A NEW APPROACH TO CHANNEL ROUTING

by

M. Marek-Sadowska and E.S. Kuh

Memorandum No. UCB/ERL M81/95

23 December 1981

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A NEW APPROACH TO CHANNEL ROUTING*

M. Marek-Sadowska† and E.S. Kuh

Department of Electrical Engineering and Computer Sciences
and the Electronics Research Laboratory
University of California, Berkeley, California 94720

ABSTRACT

This paper introduces a new approach to the 2-layer channel routing problem. The method allows horizontal and vertical connections on both layers. An order-graph similar to the traditional vertical constraint graph is used in the algorithm. The main advantage is that the method is general and does not have the usual restriction introduced by cyclic constraints. Furthermore, the channel width does not have a lower bound equal to the maximum density.

---

*Research sponsored by the National Science Foundation Grant ENG-78-24425 and the Air Force Office of Scientific Research (AFSC) United States Air Force under Contract No. F49620-79-C-0178.

†On leave from the Institute of Electron Technology, Technical University of Warsaw.

1. INTRODUCTION

Two-layer channel routing is one of the key elements in the automatic layout design of VLSI circuits. In the traditional approach of channel routing, one layer is used exclusively for horizontal connection and the other is used exclusively for vertical connection [1,2]. Via holes are used for interconnections between layers. The present paper takes a new look of the problem by allowing horizontal and vertical connections on both layers. Thus the only restrictions are that on each layer nets must not intersect and must not overlap on the same horizontal or vertical track. With this flexibility, the vertical constraint graph associated with the given net list is no longer required to be acyclic. Furthermore, the channel width is not bounded by the maximum density over the channel.

2. THE BASIC APPROACH

The problem can be specified by a net list defined on a horizontal channel. Nets which are to connect terminals evenly spaced on two sides of a channel are to be routed using horizontal and vertical tracks on two layers. A number between 0 and N is assigned to each terminal. Terminals with the same number i ($1 \leq i \leq N$) are to be connected by net i, while terminals with number 0 are left unconnected as shown in Fig. 1. Our objective is to route all the nets using a minimum number of horizontal tracks.

Our basic approach depends on the classification of nets according to their terminal positions. We assume first that all nets are 2-terminal nets. As will be shown later, multi-terminal nets can be decomposed into 2-terminal subnets. Thus our treatment is general. In Fig. 1 we show the five basic types of 2-terminal nets and their connections. Solid lines represent connections made on the first layer while dashed lines represent that on the second layer. As seen in the figure Type-A and Type-B nets require the use of two via holes each, while the others require only one via. To be more precise, the following definitions are given:

Type-A net: both terminals are on the top side of a channel.

Type-B net: both terminals are on the bottom side of a channel.

Type-C net: both terminals are on the same vertical track.

Type-D net: one terminal is on the bottom side of the channel and the other is on the top side; lower terminal is on the left of the upper terminal.

Type-E net: one terminal is on the bottom side of the channel and the other is on the top side; the lower terminal is on the right of the upper terminal.

The proposed method of routing is to take advantage of the five basic types of nets and to route the nets top down according to a definite order. The method places special emphasis on the horizontal segment of each net. Thus when we say that the routing order of nets is top down we refer to the horizontal segments of the nets. When horizontal segments overlap, we assign them to different tracks. In the following we discuss the necessary ordering of the five types of nets.

Let us first consider nets of the same type. If all nets which have horizontal overlaps are of Type-A, then nets can be routed in any order without blocking each other. This is illustrated in Fig. 2 where only solid lines and dashed lines cross each other irrespective of the ordering of routing. The same argument can be given to Type-B nets.

Consider next two Type-D nets with horizontal overlapping as shown in Fig. 3. While Fig. 3a gives a good solution, Fig. 3b leads to an intersection on the second layer. A correct order is to route first the net which has its lower terminal fartherest to the left. Thus the top-down ordering for Type-D nets must follow the order of the lower terminals from left to right. Again, using a dual argument, we can state that the proper ordering for Type-E nets is to follow the order of the upper terminals from right to left as shown in Fig. 4.

Next, we consider nets of different types. The relation between a Type-A net and a Type-D net with horizontal overlap is shown in Fig. 5. It is clear that Type-A nets must be routed before Type-D nets in order to avoid intersection on the same layer. Similarly, using a dual argument, we can show that Type-E net must be routed before Type-B nets.

In Fig. 6 we illustrate the relation between a Type-A net and a Type-C net. The proper order is to route Type-A nets before Type-C nets.

Although there are other combinations to be tested, we can easily conclude that the proper top-down ordering for nets of different types is A-D-C-E-B. With this order there will be no intersections on either layer.

The above discussion suggests a simple algorithm which is based on an order-graph G_0. $G_0 = (V,E)$ is a digraph whose vertices correspond to all the 2-terminal nets given. A directed edge $(v_i,v_j) \in E$ iff net v_i and net v_j overlap horizontally and the horizontal segment of v_i must be placed above that of v_j according to the rule discussed above. It is also clear that the order-graph G_0, different from the traditional vertical constraint graph, is always acyclic. The basic idea is to route the nets top down by following the order-graph G_0. Thus nets corresponding to those vertices without ancestors are selected first to pack the first horizontal track as densely as possible. This can be done with the left-edge algorithm [3]. Once a net is routed, its corresponding vertex is deleted from G_O together with its incident edges to form a reduced order-graph. At the completion of the routing for the first horizontal track, we consider the reduced order-graph for routing the second horizontal track. The procedure continues until all nets are routed or equivalently, the reduced order-graph is empty. Before we give the algorithm, let us first consider the problem with multi-terminal nets.

3. MULTI-TERMINAL NETS

It is easy to show that any multi-terminal net can be decomposed into 2-terminal subnets as shown in Fig. 7. We use the following rules in the decomposition:

(i) Two terminals are assigned to the same subnet if they are in the nearest vertical tracks.

(ii) If there exist two terminals on the same vertical track, then they form a Type-C subnet.

(iii) Whenever choice exists because of the presence of Type-C subnet, use Type-A and Type-B subnets rather than Type-D and Type-E subnets. [see Fig. 7 example, because of the presence of Type-C subnet (d,e), Type-B subnets (c,e) and (e,f) are used rather than (c,d) and (d,f)]

4. ROUTING ALGORITHMS

The basic routing algorithm is summarized below:

Algorithm 1.

Step 0. TRACK NUMBER = 0.

Step 1. Decompose nets into subnets.

Step 2. Assign each subnet into appropriate class.

Step 3. Obtain order-graph G_o.

Step 4. Find all subnets which correspond to vertices in G_o with no ancestors. This is the set S of subnets which can be placed on the next track.

Step 5. Apply left-edge algorithm to subnets in S, let the nets chosen be the set S_1, where $S_1 \subset S$.

Step 6. TRACK NUMBER = TRACK NUMBER + 1.

Step 7. Route subnets in S_1 on the current track.

Step 8. Remove vertices corresponding to S_1 from G_o together with incident edges to obtain reduced order-graph G_o.

Step 9. Are there any vertices left in G_o? If yes, go to Step 4; if not, go to Step 10.

Step 10. End of algorithm.

We give an example as shown in Fig. 8. Figure 8a shows the net list of the channel together with the horizontal segments of nets and subnets assigned to different tracks following Algorithm 1. Figure 8b shows the order-graph G_o. The algorithm goes through the following iterations:

Iteration 1: S = {2,17,5,14,6,13,19,7,8c}

S_1 = {17,7,14} ← Track 1

Iteration 2: S = {2,5,6,13,19,8c}

S_1 = {2,5,19,13} ← Track 2

Iteration 3: $S = \{3,6,15,8c\}$

$S_1 = \{3,6,15,8c\} \leftarrow$ Track 3

Iteration 4: $S = \{18,12,9\}$

$S_1 = \{18,12,9\} \leftarrow$ Track 4

Iteration 5: $S = \{4a,8a,8b,10a,10b,10c,20\}$

$S_1 = \{4a,8a,8b,10c\} \leftarrow$ Track 5

Iteration 6: $S = \{1,16,4b,10a,10b,11a,11b,20\}$

$S_1 = \{1,16,10a,10b,11a,11b\} \leftarrow$ Track 6

Iteration 7: $S = \{4b,20\}$

$S_1 = \{4b,20\} \leftarrow$ Track 7

Figure 8c shows the realization obtained by the basic algorithm. It is seen that a total of seven tracks is used in the realization. The maximum density for the given net list is also seven.

In Algorithm 1, all the steps with the exception of Step 3 are quite straightforward. In Step 3 we need a simple algorithm to generate the order-graph G_o. This is given as follows:

Algorithm 2.

Step 1. Find all edges in G_o which connect Type-D nets (vertices).

Step 2. Find all edges in G_o which connect Type-E nets (vertices).

Step 3. Find all Type-D vertices which do not have descendants, call the set {Y}; and all Type-E vertices which do not have ancestors, call the set {X}.

In the currently generated graph, add an edge from $y_i \in Y$ to $x_j \in X$ if horizontal segments corresponding to y_i and x_j overlap.

Step 4. Add Type-A,C and B vertices to the graph applying the rule previously discussed, i.e., to follow the order A-D-C-E-B.

5. MODIFICATION OF THE BASIC ROUTING ALGORITHM

The objective of the channel routing problem is to minimize the channel width, i.e., the number of horizontal tracks required for realizing the given net list. The proposed basic algorithm gives a realization with a channel width which has a lower bound equal to the length of the longest path of the order-graph G_0. In this section we will introduce some modifications to the basic algorithm to reduce further the channel width. Instead of giving the detailed algorithms we illustrate our ideas with the following example: In Fig. 9a there are four Type-D nets, the order graph is shown in Fig. 9b. The number of horizontal tracks required using the basic algorithm is equal to four. In Fig. 9c we demonstrate the possibility of introducing jogs at the two black dots, thus nets 2 and 3 can be divided into segments 2', 2", 3' and 3". The corresponding order-graph becomes that of Fig. 9d with a longest path length equal to two. Figure 9e shows the routing with jogs, which requires only two horizontal tracks.

Another useful modification deals with via placement. The idea is that, by moving the actual position of a via, it is sometimes possible to reduce the channel width. This is illustrated with the same example as in Fig. 9. In Fig. 10 we show that by moving the vias of nets 1 and 2 it is possible to realize the problem in Fig 9a with only one horizontal track.

These modifications have been added to the basic algorithm. In the current version of the program Vias are shifted to their opposite-most positions on the horizontal track without creating blockades for sub-nets to be routed subsequently. This allows the usage of horizontal tracks on both layers more effectively. In Fig. 11 we show the result of routing of the example in Fig. 8 after modifications. It is seen that the channel width is reduced to six although the maximum density for the problem is seven.

6. CONCLUSION

The basic algorithm together with modifications has been implemented. The program has been tested with various examples. These are shown in Figs. 12-14. As seen from the figures the results are mixed. While in one case the realization leads to channel width less than the maximum density; in others, the results are inferior to the earlier methods.

The main contribution of this paper is that we have departed from the traditional approach to the 2-layer channel routing problem and demonstrated the enormous flexibility in routing. The order-graph introduced resembles the vertical constraint graph in the traditional approach; yet with the order-graph, there exists no cyclic constraint. It is indeed possible to combine the method proposed here and the method of merging vertices on the vertical constraint graph introduced in [2]. This will be left for future research.

REFERENCES

1. D.N. Deutsch, "A dogley channel router," Proc. 13th Design Automation Conf., pp. 425-433, 1976.
2. T. Yoshimura and E.S. Kuh, "Efficient algorithms for channel routing," Electronics Res. Lab. M80-43, University of California, Berkeley, August 1980, to appear IEEE Trans. CAD, January 1982.
3. A. Hasimoto and J. Stevens, "Wire routing by channel assignment within large apertures," Proc. 8th Design Automation Workshop, pp. 155-169, 1971.

FIGURE CAPTIONS

Fig. 1. Five different types of 2-terminal nets:
Net 1: Type B, Net 2: Type A, Net 3: Type C,
Net 4: Type D and Net 5: Type E.

Fig. 2. Nets of Type A can be routed in any order. In Fig. 2a the order is 1,2 while in Fig. 2b the order is 2,1. In either case routing can be completed with no intersection on the same layer.

Fig. 3. Nets 1 and 2 are both Type D Nets. If Net 1 is routed before Net 2 then routing is completed without intersections on the same layer. If Net 2 is routed first then segments of different nets intersect on the same layer, which is not allowed.

Fig. 4. Nets 1 and 2 are of Type E. Net 2 must be routed before Net 1 as shown in Fig. 4a. Figure 4b shows incorrect order, which leads to an intersection on the first layer.

Fig. 5. Net 1 is Type A, Net 2 is Type D. In Figs. 5a and 5b the first terminal of Net 2 is between the terminals of Net 1. As shown, Net 1 has to be routed before Net 2. In Figs. 5c and 5d the first terminal of Net 2 is to the left of Net 1, either order is correct.

Fig. 6. Net 1 is Type A, Net 2 is Type C. Figure 6a gives the correct routing, i.e., the Type A Net must be routed before the Type C Net. (Note the horizontal segment of Type C Net is degenerated to a point, the via). Figure 6b represents an incorrect routing.

Fig. 7. Multi-terminal Net decomposition. Terminals a,b,c,d,e,f,g, and h form a net. It is divided into following 2-terminal subnets: (a,b),(b,c),(c,e),(d,e),(e,f),(f,g) and (g,h).

Fig. 8. Figure 8a net list specification for the channel and horizontal segments of subnets. Figure 8b: order-graph G_o. Figure 8c: channel routed applying the algorithm.

Fig. 9. Figure 9(a): Nets routed without jogs, channel width equals to 4. (b) Order-graph for the case without jogs, length of the longest path is 4. (c) Black dotts show the positions of the possible jogs. (d) Order-graph for net list with jogs, length of maximal path is 2. (e) Routing with jogs, channel width is 2.

Fig. 10. Example from Fig. 9a routed with floating via approach. Number of track becomes one.

Fig. 11. Example of Fig. 8 routed with modifications to the basic algorithm.

Fig. 12. Example 1 in [2]. Maximum density = 12, no. tracks 11.

Fig. 13. Example 4b in [2]. Maximum density = 17, no. of tracks = 18.

Fig. 14. Difficult example in [2]. Maximum density = 19, no. of tracks = 24.

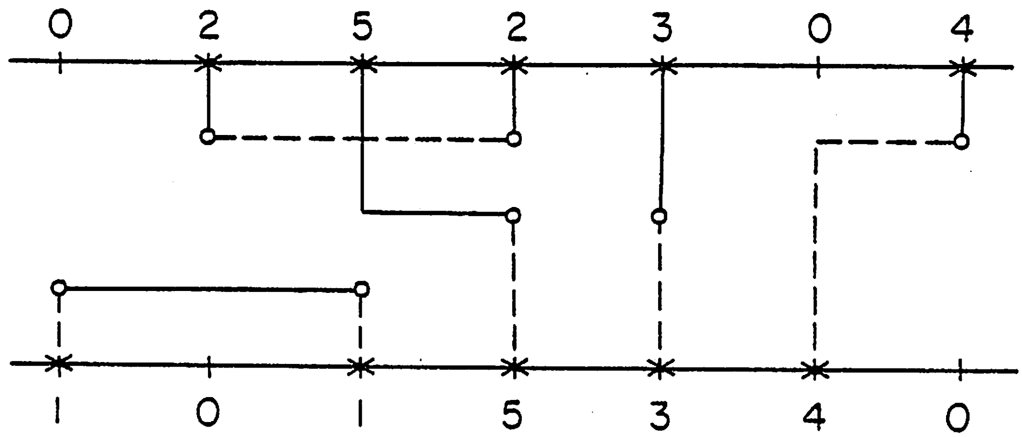


Figure 1

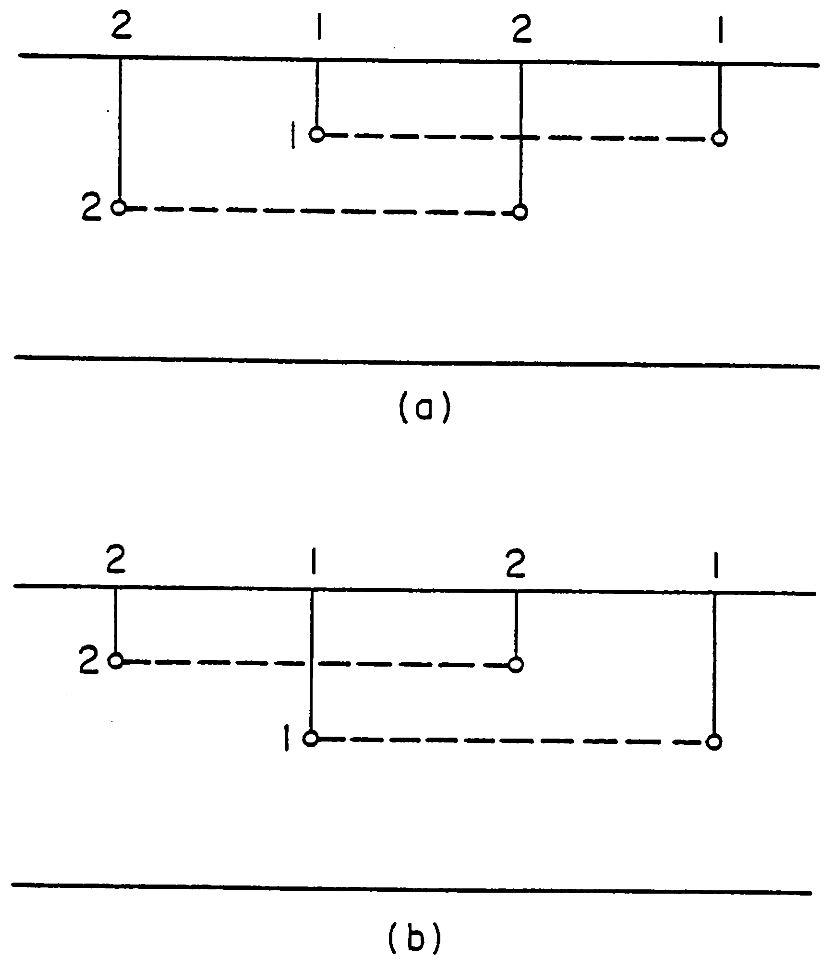
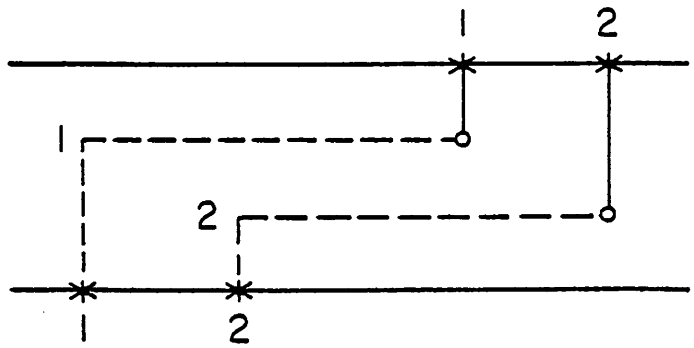
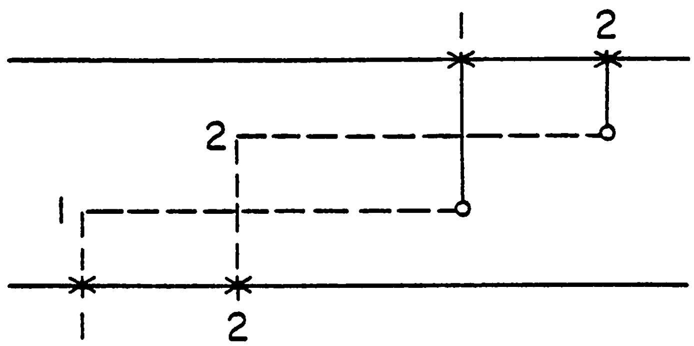


Figure 2



(a)



(b)

Figure 3

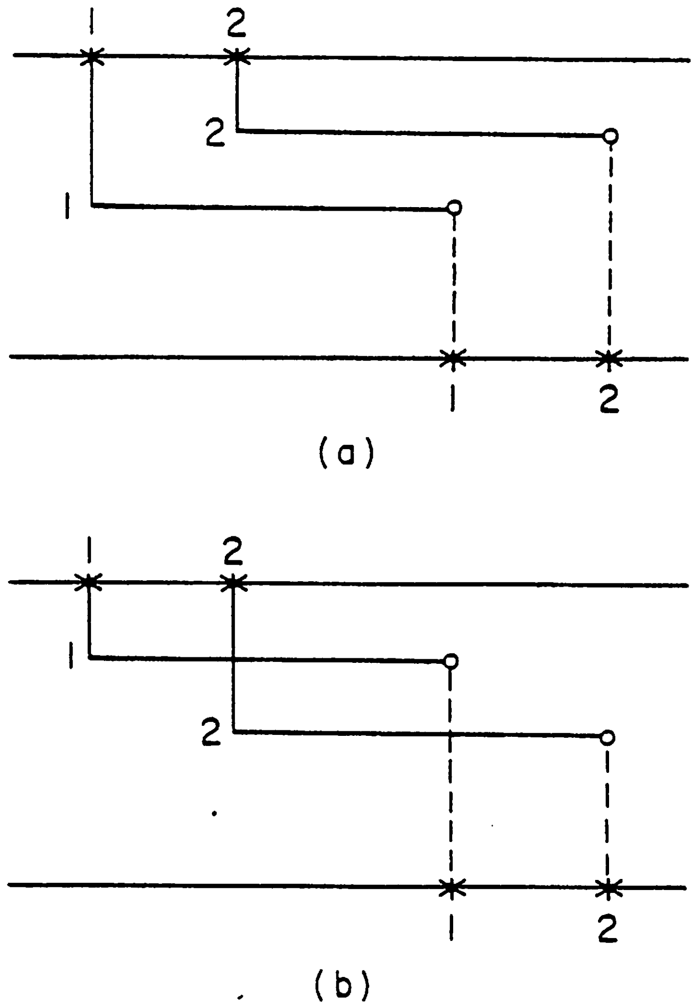


Figure 4

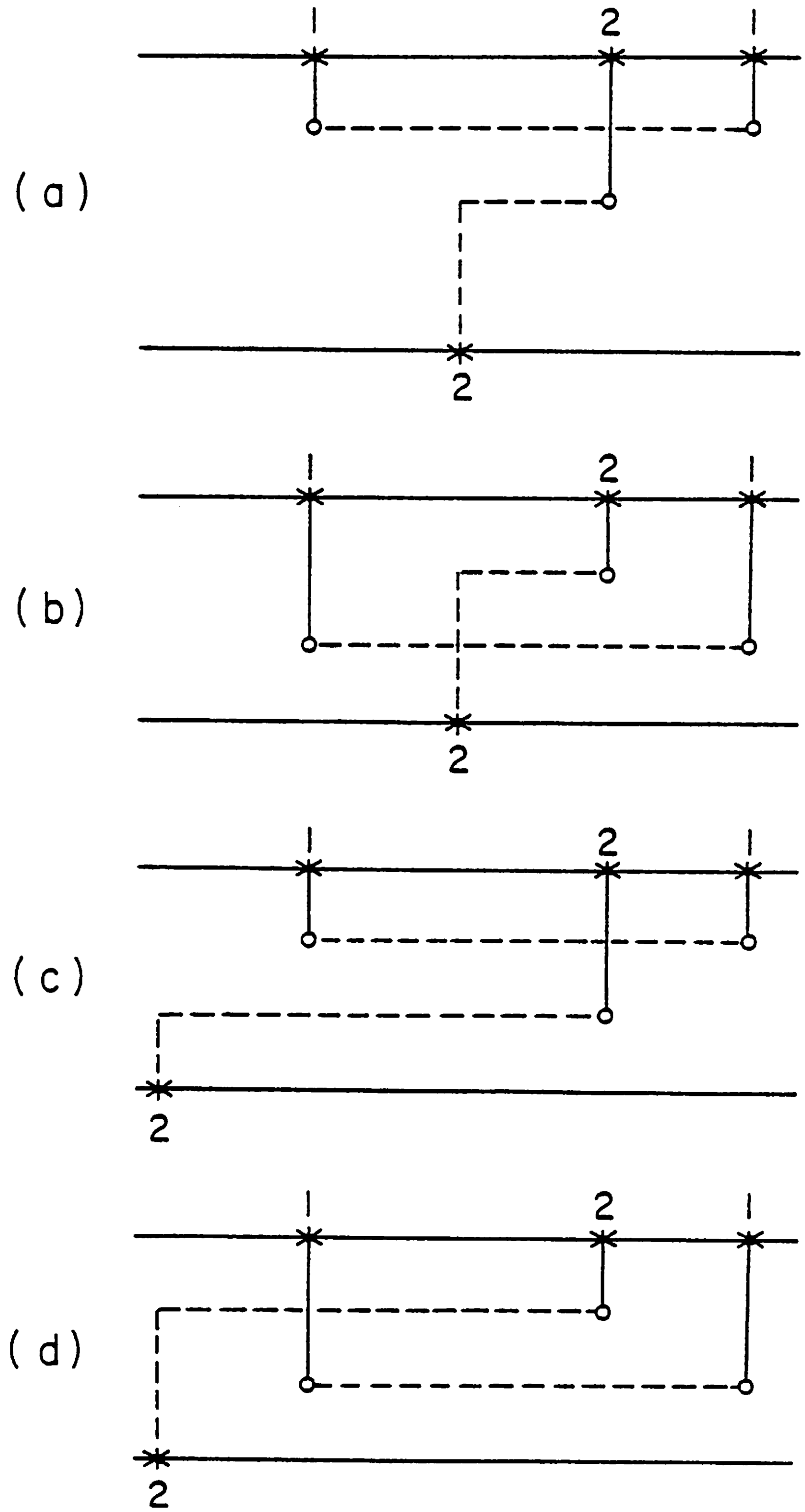


Figure 5

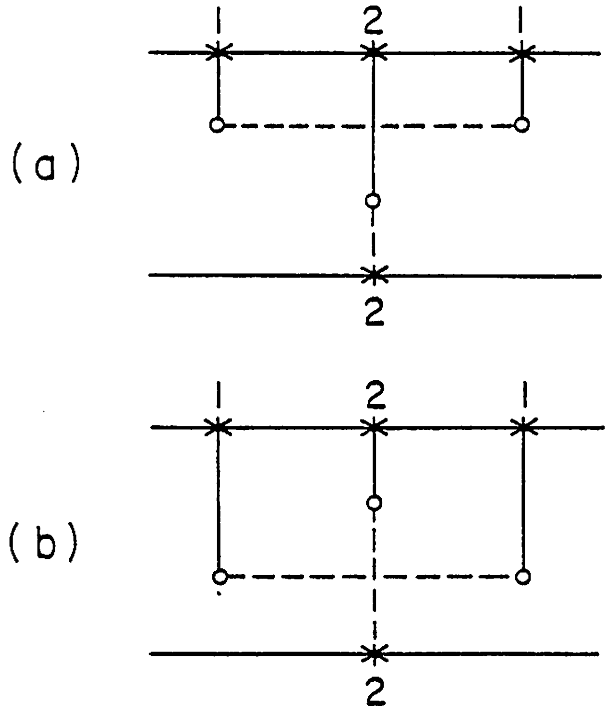


Figure 6

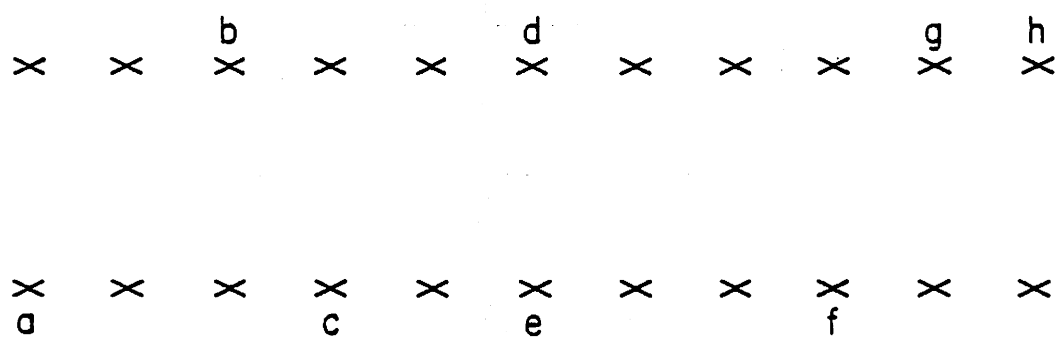
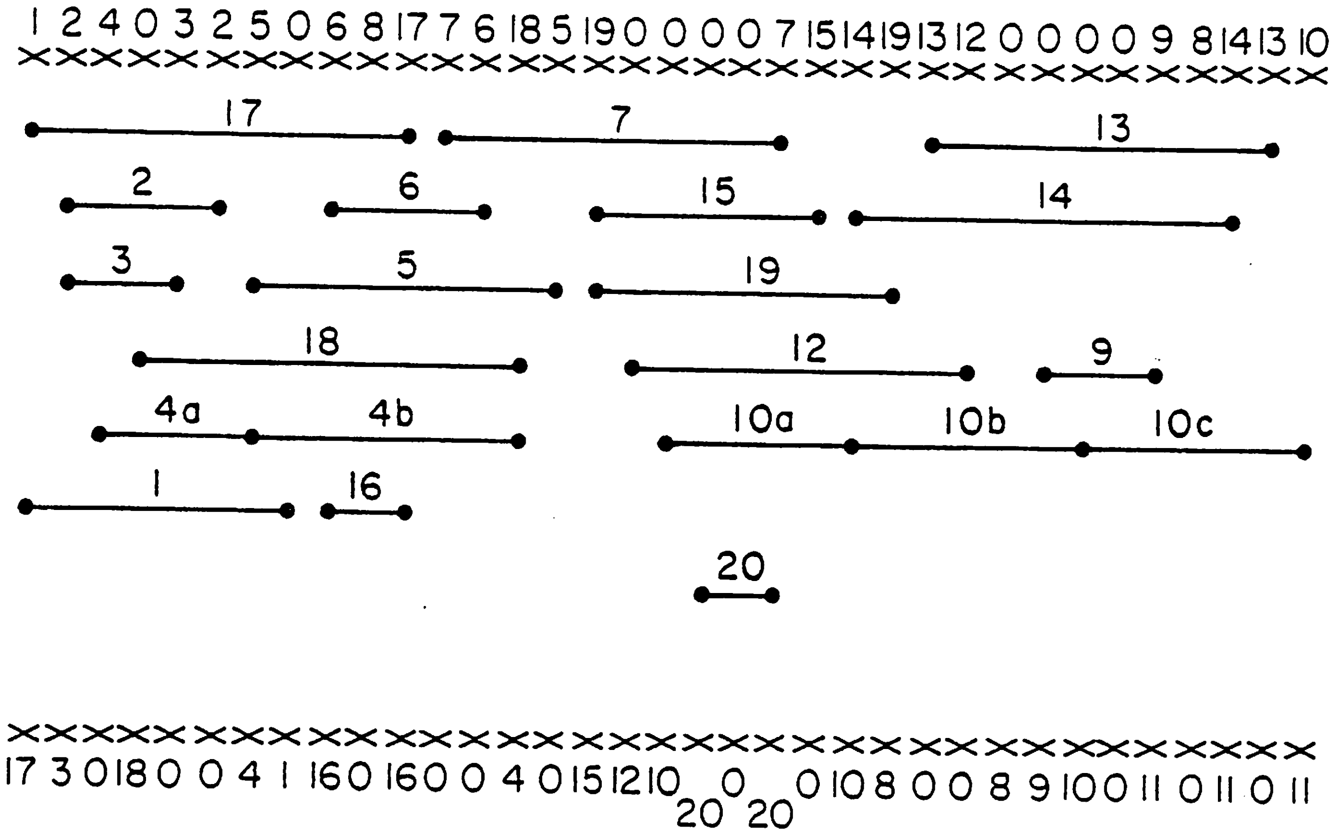
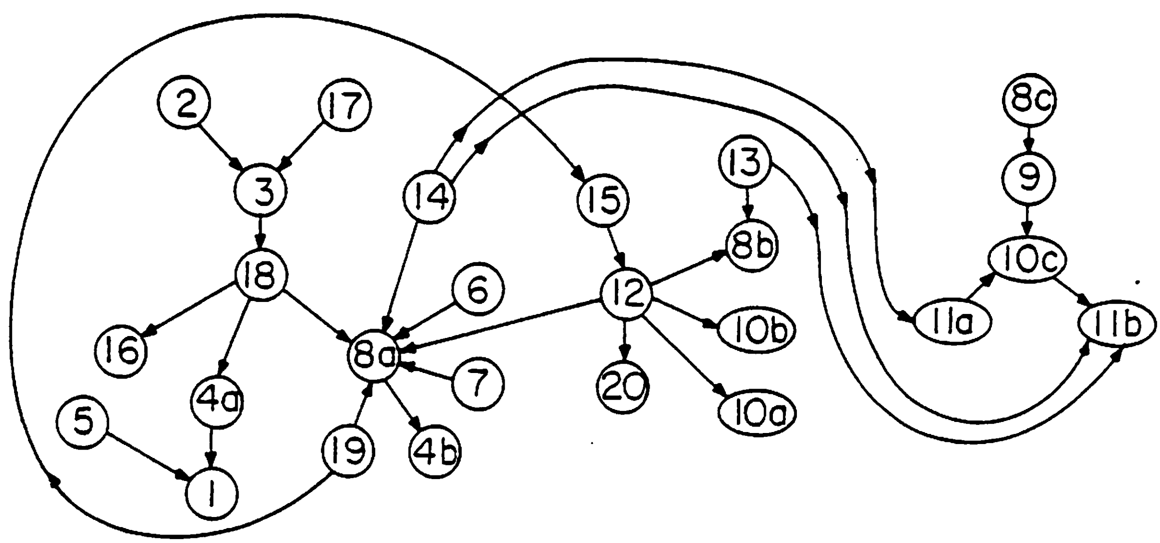


Figure 7



1 2 4 0 3 2 5 0 6 8 17 7 6 18 5 19 0 0 0 0 7 15 14 19 13 12 0 0 0 0 9 8 14 13 10
17
7
13
2
6
15
14
3
5
19
18
12
9
4a
4b
10a
10b
10c
1
16
20
17 3 0 18 0 0 4 1 16 0 16 0 0 4 0 15 12 10 20 0 20 0 10 8 0 0 8 9 10 0 11 0 11 0 11

(a)



2
17
3
18
16
5
4a
1
14
6
8a
7
19
4b
15
12
20
13
8b
10b
10a
8c
9
10c
11a
11b

(b)

Figure 8

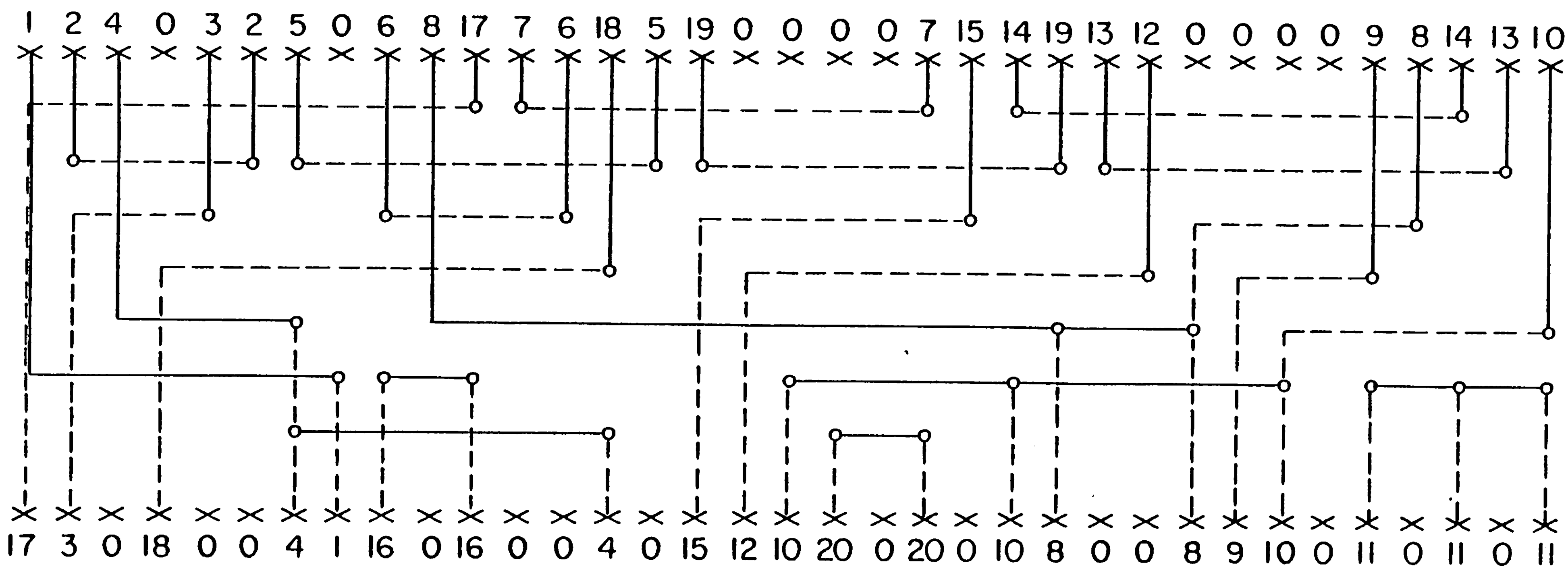

1 2 4 0 3 2 5 0 6 8 17 7 6 18 5 19 0 0 0 0 7 15 14 19 13 12 0 0 0 0 9 8 14 13 10
17 3 0 18 0 0 4 1 16 0 16 0 0 4 0 15 12 10 20 0 20 0 10 8 0 0 8 9 10 0 11 0 11 0 11

Figure 8c

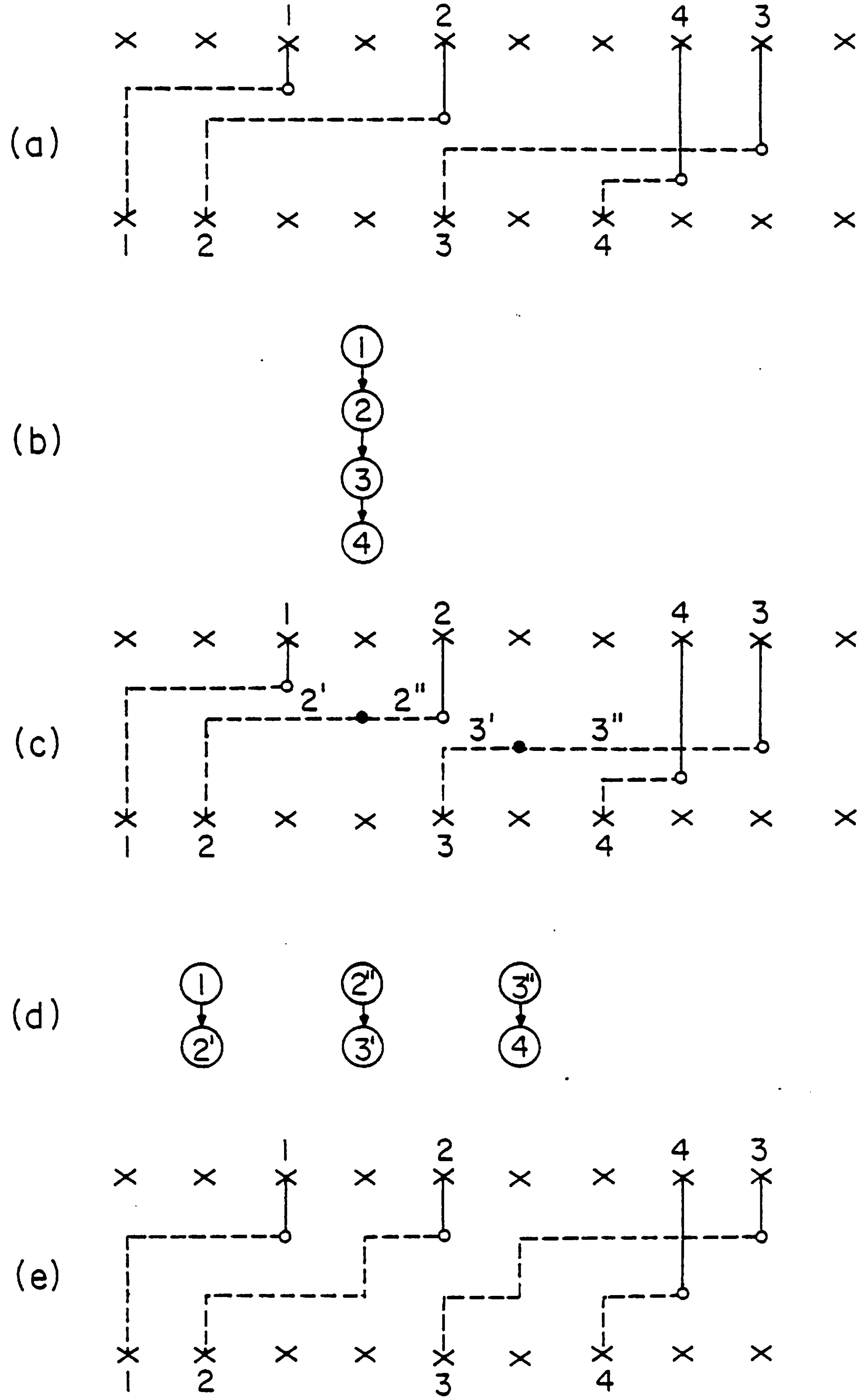


Figure 9

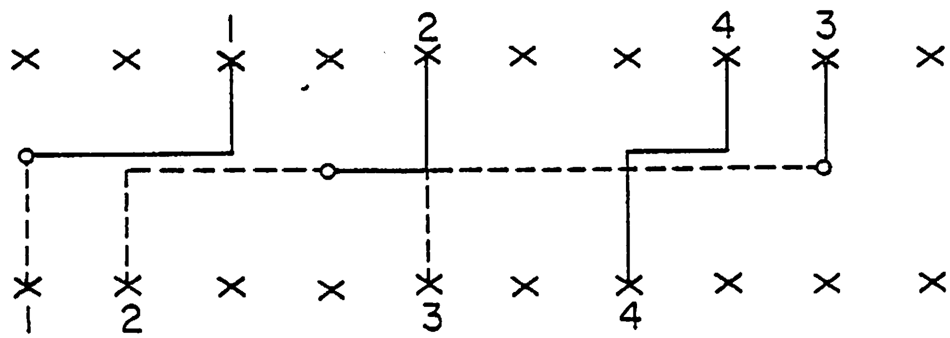


Figure 10

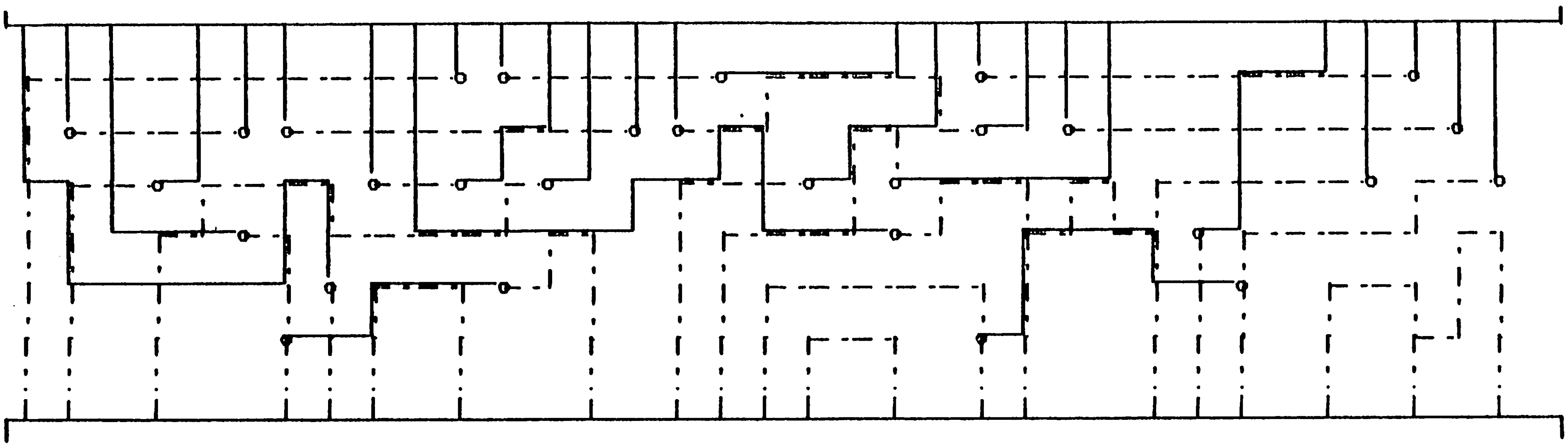


Figure 11

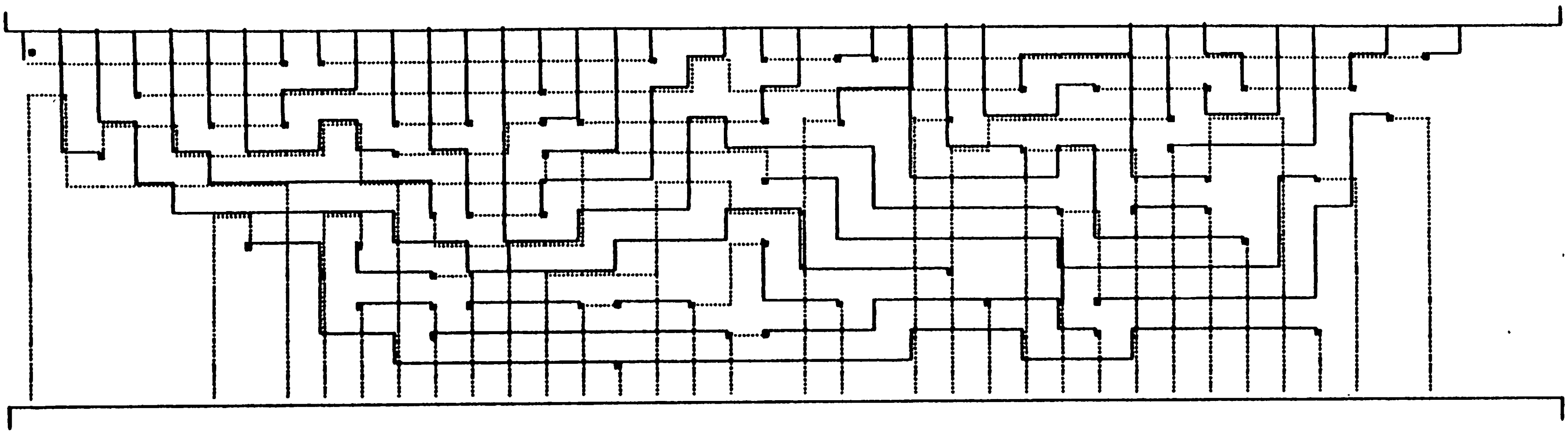


Figure 12

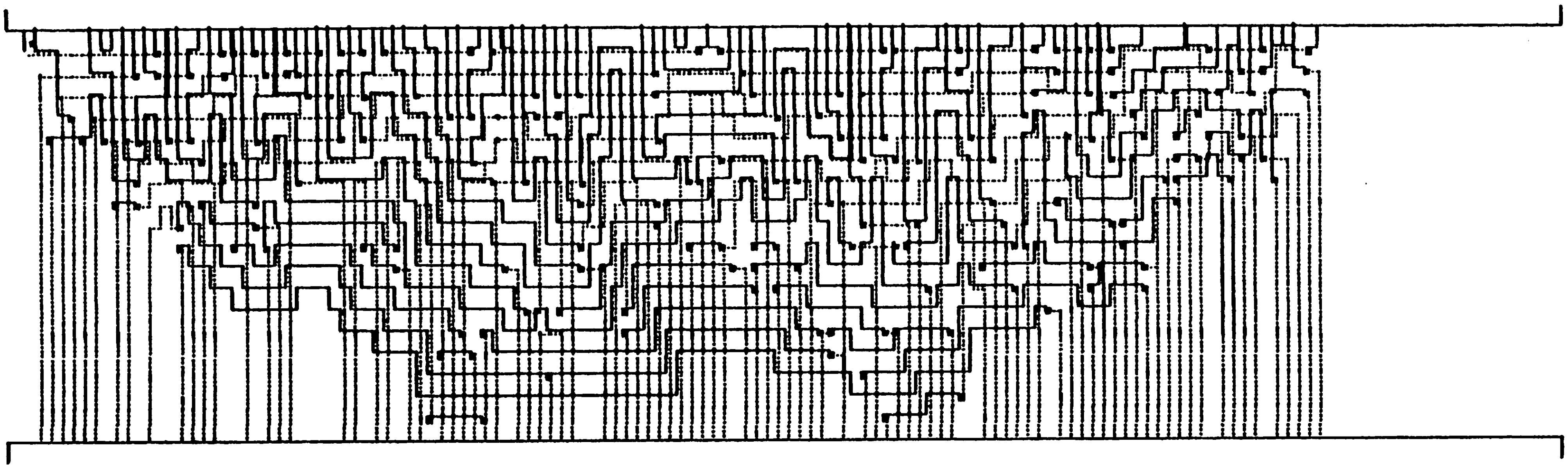


Figure 13

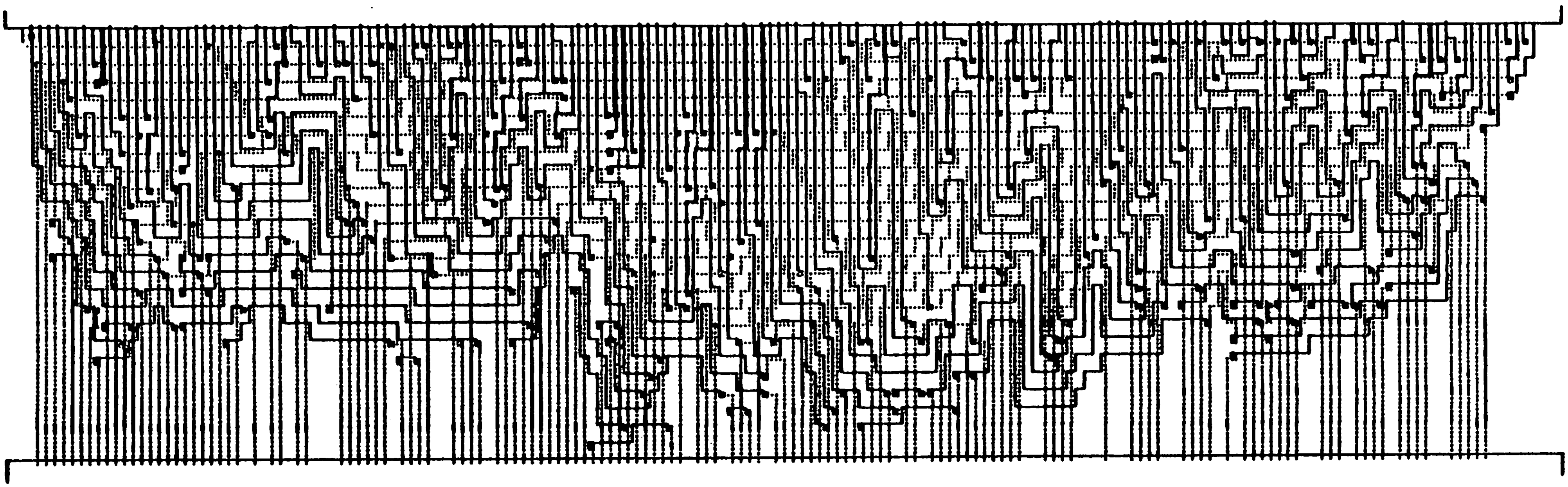


Figure 14