is *not* made.

The process can be terminated by checking either one of the following states:

1. All n nodes are in the tree.
2. $(n - 1)$ branches have been made.

In procedure A several disconnected subtrees are usually formed which must be rechecked as new branches are considered. In procedure B only *one* subtree is investigated but some of the branches which have been passed over must be rechecked as new nodes are added to the single graph. It can be shown that these procedures each produce the same minimum tree. (See appendix I.) The procedures are in fact equivalent in that the same branches are made in both cases provided the same sorted sequence is used, even though alternate branches of equal length are available. However, the branches are made in different orders in the two procedures.

Figures 3 and 4 show simple flow diagrams of the two procedures.

In the procedures described the following considerations can be noted:

1. From each node the branch to its closest neighboring node is always made. (See appendix I, theorem 2, for $k = 1$). Whenever a node first appears on the list, the corresponding branch is the shortest to that node and can always be made without forming a loop since it requires at least two branches to a node to form a loop.

2. Although there are $n(n - 1)/2$ branches between the nodes, the process can always be completed without considering the longest $(n - 2)$ branches. (See appendix II.)

PROCEDURE A

Flow Diagram

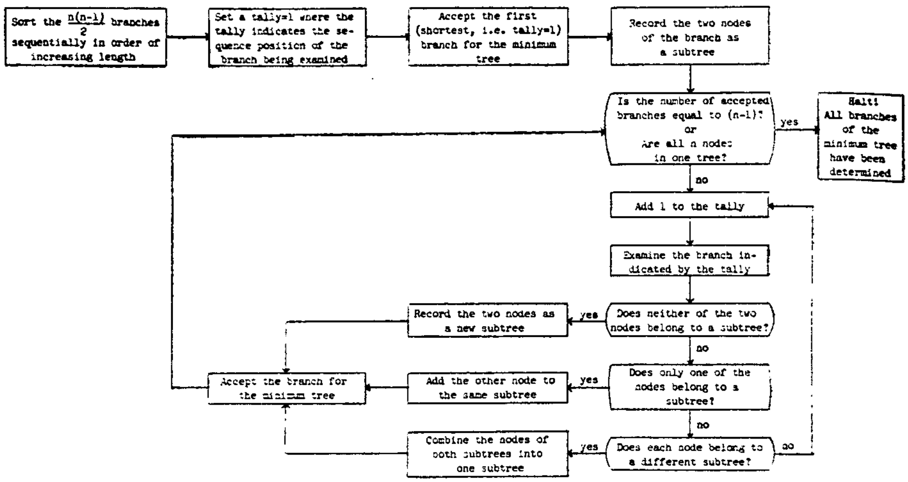

FIG. 3

PROCEDURE B

Flow Diagram

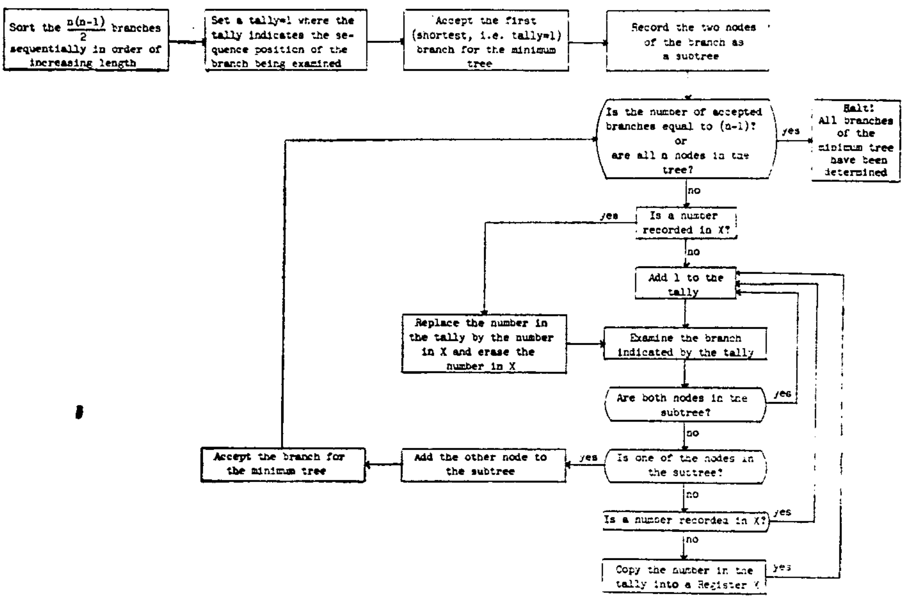

FIG. 4

3. In both procedures, if branches of equal length appear, these branches can be placed in any order with respect to each other in the sorted list and may be properly made as long as the rules to prevent loop formation are followed. The total wire length remains the same.

4. After the initial sorting into a list where the branches are of monotonically increasing length, the actual value of the length of any branch no longer appears explicitly in the subsequent manipulations. As a result, some other parameter such as the square of the length could have been used. More generally, the same minimum tree will persist for all variations in branch lengths that do not disturb the original relative order. Also, the same procedures can be applied if some criterion other than length is used to establish the relative desirability of making various branches. In using any criterion the list of branches would be ordered so that the most desirable branches would be at the top of the list and the less desirable ones near the bottom.

5. Procedures A and B allow the connectivity of a node, defined as the number of branches connected to the node, to vary from 1 to $(n - 1)$. If the connectivity of the individual nodes must be limited to a number less than $(n - 1)$ because of mechanical considerations, then procedures A and B as described will not necessarily produce a mechanically practicable solution. Where the connectivity is limited, the actual lengths of the branches must be considered directly in the process of accepting branches for the minimum tree, since different combinations of alternate branches will produce different total wire lengths. This more general problem is considerably more difficult and belongs to that class of "distribution" problems which have been attacked by linear-programming techniques. The procedures discussed in this paper are not applicable when the connectivity of the nodes is subject to restrictions.

APPENDIX I

In this appendix it will be shown that both procedures A and B yield minimum trees and, in fact, that the respective trees include the same branches.

THEOREM 1: *Given a tree of n nodes in which a path between two of its nodes a and b consists of branches* $ap_1, p_1p_2, \cdots, p_\alpha p_\beta, \cdots, p_ib$. *If one of the branches of this path is removed and the branch ab added, another tree is obtained.* The proof is self-evident.

THEOREM 2: *Given n nodes with k of these nodes connected in a subtree whose branches are also the branches of the minimum tree of n nodes. The shortest branch between any one of the k nodes and any one of the remaining* $(n - k)$ *nodes must be a branch of the minimum n-node tree.*

If a number of branches of equal length fulfill the conditions of being the shortest branch, then every one of these shortest branches is a branch of some minimum n-node tree to which the branches of the k-node subtree belong, though not necessarily of the same minimum n-node tree. Note that more than one n-node tree may qualify as a minimum if a number of branches are of equal length.

PROOF: Let there be for example in figure 5 a k-node subtree whose branches also belong to a minimum n-node tree, where $n > k$. Also let $a_1a_{k+1}, a_2a_{k+2}, \cdots, a_\alpha a_{k+\alpha}$ be the shortest branches (all of equal length) that can be connected between the k nodes and the $(n - k)$ nodes. Now, assume that one of these shortest branches, say a_1a_{k+1} is *not* a branch of any minimum tree of n nodes to which the branches of the k-node subtree belong. In that case, there must be some indirect path from a_1 to a_{k+1} as part of a minimum tree under consideration. This indirect path must include one and only one branch connecting the k group to the $(n - k)$ group, say $a_i a_{k+i}$. If $a_i a_{k+i}$ is one of the branches $a_2 a_{k+2}$, $a_3 a_{k+3}$, $\cdots$, $a_\alpha a_{k+\alpha}$, then $a_i a_{k+i}$ can be removed and $a_1 a_{k+1}$ added. The result, according to theorem 1, is a different tree, but since the branches $a_1 a_{k+1}$, $a_2 a_{k+2}$, $\cdots$, $a_\alpha a_{k+\alpha}$ are all equal, it is also a minimum tree. On the other hand, if $a_i a_{k+i}$ is *not* one of the branches $a_2 a_{k+2}$, $a_3 a_{k+3}$, $\cdots$, $a_\alpha a_{k+\alpha}$, then $a_i a_{k+i}$ could be removed and $a_1 a_{k+1}$ added in order to obtain a different but shorter tree of n nodes. This, however, contradicts the assumption that $a_1 a_{k+1}$ is not a part of any minimum tree of n nodes. Therefore branch $a_1 a_{k+1}$ must be included in at least one minimum tree of n nodes, and in fact $a_1 a_{k+1}$ must be included in some minimum tree of n nodes to which belong the branches of the k-node subtree.

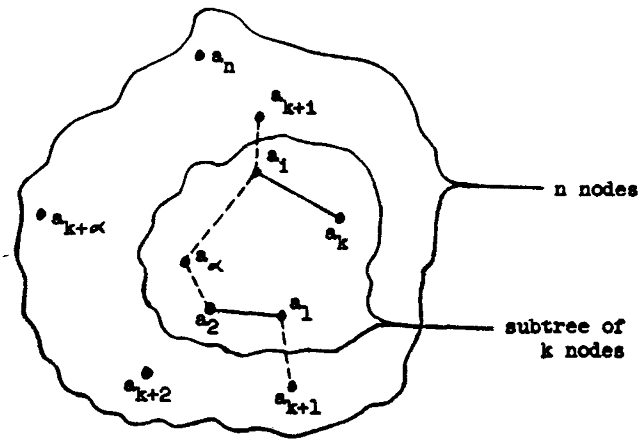

FIG. 5

COROLLARY: *Given n nodes, the shortest of all possible branches which could connect any node to any of the other (n − 1) nodes belongs to a minimum n-node tree. If a number of such branches are equal and qualify as the shortest, each belongs to at least one minimum n-node tree though not necessarily to the same minimum tree.*

By means of the above theorems, procedures A and B can be shown to yield a minimum tree.

The first step, the ordering of the branches, is the same for both procedures. The next step is also the same for both procedures A and B, namely, that the first branch (shortest or one of the shortest if a number of them are of equal length) is selected as a branch of a minimum n-node tree. The two nodes of the shortest branch are then recorded as belonging to a subtree which is in effect a tree of two nodes. The corollary to theorem 2 justifies this step.

Here the two procedures diverge.

In procedure A the branches are examined sequentially following the first (shortest) branch. For each branch the two nodes constituting its terminals are examined, and four distinct conditions are recognized. The disposition of each branch under each of the four conditions can now be justified by means of the previous theorems. The four conditions are:

Condition 1. Neither of the two nodes is present in a subtree. By the corollary to theorem 2, the branch belongs to a minimum tree because the absence of both nodes from any of the subtrees at this step in the sequence indicates that the shortest (or one of the shortest) branch from each of these two nodes is precisely this branch.

Condition 2. Only one of the two nodes is present in a subtree. According to theorem 2, the branch belongs to a minimum tree to which the branches of the existing subtree belong, because it is the shortest (or one of the shortest) of all branches which connect a node of the existing subtree to a node outside this subtree. The new graph thus formed is also a tree.

Condition 3. Each of the two nodes is present in a different subtree. By theorem 2, the branch belongs to a minimum tree to which the branches of the existing subtrees belong, because it is the shortest (or one of the shortest) of all branches which connect a node of one of these subtrees to a node outside this subtree, and it is also the shortest (or one of the shortest) of all branches which connect a node of the other subtree to a node outside this other subtree. The consolidated graph thus formed is also a tree.

Condition 4. Both nodes are present in the same subtree. Any additional branch between two of its nodes would form a second path between the two nodes. Therefore, the additional branch is rejected.

When $(n - 1)$ branches have thus been accepted as belonging to a minimum tree of n nodes or when all n nodes have been combined into one tree, the accepted branches constitute a minimum tree of n nodes.

In procedure B the initial subtree of two nodes is expanded by adding the shortest branch (or one of the shortest branches) connecting one node of the existing subtree to one node outside of the existing subtree. Just as does the single branch of the initial subtree, all subsequent branches thus added also belong to a minimum tree of n nodes, according to theorem 2.

Procedure B is terminated similarly as in procedure A.

It will now be shown that both procedures yield the same (minimum) tree. If only one minimum tree exists, both procedures must select the same branches. However, the sorted sequence of branches contains no information of the actual lengths of the branches since the branches are designated only by their nodes. Thus neither procedure can distinguish between a sequence of branches for which the minimum tree is unique and a sequence for which there is more than one minimum tree. Therefore, both procedures yield the same minimum tree.

APPENDIX II

THEOREM 3: *The longest $(n - 2)$ branches in the sorted list are never accepted for the minimum tree.*

PROOF: Consider a number of n nodes distributed in such a way that there are $(n - k)$ nodes in one group and k nodes in another group, as shown in figure 6.

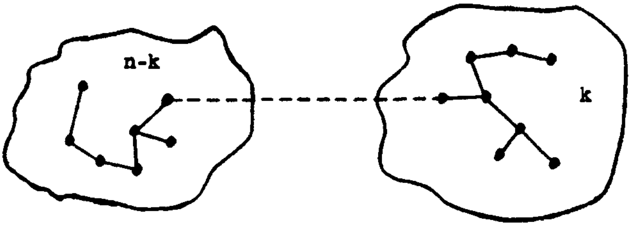

FIG. 6

In procedure A this condition will always arise since nodes are connected into subtrees and subtrees are connected until there remain only two subtrees to be connected. In the limiting case where there are $(n - 1)$ nodes in one subtree and only the last node is unconnected, k can be considered equal to 1, as is always true in procedure B.

The total number of branches in the sorted list which are considered for the minimum tree is $n(n - 1)/2$ and these are distributed as follows:

a. Total number of branches in the $(n - k)$ group $= (n - k)(n - k - 1)/2$
b. Total number of branches within the k group $= k(k - 1)/2$
c. Total number of branches between the groups $= k(n - k)$.

In the sorted list of $n(n - 1)/2$ branches from which branches are selected for the minimum tree the $k(n - k)$ branches must be at the bottom of the list; otherwise they would have been made previously. To complete the pattern one branch (the shortest) of the $k(n - k)$ branches must be chosen, leaving $k(n - k) - 1$ branches.

Now it is seen that $n - k \geqq 1$. This simply means that there is always at least one node in the $(n - k)$ group.

This can be written

$$n \geqq k + 1$$

from which easily follows that

$$n(k - 1) \geqq (k + 1)(k - 1),$$

$$nk - n \geqq k^2 - 1,$$

$$nk - k^2 \geqq n - 1,$$

$$k(n - k) \geqq n - 1,$$

$$k(n - k) - 1 \geqq n - 2.$$

The theorem is thus proved.

ACKNOWLEDGMENTS

Acknowledgments must be made to A. L. Leiner and W. W. Youden for their valuable comments during the formulation of this work, and specificially to W. W. Youden for his initial suggestion leading to procedure A.

REFERENCES

1. D. König: *Theorie der Endlichen und Unendlichen Graphen*, Chelsea Publishing Co., 1950.
2. A. Cayley: A Theorem on Trees, *Mathematical Papers*, Vol. XIII, pp. 26–28.
3. H. Prüfer, Neuer Beweis eines Satzes über Permutationen, *Arch. Math. Phys.* (3), *27* (1918), pp. 142–144.
4. J. B. Kruskal, Jr.: On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, *Proc. Amer. Math. Soc.* 7 (1956), pp. 48–50.