

**ON A PARTITION ALGORITHM FOR CRITICAL-PATH
METHOD**

TATSUO AONUMA

Kobe College of Economics

(Received Feb. 24, 1964)

§1. INTRODUCTION

In critical-path scheduling sometimes some method of activity aggregation is used for reducing the effort required to solve extremely large project scheduling problems. Quite often large portions of a project diagram form individual projects in their own right. They connect into the whole project at only two points, their origin and terminus, respectively. If each of these subprojects may be replaced by a single activity with the same utility function and duration limits, major projects will be simplified to solve however large they may be. One method for this purpose is stated in Kelley's paper [4]. Our paper also is concerned with establishing an alternative algorithm to solve easily large project scheduling problems by Critical-Path Method, where the project has a special structure that it can be partitioned into several or many subprojects.

In Kelley's aggregation method every subproject is solved separately as one CPM problem in advance and a group of its solutions, project duration limits, and a utility function obtained in each case are used as duration limits and a utility function for the corresponding aggregated activity. However, as the utility function for an aggregated activity

obtained in this manner is nonlinear (piecewise linear) and concave, in order to reduce it to the linear case the subproject must be essentially replaced by a series or parallel¹⁾ of r activities with a linear utility function, but not by one activity, in the project diagram. So it seems that Kelley's aggregation method is not suitable enough for automatic processing by the fact that the method has two computational phases, solving the subproject problems, and describing the renewed arrow diagram and then solving it, where it requires the unpredictable number of activities, denoted above by r .

It appears to us from our experience by this time that a computer equipped with tape units is necessary to solve a critical-path scheduling problem of practical size. So before establishing our algorithm we assume that a computer with tape units is always available for CPM computation, and that the possibility of partitioning a project into several subprojects is admitted for all projects that we shall deal with.

In both Kelley's and Fulkerson's CPM algorithms a principal part of computation will be the labeling process. The first aim to establish our algorithm is to reduce the time required in the labeling process as much as possible by making use of the assumed special structure of the matrix. The second aim is to make it possible to solve even a larger project scheduling problem by a computer with limited fast access working storages (for example, core storage and/or magnetic drum storage, etc.). The fast access storages are mainly used for computational operations in the computer with tape units, so the economical and computable size of the project (i.e., the maximum possible number of activities and events in the project) depends on the size of this storage. However large the size of the original project may be, we shall be able to solve its scheduling problem using our partition algorithm stated in this paper, if only the size of every subproject obtained by partitioning the project is

1) Only the series technique to handle a nonlinear CPS problem is stated in Kelley [4]. The parallel technique is illustrated in Sekine [5].

within the limit.

In Section 2 we shall deal with partitioning a project. It will be a preliminary work required for our algorithm. And in Section 3 two basic problems in our algorithm, called Partitioned Network Problem and Aggregated Network Problem, respectively, and the detailed computational steps of our algorithm will be given. In the final section a simple example will be solved by means of our algorithm.

§ 2. PARTITIONING A PROJECT

That a project P can be partitioned is that P includes at least one subproject P_k . An origin o_k and a terminus n_k belong to P_k , which connects into the whole project P at only two events, o_k and n_k , respectively. If a portion of activities and events between these two events are completely separated from P when we exclude these two events from P , then the project P will be able to be partitioned as well.

We assume that P includes m subprojects, $P_k(k=1, 2, \dots, m)$, which have no activity in common with each other. As some of the origins and termina may be common to several subprojects, each number of them will be generally less than m , and the case of $P_i \cap P_j \neq \emptyset (i \neq j)$ may occur. This fact shows that an arrow diagram obtained after the aggregation of

Fig 1 A project with two subprojects

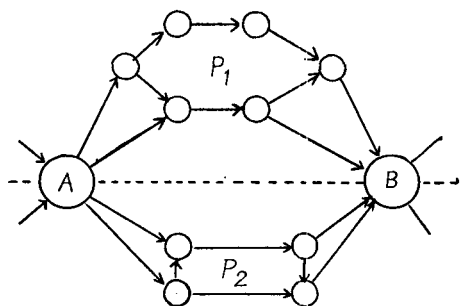
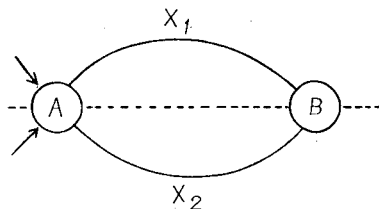


Fig 2 An aggregated project



subprojects may not satisfy the fundamental rule of it. This situation is illustrated in Fig. 1. Two groups of activities above and below the horizontal dotted line between an event A and an event B form subprojects respectively. Their common origin is the event A and their common terminus B . If we substitute two aggregated activities, X_1 and X_2 , for these two groups, an arrow diagram which does not satisfy the fundamental rule, as shown in Fig. 2, will be obtained.

In order to eliminate the inexpedience mentioned above, we shall establish a partitioning rule as follows:

Partitioning Rule

1) For every k replace the origin and the terminus of P_k by events \bar{o}_k and \bar{n}_k which satisfy the conditions, $\bar{o}_i \neq \bar{o}_j$ and $\bar{n}_i \neq \bar{n}_j$ for $i \neq j$, and $\bar{o}_i \neq \bar{n}_j$ for all i and j . Each of several activities, (o_k, \cdot) 's and (\cdot, n_k) 's, is replaced by (\bar{o}_k, \cdot) 's and (\cdot, \bar{n}_k) 's respectively. The origin of the new subproject \bar{P}_k obtained in this manner is \bar{o}_k and its terminus is \bar{n}_k .

2) Introduce two dummy activities, (o_k, \bar{o}_k) and (\bar{n}_k, n_k) . A utility of zero and durations of zero are assigned to these two activities. These activities will be called connecting activities.

3) Perform a rearranging operation, i.e., topological ordering of a list of new events, $E \sim \{\bar{o}_k, \bar{n}_k, k=1, 2, \dots, m\}$, where E is an event set of P , on the basis of the partial ordering relation defined by the new arrow diagram to which the connecting activities and events $\{\bar{o}_k, \bar{n}_k,$

$k=1, 2, \dots, m$ have been newly added. Then we shall obtain a new project \bar{P} .

These two P and \bar{P} are essentially equivalent. \bar{P} has the following property, which follows readily from the way in which \bar{P} has been generated.

Theorem 1. A rearranged project \bar{P} can be written

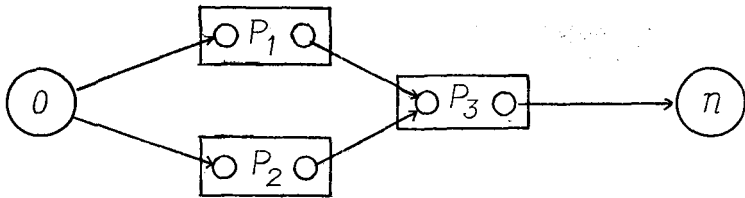
$$\bar{P} = \bigcup_{k=1}^m \bar{P}_k \smile (\text{connecting activities}) \smile R,$$

where $\bar{P}_i \smile \bar{P}_j = 0$ ($i \neq j$), and $R = (P - \bigcup_{k=1}^m P_k) \smile \bigcup_{k=1}^m (o_k : n_k)$.

Especially, in case of $R = \bigcup_{k=1}^m (o_k : n_k)$ P is called "totally partitioned," and the project consists only of its subprojects and connecting activities.

Let there be $n+1$ events in \bar{P} . The origin of \bar{P} is given the label o and the terminus is given the label n . Only the project P is required for our algorithm.

Fig 3 An example of a totally partitioned project



§ 3. PARTITION ALGORITHM

The mathematical model upon which Critical-Path Method is based is given as follows²⁾:

Find y_{ij} and t_i , $(i, j) \in \bar{P}$, that maximize the utility function

2) The notation corresponds to that used in Kelley [4].

$$\sum_{(i,j) \in \bar{P}} (a_{ij}y_{ij} + b_{ij}) \quad (1)$$

subject to

$$y_{ij} + t_i - t_j \leq 0, \quad (i, j) \in \bar{P}, \quad (2)$$

$$t_n - t_0 \leq \lambda, \quad (3)$$

$$0 \leq d_{ij} \leq y_{ij} \leq D_{ij}, \quad (i, j) \in \bar{P}. \quad (4)$$

Next we shall state the outline of Kelley's algorithm :

(i) Generating the initial schedule.

The schedule $\{\mathbf{y}, \mathbf{t}\}$, defined by $y_{ij} = D_{ij}$, $(i, j) \in \bar{P}$, $t_0 = 0$, $t_j = \max_{(i,j)} (y_{ij} + t_i)$, $1 \leq j \leq n$, is an optimal schedule, for $\lambda = t_n$.

(ii) Solving the network flow problem using labeling process.

$$\text{Maximize } \sum_{(i,n) \in Q_1} u_{in} \quad (5)$$

subject to

$$\sum_{(i,j) \in \bar{P}} u_{ij} - \sum_{(i,k) \in \bar{P}} u_{jk} = 0, \quad 1 \leq j \leq n-1, \quad (6)$$

$$0 \leq u_{ij} \begin{cases} \leq a_{ij}, & (i, j) \in Q_1 \sim Q_2 \\ = a_{ij}, & (i, j) \in Q_1 - (Q_1 \sim Q_3 \sim Q_4) \\ \geq a_{ij}, & (i, j) \in Q_1 \sim Q_4 \\ = 0, & (i, j) \in \bar{P} - Q_1, \end{cases} \quad (7)$$

where Q_i 's ($i=1, 2, 3, 4$) are the same ones that are defined in [4].

(iii) Solving the restricted dual problem of (1)–(4).

$$\sigma_{ij} = \begin{cases} 1, & \text{if } (i, j) \in Q_1 - (Q_3 \sim Q_4) \text{ and } i \in I, j \in J \\ -1, & \text{if } (i, j) \in Q_1 - (Q_2 \sim Q_3) \text{ and } i \in J, j \in I \\ 0, & \text{otherwise,} \end{cases} \quad (8)$$

and

$$\delta_i = \begin{cases} 0, & i \in I \\ 1, & i \in J \end{cases} \quad (9)$$

constitute an optimal solution of the problem, where I is the set of labeled nodes and J the set of unlabeled nodes obtained at the termination of the labeling process.

(iv) Changing the present schedule to a new schedule.
 The schedule $\{y', t'\}$, defined by

$$\begin{aligned} y'_{ij} &= y_{ij} - \theta \sigma_{ij}, \quad (i, j) \in \bar{P} \\ t'_i &= t_i - \theta \delta_i, \end{aligned} \tag{10}$$

is an optimal feasible schedule of duration $\lambda' = \lambda - \theta$, where $0 \leq \theta \leq \theta_0$ as defined in [4].

Now we shall consider the following two problems, called Partitioned Network Problem and Aggregated Network Problem respectively.

Partitioned Network Problem (PNP). Consider the following two types of network flow problem for each of the subnetworks N_k 's corresponding to the subprojects \bar{P}_k 's. The capacity restrictions of each arc and the flow conservation equations in these flow problems are the same as obtained from equations (6) and (7) by restricting all (i, j) within \bar{P}_k .

PNP-I. Find $u_{ij}, (i, j) \in \bar{P}_k$, that maximize the total flow $\sum_{j \in \bar{P}} u_{\bar{n}_k j} = \sum_{i \in \bar{P}_k} u_{i \bar{n}_k}$ subject to the network restrictions stated above using the labeling procedure. Then we denote the max-flow of the network by F_k , the set of labeled nodes by I_k , and the set of unlabeled nodes by J_k at the termination of the labeling process. The min-cut of the network will be represented by (I_k, J_k) .

PNP-II. Find $u_{ij}, (i, j) \in \bar{P}_k$, that minimize the total flow of the same network as in PNP-I, using the following modified labeling rule.

1. Let the initial feasible solution to the subnetwork N_k be $\{u_{ij}^o\}$, where $\{u_{ij}^o\}$ is an optimal solution to PNP-I of the network N_k . At first, label terminus \bar{n}_k with the label $(-, \infty)$, and then proceed to attach a label to each node using the same labeling rule as used in (I). The backward labeling will be mainly used. If origin \bar{o}_k is labeled, proceed to 2. If the origin is not labeled, this algorithm terminates, the minimum flow having been obtained.

2. If origin \bar{o}_k is labeled $(-p, h)$, change the present flow to a

new flow along the labeled chain. Namely, if node k is labeled $(-j, \cdot)$, replace u_{kj}^o by $u_{kj}^o - h$. If node k is labeled $(+j, \cdot)$, replace u_{jk}^o by $u_{jk}^o + h$. Now proceed in the same manner to consider node j . Eventually terminus \bar{n}_k will be reached. At that time this flow change terminates. Using the new value of u_{ij} and erasing all labels, Procedure 1 is repeated.

Then we denote the min-flow of the network by f_k , the set of labeled nodes at the termination of the labeling process by \tilde{I}_k , and the set of unlabeled nodes at that time by \tilde{J}_k . "Max-cut" will be represented by $(\tilde{I}_k, \tilde{J}_k)$. It will be seen from Theorem 4 to be mentioned below that we shall have not to solve PNP-II more than once throughout the whole computation.

Now we can always obtain expressions $\Delta_k = [(I_k, J_k), F_k; (\tilde{I}_k, \tilde{J}_k), F_k]$ for every subnetwork N_k as the result of the two procedures above.

Aggregated Network Problem (ANP). In a network N corresponding to the project \bar{P} , replace every subnetwork N_k by a single aggregated arc (\bar{o}_k, \bar{n}_k) with a maximum capacity F_k and a minimum capacity f_k , respectively. As a result of it, a new aggregated network N^* will be obtained. Then solve the network flow problem for N^* using the labelling process. In the network flow problem defined by (5)–(7), if we replace all the restrictions, which include flow variables u_{ij} of each subnetwork N_k as their variables, by the following only three restrictions,

$$\left. \begin{aligned} u_{\bar{o}_k \bar{o}_k} - u_{\bar{o}_k \bar{n}_k} &= 0, \\ u_{\bar{o}_k \bar{n}_k} - u_{\bar{n}_k \bar{n}_k} &= 0, \\ 0 \leq f_k \leq u_{\bar{o}_k \bar{n}_k} &\leq F_k, \end{aligned} \right\} (k=1, 2, \dots, m) \quad (12)$$

respectively, we shall be able to obtain the network flow problem for N^* . In the labeling process the aggregated arc with the capacity restriction, $f_k \leq u_{\bar{o}_k \bar{n}_k} \leq F_k$, will be able to be treated as follows;

(i) If \bar{o}_k is labeled and not yet scanned, consider it as an arc with capacity $0 \leq u_{\bar{o}_k \bar{n}_k} \leq F_k$, attaching some label to the unlabeled node \bar{n}_k .

(ii) If \bar{n}_k is labeled and not yet scanned, consider it as an arc with capacity $f_k \leq u_{\bar{o}_k \bar{n}_k}^o$, attaching some label to the unlabeled node \bar{o}_k .

(iii) Assume that every arc (\bar{o}_k, \bar{n}_k) always belongs to Q_1 .

Theorem 2. The max-flow F of the network N^* , obtained from the aggregated network flow problem, is equal to the max-flow of the network N . The min-cut of N^* can be always regarded as the min-cut of N , if only we replace every arc (\bar{o}_k, \bar{n}_k) belonging to the min-cut of N^* by either

- (i) the min-cut of N_k , (I_k, J_k) , in case of $\bar{o}_k \in I$ and $\bar{n}_k \in J$, or
- (ii) "max-cut" of N_k , $(\tilde{I}_k, \tilde{J}_k)$, in case of $\bar{o}_k \in J$ and $\bar{n}_k \in I$.

Proof. Let the max-flow of N be $F' = \sum_{i \in \bar{P}} u_{ij}^o$, where u_{ij}^o denotes an optimal solution to the flow problem for N . We shall prove $F' = F$ for each of the following four cases, which may occur at the optimal stage of the flow problem for N ;

Case (i). $\bar{o}_k, \bar{n}_k \in I$. We have always $f_k < u_{\bar{o}_k \bar{o}_k} = u_{\bar{n}_k \bar{n}_k} < F_k$ from the definitions of F_k and f_k . On the other hand, it follows from the labeling rule that the optimality of the solution to the problem for N is not changed by modification of N_k at the optimal stage if the modification does not change the values of $u_{\bar{o}_k \bar{o}_k}$ and $u_{\bar{n}_k \bar{n}_k}$. So F' does not change even though we replace N_k by an arc (\bar{o}_k, \bar{n}_k) with the capacity, $f_k \leq u_{\bar{o}_k \bar{n}_k} \leq F_k$.

Case (ii) $\bar{o}_k \in I, \bar{n}_k \in J$. We have the relation, $u_{\bar{o}_k \bar{o}_k}^o = u_{\bar{n}_k \bar{n}_k}^o = F_k$, in this case. Since \bar{n}_k also belongs to J because of $F_k - u_{\bar{o}_k \bar{n}_k}^o = 0$ even though we replace N_k by an arc (\bar{o}_k, \bar{n}_k) with the capacity, $0 \leq u_{\bar{o}_k \bar{n}_k} \leq F_k$, the optimality of the solution does not change.

Case (iii) $\bar{o}_k \in J, \bar{n}_k \in I$. We have the relation, $u_{\bar{o}_k \bar{o}_k}^o = u_{\bar{n}_k \bar{n}_k}^o = f_k$, in this case. Similarly, even though we replace N_k by an arc (\bar{o}_k, \bar{n}_k) with the capacity, $f_k \leq u_{\bar{o}_k \bar{n}_k}$, \bar{o}_k along belongs to J . So F' does not change as well.

Case (iv) $\bar{o}_k \in J, \bar{n}_k \in J$. Similarly it will clearly result that F' does

not change. In such a manner, a network which have been obtained by replacing at the optimal stage each of m subnetworks in the network N by arc (\bar{o}_k, \bar{n}_k) , respectively, is no more than N^* . So we have proved $F' = F$.

Next, we shall prove that the min-cut of N^* can be regarded as that of N . Let the flow on each arc (\bar{o}_k, \bar{n}_k) , obtained in ANP, be $u_{\bar{o}_k \bar{n}_k}$, respectively. Then such a feasible solution to the flow problem for N_k as its flow is just equal to $u_{\bar{o}_k \bar{n}_k}$, will be obtained in a way as indicated below :

(i) In case of $f_k < u_{\bar{o}_k \bar{n}_k} < F_k$. The feasible solution will be obtained, if we use a labeling rule modified as follows ;

1. Label origin with the label $(-, u_{\bar{o}_k \bar{n}_k})$, and after the p -th flow change, label origin \bar{o}_k with the label $(-, u_{\bar{o}_k \bar{n}_k} - \sum_{k=1}^p h_k)$, where h_k denotes h -part of the label of terminus \bar{n}_k labeled at the k -th labeling process. The other rules but the following one are as usual.

2. If $u_{\bar{o}_k \bar{n}_k} - \sum_{i=1}^p h_i = 0$, the labeling process terminates, and the flow at that time is the desired solution.

(ii) In case of $u_{\bar{o}_k \bar{n}_k} = F_k$. An optimal solution to the problem for N_k , obtained in PNP-I, is the desired solution.

(iii) In case of $u_{\bar{o}_k \bar{n}_k} = f_k$. An optimal solution to the problem for N_k , obtained in PNP-II, is the desired solution.

If we replace the flow on every aggregated arc in ANP by the corresponding feasible solution obtained above respectively, we have a solution which is feasible to the network N , and it is also an optimal solution. Since in case of (ii) and (iii) the aggregated arc belongs to the min-cut of N^* , (I_k, J_k) must be in the min-cut of N in case (ii), and $(\tilde{I}_k, \tilde{J}_k)$ in case (iii). This completes the proof.

Corollary 1. In ANP any entrance connecting arc, which means the arc (o_k, \bar{o}_k) for a subnetwork N_k , does not belong to any min-cut.

Because any entrance connecting arc always belongs to $Q_1 \sim Q_3$, and if $\bar{o}_k \in I$, o_k will also belong to I because of $u_{o_k, \bar{o}_k} = u_{\bar{n}_k, n_k} > 0$.

Corollary 2. In case of a totally partitioned project, the total flow F in ANP is infinite, whenever all F_k 's ($k=1, 2, \dots, m$) are infinite in PNP-I.

Because there is at least a chain, all activities in which belong to Q_1 and have an infinite capacity.

Theorem 3. In the aggregated network problem the expression Δ_k of any aggregated arcs which precede or succeed any arcs belonging to some min-cut is invariant by the succeeding schedule change.

Proof. It follows from equations (8) and (9) that σ_{ij} and δ_i , $(i, j) \in \bar{P}_\nu$, for any aggregated arc $(\bar{o}_\nu, \bar{n}_\nu)$, which precedes any arcs in min-cut, are zero. So $\{y_{ij}, t_i, (i, j) \in \bar{P}_\nu\}$ is not changed by the succeeding schedule change. For any aggregated arc $(\bar{o}_\mu, \bar{n}_\mu)$ which succeeds any arcs in min-cut, similarly we have $\sigma_{ij}=0$ and $\delta_i=1$, $(i, j) \in \bar{P}_\mu$. So $\{y_{ij}, (i, j) \in \bar{P}_\mu\}$, is not changed by the schedule change, but t_i and t_j , $(i, j) \in \bar{P}_\mu$, decrease by a same amount. In either case, Q -status of all activities included in the corresponding subprojects is not changed by the schedule change (Q -status of an activity means which set of Q_i 's the activity belongs to). This fact shows that the expression Δ_k of the arc remains as it was, after the schedule change. The theorem has been proved.

Generally, the labeling procedure must be repeated for all nodes in the network after every schedule change, but Theorem 3 shows that, in case our algorithm is applied to a partitioned project problem, we can omit the labeling process for the subnetworks which do not include any arcs belonging to min-cut after every schedule change.

Theorem 4. Let the min-flow and the max-flow of each subnetwork N_k obtained in PNP be f'_k and F'_k , respectively. And assume that those obtained in the preceding PNP were f''_k and F''_k , respectively. Then after having solved ANP, if $\bar{o}_k \in I$ and $\bar{n}_k \in J$, the new min-flow f_k to be solved in the next PNP-II will be equal to F'_k . Furthermore, we can consider the new $(\tilde{I}_k, \tilde{J}_k)$ as (J'_k, I'_k) , where I'_k is a set of labeled nodes and J'_k a set of

unlabeled nodes at the termination of PNP-I solved just before the schedule change. That is to say, we may regard new \tilde{I}_k as J'_k and new \tilde{J}_k as I'_k in a sense of set of nodes, without labeling.

If $\bar{o}_k \in J$ and $\bar{n}_k \in I$ at that time, the new min-flow f_k to be solved in the next PNP-II will be equal to f'_k , and the new $(\tilde{I}_k, \tilde{J}_k)$ can be regarded as $(\tilde{I}'_k, \tilde{J}'_k)$, where \tilde{I}'_k and \tilde{J}'_k are sets of nodes, specified as usual, corresponding to f'_k respectively.

Proof. At a stage of a schedule change, we may enumerate all possibilities for every arc belonging to min-cut as follows;

The status before a schedule change

- (i) $(i, j) \in Q_1 - (Q_3 \smile Q_4)$, $i \in I, j \in J$, $u_{ij} \leq a_{ij}$ or $= a_{ij} (u_{ij}^0 = a_{ij})$
- (ii) $(i, j) \in Q_1 - (Q_2 \smile Q_3)$, $i \in J, j \in I$, $u_{ij} \geq a_{ij}$ or $= a_{ij} (u_{ij}^0 = a_{ij})$
- (iii) $(i, j) \in (Q_1 \smile (Q_2 \smile Q_3))$, $i \in J, j \in I$, $u_{ij} \leq a_{ij}$ or $\leq \infty$, $(u_{ij}^0 = 0)$
- (iv) $(i, j) \in P - Q_1$, $i \in I, j \in J$, $u_{ij} = 0$ $(u_{ij}^0 = 0)$
- (v) $(i, j) \in P - Q_1$, $i \in J, j \in I$, $u_{ij} = 0$ $(u_{ij}^0 = 0)$

turn to,

after a schedule change

- (i) $(i, j) \in Q_1 \smile Q_4$ or $Q_1 - (Q_2 \smile Q_3 \smile Q_4)$, $u_{ij} \geq a_{ij}$ or $u_{ij} = a_{ij}$.
- (ii) $(i, j) \in Q_1 \smile Q_2$ or $Q_1 - (Q_2 \smile Q_3 \smile Q_4)$, $u_{ij} \leq a_{ij}$ or $u_{ij} = a_{ij}$.
- (iii) $(i, j) \in P - Q_1$, $u_{ij} = 0$.
- (iv) $(i, j) \in Q_1 \smile Q_2$ or $P - Q_1$, $u_{ij} \leq a_{ij}$ or $u_{ij} = 0$.
- (v) $(i, j) \in P - Q_1$, $u_{ij} = 0$.

It should be noted that the direction of the inequality in a capacity restriction of any arc belonging to min-cut has been reversed after a schedule change, as shown in the above cases. It $\bar{o}_k \in I$ and $\bar{n}_k \in J$, an arc (\bar{o}_k, \bar{n}_k) belongs to min-cut at the termination of ANP. Therefore, it is required only to show that any labeled node j in the case (ii) can not be labeled again at the termination of the modified labeling process in the next PNP-II to be solved after the schedule change. Because only such

arcs as in the case (ii) have the possibility that they might not belong to "max-cut" at the termination of the modified labeling, as we can see from their Q -status and capacity restriction modified after the schedule change. There exists at least an arc belonging to the min-cut in each of all paths from the terminus to node j . Otherwise, the terminus would have been labeled. The type of the arc belonging to the min-cut is either (I, J) or (J, I) . In case of (I, J) since Q -status of the arc is either $Q_1 - (Q_3 \cup Q_4)$ (case (i)) or $P - Q_1$ (case (iv)) before the schedule change, the node j can not be labeled by the modified labeling after the schedule change. In case of (J, I) since Q -status of the arc turns to either $P - Q_1$ (case (iii) and (v)) or $\{Q_1 \cap Q_2$ and $u^o_{ij} = a_{ij}\}$ (case (ii)) after the schedule change, the node j is also unlabeled. Thus, we can not reduce the total flow of the network problem, modified after the schedule change, less than the max-flow F'_k , and also it will be easily seen from the above arguments that $(\tilde{I}_k, \tilde{J}_k)$ may be regarded as (I'_k, J'_k) , though there may really be an unessential difference between them.

Also in case of $\bar{d}_k \in J$ and $\bar{n}_k \in I$ we can prove it in a quite similar way, but we shall omit the proof.

Now, by the several theorems mentioned above, we can establish a partition algorithm for Critical-Path Method as follows:

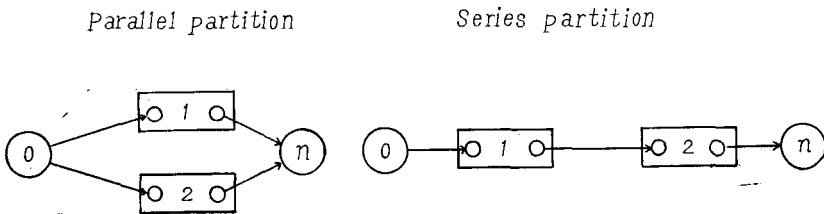
1. Reform a partitioned project \bar{P} from the original project P using the partitioning rule.
2. Obtain an initial optimal schedule by putting $y_{ij} = D_{ij}$, and then find Q -status for every activity.
3. Solve PNP's to obtain the expression Δ_k of every subnetwork N_k .
4. Solve ANP to find min-cut (I, J) of the network N .
5. Change the present schedule to a new schedule, which will be a new characteristic schedule.
6. Renew Q -status of all activities in R and in the subnetworks belonging to min-cut at the termination of the labeling process in ANP.
7. Solve PNP-I only for the subnetworks with Q -status renewed in 6. PNP-II for these subnetworks need not be solved, if only some

replacements as stated in Theorem 4 are done. Then we have a renewed expression J_k for each of these subnetworks.

8. Formulate a new ANP, on the basis of these J_k obtained in 7. Then turn to 4. We repeat these precedures from 4 to 8 untill the total flow F becomes infinite in ANP, or untill all F_k 's ($k=1, 2, \dots, m$) become infinite in PNP-I if the project is totally partitioned.

When a problem is solved by means of our partition algorithm, both the total number of events and that of activities increase, for convenience of partitioning a project, at most by $2m$ respectively more than when it is solved without partitioning, and that, in PNP we must use two kinds of labeling routines. However, it seems to us that they are not great disadvantages, since the increase of the numbers is quite insignificant in comparison with the number of events and activities in the subprojects and PNP-II need not be solved except at the initial stage, as shown in Theorem 4. On the other hand, this algorithm has a real advantage in that the labeling test to have been performed for all nodes is restricted only for a few groups of nodes in our algorithm. When we partition a project into several subprojects, it is more effective to partition into many series subprojects rather than to partition into parallel subprojects, as illustrated in Fig. 4.

Fig 4 Examples of parallel and series partitions



§4. A SIMPLE NUMERICAL EXAMPLE

The simple problem we shall solve by means of the partition algorithm is somewhat artificial, but will show the slight utility of the method even in case of simple partitions.

The example is given in an arrow-diagram form, Fig. 5. Figures on each arrows show the duration limits of the activity, and we assume that the decrease of the project cost per unit duration is two dollars for activities (C, E) and (D, E), and one dollar for activities (F, G) and (H, I). At event F, we can partition the project into two subprojects, P_1 and P_2 ,

Fig.5 An example of a project

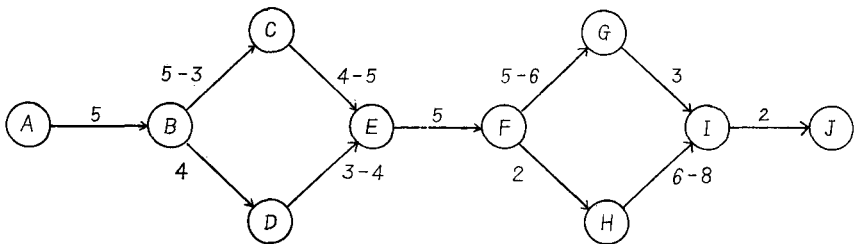
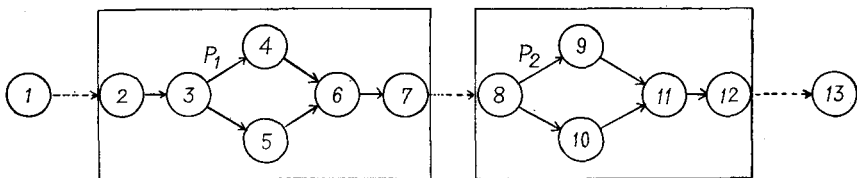


Fig.6 A partitioned project of the example

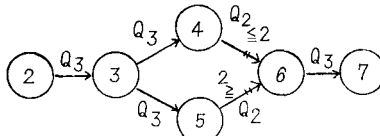


respectively. The partitioned project becomes as shown in Fig. 6. Our algorithm will be applied to this project.

Beginning with an initial feasible schedule constructed from the normal durations, the major data generated in the partition algorithm solution of this problem are given in the tables below.

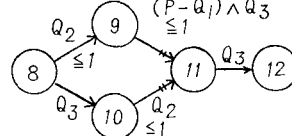
Table 1. Iterations for PNP

Iteration No. PNP for N_1



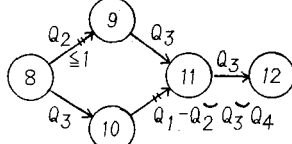
1. $I = (2, 3, 4, 5), J = (6, 7)$
 $F_1 = 4$
 $\tilde{I} = (7), \tilde{J} = (2, 3, 4, 5, 6)$
 $f_1 = 0$

PNP for N_2
 $(P - Q_1) \wedge Q_3$



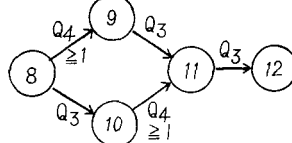
$I = (8, 9, 10), J = (11, 12)$
 $F_2 = 1$
 $\tilde{I} = (12), \tilde{J} = (8, 9, 10, 11)$
 $f_2 = 0$

2. Unchanged.



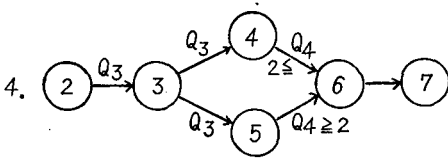
$I = (8, 10), J = (9, 11, 12)$
 $F_2 = 2$
 $I = (11, 12), J = (8, 9, 10)$
 $f_2 = 1$

3. Unchanged.



$F_2 = \infty$
 $f_2 = 2$
 $\tilde{I} = (9, 11, 12), \tilde{J} = (8, 10)$

Unchanged.



$$F_1 = \infty$$

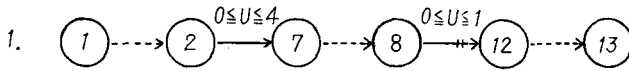
$$f_1 = 4$$

$$\tilde{I} = (6, 7) \quad \tilde{J} = (2, 3, 4, 5)$$

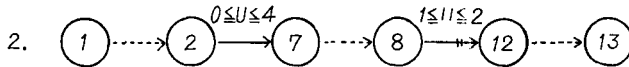
Note: shows the arc is in min-cut.

Table 2. Iterations for ANP

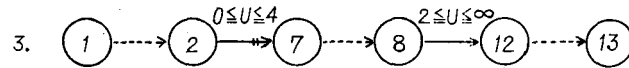
Iteration No. Min-cut Set



$$I = (1, 2, 7, 8), \quad J = (12, 13), \quad F = 1.$$



$$I = (1, 2, 7, 8), \quad J = (12, 13), \quad F = 2.$$



$$I = (1, 2), \quad J = (7, 8, 12, 13), \quad F = 4$$

4. As F_1 and F_2 become infinite in PNP-1, the algorithm ends because of a totally partitioned project.

Table 3. All characteristic schedules generated at each iteration.

	Iter. Nodes		1st.				2nd.				3rd.				4th.			
	i	j	y	t _i	t _j	Q	y	t _i	t _j	Q	y	t _i	t _j	Q	y	t _i	t _j	Q
C.A	<u>1</u>	2	0	0	0	0, Q ₃	0	0	0		0	0	0		0	0	0	
P ₁	2	3	5	0	5	0, "	5	0	5		5	0	5		5	0	5	
	3	4	3	5	8	0, Q ₃	3	5	8		3	5	8		3	5	8	
		5	4	5	9	0, "	4	5	9		4	5	9		4	5	9	
	4	6	5	8	13	0, Q ₂	5	8	13		5	8	13		<u>4</u>	8	<u>12</u>	0, Q ₄
	5	6	4	9	13	0, "	4	9	13		4	9	13		<u>3</u>	9	<u>12</u>	0, Q ₄
	6	7	5	13	18	0, Q ₃	5	13	18		<u>5</u>	13	18		5	<u>12</u>	<u>17</u>	
C.A	7	8	0	18	18	0, "	0	18	18		0	18	18		0	<u>17</u>	<u>17</u>	
P ₂	8	9	6	18	24	0, Q ₂	6	18	24		<u>5</u>	18	<u>23</u>	0, Q ₄	5	<u>17</u>	<u>22</u>	
		10	2	18	20	0, Q ₃	2	18	20		2	18	20		2	<u>17</u>	<u>19</u>	
	9	11	3	24	28	^(P-Q) 1, Q ₃	3	24	<u>27</u>	0, Q ₃	3	<u>23</u>	<u>26</u>		3	<u>22</u>	<u>25</u>	
	10	11	8	20	28	0, Q ₂	<u>7</u>	20	<u>27</u>	0, Q ₃	<u>6</u>	20	<u>26</u>	0, Q ₄	6	<u>19</u>	<u>25</u>	
	11	12	2	28	30	0, Q ₃	2	<u>27</u>	<u>29</u>		2	<u>26</u>	<u>28</u>		2	<u>25</u>	<u>27</u>	
C.A	12	13	0	30	30	0, Q ₃	0	<u>29</u>	<u>29</u>		0	<u>28</u>	<u>28</u>		0	<u>27</u>	<u>27</u>	

Note: The underlined figures in the table show figures changed by a schedule change.

As seen in the example above, only one of two subproblems, N_1 and N_2 , is solved at every stage of PNP except at the first stage, so for only about a half of all nodes the labeling process is repeated at every stage. If the project were partitioned into two parallel subprojects, such effect might be lost since it might occur that both aggregated arcs would be in min-cut at the same time.

REFERENCES

- [1] Dantzig, G. B., and P. Wolfe, "The Decomposition Algorithm for Linear Programs", *Econometrica* **29**, 4, (1961), pp. 767-778.
- [2] Fulkerson, D. R., "A Network Flow Computation for Project Cost Curves", *Management Science*, **7**, 2, (1961), pp. 167-178.
- [3] Kelley, J. E., Jr., "Parametric Programming and the Primal-Dual Algorithm", *Opns. Res.* **7**, 3, (1959), pp. 327-334.
- [4] _____, "Critical-Path Planning and Scheduling: Mathematical Basis", *Opns. Res.* **9**, 3, (1961), pp. 296-320.
- [5] Sekine, T., et al., PERT/CPM no riron to jissai (Theory of PERT/CPM and pits Applications), JUSE, Tokyo, 1964 (in Japanese). (To appear.)