<Research paper>

Journal of the Korean Institute of Industral Engineers
Vol. 3, No. 2, December 1977

An Algorithm for Changing Network Configurations

Young Hui Kim*

ABSTRACT

PERT/CPM type scheduling systems use a network to graphically portray the interrelationship among the activities of a project. This network representation of the project schedule shows all the precedence relationship regarding the order in which tasks must be performed.

In this paper an algorithm is presented that changes the network configurations without violating precedence relationship of the original network.

Introduction

The critical path problem takes the simplest from when we have unlimited resources available for a given project. In this case the problem reduces to finding the critical path through the network of the project and computing the project completion time. However, in reality, many situations arise when we have to deal with limited resources and restrictions on time. Basically these situations can be classified into the following two cases:

(1) The project completion date is not negotiable; however, we can increase the scarce resources at certain cost. (say-by doing some overtime work, employing additional manpower or subcontracting etc.). In this case, we are essentially buying time along the critical path, so as to meet the projet completion time. Then the problem is to find the minimum project cost.

(2) The increase of resources is not permissible; however, the project completion date is negotiable. In this case we are seeking the minimum project completion time under the resource constraint. This is the type of problem we will discuss further in this paper.

Any project can be represented as a directed acyclic network. Let this network consists of n nodes and a arcs, and denote it by G (N,A). To each arc a_i ($i=1, 2\cdots\cdots, a$), which represents the activities of the project, there associates two variables r_i and t_i which represents resource and time requirement for that activity.

Furthermore, let R_0 be the limit of resource available and $max\ R(G)$ be the maximum resource requirement at any point in time during the project. Where $R(G)$ is some function of G and r_i. Now, let us assume that the configuration of the graph (or network) can be changed according to certain rules (these rules will be discussed later), without violating technical interdependencies amongst activities. Then, our problem may be stated as finding $G^*(N,A)$ that minimizes the project completion time $T(G)$, under the restriction of $R_0 \geq max\ R(G)$. Where $T(G)$ is some function of

*Korea University

G and t_i.

Changing the Network Configuration

In order to solve the problem previously stated, we must be able to generate all the possible configurations of given network that does not violate the interdependencies amongst activities. We will first consider some basic properties of the network.

Any directed acyclic network is composed of the following three basic types of elements (subnet).

a. Series subnet.

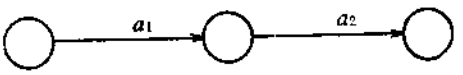
b. Parallel subnet.

c. Cross-connecting subnet.

In the following we will examine the effects of changing the configuration of the network in each of these subnets.

Postulate 1: Change of configuration in the series subnet is not permissible.

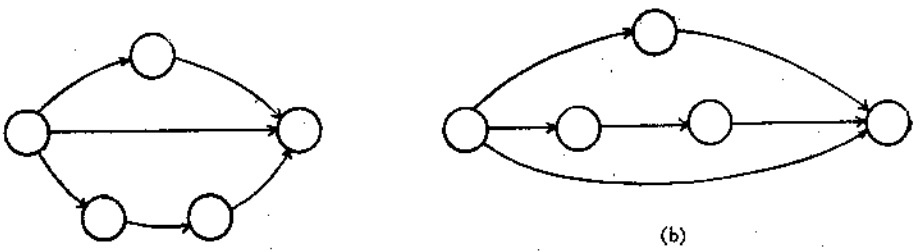
In Figure 1 activity a_2 is directly depend upon a_1, *i.e.*, activity a_1 must be completed before activity a_2 can start. Therefore in a series subnet, change of configuration would violate the technical interdependencies.



Fieure 1

Postulate 2: Change of configuration in the parallel subenet has no effect on the properties of network.

Figure 2 illustrates typical parallel subnet. As it is illustrated in this figure, interchange of parallel elements do not bring any changes to graph. From graph theoretical point of view, they are equivalent graphs. Therefore, in the parallel subnet, interchange of parallel elements has no effect on the configuration of the network.



Figure

Postulate 3: Configuration of a network can be changed in the cross connecting subnet without violating technical interdependencies.

For example, in Figure 3, arc (1, 4) may be replaced by (1, 3) or (1, 2) represented by dotted lines. Such a changes do not violate technical interdependencies, since replacement of arc (1, 4) by (1, 3) or (1, 2) means that the given activity (1, 4) will be finished earlier than it is actually required. However, replacing arc (1, 4) by (1, 3) would impose additional restrictions on node 3.

In the network given in Figure 3, the grapht $G(N,A)$ forms a cross connecting subnet, where $N=(1, 2, 3, 4)$, $A=\{(1, 2), (2, 3), (3, 4), (1, 4)\}$. In this subnet, arc (1, 4) is a cross connecting

arc which is a candidate to be replaced. Note that when cross connecting arc (1, 4) is eliminated, the cross connecting subnet reduces to a series subnet.

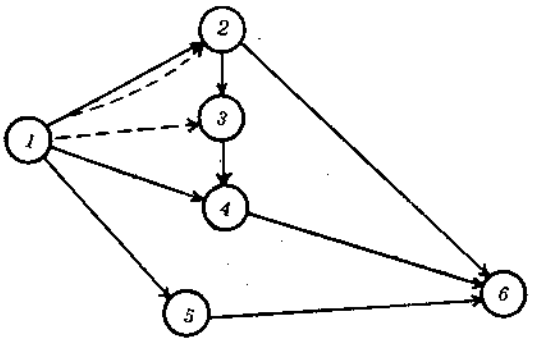


Figure 3

Based on the previous observations, we conclude that cross connecting subnets in a given network are the candidates for the change of configuration. Therefore, to generate all possible configurations from the original network, we need to proceed the following steps.

(1) Search for cross connecting subnets in the network.

(2) Scan for the number of predecessor nodes. In Figure 3, (1, 4) is cross connecting arc and 3, 2 are predecessor nodes.

(3) Generate new configuration by replacing the original cross connectsng arc with alternative cross connecting or parallel arc. In Figure 3, for example, replace (1, 4) by (1, 3) or (1, 2). Repeat this procedure until all the predecessor nodes are exhausted.

(4) GO TO (1) and repeat steps through (1)—(4) until all the cross connecting subnets in the given network are exhausted.

In order to identify cross connecting subnets in a network, depict the network as a tree (ref.1) Then the occurrence of two or more nodes of the tree with identical labels, n_j, corresponds to the entrance of two or more arcs into a node n_j in the network. Therefore if identically labelled nodes of the tree all branch from a common node n_i on the immediately preceding level, n_i is the source of a parallel subnet and n_j its sink.

In case where the parallel elements are composed of more than two arcs in series n_i refers to source and n_j the sink of the parallel elements.

On the other hand, if two of the nodes labelled n_j branch from a common node n_i at some level higher than that immediately preceeding and any arc branch out from an intermediate node on the path connecting n_i and n_j then n_j is the terminal node for a cross connection between two paths emerging from n_i.

An Algorithm

An algorthm to change configuration of a given network will be explained by taking a network illustrated in Figure 4.

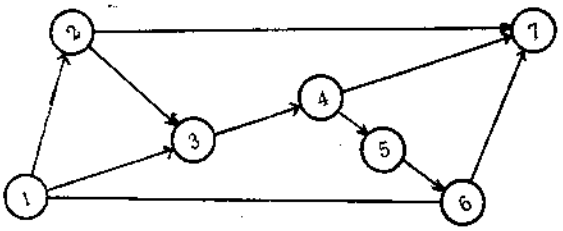


Figure 4

In numbering nodes defining activity(i,j), the number assigned to predecessor node, i, of an arc should be always less than the number assigned to successor node, j.

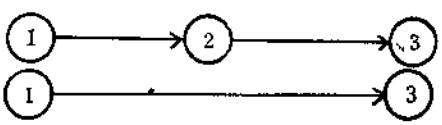
Step 1: Build the connectivity matrix of the network. The matrix of the network in Figure 4 is given in Figure 5.

Step 2: Starting from column 1 of the matrix, mark every column to which more than two arcs are entering. In our example, these columns are ③, ⑥, ⑦.

	1	2	3	4	5	6	7
1	0	1	1	0	0	1	0
2	0	0	1	0	0	0	1
3	0	0	0	1	0	0	0
4	0	0	0	0	1	0	1
5	0	0	0	0	0	1	0
6	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0

Figure 5

Step 3: Check if each of nodes marked in Step 2 is terminal node for cross connecting subnet. In column 3 of the matrix, two arcs are entering node 3, *i.e.*, we have,



Furthermore, an arc is branched out from node 2 to node 7. Note '1' in the cell$(2,7)$. Therefore node 3 is a terminal node of a cross connecting subnet.

Similarly, we find nodes 6, 7 are terminal nodes of cross connecting subnets.

Step 4: Check predecessor nodes of the terminal nodes for each of the cross connecting subnets, and candate arcs to be replaced.

In Figure 4, we have,

terminal node of cross connection	predecessor node	candidate arc to be replaced
3	2	(1, 3)
6	5, 4, 3, 2	(1, 6)
7	6, 5, 4, 3	(2, 7)
	6, 5	(4, 7)

Step 5: Starting from the cross connecting terminal node that has the largest number, generate a new alternative by replacing the candidate arc with alternative arc that terminates at a predecessor node of the terminal node previously checked.

In the example given, starting from column 7, we replace (4, 7) by (4, 6) thus creating the following matrix.

	1	2	3	4	5	6	7
1	0	1	1	0	0	1	0
2	0	0	1	0	0	0	1
3	0	0	0	1	0	0	0
4	0	0	0	0	1	1	1
5	0	0	0	0	0	1	0
6	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0

Figure 6

Next we create another new network by replacing (4, 6) by (4, 5).

We continue this procedure until all the possible configuration of the given network is obtained.

Summary and Conclusion

In this paper an algorithm to change a configuration of a given network is presented. By applying this algorithm, it is possible to generate a set of networks that do not violate precedence relationship of the original graph.

Depending on the situation, one way to formulate a network scheduling problem is that of minimizing the project completion time under resource constraints. Solution of such a problem would require understanding relationship between resource requirement and the network structure.

It is hoped that the algorithm presented here provide one method of analyzing the network in the context of problem stated above. However, further study is required before the algorithm could effectively be used for such a purpose.

Reference

J.J Martin, "Distribution of the Time Through a Directed Acyclic Network." Journal of ORSA, Vol. 13, pp. 46–66(1965)