SOME PROBLEMS OF THE DISTRIBUTION OF INFORMATION FLOW IN COMMUNICATION NETWORKS

Alexandre Butrimenko

July 1974 WP-74-25

SOME PROBLEMS OF THE DISTRIBUTION OF INFORMATION FLOW IN COMMUNICATION NETWORKS*

Alexandre Butrimenko

The last few years have brought forth more and more control and information systems which have emerged as a result of information being given in digital form. In my lecture I shall look at some problems of the elements of such systems, namely, problems of the distribution of information flow in communication networks.

I particularly want to refer to the so-called data communication networks, i.e. when a sender dispatches a message to a receiver and it is not necessary to establish a direct connection between them.

The message can be transmitted in various ways:-

(a) We can use the so-called "channel switching". This is in fact exactly what we use in telephone connections. Fig. 1

(b) We can use "store-and-forward" or "message switching". The information is communicated in relays and can be delayed to a smaller or greater degree in the separate nodes. Fig. 2

Regardless of which network we look at, there are problems with the optimization of its working procedure. In this lecture I will not talk about the problems of optimization of the network; the network itself is taken as given.

Certainly, we normally know relatively little about how the load of the network, and perhaps the network itself, changes with time. This means that we should adapt ourselves

---

* A lecture given to the Austrian Society for Cybernetic Studies on 29 May, 1974.

to ever-changing conditions; the network--or rather the routing technique--should be adaptable. On the other hand, the routing technique or the control of the information flow must be reliable and resilient. One could achieve this through decentralization. Decentralization means that no node has access to information on the entire network and on the state of the traffic. Such decentralization prevents paralysis of the whole network when the information control or part of the network fails.

The chosen criteria of the network performance play a special role in optimization. In this lecture I will show three methods, each of which is based on different criteria. Of course, all of these methods have several similar characteristics, and they are adaptive and decentralized.

First of all we will look at "channel switching networks". If we can find a way from sender to receiver (i.e. a free path) then the appropriate message can be transmitted. If such a path cannot be found, then the attempt at connection will fail and we talk of a loss. Our aim is to reduce the number of these losses. However, I wish to stress that the failed connection does not mean that there is no free path in the network between sender and receiver. The reason for the loss is more likely to be that we could not find this path.

Let us look at an example. Let us assume that the sender in Node 2 should be connected to the receiver in Node 15. The dot-dash lines in the diagram are fully occupied and the dash lines have at least one channe free. If we have this information, as shown in the diagram, then we could find a path. However, for several reasons we do not have access to this information--firstly because the information changes too rapidly. Fig. 3

Let us now look at the networks which permit a connection via various paths, using the so-called routing technique. In such networks there are special instructions in every node that indicate which direction should be chosen.

In the case of the adaptive routing technique these instructions can be changed. In our example, our instructions bring us to a dead end; we cannot use the first exit because there is no free line to Node 10, and the second exit leads us to Node 1 where there are no free exits at all. The adaptive routing technique alters the instructions, i.e. it would show that there is no exit from Node 1. As mentioned earlier, we cannot avoid all mistakes but the number of such mistakes should be minimized. And that is the task of the adaptive routing technique. Fig. 4

Now I will describe an algorithm which seems to have such characteristics.

Let us assume that from every node the probability to reach the destination node is taken as given. These probabilities are dependent on line blocking and on routing. The optimal route is always the one which leads to the nearest node with the highest probability. In our example, Node 10 will be tried first and then Nodes 1 and 4.

However, the problem is that the blocking probabilities cannot be determined and they change with the course of time. It is proposed that we continually evaluate the blocking probabilities. These are known for the exiting channels in every node (because they are also measured in each node). We assume that in every node the probabilities of the neighbouring nodes are also known. Then we can calculate the probabilities for the considered node.

For example:

$$P_2 = (1 - p_{2,10}) \cdot P_{1,0} + p_{2,10}(1 - p_{2,1}) \cdot P_1 +$$

$$+ p_{2,10} \cdot p_{2,1} (1 - p_{2,4}) \cdot P_4 =$$

$$= (1 - 0.9) \cdot .0,9 + 0,1 \cdot (1 - 0,9) \cdot 0,8 +$$

$$+ 0,1 \cdot 0,1 \cdot (1 - 0,99) \cdot 0,7 = 0,00987$$

It is obvious that the probabilities of the nodes can be wrong. This means that the calculated probability is also wrong. We can imagine this process as an iterative method for the solution of the following equational system.

$$P_i^{(k,t+1)} = (1 - p_{i,max_1}) \cdot P_{max_1}^{(k,t)} + p_{max_1} \cdot (1 - p_{i,max_2}) \cdot p_{max_2}^{(k,t)} + \ldots + p_{i,max_1} \cdot p_{i,max_2} \cdot \ldots \cdot p_{i,max_{n-2}} (1 - p_{i,max_{n-1}} \cdot P_{max}^{(k,t)}$$

where P_{max_ℓ} is ℓ-th maximum in $\{P_i\}$

We repeat this process many times for each node in the network. It should still be mentioned that for each destination node this probability is constant and equal to one.

A colleague from our Institute, Dr. Bell, has shown that this process converges with the solution to any initial numbers and any blocking probabilities. Mr. Gradischnig has simulated this system with me and the simulation showed that the system converges very quickly. Fig.5

We now come to the second method - the so-called store-and-forward or message switching method. Using this method we do not have to have a switched through channel from sender to receiver because the information is transmitted in relays and can be delayed to a smaller or greater degree in every transit node.

When all the channels in the system have the same speed and no queues are formed, then the shortest paths are also the quickest paths. In practice, however, queues do form and the shortest paths are not necessarily the quickest.

As in the former case, the information on the routing is stored in a matrix in every node. The number of rows in the matrix is equal to the number of nodes in the network. The number of columns in the matrix is equal to the number of exits of this node. In our example Matrix A consists of four rows and three columns, Matrix B of four rows and only two columns, corresponding to the two exits from B to Nodes A and C. Every row in such a matrix corresponds to one destination node and the elements give the minimum distance via the appropriate exit direction to the destination node. Fig. 6

At the same time, the distance "1" means that the duration of the information transfer between these two nodes requires one time unit. For example, Element CC in Matrix A gives the shortest path from Node A to Node C. If the message were sent from A to C in the direction of B, then it would require two time units to Node C. Obviously, every matrix has a row which consists completely of zeroes. The index of this row coincides everywhere with the index of the considered matrix.

In the diagram we see a vector in every node. The elements of this vector give the minimum distance from the considered node to all destination nodes. For example, if the minimum distance from Node B to Node A requires one time unit, then these two nodes are connected via one channel directly to each other. Or, the shortest distance from Node B to Node D requires two time units.

If there is no message waiting to be sent, then we can make use of the matrices directly to establish the shortest and quickest paths. If we now send a message from Node B to Node D, then we choose the lowest figure in Node B from row D of the destination node; the index of the column indicates the neighbouring node to us to which the message must be sent.

Since in this case both elements in row D are equal to two, let us choose here the exit to Node C. If the message moves into Node C, then we must ascertain the lowest figure in row D (destination node) of Matrix; the column again shows us the exit direction. In this case we send the message directly to D. We see that this procedure is feasible if there are no queues.

Now we examine the case when three messages are waiting in Node C for the direction of D. Let us choose once more in Node B the path via Node C; then our message needs three time units to reach destination node D via the path BCAD, because now the minimal time path from Node C to Node D passes through Node A.

In order to find the correct exit from Node C, we must correct column D in Matrix C, i.e. we must in our example increase all elements in this column by three, since all paths via channel CD have become slower by three time units. This alteration is shown on our diagram under the matrices and vectors.

If the information on the position in Node C were known in Node B, then we would of course send this message from Node B in the direction of A. Therefore, we must also correct column C in Matrix B accordingly.

Therefore, we again establish in Node C the vector of the minimum distances to all destination nodes. In our example we obtain the vector {1, 1, 0, 2}. So we see that due to these waiting messages the shortest route from Node C to Node D has increased from one to two.

Thus, this vector indicates the minimum distance from Node C to all destination nodes, taking into account the current message traffic in Node C. If this vector was known in Node B, we could correctly decide which channel to choose. For this purpose we write the vector of C into column C of Matrix B. We must however bear in mind that the transfer of a message

from Node B to Node C will require one time unit. Accordingly we must increase all elements of vector C by one. The 0-elements of the matrix remain unchanged.

Thus the procedure can be formulated: in every node one (periodically) establishes the vector of the "minimum distance" taking into account the number of waiting messages. Then every element of the vector is increased by one and the vector is sent to all neighbouring nodes. There it will be stored in the appropriate column.

This procedure was simulated and compared to the non-adaptive method, i.e. when only the shortest paths are chosen. The simulation was carried out for the network shown in Fig.3. Using the non-adaptive routing technique, endless queues had already formed by load 7.9. This means that the network cannot let through more than 7.9 messages in one time unit. When we use the proposed adaptive method, then the same network can let through 16.6 messages. This means that the proposed method allows a much higher load on the network, so that with the same load several channels can be saved.

Now we come to the third method. As mentioned earlier, we have so far only considered messages of equal importance. However, such an assumption very seldom corresponds to clients' demands. I should like to illustrate this problem by a couple of examples. If, for instance, one sends a greetings telegram for a birthday, then this telegram should arrive on the birthday. It should certainly not arrive earlier and the later it arrives the less pleasure it gives. In other words, its importance increases up to the birthday and then drops very quickly. A week after the birthday the telegram is completely unimportant. Another example--when one sends off information for the weather forecasting system, then this information should arrive within a fixed time at the weather center, in order to ensure that we do not get yesterday's weather forecast today. The number of such examples is unlimited.

In order to reflect the differing urgency of the messages in the control algorithms of the data communication system, one usually established priorities. A message with a higher priority therefore has precedence over a message with a lower priority. However, such a priority philosophy does not seem to be very suitable for large communication systems.

For example, it could happen that the system is fully blocked by obsolete messages which originally had a high priority. These messages, which have in the meantime lost all importance to the client, are however transmitted because of their high priority. Thus, the customers will subsequently receive only obsolete messages, since all new incoming messages with a lower priority will be transmitted only after a delay. In order to avoid such a development, one would have to introduce priorities that are not only dependent on the initial urgency of the message but also on its age and the entire state of the system. I must stress that the customer is not interested in the priority of his telegram, but only that his message arrives in the right place at the right time. Thus, his interests can more adequately be dealt with by a queuing discipline based on the altering importance of the messages.

Let us further assume that the customer is ready to pay according to the importance of the message at the moment of its arrival. Under this condition, our system is given a clear criterion for the quality of its work: the income earned in the system. Maximization of income means at the same time best satisfaction of the customer's needs.

This method, which takes into account the altering importance of the messages and the traffic, has many more complicated details than the former, and unfortunately it is not possible for me to explain them quickly. Basically, this method differs very little from the other. Instead of figures, which represented the expected delivery time, we have here functions which show us what income we can expect for the chosen queues, according to the age of the message.

Let us again look at the network with four nodes. We see that here, instead of the constant zero, we have basic (pay) functions. The other functions are estimated and help in choosing the direction. Exactly as in the earlier method a vector of the functions is established in every node, is sent to the neighbouring nodes and stored in the matrix.

This method was also simulated and compared to the method for minimization of delivery time. The results are very much dependent on the basic functions. When the functions drop off very slowly then the results are more or less equal to the previous method, but when they fall off very quickly then the latter method has many advantages.

It could also be mentioned that the latter method is equivalent to the method for minimization of delivery time when the basic function is linear. Of course this function should reach zero but not earlier than the maximum delivery time.

In this lecture I wished to present the basic principles of three adaptive and decentralized methods. The methods differ according to the criteria of the work, although they do show the same characteristics, e.g. great adaptability, low sensitivity and simple implementation.

It seems that the presented methods are far more general--or rather generalized--and can be applied to many other systems which need adaptive and decentralized control, e.g. to city traffic or control of production lines. I would be pleased if these methods appear interesting for the solution of someone's problems and am at your disposal for further talks and discussions.

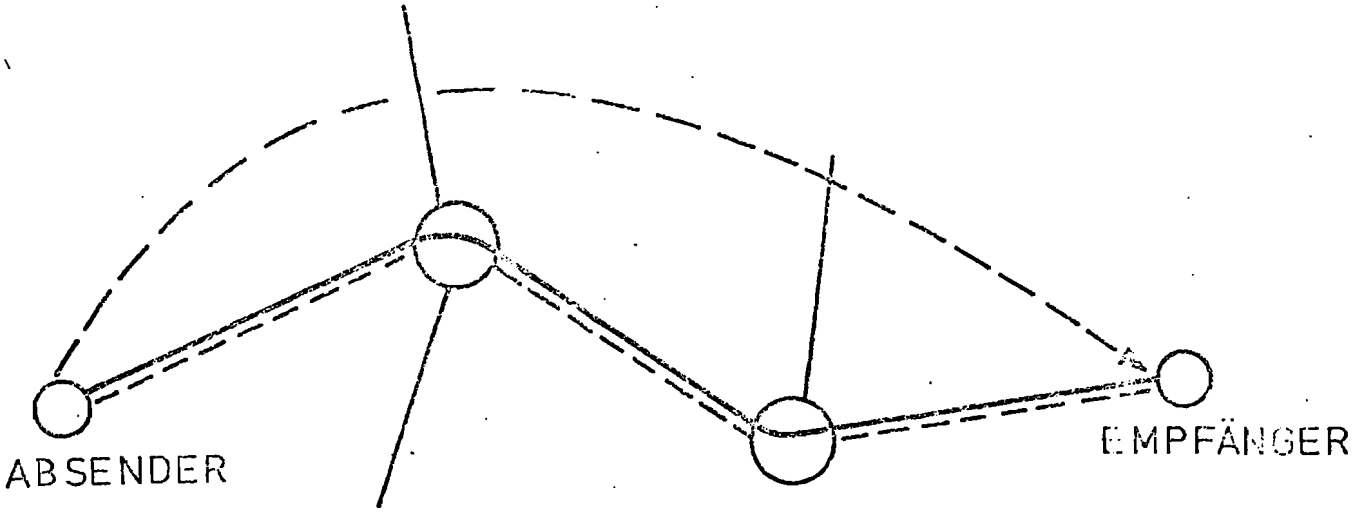


FIG. 1 CHANNEL SWITCHING

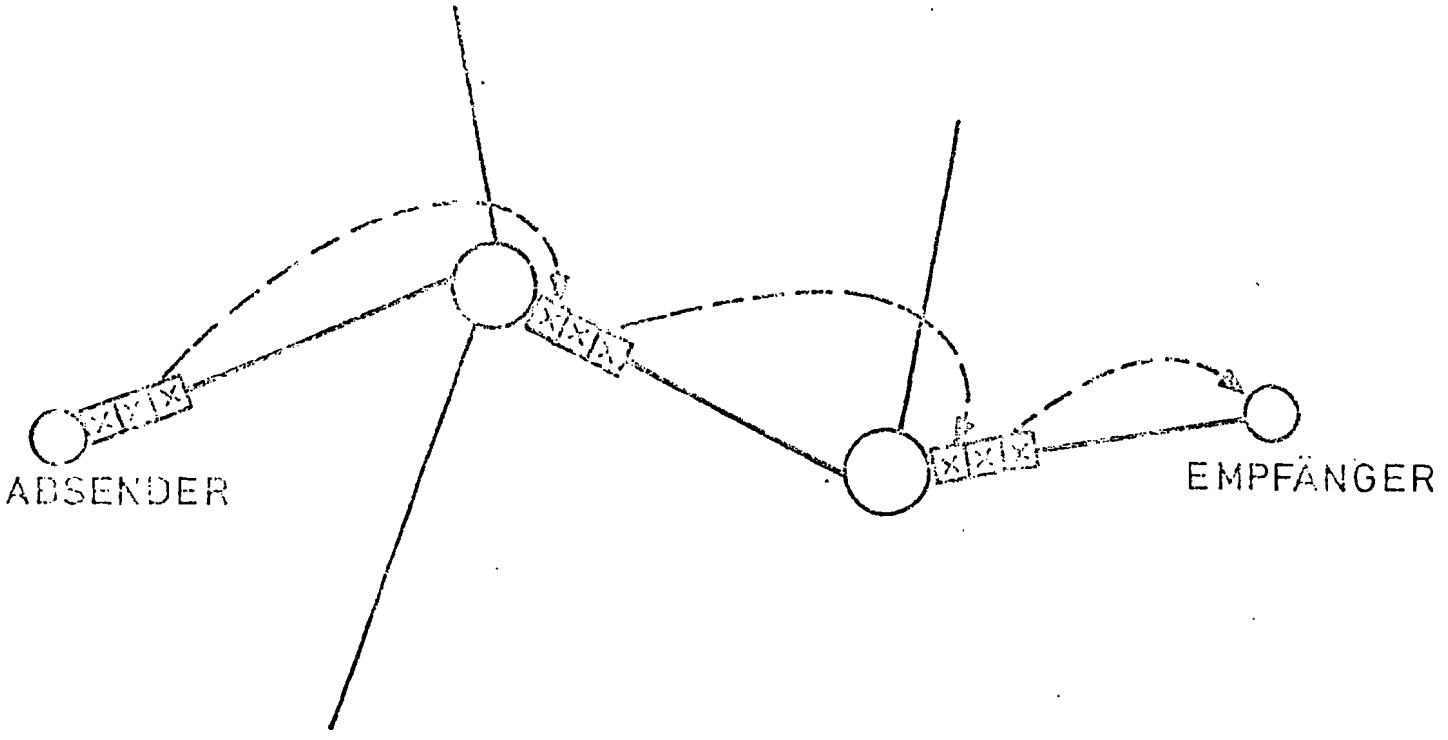


FIG. 2 STORE-AND-FORWARD (MESSAGE SWITCHING)

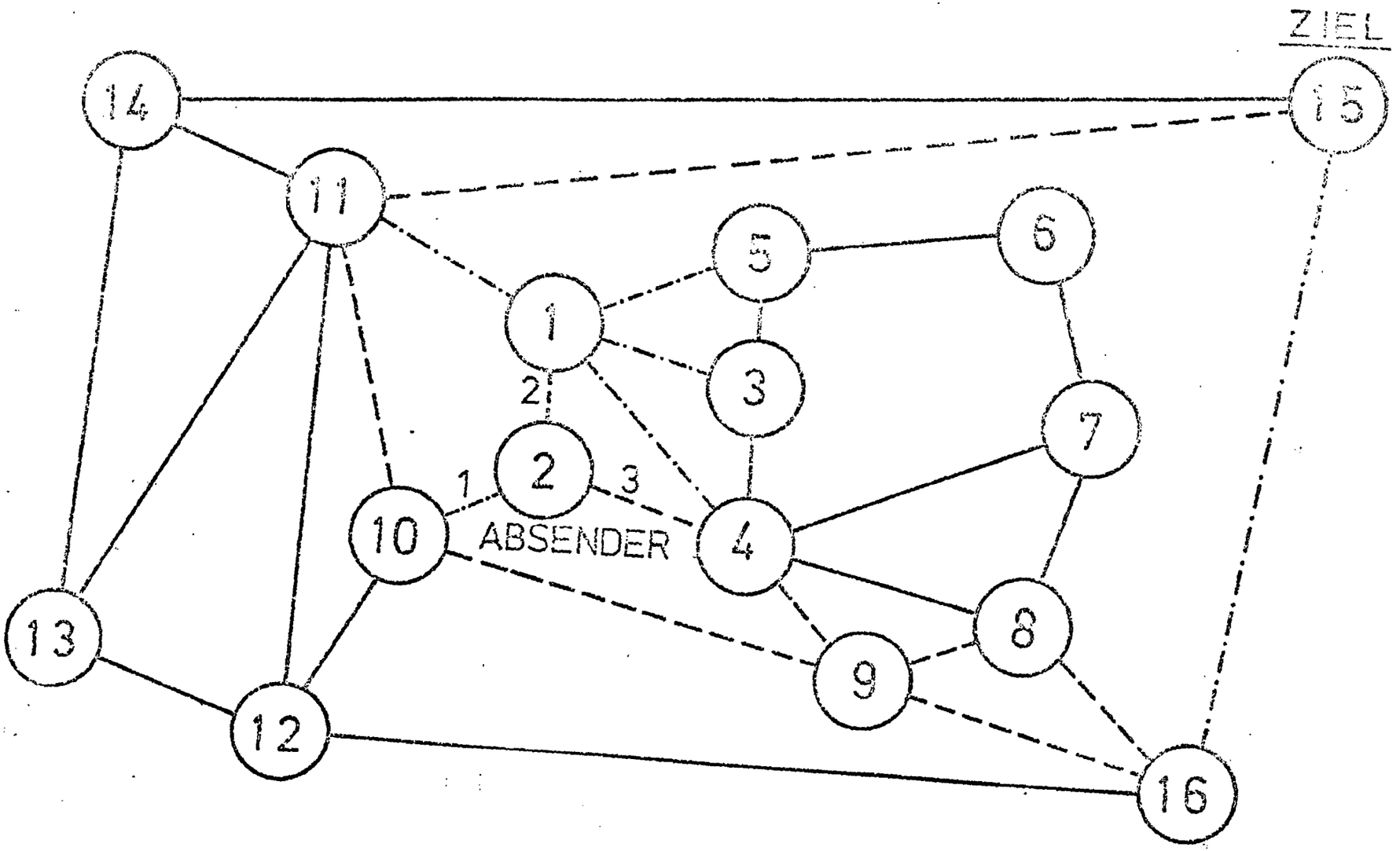


FIG. 3 AN EXAMPLE OF A NETWORK

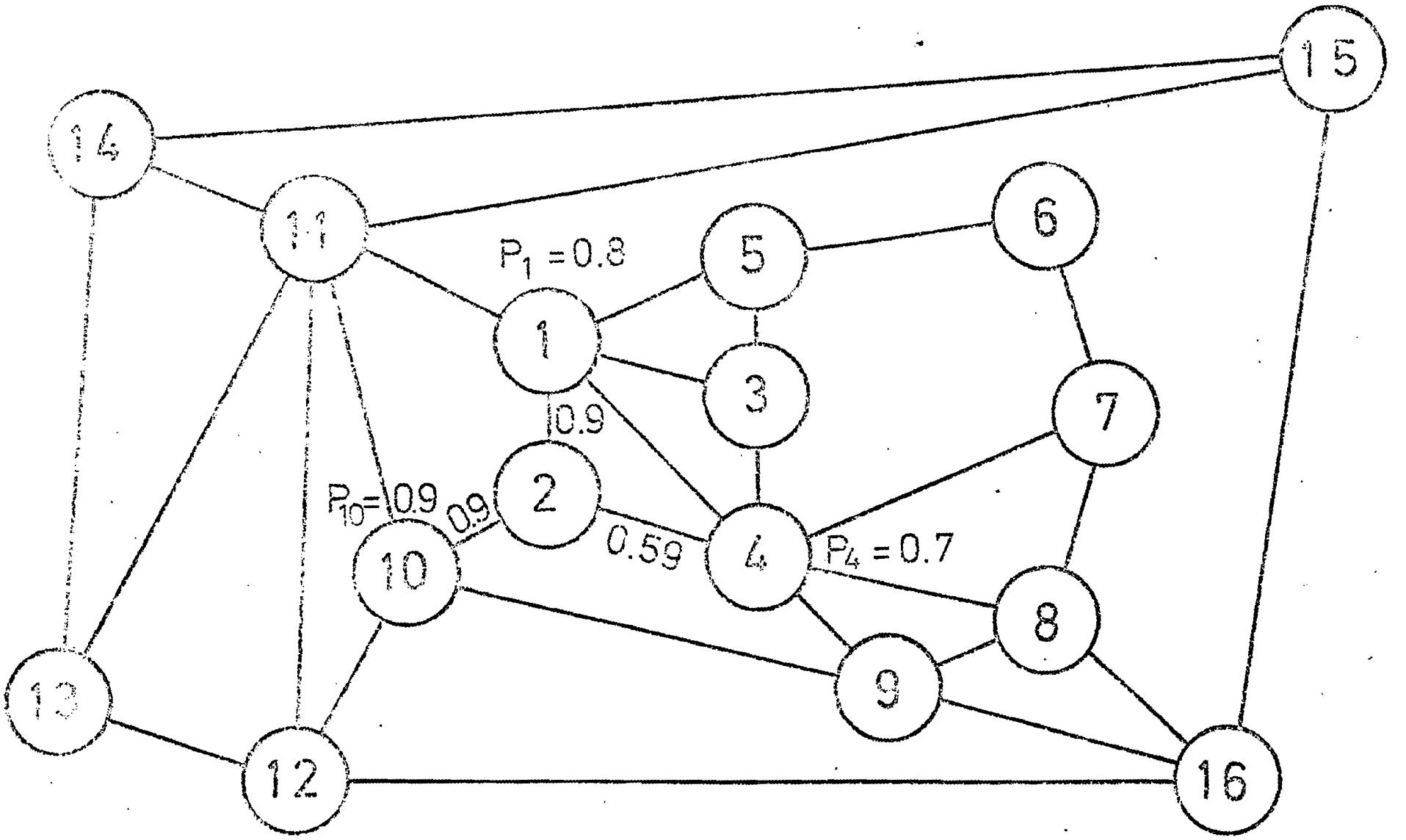


FIG. 4 AN EXAMPLE OF A NETWORK

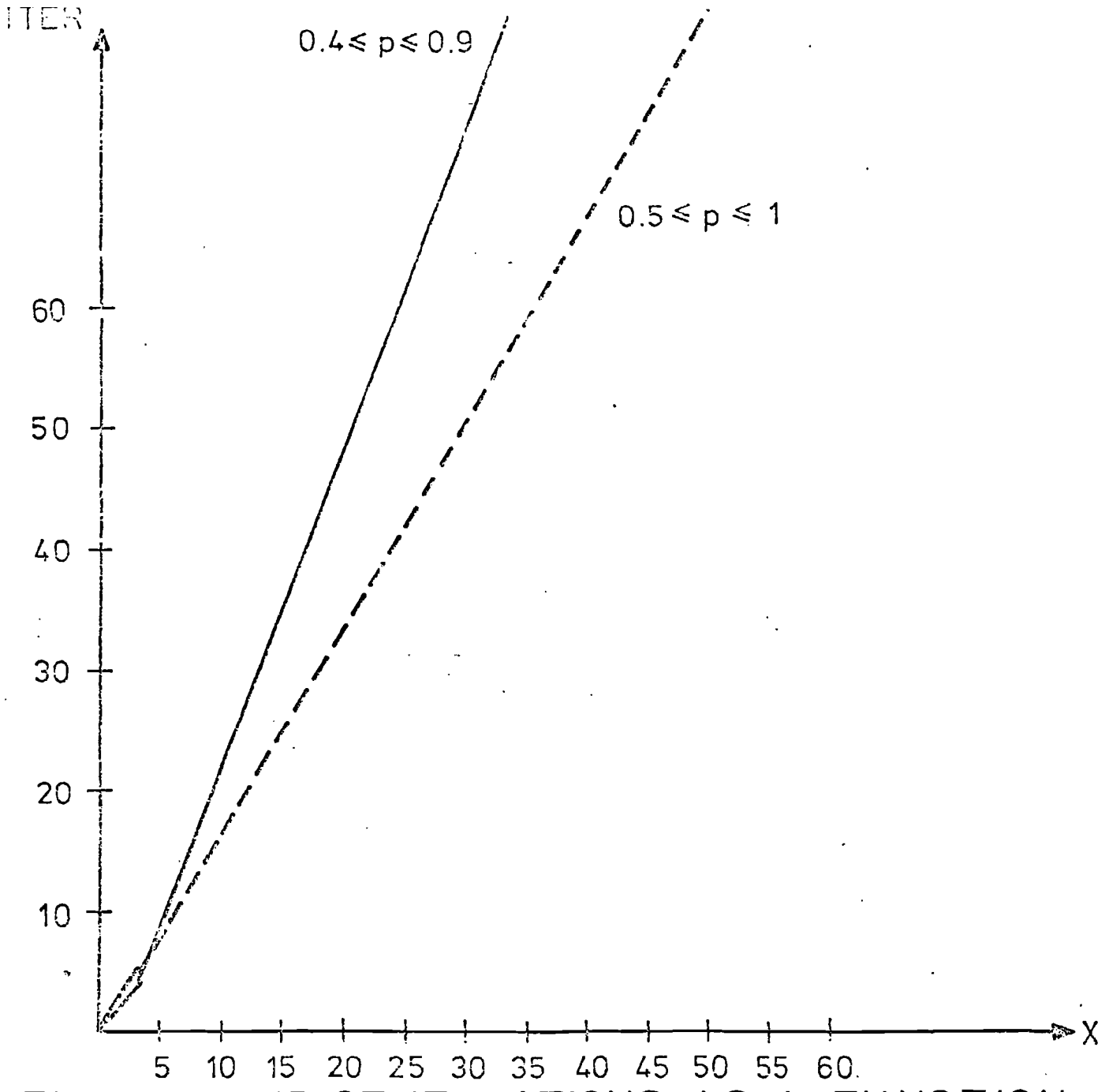
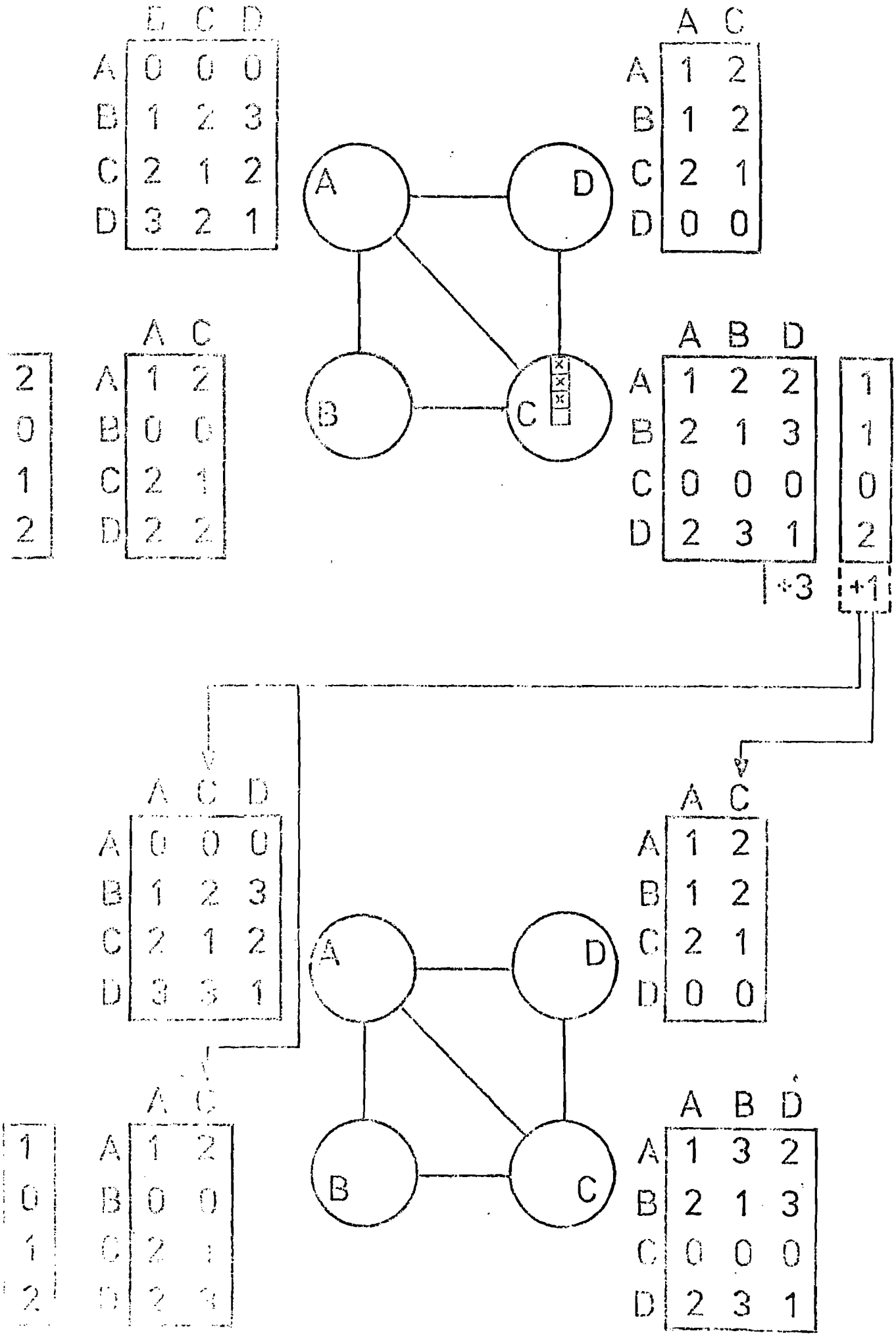


FIG. 5 THE NUMBER OF ITERATIONS AS A FUNCTION OF PRECISION = 2^{-X}


E C D
A 0 0 0
B 1 2 3
C 2 1 2
D 3 2 1
A C
A 1 2
B 1 2
C 2 1
D 0 0
A
D
B
C
A C
2
0
1
2
A 1 2
B 0 0
C 2 1
D 2 2
A B D
A 1 2 2
B 2 1 3
C 0 0 0
D 2 3 1
1
1
0
2
+3
+1
A C D
A 0 0 0
B 1 2 3
C 2 1 2
D 3 3 1
A C
A 1 2
B 1 2
C 2 1
D 0 0
A
D
B
C
A C
1
0
1
2
A 1 2
B 0 0
C 2 1
D 2 3
A B D
A 1 3 2
B 2 1 3
C 0 0 0
D 2 3 1

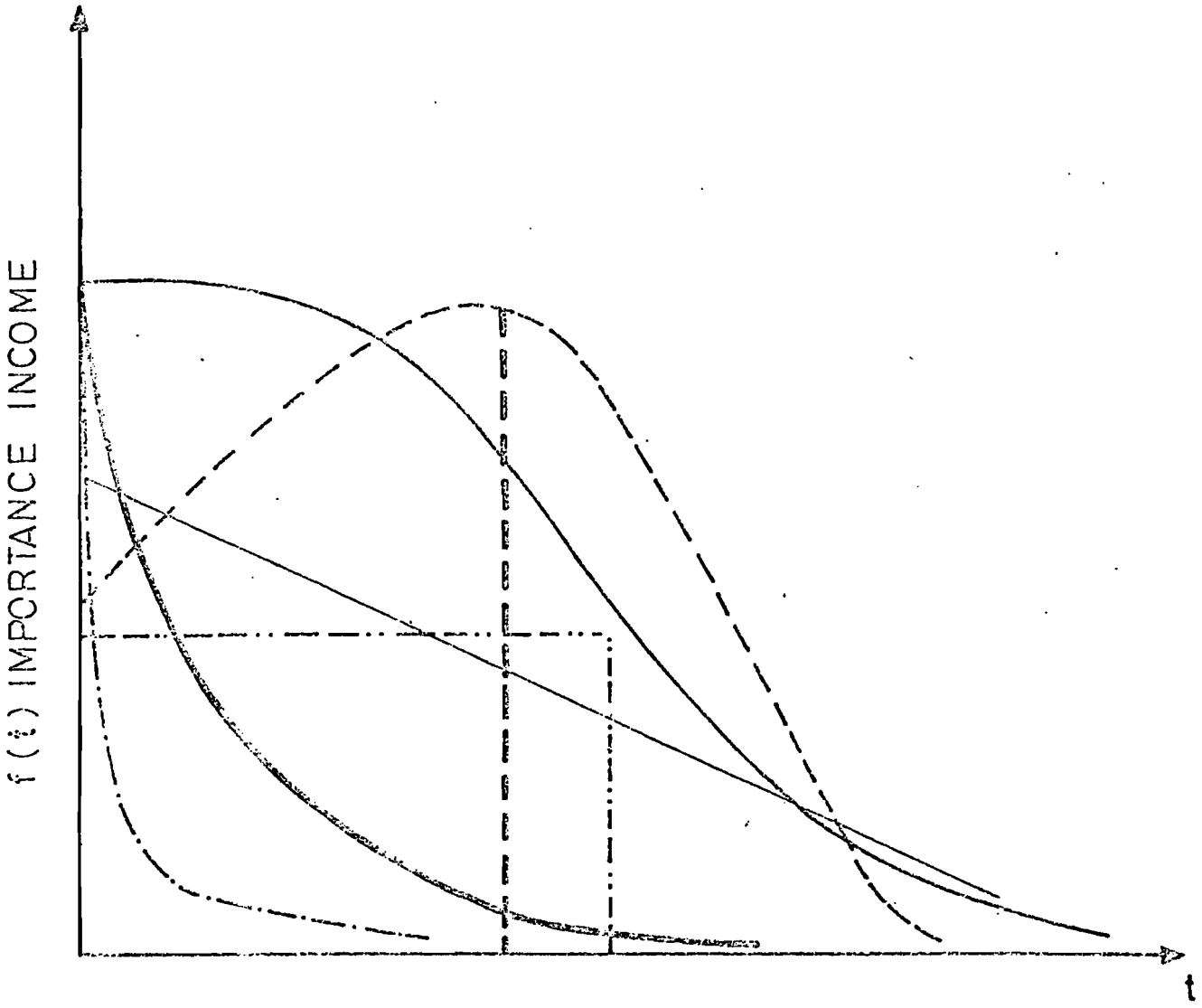


FIG. 7 DIFFERENT KINDS OF OUTDATING (PAY) FUNCTIONS

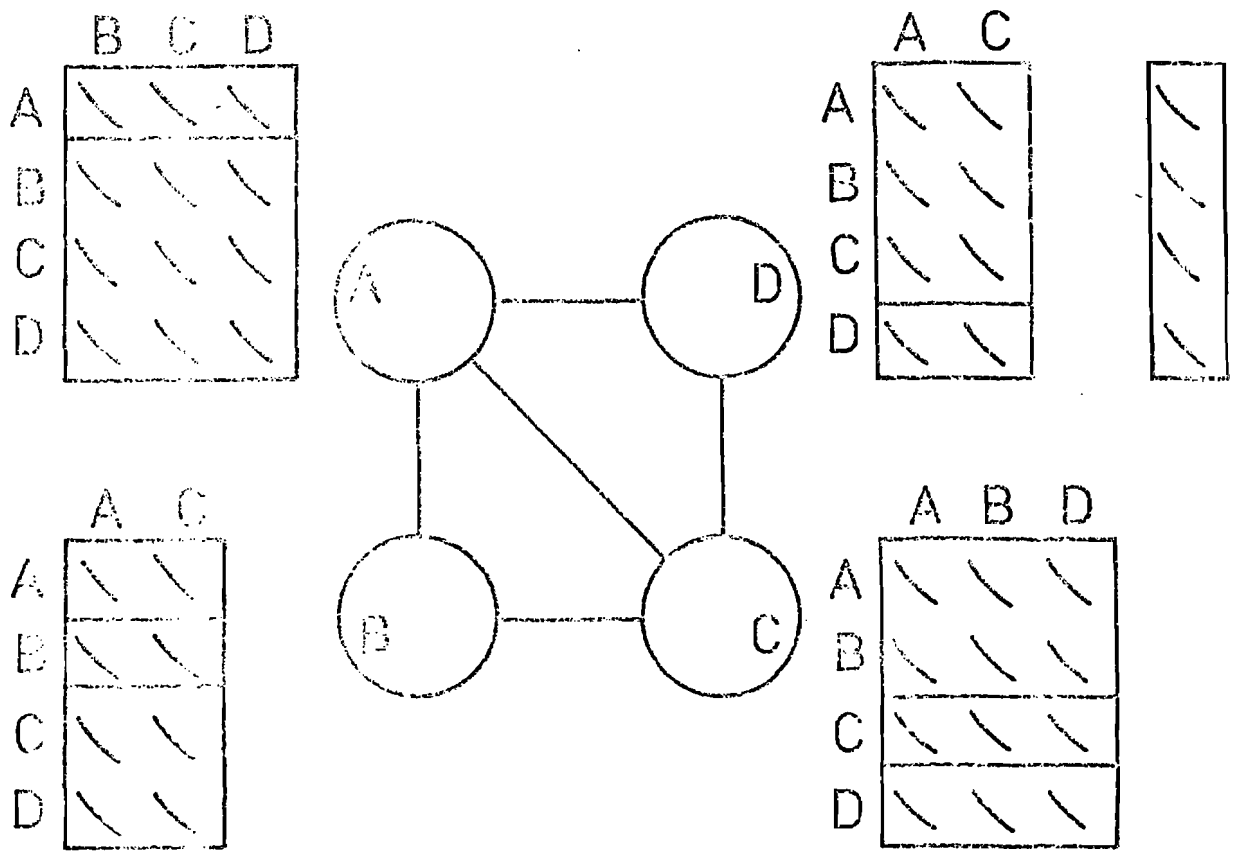


FIG. 8