J. Operations Research Soc. of Japan
Vol. 19, No. 1, March, 1976

AN ASSIGNMENT PROBLEM ON A NETWORK

TOSHIHIDE IBARAKI, HIROAKI ISHII
AND HISASHI MINE
Kyoto University

(Received December 21, 1974)

Abstract

This paper deals with a problem of assigning n persons to n nodes (jobs) in a given network, whose arc lengths are dependent upon the assignment, so that the length of a critical path in the network may be minimized. The problem is first formulated as an integer programming probem, and then a branch-and-bound algorithm is proposed. The algorithm makes use of lower bounds obtained from the generated LP (linear programming) problems. Each LP problem is efficiently solved by reducing them to a series of rather small LP problems with two types of auxiliary problems : the ordinary assignment problems and the critical path problems.

1. Introduction

The ordinary assignment problem assigns n persons to n jobs so that the sum of the assignment costs may be minimized. A variant of this assignment problem will be discussed in this paper. Consider the assignment of n persons (machines) to n jobs, the execution of which must satisfy certain precedence relations represented as a network. The objective to be minimized is the time of completing all jobs. It is formally defined as follows.

Let $G = [\ V, A\ ]$ be an acyclic network with a set of vertices $V = \{\ v_0, v_1, v_2, \ldots, v_{n+1}\ \}$ and a set of (directed) arcs $A = \{\ a_1, a_2, \ldots, a_{t_0}\ \} \subset V \times V$. Each vertex represents a job and each arc a precedence relation between jobs ; arc $a_t = (\ v_i, v_j\ )$ implies that the execution of v_j cannot be initiated prior to the completion of v_i. v_0 is the dummy job representing the start ; hence no arc is incident to v_0. v_{n+1} is the dummy job representing the completion of all jobs ; hence no arc is incident from v_{n+1}. We assume that all (directed) paths start from v_0 and end at v_{n+1}.

Let $M = \{\ m_1, m_2, \ldots, m_n\ \}$ be the set of n persons (machines) to be assigned to jobs, and let $d^t_{k\alpha(t)}$ be the length of $a_t \in A$ when m_k is assigned to $v_{\alpha(t)}$ where $v_{\alpha(t)}$ is the vertex in V from which the arc a_t is incident, i.e., $a_t = (v_{\alpha(t)}, v_j)$ for some v_j. The length of $a_t = (v_{\alpha(t)}, v_j)$ may be considered as the execution time of $v_{\alpha(t)}$ plus the setup time of v_j, that generally depends on both $v_{\alpha(t)}$ and v_j. For simplicity d^t_{k0} (lengths of arcs from v_0) are considered to be 0.

Our problem P is now defined as follows : Find an assignment of n elements in M to n jobs in V (i.e., one-to-one mapping : $M \to V$) that minimizes the length of the critical path (longest path) from V_0 to V_{n+1}. As a special case, this is equivalent to the ordinary assignment problem if $G = [V, A]$ is given by $V = \{ v_0, v_1, \ldots, v_{n+1} \}$ and $A = \{(v_0, v_1), (v_1, v_2), \ldots, (v_n, v_{n+1})\}$.

The above problem naturally arises in optimal scheduling of a set of n jobs, on which a technological ordering described by a PERT-like network G is imposed, if the selection of a person (or a machine) to execute each job is permitted. For a given G, the problem asks to find an assignment of n persons with different ability, experience, education and so forth (or n machines of different performance) in such a way that minimizes the total completion time of all n jobs. Note that the differences in persons (or machines) are represented solely by the lengths $d^t_{k\alpha(t)}$ of arc a_t.

As an example, consider that G represents the process of painting a house (or a toy, an automobile, etc.) in n colours. Each node represents to paint one given colour. The preceding relation specified by G shows the order in which n colours are painted ; e.g., black is painted after white, etc. The length $d^t_{k\alpha(t)}$ of $a_t =(v_{\alpha(t)}, v_j)$ is the sum of the time of painting colour $v_{\alpha(t)}$ and the time of waiting until it dries enough to start painting the next colour v_j. Then the above problem is to assign n spray guns of varied performance to n nodes so that the entire process may be completed in the minimum time. (It is natural to assume that one spray gun is prepared for each colour .)

The dynamic assignment models discussed in Charnes et al.[1] is also similar to ours in that an assignment on an n-stage network is considered, though they use somewhat different constraints and objective function.

In case the length $d^t_{k\alpha(t)}$ of $a_t =(v_{\alpha(t)}, v_j)$ is dependent on $v_{\alpha(t)}$ only, it is sometimes more convenient to use the network in which each arc (rather than node) represents a job, as in the ordinary PERT network. It would be obvious that the following theory can be easily modified to this model ; hence the discussion of this model will be omitted.

2. Formulation as an IP Problem

The above assignment problem P is formulated as an IP (integer programming) problem in this section. An algorithm based on the branch-and-bound principle will then be developed in the subsequent sections. Branch-and-bound algorithms have attained certain success for other variants of the assignment problem, such as the quadratic assignment problem [4][6] and the multidimensional assignment problem [10].

Let the path-arc matrix of G be given by $\Pi = (\pi_{st})$, where

$$\pi_{st} = \begin{cases} 1 & \text{if } a_t \in A \text{ is on the } s\text{-th path of } G \\ 0 & \text{otherwise,} \end{cases}$$

$$s = 1,2,\ldots, s_0 \quad (= \text{the number of paths in } G)$$

$$t = 1,2,\ldots, t_0 \quad (= \text{the number of arcs}).$$

Introduce 0-1 variables x_{ki} which specify an assignment in the following manner :

$$x_{ki} = \begin{cases} 1 & \text{if } m_k \in M \text{ is assigned to } v_i \in V \\ 0 & \text{otherwise.} \end{cases}$$

The length of the s-th path is then written as follows, for the assignment given by $x = (x_{11}, x_{12}, \ldots, x_{nn})$.

$$\sum_{t=1}^{t_0} \pi_{st} \sum_{k=1}^{n} d^t_{k\alpha(t)} \; x_{k\alpha(t)}, \qquad s = 1,2,\ldots, s_0 .$$

(For notational convenience, dummy variables $x_{k0}=0$ are introduced.)
Thus P is formulated as the following IP problem with n^2 0-1 variables x_{ki} and one real variable λ.

$$\begin{aligned}
P : \text{minimize} \quad & \lambda \\
\text{subject to} \quad & \sum_{t=1}^{t_0} \pi_{st} \sum_{k=1}^{n} d^t_{k\alpha(t)} \; x_{k\alpha(t)} \le \lambda, \qquad s = 1,2,\ldots, s_0 \\
& \sum_{i=1}^{n} x_{ki} = 1, \qquad k = 1,2,\ldots, n \\
& \sum_{k=1}^{n} x_{ki} = 1, \qquad i = 1,2,\ldots, n \\
& x_{ki} = 0 \text{ or } 1, \qquad k,i = 1,2,\ldots, n .
\end{aligned}$$

Note that P is the ordinary assignment problem if the first s_0 constraints having λ on their right hand sides are removed and the objective function is appropriately modified.

3. Solving P by a Branch-and-Bound Method

Since the IP problem P is of considerably large size, we attempt to solve it by using a branch-and-bound method, rather than directly applying the existing integer programming algorithms. A branch-and-bound method (e.g., [7] [8]) is usually determined by the following two ingredients :

(a) Branching rule ; that specifies how to decompose a given (partial) problem P_j (P_j is either the original P or one of problems generated by this branching operation) into smaller partial problems $P_{j_1}, P_{j_2}, \ldots, P_{j_k}$ such that P_j can be solved if all $P_{j_1}, P_{j_2}, \ldots, P_{j_k}$ are solved ;

(b) Bounding strategy ; that specifies how to attempt to solve P_j and, in case P_j is not solved by this attempt, how to obtain a lower bound of its optimal objective value.

Now we give a description of (a) and (b) for our problem, and after that a description of the whole branch-and-bound method will follow.

First the branching is made from a given (partial) problem P_j to two partial problems P_{j_1} and P_{j_2} such that

(i) P_{j_1} is obtained from P_j by adding the constraint that a certain $m_{\bar{k}} \in M$ is assigned to $v_{\bar{i}} \in$ V, where no fixed assignment was given to $m_{\bar{k}}$ or $v_{\bar{i}}$ in P_j, and the assignment $m_{\bar{k}} \to v_{\bar{i}}$ was not prohibited in P_j ;

(ii) P_{j_2} is obtained from P_j by adding the constraint that the assignment $m_{\bar{k}} \to v_{\bar{i}}$ is prohibited.

This is formally described as follows. With a (partial) problem P_j, let two sets $F_j \subset M \times V$ and $H_j \subset M \times V$ be associated ; P_j is denoted by $P_j = (F_j, H_j)$. $(m_k, v_i) \in F_j$ denotes that the assignment $m_k \to v_i$ is fixed in P_j, while $(m_k, v_i) \in H_j$ denotes that the assignment $m_k \to v_i$ is prohibited in P_j. The orignal problem P is defined by

$$P = (\phi, \phi).$$

The branching is then defined by a pair $(m_{\bar{k}}, v_{\bar{i}})$ such that

$$\begin{aligned} &(m_{\bar{k}}, v_i) \notin F_j, && i = 1,2,\ldots, n \\ &(m_k, v_{\bar{i}}) \notin F_j, && k = 1,2,\ldots, n \\ &(m_{\bar{k}}, v_{\bar{i}}) \notin H_j. && \end{aligned}$$

The resulting P_{j_1} and P_{j_2} are given by

$$P_{j_1} = (F_{j_1} , H_{j_1})$$

$$P_{j_2} = (F_{j_2} , H_{j_2}),$$

where

$$F_{j_1} = F_j \cup \{ (m_{\bar{k}} , v_{\bar{i}}) \}$$

$$H_{j_1} = H_j$$

$$F_{j_2} = F_j$$

$$H_{j_2} = H_j \cup \{ (m_{\bar{k}} , v_{\bar{i}}) \}.$$

Since the way of selecting $(m_{\bar{k}} , v_{\bar{i}})$ for branching is crucial for the algorithm efficiency, it will be further discussed in Section 5.

To determine a bounding strategy, next, note that a partial problem P_j is also an IP problem obtained from P by fixing some variables to 0 or 1. Let $\bar{P}_j$ be the LP (linear programming) problem obtained from P_j by removing the integrality constraint. Obviously an optimal solution of $\bar{P}_j$ is an optimal solution of P_j if it is an integer solution. Furthermore, denoting the values of optimal solutions of P_j and $\bar{P}_j$ by $\lambda(P_j)$ and $\lambda(\bar{P}_j)$ respectively, the following relation is easily proved :

$$\lambda(\bar{P}_j) \leq \lambda(P_j).$$

Thus $\lambda(\bar{P}_j)$ is used as a lower bound of $\lambda(P_j)$.

Although $\bar{P}_j$ is usually a large LP problem, a decomposition technique can be applied since $\bar{P}_j$ is highly structured. The resulting computational procedure will be described in the next section.

A branch-and-bound algorithm for solving P is now given. In the algorithm, a problem P_j is called active if it is to be examined further, while it is called terminated if a conclusion that P_j can be discarded from the subsequent computation has been drawn for some reason, or a necessary step was already taken for P_j. The algorithm terminates whenever no active problem exists.

Branch-and-Bound Algorithm for solving P (Algorithm A)

Step 1. (Initialization) Let $P_0(=P)$ be active. $\lambda^*:=\infty$. Solve $\bar{P}_0$. If $\bar{P}_0$ has an integer optimal solution, go to Step 4 after letting $\lambda^*:=\lambda(\bar{P}_0)$; otherwise go to Step 2.

Step 2. Select one active problem P_j, and go to Step 3. If no active problem exists, go to Step 4.

Step 3. (Branching and Bounding). Decompose P_j into P_{j_1} and P_{j_2} by the branching rule described above, and terminate P_j. Solve $\bar{P}_{j_1}$ and $\bar{P}_{j_2}$. If $\bar{P}_{j_1}$ ($\bar{P}_{j_2}$) has an integer optimal solution, terminate P_{j_1} (P_{j_2}) and replace λ^* by $\lambda(\bar{P}_{j_1})$ ($\lambda(\bar{P}_{j_2})$) if $\lambda(\bar{P}_{j_1}) < \lambda^*$ ($\lambda(\bar{P}_{j_2}) < \lambda^*$). Terminate all active problems P_k (including P_{j_1} and P_{j_2}) such that

$$\lambda(\bar{P}_k) \geq \lambda^*$$

holds. Let P_{j_1} and /or P_{j_2} not terminated in this process be active. Return to Step 2.

Step 4. Terminate the computation. λ^* gives the value of optimal assignments.

The selection rule of an active problem P_j in Step 2 is also important in determining the algorithm efficiency. Although any rule discussed in the framework of general branch-and-bound methods would be applicable, the so-called linear search rule that selects the active problem which was most recently generated is employed in our computation of Section 8. The linear search rule consumes less amount of memory space compared with others.

4. Solving $\bar{P}_j$ for Obtaining Lower Bounds

For simplicity, an algorithm for solving $\bar{P}$ instead of $\bar{P}_j$ will be described. The modification necessary for general $\bar{P}_j$ is obvious. The essence of the following procedure is to apply the Dantzig-Wolfe decomposition technique [2] to $\bar{P}$ and reduce it to a series of small size LP problems (compared with $\bar{P}$) with two types of auxiliary problems one of which is the ordinary assignment problem and the other is the critical path problem. It is well known that considerably efficient algorithms exist for both the assignment problem [5][9] and the critical path problem.

Let $x = (x_{11}, x_{12}, \ldots, x_{nn})$ and let

$$D = \{x \mid \sum_{i=1}^{n} x_{ki} = 1, \ \sum_{k=1}^{n} x_{ki} = 1, \ \ x_{ki} \geq 0\}.$$

Denote extreme points of D by

$$x^r, \quad r = 1, 2, \ldots, r_0 \ (=n!),$$

i.e., x^r is an n^2 dimensional 0-1 vectors corresponding to assignments : $M \to V$. Any $x \in D$ can be represented as follows :

$$x = \sum_{r=1}^{r_0} \rho_r x^r$$

where

$$\sum_{r=1}^{r_0} \rho_r = 1, \quad \rho_r \geq 0, \quad r = 1, 2, \ldots, r_0.$$

By substituting these expressions, $\bar{P}$ is transformed into

$\bar{P}$: minimize λ

$$\text{subject to } \sum_{r=1}^{r_0} \psi_{sr}\rho_r \leq \lambda, \quad s = 1, 2, \ldots, s_0$$

$$\sum_{r=1}^{r_0} \rho_r = 1$$

$$\rho_r \geq 0, \quad r = 1, 2, \ldots, r_0,$$

where ψ_{sr} is a constant defined by

$$\psi_{sr} = \sum_{t=1}^{t_0} \pi_{st} \sum_{k=1}^{n} d^t_{k\alpha(t)} \; x^r_{k\alpha(t)},$$

which is the length of the s-th path in G when assignment x^r is made.

$\bar{P}$ is usually a very large LP problem. However the following column and row generation technique makes it possible to solve $\bar{P}$ by a series of rather small LP problems. We will first show how the dual and primal feasibility can be checked without maintaining the entire simplex tableau of $\bar{P}$, and then give an algorithm for $\bar{P}$.

(A) Dual feasibility : For $S \subset S_0$ ($= \{1, 2, \ldots, s_0\}$), let $\bar{P}(S)$ (the role of S will become clear later) be $\bar{P}$ with constraints

$$\sum_{r=1}^{r_0} \psi_{sr} \; \rho_r \leq \lambda, \quad s \notin S$$

delated. The modified costs $\bar{c}_r$ of $\bar{P}(S)$ are then given by

$$\bar{c}_r = \sum_{s \in S} \sigma_s \psi_{sr} - \sigma_0, \quad r = 1, 2, \ldots, r_0,$$

where σ_s and σ_0 are the current values of dual variables (simplex multipliers) corresponding to

$$\sum_{r=1}^{r_0} \psi_{sr}\rho_r \leq \lambda \quad \text{and} \quad \sum_{r=1}^{r_0} \rho_r = 1,$$

respectively. Note that σ_s corresponding to

$$\sum_{r=1}^{r_0} \psi_{sr}\rho_r < \lambda$$

for the current values of ρ_r always assumes the value 0 due to the complementary slackness relation. Thus it is possible to assume that $s \in S$ always represents a path satisfying

$$\sum_{r=1}^{r_0} \psi_{sr}\rho_r = \lambda,$$

since the addition of $\sigma_s=0$ does not change $\bar{c}_r$.

The dual feasibility of $\bar{P}(S)$, i.e.,

$$\bar{c}_r \geq 0, \quad r = 1,2,\dots, r_0$$

is satisfied if and only if

$$\bar{z} = min \; \{ \sum_{s\in S} \sigma_s \psi_{sr} \mid r = 1,2,\dots, r_0 \} \geq \sigma_0$$

holds. Note that $\bar{z}$ is the optimal objective value of the following assignment problem :

$$Q(\sigma) : \text{minimize} \quad z = \sum_{s\in S} \sigma_s \sum_{t=1}^{t_0} \pi_{st} \sum_{k=1}^{n} d^t_{k\alpha(t)} x_{k\alpha(t)}$$

$$\text{subject to} \quad \sum_{k=1}^{n} x_{ki} = 1, \qquad i = 1,2,\dots, n$$

$$\sum_{i=1}^{n} x_{ki} = 1, \qquad k = 1,2,\dots, n$$

$$x_{ki} \geq 0, \qquad k = 1,2,\dots, n, \qquad i = 1,2,\dots, n,$$

where σ_s, π_{st}, $d^t_{k\alpha(t)}$ are constants and x_{ki} are variables. Thus the dual feasibility of $\bar{P}(S)$ can be checked by solving an ordinary assignment problem.

(B) Primal feasibility : For the current values of ρ_r, let the index set $R \subset R_0$ ($= \{1,2,\dots, r_0\}$) be given such that $\rho_r = 0$ if $r \notin R$ (the role of R will become clear later). Assume that

$$\sum_{r\in R} \rho_r = 1 \quad \text{and} \quad \rho_r \geq 0, \quad r\in R$$

hold. Define the weighted arc length of $a_t \in A$ for ρ by

$$d^t(\rho) = \sum_{r\in R} \rho_r \sum_{k=1}^{n} d^t_{k\alpha(t)} x^r_{k\alpha(t)}, \qquad t = 1,2,\dots, t_0.$$

Let $G(\rho)$ be the network G with each arc $a_t \in A$ having length $d^t(\rho)$. Then the primal feasibility of $\bar{P}$ is satisfied if and only if all path lengths in $G(\rho)$ are not greater than λ(= the current objective value). The last condition is satisfied if and only if the length $\mu(\rho)$ of critical paths in $G(\rho)$ satisfies

$$\mu(\rho) \leq \lambda .$$

Thus the primal feasibility of $\bar{P}$ can be checked by solving a critical path problem.

Before proceeding to an algorithm for solving $\bar{P}$, define the LP problem $\bar{P}(S, R)$ for $S \subset S_0$ and $R \subset R_0$ as follows :

$\bar{P}(S, R)$: minimize λ

$$\text{subject to } \sum_{r \in R} \psi_{sr} \rho_r \leq \lambda, \quad s \in S$$
$$\sum_{r \in R} \rho_r = 1$$
$$\rho_r \geq 0 , \quad r \in R .$$

Algorithm for solving $\bar{P}$(Algorithm B)

Step 1. Start with $S \subset S_0$ and $R \subset R_0$ heuristically obtained. Go to Step 2.

Step 2. Solve $\bar{P}(S, R)$ and denote its optimal tableau by T. Redefine R by $R := R - R'$ (i.e., delete from T all columns corresponding to $r \in R'$), where R' is the set of indices r such that ρ_r is nonbasic and $\bar{c}_r > 0$ in T. Let the resulting tableau be T and go to Step 3.

Step 3. If T is dual feasible (this can be checked by solving the assignment problem $Q(\sigma)$), go to Step 4. Otherwise let $R := R \cup \{r'\}$ (i.e., augment T by column r'), where $x^{r'}$ is the optimal assignment of $Q(\sigma)$, and return to Step 2.

Step 4. If T is primal feasible (this can be checked by obtaining a critical path of $G(\rho)$), terminate. T gives an optimal solution of $\bar{P}$. Otherwise, let $S := S \cup \{s'\} - S'$ (i.e., add row s' and delete the set of rows S'), where s' is the index of the critical path of $G(\rho)$, and S' is defined as follows : S' is the set of indices corresponding to non-binding constraints of T (i.e., with positive slack variables) if the objective value λ was improved after the previous dual feasible solution ; $S' = \phi$ if the objective value λ was not improved or the current solution is the first dual feasible solution. Return to Step 2.

The finite convergence of this algorithm under the usual nondegeneracy assumption can be proved in a manner similar to the one used by Geoffrion [3] for proving the convergence of the general relaxation strategy.

A good heuristic rule for defining the initial S and R in Step 1 may be to let S be the set of indices for critical paths of G with all arc lengths considered to be 1, and to let R be a set of indices for assignments which make these paths in S reasonably short. In the experiment of Section 8, however, a much simpler rule was employed.

5. Branching Rule

Various methods would be conceivable for determining the pair $(m_{\bar{k}}, v_{\bar{i}})$ which defines the branching operation in Step 3 of the branch-and-bound method outlined in Section 3. The following used in our computation should be one of the most reasonable ones.

Let LP problem $\bar{P}_j$ have the optimal solution :

$$\bar{\rho} = (\bar{\rho}_1, \bar{\rho}_2, \ldots, \bar{\rho}_{r_0}).$$

Based on ρ, define

$$x_{ki}(\bar{\rho}) = \sum_{r=1}^{r_0} \bar{\rho}_r x_{ki}^r$$

$$x_{\bar{k}\bar{i}}(\bar{\rho}) = \max_{k,i}\{ x_{ki}(\bar{\rho}) \mid 0 \le x_{ki}(\bar{\rho}) < 1 \}.$$

$x_{\bar{k}\bar{i}}(\bar{\rho})$ is considered to indicate the most promising assignment $m_{\bar{k}} \to v_{\bar{i}}$, as far as the information contained in $\bar{\rho}$ is concerned. Thus $(\bar{k}, \bar{i})$ is used to define the branching from P_j. The case of $x_{ki}(\bar{\rho})$=1 is excluded since it is likely that the partial problems generated from P_j also satisfy x_{ki}=1 even without being fixed. (If x_{ki} was already fixed to 1 by the branching operation applied to P_k, from which P_j resulted, we obviously need not fix it again. The above rule also excludes this possibility.)

6. Further Computational Considerations

In implementing the algorithm outlined so far, the following remarks may be useful from the view point of computational speed.

(i) The value of each dual feasible solution of $\bar{P}_j$ obtained in Step 3 of Algorithm B (Section 4) is not greater than the optimal value $\lambda(\bar{P}_j)$. Thus the value of any dual feasible solution can be used as a lower bound of $\lambda(P_j)$. This suggests the termination of the LP computation (Algorithm B) before the primal feasibility is attained. A reasonable cutoff rule may be to terminate the computation of $\bar{P}_j$ whenever a dual feasible tableau T with $\mu(\rho) - \lambda \le \varepsilon$ is obtained, where λ is the objective value of T, $\mu(\rho)$ is the length of a critical path of $G(\rho)$ and $\varepsilon(\geq 0)$ is a given constant (note $\mu(\rho)-\lambda\leq 0$ implies primal feasibility). $\varepsilon=10$ was used in our experiment of Section 8. (See also the discussion given to Tables 3 and 4 of Section 8.)

(ii) In solving $\bar{P}_j$ by Algorithm B, or in passing from one problem $\bar{P}_j$ to $\bar{P}_k$, where $\bar{P}_k$ is obtained from $\bar{P}_j$ by branching or backtracking, the technique of parametric programming (or sensitivity analysis) known in the LP theory can be fully utilized. Details are, however, omitted.

(iii) The optimal solution (or a good feasible solution) of $\bar{P}_j$ may be used to generate good assignments ; assignments x^r with relatively large $\bar{\rho}_r$ in $\bar{\rho}$ would have a high probability of being close to an optimal assignment of P. Thus in Step 3 of the branch-and-bound algorithm (Algorithm A) in Section 3, a certain number of assignments considered promising according to optimal LP solutions of $\bar{P}_{j_1}$ and $\bar{P}_{j_2}$ may be actually constructed and tested. If an assignment with a smaller objective value than the current λ^* is found, λ^* is immediately replaced by the new value. This modified algorithm has a tendency of keeping the value of λ^* smaller than that without this modification. Hence a speed up in computation time can be expected. In our experiment of Section 8, all new assignments x^r with positive ρ_r were actually tested.

7. Example

Consider the problem with the network of Fig. 1 and arc lengths $d^t_{k\alpha(t)}$ given by Table 1.

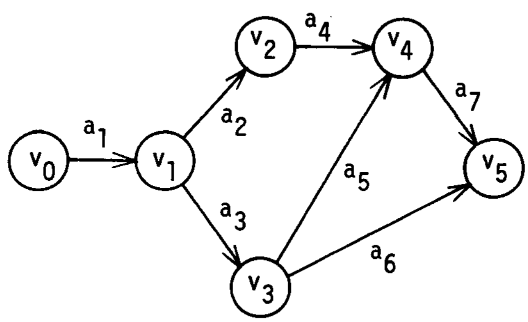


Fig. 1. Precedence relations between jobs.

t	2	3	4	5	6	7
$\alpha(t)$	1	1	2	3	3	4
k=1	701	542	801	422	0	404
2	408	308	552	365	373	225
3	224	947	874	8	505	297
4	715	117	512	186	324	636

Table 1. Arc lengths $d^{t}_{k\alpha(t)}$ of the assignment problem on network of Fig.1.

After letting $\lambda^*=\infty$ in Step A-1 (i.e., Step 1 of Algorithm A), Step B-1 (i.e., Step 1 of Algorithm B) is entered to solve $\bar{P}_0$. The initial S of Step B-1 is given by path 1 and path 2 of Fig.2, and the initial R by assignment 1 and assignment 2 of Fig.2, where $(i_1\ i_2\ i_3\ i_4)$ denotes the assignment $\{m_1 \to v_{i_1}, m_2 \to v_{i_2}, m_3 \to v_{i_3}, m_4 \to v_{i_4}\}$. The arc lengths for these assignments are also shown in Fig.2. This defines LP problem $\bar{P}(S, R)$ as follows:

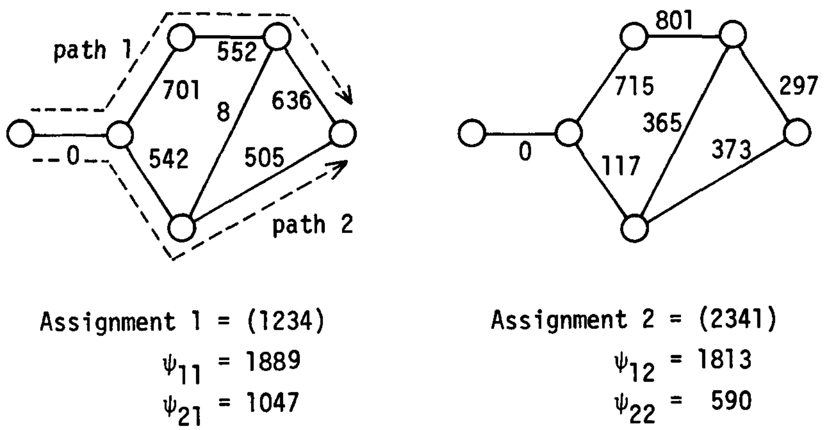


Fig.2. Initial paths and assignments.

$\bar{P}(\{1, 2\}, \{1, 2\})$: minimize λ

$$\begin{aligned} \text{subject to} \quad & 1889\rho_1 + 1813\rho_2 \le \lambda \\ & 1047\rho_1 + 590\rho_2 \le \lambda \\ & \rho_1 + \rho_2 = 1 \\ & \rho_1, \rho_2 \ge 0. \end{aligned}$$

An optimal solution (obtained by the simplex method) is

$$\lambda = 1813, \quad \rho_1 = 0, \quad \rho_2 = 1, \quad \sigma_0 (= \lambda) = 1813, \quad \sigma_1 = 1, \quad \sigma_2 = 0,$$

where σ_i are dual variables. The nonbasic column ρ_1 is then deleted since $\bar{c}_1 > 0$ as easily calculated (Step B-2).

To check the dual feasibility of this solution (Step B-3), assignment problem $Q(\sigma)$ with $\sigma_1 = 1$ is then solved. Coefficients of $Q(\sigma)$ are shown in Table 2. The (k, i)-th element of Table 2 shows the coefficient of x_{ki}

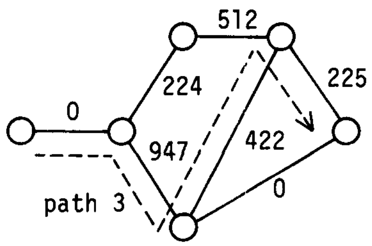
k \ i	1	2	3	4
1	701	801	0*	404
2	408	552	0	225*
3	224*	874	0	297
4	715	512*	0	636

Table 2. Coefficients of the assignment problem to check the dual feasibility.

in the objective function of $Q(\sigma)$. An optimal assignment of $Q(\sigma)$ (indicated by * in Table 2) is

$$\text{assignment } 3 = (3412)$$

and the optimal value is 961. Arc lengths for assignment 3 are shown in Fig.3.



Assignment 3 = (3412)

$\psi_{13} = 961$

$\psi_{23} = 947$

Fig.3. Arc lengths for assignment 3.

Since $\bar{z} = 961 < 1813 = \sigma_0$ holds, the present solution is not dual feasible. Thus assignment 3 is added to form the new $\bar{P}(S, R)$, that is

$$\bar{P}(\{1, 2\}, \{2, 3\}): \text{minimize } \lambda \quad \text{subject to} \quad \begin{aligned} 1813\rho_2 + 961\rho_3 &\le \lambda \\ 590\rho_2 + 947\rho_3 &\le \lambda \\ \rho_2 + \rho_3 &= 1 \\ \rho_2, \rho_3 &\ge 0. \end{aligned}$$

An optimal solution obtained in Step B-2 is

$$\lambda = 961, \quad \rho_2 = 0, \quad \rho_3 = 1, \quad \sigma_0 = 961, \quad \sigma_1 = 1, \quad \sigma_2 = 0.$$

(Although $\bar{P}(\{1, 2\}, \{2, 3\})$ is actually solved starting with the optimal tableau of $\bar{P}(\{1, 2\}, \{1, 2\})$ to facilitate the computation, the details are omitted for simplicity.) The nonbasic colum ρ_2 is then deleted since $\bar{c}_2 > 0$ (Step B-2).

The assignment problem $Q(\sigma)$ to check this dual feasibility is the same as the above one with $\bar{z} = 961$. Thus $\bar{z} = \sigma_0$ holds and the present solution is dual feasible (Step B-3)

Now it is required to check the primal feasibility (Step B-4). Since $\rho_3 = 1$ implies that network $G(\rho)$ is the same as the one in Fig.3 corresponding to assignment 3, this is done by calculating its critical path (path 3). Path 3 has the length $\mu(\rho) = 1594$. $\mu(\rho) = 1594 > 961 = \lambda$ shows that the present solution is not primal feasible.

We again return to Step B-2 and solve $\bar{P}(\{1, 2, 3\}, \{3\})$ augmented by path 3. Repeating Step B-2 and Step B-3, the next dual feasible solution is obtained after adding two more assignments :

assignment 4 = (*2431*), and assignment 5 = (*4132*).

In this case, the resulting solution is also primal feasible. Thus its optimal solution

$$\lambda(=\lambda(\bar{P}_0)) = 1146.7, \quad \rho_3 = 0.49, \quad \rho_5 = 0.51$$

also solves $\bar{P}_0$. Fig.4 shows network $G(\rho)$ for this solution, in which path 1 and path 3 are both critical.

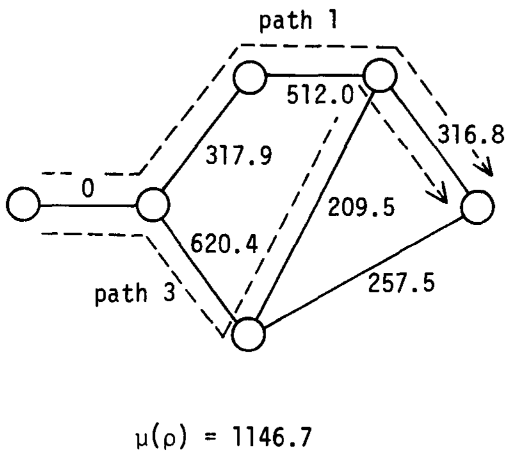


Fig.4. Network $G(\rho)$ corresponding to the optimal solution of $\bar{P}_0$.

At this point, assignment 3 and assignment 5 (with positive ρ_r) are tested to improve λ^* (see remark (iii) of Section 6). Assignment 5 gives a smaller value than assignment 3. Its value *1324* is hence stored as the new λ^*. This completes the computation associated with the initial problem P_0.

Returning to Algorithm A, P_0 is selected in Step A-2 since it is the only active problem. The branching of Step A-3 is done according to the rule in Section 5. $x_{ki}(\bar{\rho})$ for $\bar{\rho}_3 = 0.49$, $\bar{\rho}_5 = 0.51$ (optimal solution of $\bar{P}_0$) are first calculated. They are

$$x_{13}(\bar{\rho}) = x_{24}(\bar{\rho}) = x_{31}(\bar{\rho}) = 0.49, \quad x_{14}(\bar{\rho}) = x_{21}(\bar{\rho}) = x_{33}(\bar{\rho}) = 0.51,$$

$$x_{42}(\bar{\rho}) = 1, \quad x_{ki}(\bar{\rho}) = 0 \text{ for other } k \text{ and } i.$$

Thus $x_{14}(\bar{\rho}) = 0.51$ which is the first one with the highest value (excluding $x_{42}(\bar{\rho})=1$) is selected to define the branching $m_1 \to v_4$ and $m_1 \not\to v_4$.

The subsequent branch-and-bound computation proceeds as illustrated in

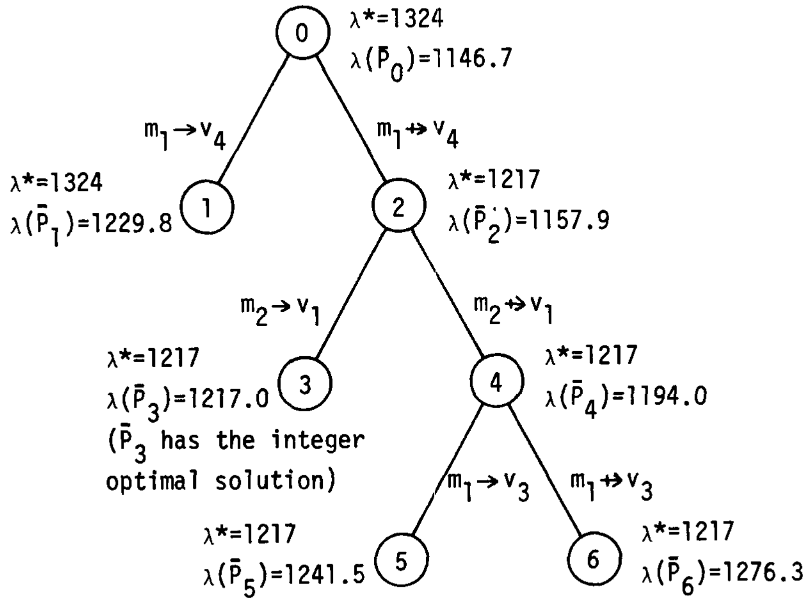


Fig.5. Illustration of the branch-and-bound computation.

Fig. 5. Node numbers indicate the order in which partial problems are tested. λ^* attached to each node is the value when the computation of that node is completed. $\lambda(\bar{P}_j)$ is the LP optimal value of P_j and it works as a lower bound. When node 6 is tested, no active node exists since all nodes have lower bounds greater than $\lambda=1217$. Hence the whole computation terminates (Step A-4).

The optimal solution is

assignment 7 = (*3142*)

with $\lambda=1217$, which was obtained at node 2 in the computation of the new λ^*.

This optimal solution is shown in Fig.6.

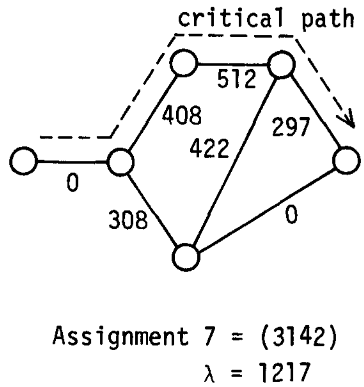


Fig.6. Optimal assignment

8. Computational Experiment

To see the efficiency of our algorithm, the entire procedure was coded in FORTRAN and run on the FACOM *230/75* computer at Kyoto University. The machine roughly corresponds to IBM *370/165*. The code for ordinary assignment problems was obtained by translating the Silver's code [11] written in ALGOL.

Before determining the details of the branch-and-bound algorithm (Algorithm A), the possibility of strategy (i) of Section 6 was tested by solving LP problems $\bar{P}_j$ (using Algorithm B) and examining the quality of the values of dual feasible solutions which are not necessarily primal feasible. Results are shown in Table 3. Each figure is the average of *11* to *40* problems as indicated in the bottom row. All problems were generated randomly by assigning lengths taken from the uniform distribution between *0* and *1* to arcs of three networks : 10A, 10B and 20A of Table 3. The initial S and R of Step B-1 were defined as follows.

- S represents two paths : One with the largest number of arcs, and the other with the largest number of arcs in the graph resulted by removing the first path.
- R represents two assignments : $(1, 2, \ldots, n)$ and $(2, n-1, 4, n-3, 6, n-5, \ldots, n, 1)$.

Problem type		10A	10B	20A
Number of vertices (persons) n		10	10	20
Number of arcs t_0		13	18	26
Number of generated columns (assignments) (a)		13.0	37.2	63.0
Number of generated rows (paths) (a)		3.2	6.5	5.9
90% point$^{(b)}$	colums	10.1	17.0	35.7
	rows	2.7	3.5	4.4
Number of test problems		40	23	11

Table 3. Computational results for LP problems $\bar{P}_j$.

Notes to tables

(a) Some columns (rows) may be counted repeatedly if deleted columns (rows) are again added back to set R (S).

(b) Numbers of columns and rows when a dual feasible solution with objective value $\lambda \geq 0.9\lambda(\bar{P})$ is attained.

(c) This number contains those partial problems which are terminated in Step B-3.

(d) The number of columns generated before the column corresponding to an optimal assignment is generated.

Problem type	10B		10B	
ε	0	10	0	10
Computation time in seconds	2.43	2.25	4.84	4.39
Number of generated partial problems (c)	42	48	52	54

Table 4. Results for assignment problems P with $\varepsilon=0$ and $\varepsilon=10$.

Problem type		10A	10B	20A
Number of generated partial problems (c)	Mean	73.9	140.2	891.6
	Standard deviation	107.9	152.5	1013.8
Number of generated columns (assignments) (a)		223	631	7256
Number of generated rows (paths)[(a)]		5	49	27
Number of privots		390	1507	13491
When an optimal assignment generated (d)		126	104	436
Computation time in seconds		1.9	7.2	193.0
Number of test problems		30	20	10

Table 5. Computational results for assignment problems on networks by branch-and-bound.

(These are adopted mainly for the sake of simplicity.) It may be concluded that rather accurate lower bounds can be expected even if the computation is cut off prior to the primal feasibility (*90%* estimation attained in about a half of the total number of columns).

Strategy (i) of Section 6 is further justified by Table 4, which shows the computational results for two type 10B problems (type 10A is too simple to see the difference) using both $\varepsilon=0$ and $\varepsilon=10$. (Arc lengths are defined in the same manner as in Table 5 described next.) It seems that setting $\varepsilon=10$ tends to consume less computation time although it generates a slightly larger number of partial problems.

Based on these preliminary results, ε in the cutoff rule was set to *10*. Then a number of problems was solved by Algorithm A with LP subalgorithm (Algorithm B). All test problems were also generated randomly, but in this case arcs were assigned integer lengths taken from the uniform distribution between *0* and *999*. Since each problem in this case has an integer optimal value, λ of each dual feasible solution can be used as a lower bound after being rounded up to the smallest integer not smaller than λ. (This gimmick was quite useful to increase the efficiency.)

Computational results are summarized in Table 5. Problem type in the first row refers to the same network as Table 3. Each figure in Table 5 shows the average of *10~30* problems. It was noticed that the behavior of the algorithm was quite erratic. The standard deviations of the number of partial problems are also included to indicate this.

In most cases, relatively good assignments (if not optimal) were obtained in the early stage of the computation (see (iii) of Section 6 and the row labelled "when an optimal assignment generated" in Table 5). This indicates that our algorithm is also useful as a suboptimal algorithm in case enough computation time is not available.

Unfortunately it seems difficult to obtain exact optimal solutions of large problems (say, $n=30$), due to the rapid increase in the computation time. Probably this is because the quality of lower bounds provided by LP optimal solutions are not accurate enough. Thus it would be a direction of future research to find methods for obtaining better lower bounds.

Acknowledgement

The authors thank Professor T. Hasegawa of Kyoto University for his comments on the subject. They are also indebted to Miss T. Kanazawa for her excellent typing.

References

[1] Charnes, A., W.W. Cooper, R.J. Niehaus and A. Stedry, "Static and dynamic assignment models with multiple objectives, and some remarks on organization design," Management Science, 15 (1969), B365-B375.

[2] Dantzig, G.B., and P. Wolfe, "Decomposition principle for linear programs," Operations Research, 8 (1960), 101-111.

[3] Geoffrion, A.M., "Elements of large-scale mathematical programming," Management Science, 16 (1970), 652-691.

[4] Gilmore, P.C., "Optimal and suboptimal algorithms for the quadratic assignment problem," J. SIAM, 10 (1962), 305-313.

[5] Kuhn, H.W., "The Hungarian method for the assignment problem," Naval Res. Log. Quarterly, 2 (1955), 83-97.

[6] Lawler, E.L., "The quadratic assignment problem," Operations Research, 9 (1963), 586-599.

[7] Lawler, E.L., and D.E. Wood, "Branch and bound method : A survey," Operations Research, 14 (1966), 699-719.

[8] Mitten, L.G., "Branch-and-bound methods : General formulation and properties," Operations Research, 18 (1970), 24-34.

[9] Munkres, J., "Algorithms for the assignment and transportation problems," J. SIAM, 5 (1957), 32-38.

[10] Pierskalla, W.P., "The multidimensional assignment problem," Operations Research, 16 (1968), 422-431.

[11] Silver, R., "Algorithm 27 : Assignment," Commun. ACM, 3 (1960), 603-604.

(Authors' address: Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Kyoto Japan.)