ELSEVIER European Journal of Operational Research 110 (1998) 314–325

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

A variant of time minimizing assignment problem

Shalini Arora *, M.C. Puri [1]

Department of Mathematics, IIT Delhi, Hauz Khas, New Delhi-110016, India

Received 13 June 1996; accepted 6 April 1997

Abstract

A time minimizing assignment problem (TMAP) dealing with the allocation of n jobs to $m(< n)$ persons is considered in this paper. One job is to be allocated to exactly one person and each person does at least one job. All the persons start working on the jobs simultaneously. If a person is to do more than one job, he does them one after the other in any order. The aim of the present study is to find that feasible assignment which minimizes the total time for completing all the jobs. A lexi-search approach is proposed to find an optimal feasible assignment. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: Imbalanced assignment problem; Time minimizing assignment problem; Bottleneck assignment problem

1. Introduction

The cost minimizing assignment problem (CMAP) has been extensively considered in the literature [28] and several polynomial algorithms [5,9,25] for n jobs and m persons, are available to solve it. Because of the widespread applicability, considerable time and effort has been devoted in developing and evaluating efficient algorithms for solving large scale assignment problems [23,24,27,34,43,44]. The CMAP often arises as a subproblem of difficult combinatorial programming problems. For example, generalized assignment problem (GAP), which is NP-complete, can be solved by the repeated application of the Hungarian method [46]. A large number of problems occurring within resource-constrained assignment scheduling can be modelled as 0-1 assignment problems with side constraints (APSC). APSC, which is NP-complete, can be solved by generating all the assignments of the related CMAP in order of the decreasing cost and checking the feasibility of each assignment as it is generated [35]. Mazzola et al. [33] developed a branch and bound algorithm for producing exact solutions as well as a heuristic procedure for producing an approximate solution to APSC.

* Corresponding author. E-mail: shalinia@maths.iitd.ernet.in.
[1] E-mail: mcpuri@maths.iitd.ernet.in.

PII S 0 3 7 7 - 2 2 1 7 (9 7) 0 0 2 6 6 - X

The CMAP, when n jobs can be grouped into $p(<n)$ distinct categories, can be solved for the case $p = 3$ by the approach of Brandt et al. [12] and for the case $3 < p < n$ by the algorithm developed by Aggarwal [2].

The quadratic assignment problem (QAP) is an important non-linear problem studied in the literature [1,6,7,18,32]. The QAP finds application in some important areas like facility location problems [26] and parallel and distributed computing [11]. Recently, an application of the biquadratic assignment problem has been reported in the literature in the field of VLSI synthesis [13]. Parallel branch and bound algorithms have been proposed for the exact solution of QAP [38,41] based on the Gilmore–Lawler lower bounding technique [22,29]. The main difficulty in these algorithms [38,41] is that the quality of the lower bounds is very poor for the QAP. Strengthening of the lower bounds is discussed by Balas [4]. However, the effective strategies for QAP still remain elusive.

Another important non-linear assignment problem is the time minimizing assignment problem (TMAP), also popularly known as the bottleneck assignment problem. TMAP has been considered by many researchers like Garfinkel [19], Ravindran and Ramaswamy [40] and Bhatia [10] under the usual assumption that work on all the n jobs commence simultaneously. Seshan [42] considered a generalized version of TMAP when n jobs are considered to be partitioned into $p(< n)$ blocks with precedence constraints on the jobs and the blocks. In one case, jobs in each block are performed in some sequential order but the various blocks can be commenced simultaneously, whereas in the other case the jobs within each block are commenced simultaneously but the p blocks are completed in a sequential order. These two generalizations, when either $p = 1$ or $p = n$ are equivalent to either the CMAP or the TMAP, and thus solvable in polynomial time. However, when $1 < p < n$, the usual solution procedures are not applicable. Using the domination characters, Aggarwal et al. [3] developed algorithms that are more efficient than the known branch-and-bound algorithms [42] for these TMAP under categorization. As a special case of constrained bottleneck problems in networks, Bermen et al. [8] have studied TMAP subject to an additional constraint on the total cost. The stochastic version of TMAP has been studied by De et al. [39]. Their primary objective is to maximize the probability that the bottleneck (time) value satisfies a specified target. De et al. [39] also examined the situation where the target is to be minimized, given that the probability of satisfying the target exceeds a specified threshold.

Some special cases of CMAP were presented by Subrahmanyam [45] where the number of persons is less than the number of jobs and a person is allowed to do more than one job, but a job is to be done by exactly one person. Such variants of TMAP are not so easy to handle and the solution strategy developed by Subrahmanyam [45] is not applicable. A variant of TMAP in which all the n jobs are to be done by a lesser number of persons, say $m(< n)$, each of which has to do at least one job, is studied in the present paper. It is assumed that (i) each job is done by exactly one person and, (ii) all the persons start their work on various jobs simultaneously but a person doing more than one job has to do them one after the other in any order. Unlike in assignment problems under categorization [2,3,12] in the present case categories are not known in advance. To the best of the authors' knowledge, no study has yet appeared in this type of TMAP.

Dynamic Programming approach may be used to solve this problem but the main drawback would be that all possible feasible assignments will be generated. Thus Dynamic Programming approach is quite inefficient.

An algorithm is developed based on the lexicographic search approach [21,36,37]. In this approach each solution to the problem is defined as a word and the solutions are generated in some hierarchy of their values. Each partial word (defined in Section 2) defines a block of words with itself as leader and a bound on the value of the objective function for this block of words is calculated. If this bound is more than the value of the objective function for a known feasible assignment, then the entire block of words is rejected.

The algorithm starts with a feasible assignment which provides a reasonably good upper bound, say T_0, on the value of the objective function. The initial feasible assignment is obtained using a heuristic approach in scheduling theory for processing jobs on parallel machines minimizing the makespan [14–16,20,30]. Our algorithm examines the optimality of this feasible assignment and if found non-optimal, proceeds to find an optimal feasible assignment in a finite number of iterations.

The mathematical model of the problem and some relevant definitions are given in Section 2 in which some results are established. Based upon these, an algorithm is proposed in Section 3 and illustrations to explain it, given in Section 4. In Section 5 some general comments are made on the study.

2. Theoretical development

There is a set $J = \{1, 2, \ldots, n\}$ of n jobs and a set $I = \{1, 2, \ldots, m\}$ of $m(< n)$ persons who have to complete all the n jobs with a restriction that one job will be allocated to a unique person and each person has to do at least one job. All the persons start doing the jobs simultaneously but a person doing more than one job has to do them one after the other. Let t_{ij} be the time that the ith person takes to complete the jth job. The aim is to find that assignment of persons to the jobs for which the corresponding time of completion of all the jobs is the minimum. If the decision variable x_{ij}, $(i,j) \in I \times J$ takes the value 1 when the ith person does the jth job and 0 otherwise, then the problem under study can be mathematically modelled as:

$$\text{(ITMAP)} \qquad \min \quad T(X) = \max_{i \in I} \left(\sum_{j=1}^{n} (t_{ij} \colon x_{ij} > 0) \right)$$

$$\text{s.t.} \quad \sum_{i \in I} x_{ij} = 1, \quad j \in J, \tag{1}$$

$$\sum_{j \in J} x_{ij} \geqslant 1, \quad i \in I, \tag{2}$$

$$x_{ij} = 0 \text{ or } 1 \quad (i,j) \in I \times J. \tag{3}$$

Since the number of persons is less than the number of jobs, we call this problem an *Imbalanced Time Minimizing Assignment Problem (ITMAP)*. Clearly it always has a feasible solution.

An assignment, $X = \{x_{ij}\}$, is one which satisfies (1) and (3), and $T(X)$ is the corresponding time of completion of the jobs. An assignment is called a feasible assignment if (2) is also satisfied.

For an assignment, $X = \{x_{ij}\}$, $\exists$ exactly one $i_j \in I$ for each $j \in J$ such that $x_{i_j j} = 1$ and $x_{ij} = 0$, $i \in I - \{i_j\}$. This assignment can also be represented by a row vector as follows:

$$w = (i_1, i_2, \ldots, i_n), \tag{4}$$

where, all i_j's are not distinct, clearly $|w| = |J| = n$. Thus, the assignment represented by (4) implies that the jth job is done by the i_j th person, $j = 1, 2, \ldots, n$.

Each assignment in its vector form (4) can be thought of as a word, w, of length n, with letters i_j's from the set I. Let $W = \{w\}$ be the set of all feasible words of length n. Then for a feasible word, say w, given by (4), the corresponding feasible assignment $X^w = \{x^w_{ij}\}$ is given by:

$$x^w_{i_j j} = 1, \quad j = 1, 2, \ldots, n$$

$$x^w_{ij} = 0 \quad (i,j) \in I \times J - \{(i_j, j) \colon j \in J\}.$$

The value of $T(X^w)$ for this feasible assignment corresponding to w is

$$T(X^w) = \max_{i \in w} \left(\sum_{j=1}^{n} (t_{ij} \colon x^w_{ij} = 1) \right).$$

2.1. Definitions

2.1.1. Alphabet matrix (AB)

It is an $m \times n$ matrix formed by the positions of the elements of the given $m \times n$ matrix $\{t_{ij}\}$ of times. The jth column of AB consists of the positions of the entries in the jth column of the matrix $\{t_{ij}\}$ when they are arranged in the non-decreasing order of their values. Let $ab(y,j)$ stands for the yth entry in the jth column of AB. Therefore, $ab(1,j)$ corresponds to the smallest entry in the jth column of the matrix $\{t_{ij}\}$ that is, $\min_i\{t_{ij}\} = t_{ab(1,j)j}$. If $y < z$, then $t_{ab(y,j)j} \leqslant t_{ab(z,j)j}$.

Thus, the jth column of AB is $[ab(1,j), ab(2,j), \ldots, ab(m,j)]'$ where, $(')$ stands for the transpose. Clearly $t_{ab(1,j)j} \leqslant t_{ab(2,j)j} \leqslant \cdots \leqslant t_{ab(m,j)j}$.

All the words in W can be systematically generated by considering the elements of the jth column of AB in the jth position $(j = 1,2,\ldots,n)$ of a word, i.e., $i_j \in \{ab(q,j), q = 1,2,\ldots,m\}$.

2.1.2. Partial word (Pw)

$\text{Pw} = (i_1, i_2, \ldots, i_r)$, $r \leqslant n$, represents a partial word. A partial assignment corresponding to it consists of assigning the jth job to the i_jth person, $j = 1,2,\ldots,r$ (jobs $r+1, r+2, \ldots, n$ are still to be assigned). Pw defines a block of words each of which has first r letters as $i_1, i_2, \ldots, i_r$. In this sense Pw is called the leader of this block of words. If a partial word is such that $|\bar{I}| > n - |\text{Pw}|$, then clearly this partial word cannot contain a feasible word, where $|\bar{I}|$ is the index set of unassigned persons. Such a partial word is called an *infeasible partial word.* On the other hand, $|\bar{I}| \leqslant n - |\text{Pw}|$ then Pw is called a *feasible partial word.*

Contribution to the objective function $T(\cdot)$ by the partial assignment, say X^{Pw}, corresponding to Pw is given by

$$T(X^{\text{Pw}}) = \max_{i \in \text{Pw}} \left(\sum_{j=1}^{r} (t_{ij}\colon x_{ij}^{\text{Pw}} = 1) \right).$$

Clearly, for a word w whose leader is Pw, we have $T(X^w) \geqslant T(X^{\text{Pw}})$.

2.2. Notations

T_0	starting upper bound on the value of the objective function $T(\cdot)$
J_s	$J - \{j_1, j_2, \ldots, j_{s-1}\}$ (clearly $J_1 = J$)
$\lvert\bar{\mathrm{I}}\rvert$	index set of unassigned persons
$\boxed{*}$	augmentation $\boxed{\cancel{*}}$: negation of augmentation
T_u	updated upper bound on the value of the objective function $T(\cdot)$
ϕ	emptyset

2.3. Finding of the starting upper bound on the value of the objective function $T(\cdot)$

For each $i \in I$, find $\min_{j \in J_i} t_{ij} = t_{ij_i}$ (say), and set $x_{ij_i} = 1$, $i \in I$. Thus each of the m persons is assigned to a unique job in the set $\{j_1, j_2, \ldots, j_m\}$. For allocation of the remaining jobs in J_{m+1}, proceed as follows: For each $k = 1,2,\ldots,(n-m)$

$$\min_{i \in I} \left(\sum_{j \in J - J_{m+k}} (t_{ij}\colon x_{ij} = 1) + \min_{j \in J_{m+k}} t_{ij} \right) = \sum_{j \in J - J_{m+k}} (t_{i_{m+k}j}\colon x_{i_{m+k}j} = 1) + t_{i_{m+k}j_{m+k}} \quad \text{(say)}.$$

Then allocate job j_{m+k} to person i_{m+k}, $k = 1,2,\dots,(n-m)$. T_0 will be given by

$$T_0 = \max_{i\in I}\left(\sum_{j\in J}(t_{ij}: x_{ij} = 1)\right).$$

This heuristic will provide the starting upper bound on the value of $T(\cdot)$ quite close to its optimal value.
A feasible assignment thus obtained is:

$$x_{ij} = \begin{cases} 1, & j = j_i \\ 0, & j \neq j_i \end{cases} \quad i \in I,$$

$$x_{ij} = \begin{cases} 1, & i = i_j \\ 0, & i \neq i_j \end{cases} \quad j \in J_{m+1}.$$

Let the feasible word corresponding to this feasible assignment be $w = (ab(y_1,1), ab(y_2,2),\dots,ab(y_n,n))$. The above feasible assignment can then be given as

$$x_{ab(y_j,j)j} = 1, \quad j = 1,2,\dots,n,$$
$$x_{ij} = 0, \quad (i,j) \in I\times J - \{(ab(y_j,j),j), \quad j = 1,2,\dots,n\}.$$

Theorem 1. $\mathrm{Pw} = (ab(y_1,1), ab(y_2,2),\dots,ab(y_r,r))$, $r \leqslant n$ *is a partial word for which* $T(X^{\mathrm{Pw}}) \geqslant T_{\mathrm{u}}\ \forall y_r = 1,2,\dots,m$ *where,* T_{u} *is the current upper bound on the optimal value of the objective function* $T(\cdot)$. *Then the partial word* $\overline{\mathrm{Pw}} = (ab(y_1,1), ab(y_2,2),\dots,ab(y_{r-1},r-1))$ *cannot contain a word with the corresponding value of* $T(\cdot)$ *less than* T_{u}.

Proof. $T(X^{\mathrm{Pw}}) \geqslant T_{\mathrm{u}}\ \forall y_r = 1,2,\dots,m$ implies that assigning the rth job to any of the m facilities, when the first $(r-1)$ jobs are being assigned respectively to the persons $ab(y_1,1), ab(y_2,2),\dots,ab(y_{r-1},r-1)$ yields partial words corresponding to each of which the contribution to the objective function $T(\cdot)$ is not less than T_{u}. This implies that the partial word $\overline{\mathrm{Pw}} = (ab(y_1,1), ab(y_2,2),\dots,ab(y_{r-1},r-1))$ cannot be augmented to generate a word corresponding to which the associated assignment yield value of the objective function $T(\cdot)$ is less than T_{u}. □

Remark 1. Whenever for a partial word Pw, we have $T(X^{\mathrm{Pw}}) < T_{\mathrm{u}}$ and $|\bar{I}| \leqslant n - |\mathrm{Pw}|$ then the partial word Pw may contain word with the corresponding value of $T(\cdot)$ less than T_{u}. Also when $T(X^{\mathrm{Pw}}) \geqslant T_{\mathrm{u}}\ \forall y_r = 1,2,\dots,m$, where, $\mathrm{Pw} = (ab(y_1,1), ab(y_2,2),\dots,ab(y_r,r))$, $r \leqslant n$, then the partial word Pw is abandoned and allocations of some or all the first $(r-1)$ jobs respectively to the persons $ab(y_1,1), ab(y_2,2),\dots,ab(y_{r-1},r-1)$ must be altered.

Alteration in the $\overline{\mathrm{Pw}} = (ab(y_1,1), ab(y_2,2),\dots,ab(y_{r-1},r-1))$ means examining the possibility of assigning the $(r-1)$th job to the person $ab(z,r-1)$ for $y_{r-1} < z \leqslant m$. If $y_{r-1} = m$, then we examine the possibility of assigning the $(r-2)$th to the person $ab(z,r-2)$, $y_{r-2} < z \leqslant m$ and so on. Thus if $y_1 = y_2 = \cdots = y_r = m$, then the partial word $\overline{Pw}$ cannot under go any alteration.

Theorem 2. *Consider a partial word* $Pw = (ab(y_1,1))$. *Let* $T(X^{\mathrm{Pw}}) \geqslant T_{\mathrm{u}}$, T_{u} *being the current upper bound on the optimal value of* $T(\cdot)$. *Then* T_{u} *is the optimal value of* $T(\cdot)$.

Proof. As $T(X^{\text{Pw}}) \geqslant T_{\text{u}}$, then in virtue of Remark 1 the partial word $\text{Pw} = (ab(y_1, 1))$ cannot contain a word with the corresponding value of $T(\cdot)$ less than T_{u}. Also as $t_{ab(z,1))1} \geqslant t_{ab(y_1,1))1}$ for all $z > y_1$, it follows that $T(X^{\widehat{\text{Pw}}}) \geqslant T_{\text{u}}$ for every $\widehat{\text{Pw}} = (ab(z, 1)), z \geqslant y_1$. Therefore, no partial word $\widehat{\text{Pw}} = (ab(z, 1))$, $z \geqslant y_1$ can contain a word with the corresponding value of $T(\cdot)$ less than T_{u}. Further as the words are generated from the partial words in a systematic manner in the decreasing order of their contribution to $T(\cdot)$, it follows that any word derived from the partial word $(ab(y, 1))$, $1 \leqslant y \leqslant y_1$ must have the corresponding value of $T(\cdot)$ not less than T_{u}. Therefore, there does not exist a word with the corresponding value of $T(\cdot)$ less than T_{u} and hence T_{u} is the optimal value of $T(\cdot)$. □

Theorem 3. *Let* $\text{Pw} = (ab(m, 1), ab(m, 2), \ldots, ab(m, r))$, $r \leqslant n$ *be a partial word satisfying* $T(X^{\text{Pw}}) < T_{\text{u}}$, T_{u} *being the current upper bound on the value of the objective function* $T(\cdot)$. *Let* $\overline{\text{Pw}} = \text{Pw} \uplus (ab(y_{r+1}, r + 1))$ *be the partial word derived from Pw such that* $T(X^{\overline{\text{Pw}}}) \geqslant T_{\text{u}}$ *for all* $y_{r+1} = 1, 2, \ldots, m$ *then* T_{u} *is the optimal value of* $T(\cdot)$.

Proof. As $T(X^{\overline{\text{Pw}}}) \geqslant T_{\text{u}}$ for all $y_{r+1} = 1, 2, \ldots, m$ it follows, in virtue of the Theorem 1, that Pw cannot contain a word with the corresponding value of the objective function $T(\cdot)$ less than T_{u}. Therefore, in virtue of the Remark 1, allocation of the first r jobs respectively to the persons $ab(m, 1), ab(m, 2), \ldots, ab(m, r)$ must be altered. But as these persons correspond to the last entries in the first r columns of the Alphabet Matrix, it follows that Pw cannot undergo any alteration. Also as all partial words are generated in a systematic manner in the decreasing order of their contribution to the objective function $T(\cdot)$, it follows that prior to Pw all the partial words $(ab(y_1, 1), ab(y_2, 2), \ldots, ab(y_r, r))$ for all the permutations of $y_1, y_2, \ldots, y_r$ in the set $\{1, 2, \ldots, m\}$ except the permutation $y_1 = y_2 = \cdots = y_r = m$ have their corresponding contribution to $T(\cdot)$ not better than that of Pw. Hence we cannot have a word with corresponding objective function value less than T_{u} implying that the optimal value of $T(\cdot)$ is T_{u}. □

3. Algorithm

Throughout the algorithm J is used to indicate the position of a column in the matrix AB. For example, $J = r$ would mean that currently column r of AB is being examined. K is used to indicate the position of an entry in the column of AB currently under investigation. The algorithm runs into the following steps:

Step 0 (Initialization): Construct the Alphabet Matrix $AB = \{ab(i, j)\}$. Compute T_0 and find the corresponding feasible assignment $w = (ab(y_1, 1), ab(y_2, 2), \ldots, ab(y_j, j), \ldots, ab(y_n, n))$. $T_{\text{u}} = T_0$. Set $\text{Pw} = w$ and go to step 1.

Step 1: Set $J = n$, $K = y_J$ $(= y_n)$, $I' = \phi$. Go to step 2.

Step 2: (*a*) If $K < m$, set $\text{Pw} = \text{Pw} \mathbin{\cap\!\!\!\!\!\ast\;} (ab(K, J))$. $\bar{\text{I}} = \bar{\text{I}} \uplus (ab(K, J))$, if $ab(K, J) \neq ab(y, J)$ for $y_1 \leqslant y \leqslant y_{J-1}$ and also set $x_{ab(K,J)J} = 0$. Next set $K = K + 1$ and go to step 3. (*b*) If $K = m$, set $\text{Pw} = \text{Pw} \mathbin{\cap\!\!\!\!\!\ast\;} (ab(K, J))$ and set $\bar{I}$ as in case (a) also set $x_{ab(K,J)J} = 0$. Next set $J = J - 1$ and go to step 4.

Step 3: Set $x_{ab(K,J)J} = 1$. If $ab(K, J) \in \bar{I}$ then update it as $\bar{\text{I}} = \bar{\text{I}} \mathbin{\cap\!\!\!\!\!\ast\;} (ab(K, J))$. Update the partial word as $\text{Pw} = \text{Pw} \uplus (ab(K, J))$. If this partial word is such that $\bar{I} \neq \phi$ and $|\bar{I}| > n - |\text{Pw}|$, then abandon this partial word as this is an infeasible partial word and go back to step 2. Otherwise compute $T(X^{\text{Pw}})$.

If $T(X^{\text{Pw}}) < T_{\text{u}}$, go to step 5. (Ref: Remark 1).

If $T(X^{\text{Pw}}) \geqslant T_{\text{u}}$ and $J > 1$, go to step 2. (Ref: Theorem 1).

If $T(X^{\text{Pw}}) \geqslant T_{\text{u}}$ and $J = 1$, go to step 6. (Ref: Theorem 2).

Step 4: If $J > 1$ and each letter in the updated partial word Pw corresponds to the last entry in the respective columns of the Alphabet Matrix (i.e., each letter $ab(y_j, j)$ of Pw has each $y_j = m$), then go to step 6 (Ref: Theorem 3). Otherwise when $J > 1$, then for this J set $K = y_J$ and go to step 2. If the updated $J = 1$, then for this J set $K = y_J (= y_1)$. If $K = y_1 < m$ then go to step 2(a) and if $K = y_1 = m$ then go to step 6.

Step 5: If $J < n$, then in virtue of Remark 1, set $J = J + 1$, if this updated $J = n$ and $\bar{I} = \{i'\}$ (say), and $t_{i'n}$ is the qth entry in the nth column of AB then set $K = q$ and go to step 3. Otherwise set $K = 1$ and go to step 3. If $J = n$, we get a new feasible word, say w, the assignment corresponding to which yields value of the objective function $T(\cdot)$ strictly less than T_u. Thus a tighter bound on the value of the objective function $T(\cdot)$ is obtained. Set PW $= w$ and go to step 1.

Step 6: Stop. The current upper bound T_u is the optimal value of $T(\cdot)$ and the corresponding feasible assignment is an optimal assignment.

Remark 2. It may be noted that Step 1 of the algorithm is executed only when a word is obtained whose optimality is still not established.

Step 2(a) of the algorithm is carried out when either (i) $K < m$ and $J > 1$, in which case the next entry in the column J of AB is considered or (ii) $T(X^{Pw}) \geqslant T_u$ or $|\bar{I}| > n - \text{Pw}$, in which case alteration in the last letter of Pw is made. Step 2(b) of the algorithm is executed when $K = m$ and $T(X^{\text{Pw}}) \geqslant T_u$ or $|\bar{I}| > n - \text{Pw}$, in which case alterations in the last two letters of Pw are made.

Step 3 is implemented only when either (i) $K < m$, in which case we examine the possibility of allocating the current job J to the next person with the minimum possible rise in the corresponding completion time or (ii) $J < n$ in which case allocation of the next job (i.e., job $J + 1$) is considered. Thus, step 3 is carried out to augment the current Pw with a new letter.

Step 4 is carried out only when $K = m$ and J is set as $J - 1$. Whenever $T(X^{\text{Pw}}) < T_u$ and $J < n$, the process iterates between step 3 and step 5 until either (i) a partial word, say Pw, is obtained for which $T(X^{\text{Pw}}) \geqslant T_u$ or (ii) a word contained in Pw is obtained.

Remark 3 (*Convergence*). The algorithm is bound to converge in a finite number of steps because (i) maximum number of feasible words is n^{m}, (ii) every new word constructed in the process yields a tighter upper bound on $T(\cdot)$ that is, the upper bound on the optimal value of $T(\cdot)$ is lowered by an integral value whenever a new word is obtained, and (iii) all the partial words obtained in the process yielding value of $T(\cdot)$ not less than T_u are abandoned. Infeasible partial words as and when encountered are also abandoned.

Remark 4 (*Memory*). It may be observed that only partial word Pw which is under consideration needs to be stored in the active memory and the Alphabet Matrix is prepared once for all and stored. Hence the active memory required is almost negligible.

Remark 5. Although the problem is NP-hard, the proposed algorithm can solve to optimality all the instances that can be generated. This is so because of the tightness of the upper bound T_0 computed initially. In very many instances the starting upper bound T_0 will be equal to the optimal value of the instance.

Remark 6. The proposed algorithm does not generate feasible assignments with the corresponding value of $T(\cdot)$ not less than T_0. In this respect this algorithm is better than the Dynamic Programming approach in which all feasible assignments will have to be obtained.

4. Numerical

Consider the following imbalanced TMAP which consists of three persons and five jobs. The (i, j)th entry of the given matrix $\{t_{ij}\}$ represents the time taken by the ith person to complete the jth job.

$$\{t_{ij}\} = \begin{array}{|c|c|c|c|c|}\hline 6 & 3 & 1 & 5 & 4 \\ \hline 5 & 5 & 4 & 3 & 3 \\ \hline 1 & 2 & 2 & 6 & 4 \\ \hline \end{array}$$

Step 0 (Initialization): Alphabet Matrix is given as

$$AB = \begin{array}{|c|c|c|c|c|}\hline 3 & 3 & 1 & 2 & 2 \\ \hline 2 & 1 & 3 & 1 & 1 \\ \hline 1 & 2 & 2 & 3 & 3 \\ \hline \end{array}$$

To find the initial upper bound on the value of the objective function $T(\cdot)$
$J_1 = \{1,2,3,4,5\}$. For $i = 1$, $\min_{j\in J_1}\{t_{1j}\} = t_{13} = 1$,
$J_2 = J_1 - \{3\} = \{1,2,4,5\}$. For $i = 2$, $\min_{j\in J_2}\{t_{2j}\} = t_{24} = 3$,
$J_3 = J_2 - \{4\} = \{1,2,5\}$. For $i = 3$, $\min_{j\in J_3}\{t_{3j}\} = t_{31} = 1$.
For the allocation of the remaining jobs in J_{m+1}, proceed as follows:
$J_4(= J_{m+1}) = \{2,5\}(= \{j_{m+1}, j_{m+2}, \ldots, j_n\})$.
For $j = 2 \in J_4$,

$$\min_{i\in I}\left(\sum_{j\in\{1,3,4\}} (t_{ij}\colon\ x_{ij} = 1) + \min_{j\in J_4} t_{ij}\right) = \sum_{j\in\{1,3,4\}} (t_{3j}\colon\ x_{3j} = 1) + t_{32} = t_{31} + t_{32} = 1 + 2\,.$$

Now allocate the second job to the third person and now $J_5 = \{5\}(= j_{m+2})$.
For $j = 5 \in J_5$,

$$\min_{i\in I}\left(\sum_{j\in\{1,2,3,4\}} (t_{ij}\colon\ x_{ij} = 1) + \min_{j\in J_5} t_{i5}\right) = \sum_{j\in\{1,2,3,4\}} (t_{1j}\colon\ x_{1j} = 1) + t_{15} = t_{13} + t_{15} = 1 + 4 = 5.$$

A feasible assignment thus obtained is given by $w = (3,3,1,2,1)$ and the objective function value corresponding to this feasible assignment is $T(X^w) = 5$.

Set $\mathrm{Pw} = (3,3,1,2,1) = (ab(1,1), ab(1,2), ab(1,3), ab(1,4), ab(2,5))$. Go to step 1 to examine the optimality of this word.

Step 1: $J = 5, K = y_5 = 2$, $\mathrm{Pw} = (3,3,1,2,1) = (ab(1,1), ab(1,2), ab(1,3), ab(1,4), ab(2,5))$, $\bar{I} = \phi$.

Step 2(a): $K = 2 < m(= 3), \mathrm{Pw} = (3,3,1,2,1) \overset{*}{\cap} (1) = (3,3,1,2)$, $\bar{I} = \phi$, $x_{ab(1,5)5} = 0$, $K = K + 1 = 3$. Go to step 3.

Step 3: $x_{ab_{(3,5)5}} = 1$, $\mathrm{Pw} = (3,3,1,2) \overset{*}{\cup} (ab(3,5)) = (3,3,1,2,3)$, $T(X^{\mathrm{Pw}}) = 7 > T_{\mathrm{u}}(= 5)$, $J > 1$. Go to step 2.

Step 2(b): $K = 3(= m), \mathrm{Pw} = (3,3,1,2,3) \overset{*}{\cap} (3) = (3,3,1,2)$, $\bar{I} = \phi$ and set $x_{ab_{(3,5)5}} = 0$. Next set $J = J - 1 = 4$ and go to step 4.

Step 4: Updated $J(=4) > 1$, $\mathrm{Pw} = (ab(1,1), ab(1,2), ab(1,3), ab(1,4))$, set $K = y_4 = 1(< m)$. Go back to step 2(a).

Step 2(a): $K = 1(< m)$ $\mathrm{Pw} = (3,3,1,2) \mathbin{\cap\!\!\!\!\ast} (2) = (3,3,1)$ $\bar{I} = \{2\}$ set $K = K + 1 = 2$. Go to step 3.

Step 3: $x_{ab_{(2,4)}4} = 1$, $\mathrm{Pw} = (3,3,1) \mathbin{\cup\!\!\!\!\ast} (ab(2,4)) = (3,3,1,1)$, $\bar{I} = \{2\}$, $T(X^{\mathrm{Pw}}) = 6 > T_{\mathrm{u}}(=5)$. Go to step 2.

Continuing like this we reach the first column of the Alphabet Matrix AB $J = 1$, $K = y_1 = 1$, $\mathrm{Pw} = \phi$, $K = K + 1 = 2$, $\bar{I} = \{1,2,3\}$.

Step 3: Set $x_{ab(2,1)1} = 1$, $\mathrm{Pw} = \phi \mathbin{\cup\!\!\!\!\ast} \{ab(2,1)\} = \{2\}$, $T(X^{\mathrm{Pw}}) = 5 = T_{\mathrm{u}}$. $J > 1$. Go to step 6. (Ref: Theorem 2).

Step 6: Stop. The current upper bound $T_{\mathrm{u}} = 5$ is the optimal value of $T(\cdot)$ and the feasible assignment corresponding to the feasible word yielding the value T_{u} given by $(3,3,1,2,1)$ is an optimal feasible assignment.

Consider another imbalanced TMAP

$$\{t_{ij}\} = \begin{array}{|c|c|c|c|c|} \hline 3 & 2 & 3 & 2 & 4 \\ \hline 1 & 3 & 2 & 3 & 3 \\ \hline 2 & 2 & 3 & 1 & 3 \\ \hline \end{array}$$

Step 0 (Initialization): Alphabet Matrix is given as

$$AB = \begin{array}{|c|c|c|c|c|} \hline 2 & 1 & 2 & 3 & 2 \\ \hline 3 & 3 & 1 & 1 & 3 \\ \hline 1 & 2 & 3 & 2 & 1 \\ \hline \end{array}$$

To find the initial upper bound on the value of the objective function $T(\cdot)$:
$J_1 = \{1,2,3,4,5\}$. For $i = 1$, $\min_{j \in J_1}\{t_{1j}\} = t_{12} = 2$,
$J_2 = J_1 - \{2\} = \{1,3,4,5\}$. For $i = 2$, $\min_{j \in J_2}\{t_{2j}\} = t_{21} = 1$,
$J_3 = J_2 - \{1\} = \{3,4,5\}$. For $i = 3$, $\min_{j \in J_3}\{t_{3j}\} = t_{34} = 1$.
For the allocation of the remaining jobs in J_{m+1}, proceed as follows:
$J_4(= J_{m+1}) = \{3,5\}$ $(= \{j_{m+1}, j_{m+2}, \ldots, j_n\})$.
For $j = 3 \in J_4$,

$$\min_{i \in I}\left(\sum\nolimits_{j \in \{1,2,4\}}(t_{ij} : x_{ij} = 1) + t_{i3}\right) = \sum\nolimits_{j \in \{1,2,4\}}(t_{2j} : x_{2j} = 1) + t_{23} = t_{21} + t_{23} = 1 + 2 = 3.$$

Now allocate the second job to the third person and now $J_5 = \{5\}(= j_{m+2})$.
For $j = 5 \in J_5$,

$$\min_{i \in I}\left(\sum\nolimits_{j \in \{1,2,3,4\}}(t_{ij} : x_{ij} = 1) + t_{i5}\right) = \sum\nolimits_{j \in \{1,2,3,4\}}(t_{3j} : x_{3j} = 1) + t_{35} = t_{34} + t_{35} = 1 + 3 = 4.$$

The feasible assignment thus obtained is given by $w = (2,1,2,3,3)$ and the objective function value corresponding to this feasible assignment is $T(X^{w}) = 4$.

Set Pw $= (2,1,2,3,3) = (ab(1,1), ab(1,2), ab(1,3), ab(1,4), ab(2,5))$. Go to step 1 to examine the optimality of this word.

Step 1: $J = 5, K = y_5 = 2$, Pw $= (2,1,2,3,3) = (ab(1,1), ab(1,2), ab(1,3), ab(1,4), ab(2,5))$, $\bar{I} = \phi$ and go to step 2(a).

Step 2(a): $K = 2 < m(= 3)$, Pw $= (2,1,2,3,3) \dot{\cap}_{*} (3) = (2,1,2,3)$, $\bar{I} = \phi$, $K = K + 1 = 3$. Go to step 3.

Step 3: $x_{ab_{(3,5)}5} = 1$, $\bar{I} = \phi$, Pw $= (2,1,2,3) \dot{\cup}_{*} (ab(3,5)) = (2,1,2,3,1)$, $T(X^{\text{Pw}}) = 6 > T_{\text{u}}(= 4)$, $J > 1$. Go to step 2.

Step 2(b): $K = 3(= m)$, Pw $= (2,1,2,3,1) \dot{\cap}_{*} (1) = (2,1,2,3)$, $\bar{I} = \phi$. Next set $J = J - 1 = 4$ and go to step 4.

Step 4: Updated $J(= 4) > 1$, Pw $= (ab(1,1), ab(1,2), ab(1,3), ab(1,4))$. Set $K = y_4 = 1(< m)$. Go back to step 2(a).

Step 2(a): $K = 1(< m)$, Pw $= (2,1,2,3) \dot{\cap}_{*} (3) = (2,1,2)$, $\bar{I} = \{3\}$. Set $K = K + 1 = 2$. Go to step 3.

Step 3: $x_{ab_{(2,4)}4} = 1$, Pw $= (2,1,2) \dot{\cup}_{*} (ab(2,4)) = (2,1,2,1)$, $\bar{I} = \{3\}$, $T(X^{\text{Pw}}) = 4 > T_{\text{u}}$. Go to step 2. Continuing like this we reach at the following stage.

Step 3: Pw $= (2,3,2,3)$, $\bar{I} = \{1\}$, $T(X^{\text{Pw}}) = 3 < T_{\text{u}}$. Go to step 5.

Step 5: As $J = 5$ and $\bar{I} = \{1\}$, set $K = 1$ and go to step 3.

Step 3: Set $x_{ab(1,5)5} = 1$, Pw = Pw $\dot{\cup}_{*} \{ab(1,5)\} = (2,3,2,3,1)$, $T(X^{\text{Pw}}) = 4 = T_{\text{u}}$. Go to step 2.

Proceeding like this we reach the stage where the partial word is as under Pw $= (1,2,3) = (ab(3,1), ab(3,2), ab(3,3))$ (i.e., each letter in the partial word corresponds to the last entry in the respective columns of the Alphabet Matrix). Go to step 6 (Ref: Theorem 3).

Step 6: Stop. The current upper bound $T_{\text{u}} = 4$ is the optimal value of $T(\cdot)$ and the feasible assignment corresponding to the feasible word yielding the value T_{u} given by $(2,1,2,3,3)$ is an optimal feasible assignment.

5. Concluding remarks

1. As the authors are not aware of any study on the ITMAP discussed in this paper, the main aim is to develop some solution strategy for the same. In the absence of any other algorithm for the ITMAP, the authors are unable to present any comparative results.
2. The proposed study can be easily extended to the case when only $n'(m < n' < n)$ jobs are to be done or when each person has to do at least a specified number of jobs.
3. Although the present study focuses on two dimensional TMAP, the results established in the paper hold for three dimensional assignment problem [17,31]. As three dimensional assignment problem is known to be NP-hard [17], the computational advantages of the established results are somewhat diminished.

Acknowledgements

The authors are thankful to the honourable referees for their critical and constructive comments which have helped them to improve the paper. This research was carried out under the financial support provided by Council of Scientific and Industrial Research, India.

References

[1] W.P. Adams, T.A. Johnson, Improved linear programming based lower bounds for the quadratic assignment problem, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 16 (1994) 43–75.

[2] V. Aggarwal, The assignment problem under categorized jobs, European Journal of Operational Research 14 (1983) 193–195.

[3] V. Aggarwal, V.G. Tikekar, L.-F. Hsu, Bottleneck assignment problems under categorization, Computers and Operations Research 13 (1) (1986) 11–26.
[4] E. Balas, S. Ceria, G. Cornuejols, A lift-and-project cutting plane algorithm for mixed 0-1 programs, Mathematical Programming 58 (1993) 295–324.
[5] R.S. Barr, F. Glover, D. Klingman, The alternating basis algorithm for assignment problems, Mathematical Programming 13 (1977) 1–13.
[6] M.S. Bazzara, H.D. Sherali, Bender's partitioning scheme applied to a new formulation of the quadratic assignment problem, Naval Research Logistics Quarterly 27 (1980) 29–41.
[7] M.S. Bazzara, H.D. Sherali, On the use of exact and heuristic cutting plane methods for the quadratic assignment problem, Journal of Operational Research Society 33 (1982) 991–1003.
[8] O. Berman, D. Einav, G. Handler, The constrained bottleneck problem in networks, Operations Research 38 (1990) 178–181.
[9] D.P. Bertsekas, A new algorithm for the assignment problem, Mathematical Programming 21 (1981) 152–171.
[10] H.L. Bhatia, Time minimizing assignment problem, Systems and Cybernetics in Management 6 (1977) 75–83.
[11] S.H. Bokhari, Assignment problems in distributed and parallel computing, Kluwer Academic Publishers, Boston, 1987.
[12] A. Brandt, Y. Intrator, The assignment problem with three job categories, Casopis Pro Pestovani Matematiky 96 (1971) 8–11.
[13] R.E. Burkard, E. Cela, B. Klinz, On the biquadratic assignment problem, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 16 (1994) 117–146.
[14] T.C.E. Cheng, C.C.S. Sin, A state-of-art review of parallel-machine scheduling research, European Journal of Operational Research 47 (1990) 271–292.
[15] P. De, T.E. Morton, Scheduling to minimum makespan on unequal parallel processors, Decision Science 11 (1980) 586–602.
[16] S. French, Sequencing and Scheduling – Introduction to the Mathematics of the Job Shop, Ellis Horwood, Chichester, 1982.
[17] A.M. Frieze, Complexity of a 3-dimensional assignment problem, European Journal of Operational Research 14 (1983) 161–164.
[18] A.M. Frieze, J. Yadegar, On the quadratic assignment problem, Discrete and Applied Mathematics 5 (1983) 89–98.
[19] R.S. Garfinkel, An improved algorithm for bottleneck assignment problem, Operations Research 18 (1971) 1717–1751.
[20] R.G. Parker, Deterministic Scheduling Theory, Chapman and Hall, 1995.
[21] S. Ghose, The maximum capacity routes: A lexisearch approach, Opsearch 8 (3) (1971) 209–225.
[22] P.C. Gilmore, Optimal and suboptimal algorithms for the quadratic assignment problem, SIAM Journal on Applied Mathematics 10 (1962) 305–313.
[23] F. Glover, D. Karney, D. Klingman, Implementation and computational comparisons of primal, dual and primal-dual computer codes for minimum cost network flow problems, Networks 4 (3) (1974) 191–212.
[24] R.S. Hatch, Bench marks comparing transportation codes based on primal simplex and primal-dual algorithms, Operations Research 23 (1975) 1167–1171.
[25] M.S. Hung, W.O. Rom, Solving the assignment problem by relaxation, Operations Research 28 (1980) 969–982.
[26] T.C. Koopmans, M.J. Beckmann, Assignment problems and the location of economic activities, Econometrica 25 (1957) 53–76.
[27] H.W. Kuhn, The Hungarian method for assignment problem, Naval Research Logistics Quarterly 2 (1956) 83–97.
[28] E.L. Lawler, Combinatorial optimization, Networks and Matroids, Holt, Reinhart and Winston, New York, 1976.
[29] E.L. Lawler, The quadratic assignment problem, Management Science 9 (1963) 586–599.
[30] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnoy Kan, D.B. Shmoys, Sequencing and Scheduling: Algorithms and Complexity, Report BS-R8909, Centre for Mathematics and Computer Science, Amsterdam, 1989.
[31] R. Malhotra, H.L. Bhatia, M.C. Puri, The three dimensional bottleneck assignment problem and its variants, Optimization 16 (1985) 245–256.
[32] T. Mautor, C. Roucairol, Difficulties of exact methods for solving the quadratic assignment problem, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 16 (1994) 263–274.
[33] J.B. Mazzola, A.W. Neebe, Resource constraint assignment scheduling, Operations Research 34 (4) (1986) 560–572.
[34] L.F. McGinnis, Implementation and testing of a primal-dual algorithm for the assignment problem, Operations Research 31 (1983) 277–291.
[35] K.G. Murty, An algorithm for ranking all the assignments in order of increasing cost, Operations Research 16 (1968) 628–687.
[36] S.N.N. Pandit, Y.V. Subrahmanyam, Enumeration of all optimal job sequence, Opsearch 12 (1-2) (1975) 35–39.
[37] S.N.N. Pandit, M.S. Murthy, Allocation of sources and destinations, 8th Annual Convention of Operational Research Society of India, 1975, pp. 22–24.
[38] P.M. Pardalos, J.V. Crouse, A parallel algorithm for the quadratic assignment problem, Proceedings Supercomputing 89 (1989) 351–360.
[39] P. De, J.B. Ghosh, C.E. Wells, On the solution of a stochastic bottleneck assignment problem and its variants, Naval Research Logistics 39 (1992) 389–397.
[40] A. Ravindran, V. Ramaswamy, On the bottleneck assignment problem, Journal of Optimization Theory And Applications 21 (1977) 451–458.

[41] C. Roucairol, A parallel branch and bound algorithm for the quadratic assignment problem, Discrete and Applied Mathematics 18 (1987) 211–225.
[42] C.R. Seshan, Some generalisations of time minimizing assignment problem, Journal of Operational Research Society 32 (1981) 489–494.
[43] V. Srinivasan, G.L. Thompson, Accelerated algorithms for labeling and relabeling of trees with applications to distribution problems, Journal of Association for Computing Machinery 19 (1972) 712–726.
[44] V. Srinivasan, G.L. Thompson, Benefit cost analysis of coding techniques for the primal transportation algorithm, Journal of Association for Computing Machinery 20 (1973) 194–213.
[45] Y.V. Subrahmanyam, Some special cases of assignment problems, Opsearch 16 (1) (1979) 45–47.
[46] V. Verma, Optimization over a polyhedral domain: Some Aspects, Ph.D. Thesis, IIT Delhi, 1990.