ELSEVIER

Discrete Applied Mathematics 65 (1996) 123–139

DISCRETE APPLIED MATHEMATICS

Three-dimensional axial assignment problems with decomposable cost coefficients

Rainer E. Burkard*, Rüdiger Rudolf, Gerhard J. Woeginger

Institut für Mathematik B, TU Graz, Steyrergasse 30, A-8010 Graz, Austria

Received 19 October 1992; revised 16 December 1993

Abstract

Given three n-element sequences a_i, b_i and c_i of nonnegative real numbers, the aim is to find two permutations ϕ and ψ such that the sum $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$ is minimized (maximized, respectively). We show that the maximization version of this problem can be solved in polynomial time, whereas we present an NP-completeness proof for the minimization version. We identify several special cases of the minimization problem which can be solved in polynomial time, and suggest a local search heuristic for the general case.

Keywords: Three-dimensional assignment problems; Decomposable cost coefficients; Complexity; Special cases; Heuristics

1. Introduction

The general axial three-dimensional assignment problem, 3AP for short, is well known to be NP-hard [13]. For this type of problem, only implicit enumeration methods are known. The first branch and bound methods are due to Vlach [16] and Pierskalla [14], a primal–dual implicit enumeration method based on a graph theoretic approach was designed by Hansen and Kaufman [11]. Fröhlich [9] and Burkard and Fröhlich [6] improved the bounding technique by using subgradient optimization. Recently polyhedral approaches were applied to this problem by Balas and Saltzman [3] (see [7] for further literature concerning 3APs).

Since 3AP is NP-hard, the computational complexity of the 3AP when restricted to special cases is of interest:

Crama and Spieksma [8] considered 3APs where the cost coefficients fulfill some special triangle inequalities. Though the triangle inequalities make the problem easier to approximate, it does not remove the NP-hardness from the problem [8].

Since the classical two-dimensional assignment problem becomes simpler, if its cost coefficients c_{ij} can be decomposed into $c_{ij} = a_i b_j$ (see Section 2), we consider in this

* Corresponding author.

SSDI 0166-218X(95)00031-3

paper the three-dimensional case with decomposable cost coefficients $d_{ijk} = a_i b_j c_k$. So, given three n-element sequences a_i, b_i and c_i of nonnegative numbers, our problem consists in finding two permutations ϕ and ψ such that the sum $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$ attains its minimum (maximum, respectively). This problem can be formulated as 3AP in the following way:

$$\text{min (max)} \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} a_i \cdot b_j \cdot c_k \cdot x_{ijk}$$

$$\text{subject to} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ijk} = 1 \quad \forall k = 1,\ldots,n,$$

$$\sum_{i=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \forall j = 1,\ldots,n,$$

$$\sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \forall i = 1,\ldots,n,$$

$$\text{where } x_{ijk} \in \{0,1\}.$$

Although the minimization and the maximization look very similar at a first glance, their computational complexity is totally different. In Section 2, we present a polynomial time algorithm for the maximization version. In Section 3, we give an NP-completeness proof for the minimization version, and show that this version of the problem is inherently hard to approximate. In Section 4, various polynomially solvable cases for the minimization version are inspected. Section 5 deals with some structural properties of the cost coefficients and Section 6 contains a promising local search heuristic and computational results.

2. The maximization case

In order to achieve a "good" solution method for maximizing $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$, we turn back to the simpler two-dimensional problem. For two dimensions, the way of rearranging two n-element sequences a_i and b_i such that their scalar product becomes maximum is well understood and summarized in the following proposition.

Proposition 2.1 (Hardy, et al. [12]). *Let a_i and b_i be two n-element sequences of real numbers sorted in nondecreasing order and let ϕ be an arbitrary permutation of $\{1,\ldots,n\}$. Then we have*

$$\sum_{i=1}^{n} a_i b_{n-i+1} \leqslant \sum_{i=1}^{n} a_i b_{\phi(i)} \leqslant \sum_{i=1}^{n} a_i b_i$$

for all permutations ϕ.

We use Proposition 2.1 to prove the following lemma.

Lemma 2.2. *Given three sequences a_i, b_i and c_i consisting of n nonnegative numbers sorted in nondecreasing order, then*

$$\sum_{i=1}^{n} a_i b_i c_i = \max_{\phi,\psi} \sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}.$$

Proof. For two permutations ϕ and ψ we define a function

$$h(\phi,\psi) := |\{(i,j) : 1 \leqslant i < j \leqslant n, \phi(i) > \phi(j)\}| + |\{(i,j) : 1 \leqslant i < j \leqslant n, \psi(i) > \psi(j)\}|$$

to count the total number of inversions in ϕ and ψ. The following three observations are obvious.

(i) $h(\phi,\psi) \geqslant 0$ for all permutations ϕ and ψ,
(ii) $h(\phi,\psi) = 0$ iff ϕ and ψ are the identical permutation I,
(iii) $h(\phi,\psi) \leqslant n(n-1)$.

Let us assume that (ϕ,ψ) gives an optimal solution to the maximization problem with $(\phi,\psi) \neq (I,I)$. Then we have $h(\phi,\psi) > 0$ telling us that there are at least two indices $i_1 < i_2$ such that in the corresponding terms $a_{i_1} b_{\phi(i_1)} c_{\psi(i_1)}$ and $a_{i_2} b_{\phi(i_2)} c_{\psi(i_2)}$ the following holds: $\phi(i_1) > \phi(i_2)$ or $\psi(i_1) > \psi(i_2)$. But then we construct a new solution $(\tilde{\phi},\tilde{\psi})$ by :

$$\tilde{\phi}(i) := \begin{cases} \min\{\phi(i_1),\phi(i_2)\}, & i = i_1, \\ \max\{\phi(i_1),\phi(i_2)\}, & i = i_2, \\ \phi(i), & \text{otherwise.} \end{cases}$$

and

$$\tilde{\psi}(i) := \begin{cases} \min\{\psi(i_1),\psi(i_2)\}, & i = i_1, \\ \max\{\psi(i_1),\psi(i_2)\}, & i = i_2, \\ \psi(i), & \text{otherwise.} \end{cases}$$

But then $h(\tilde{\phi},\tilde{\psi}) < h(\phi,\psi)$ and using Proposition 2.1 and the nonnegativity of the elements, we get

$$\sum_{i=1}^{n} a_i (b_{\tilde{\phi}(i)} c_{\tilde{\psi}(i)}) \geqslant \sum_{i=1}^{n} a_i (b_{\phi(i)} c_{\psi(i)}).$$

So $(\tilde{\phi},\tilde{\psi})$ is an optimal solution, too. Hence, it is possible to construct a sequence $\{(\tilde{\phi}_i,\tilde{\psi}_i)\}_{i=1}^{N}$ of optimal solutions with strictly decreasing values $h(\cdot,\cdot)$. By observation (iii) above, after a finite number of transformations the optimal solution (I,I) is reached. □

Lemma 2.2 can be generalized in a straightforward way to higher-dimensional axial assignment problems with m sequences $p_i^{(j)}$.

Theorem 2.3. *Let $p_i^{(j)}$ be m n-element sequences with nonnegative elements sorted increasingly and let ϕ_j be m arbitrary permutations of $\{1,\dots,n\}$. Then we have that*

$$\sum_{i=1}^{n}\prod_{j=1}^{m} p_i^{(j)} = \max_{\phi_1,\dots,\phi_m} \sum_{i=1}^{n}\prod_{j=1}^{m} p_{\phi_j(i)}^{(j)}.$$

We conclude this section with the following two remarks.

Remark 1. The array D defined as $d_{ijk} := -a_i b_j c_k$ fulfills the three-dimensional Monge property as first proposed by Aggarwal and Park [1, 2]. Therefore, Lemma 2.2 and also Theorem 2.3 can be seen as a special case of a more general result of Bein et al. [4]. These authors have shown that the lexicographical greedy algorithm solves the d-dimensional transportation problem if and only if the cost array possesses the Monge property.

Remark 2. If we look at the corresponding bottleneck problem, where the sum in the objective function is replaced by a maximum, the following interesting property holds: the optimal solution of the sum problem is also an optimal solution to the corresponding bottleneck problem. This is due to the fact that both optimal solutions are just determined by ordering the given cost coefficients in the same way.

3. NP-completeness of the minimization problem

In this section we will prove that the minimization version of the 3AP with decomposable cost coefficients is NP-hard.

Theorem 3.1. *Let three n-element sequences a_i, b_i and c_i of nonnegative rational numbers and a bound S be given. Then:*

(i) *It is* NP*-complete to decide whether there exist permutations ϕ and ψ such that $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)} \leqslant S$.*

(ii) *For each $k \geqslant 1$, it is* NP*-hard to approximate the optimum solution within a factor of n^k.*

Proof. The proof is done by reducing the NP-complete NUMERICAL THREE-DIMENSIONAL MATCHING problem (N3DM) to our problem (see [10, Problem SP16]).

Instance: A bound B; three n-element pairwise disjoint sets $\{w_i\}$, $\{x_i\}$ and $\{y_i\}$ of positive integers with total sum nB, for some $n \geqslant 2$.

Question: Can $\{w_i\} \cup \{x_i\} \cup \{y_i\}$ be partitioned into n disjoint sets $\{w_i, x_{\phi(i)}, y_{\psi(i)}\}$ such that $w_i + x_{\phi(i)} + y_{\psi(i)} = B \,\forall i$?

Now let $\{w_i\}$, $\{x_i\}$ and $\{y_i\}$ together with B constitute an N3DM instance. Since the N3DM problem even is NP-complete *in the strong sense* (cf. [10]), we may assume

that all numbers in the instance are represented in *unary* notation (i.e. a string of m 1's represents the number m). We use this representation to ensure that our reduction indeed is a polynomial time reduction.

For some $k \geqslant 1$, we let $Z = n^{k+1}$ and we define $a_i = Z^{w_i}$, $b_i = Z^{x_i}$ and $c_i = Z^{y_i}$, for $1 \leqslant i \leqslant n$. Moreover, we set S to nZ^B. It is straightforward to see that the *binary* representation of all these new numbers can be calculated in an overall time that is polynomial in the length of the N3DM instance. (This would not be possible if the numbers in the N3DM instance were *binary* encoded. For example, the representation of a number m in binary representation has length $O(\log m)$, but the binary representation of Z^m has length $m \log Z$, which gives an exponential increase in the length and, consequently, in the time.)

We claim that the constructed assignment problem has a solution if and only if the N3DM instance has a solution.

(if) Let $\{w_i, x_{\phi(i)}, y_{\psi(i)}\}$ be a solution for the N3DM problem. But then it is easy to see that $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$ equals $nZ^B = S$.

(only if) Now we assume that N3DM is not solvable. This implies that $\forall \phi, \psi \exists\, i$: $w_i + x_{\phi(i)} + y_{\psi(i)} \geqslant B + 1$. Consequently $\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$ contains always one term that is at least Z^{B+1}. But then the total sum is bounded from below by $Z^{B+1} + 1 = (n^{k-1})^{B+1} + 1 > n^k (n^{kB+B+1}) = n^k S > S$.

Summarizing, the minimum sum either equals S (if the N3DM is solvable) or is at least a factor of n^k away from S, if the N3DM is not solvable. □

Not only the minimization sum problem is NP-hard, but also the corresponding bottleneck problem is NP-hard. This is summarized in the subsequent theorem.

Theorem 3.2. *Let three n-element sequences a_i, b_i and c_i of nonnegative rational numbers and a bound S be given. Then:*

(i) *It is* NP*-complete to decide whether there exist permutations ϕ and ψ such that* $\max_i \{a_i b_{\phi(i)} c_{\psi(i)}\} \leqslant S$.

(ii) *For each $k \geqslant 1$, it is* NP*-hard to approximate the optimum solution within a factor of n^k.*

Proof. The proof is done in a similar way as the proof for Theorem 3.1. Again we reduce N3DM to our problem. Let $\{w_i\}$, $\{x_i\}$ and $\{y_i\}$ together with B form an instance of N3DM. For some $k \geqslant 1$, we define $Z = n^{k+1}$ and $a_i = Z^{w_i}$, $b_i = Z^{x_i}$ and $c_i = Z^{y_i}$, for $1 \leqslant i \leqslant n$ and fix S to Z^B.

We claim that the constructed assignment problem has a solution if and only if the N3DM instance has a solution.

(if) Let $\{w_i, x_{\phi(i)}, y_{\psi(i)}\}$ be a solution for the N3DM problem. But then it is easy to see that $\max_i \{a_i b_{\phi(i)} c_{\psi(i)}\}$ equals $Z^B = S$.

(only if) Now we assume that N3DM is not solvable. This implies that $\forall \phi, \psi \exists i$: $w_i + x_{\phi(i)} + y_{\psi(i)} \geqslant B + 1$. Consequently $\max_i \{a_i b_{\phi(i)} c_{\psi(i)}\}$ contains always one term that is at least Z^{B+1}. But then the maximum is at least $Z^{B+1} > n^k S$.

So we have shown that the maximum either equals S (if the N3DM is solvable) or is at least a factor of n^k away from S, if the N3DM is not solvable. □

4. Polynomially solvable special cases

As the minimization problem is NP-complete and hopelessly hard to approximate, we are interested in special cases which are computationally better tractable. In this section we will investigate some restrictions that make the problem solvable in polynomial time.

Before formulating our restrictions let us make two observations concerning the given sequences. We only have to consider instances with strictly positive sequence elements, since a 0-element always has to be matched with the largest elements of the other two sequences. A second observation is that only sequences with smallest element equal to 1 have to be considered. This is no restriction, since this can always be reached by scaling, i.e. division by the smallest value.

So, after these evident observations, a first possibility is to bound the number of distinct values in the sequences A, B and C. This suggests the following restriction which can be seen as a preparation for further more general cases.

Restriction 1. The sequences are of the form $A = (1,\ldots,1,x,\ldots,x)$, $B = (1,\ldots,1,y,\ldots,y)$ and $C = (1,\ldots,1,z,\ldots,z)$ with $1 < x \leqslant y \leqslant z$. Let a, b and c denote the number of x's occurring in the sequence A, of y's in B and of z's in C, respectively.

Lemma 4.1. *Let* $\mathcal{S} \subseteq \{1\cdot 1\cdot 1, x\cdot 1\cdot 1, 1\cdot y\cdot 1, 1\cdot 1\cdot z, x\cdot y\cdot 1, x\cdot 1\cdot z, 1\cdot y\cdot z, x\cdot y\cdot z\}$ *denote the set of terms occurring in the objective function in an optimum solution of a problem fulfilling Restriction* 1. *Then there is an optimal solution such that* $\mathcal{S}$ *does not contain*

(i) $\{1\cdot 1\cdot 1\}$ *and an element of* $\{x\cdot y\cdot 1, x\cdot 1\cdot z, 1\cdot y\cdot z, x\cdot y\cdot z\}$ *at the same time,*
(ii) $\{x\cdot 1\cdot 1\}$ *and an element of* $\{1\cdot y\cdot z, x\cdot y\cdot z\}$ *at the same time,*
(iii) $\{1\cdot y\cdot 1\}$ and an element of $\{x\cdot 1\cdot z, x\cdot y\cdot z\}$ at the same time,
(iv) $\{1\cdot 1\cdot z\}$ *and* $\{x\cdot y\cdot z\}$ *at the same time.*

Proof. Suppose $\mathcal{S}$ would contain the terms $1\cdot 1\cdot 1$ and $x\cdot y\cdot 1$. Replacing them by $x\cdot 1\cdot 1$ and $1\cdot y\cdot 1$ would decrease the sum, as $1+xy > x+y$ holds. The other cases are settled by analogous arguments. □

To derive a polynomial time algorithm, we apply Lemma 4.1 in the following cases and subcases.

Case I: $a+b+c \leqslant n$. This case is demonstrated in Fig. 1. If one of the terms $x\cdot y\cdot 1$, $x\cdot 1\cdot z$, $1\cdot y\cdot z$ or $x\cdot y\cdot z$ occurs in the optimum solution, the term $1\cdot 1\cdot 1$ must occur at the same time, a contradiction to Lemma 4.1(i). The value v of the optimal solution

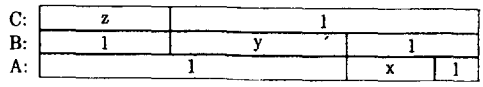

Fig. 1. Situation of Case I.

Fig. 2. Situation of Case II(i).

is given by

$$v := a \cdot x + b \cdot y + c \cdot z + n - (a + b + c).$$

Case II: $a + b + c > n$ and $a + b + c < 2n$. Here we distinguish four subcases in the following way.

(i) $b + c \geqslant n$ and $a + c \leqslant n$: This situation is depicted in Fig. 2. Since $b + c \geqslant n$ holds, at least $b + c - n$ times an element $y \in B$ and an element $z \in C$ must be paired together. The remaining elements x are fixed to y since $y \leqslant z \leqslant yz$. The optimal solution value is

$$v := (b + c - n) \cdot yz + a \cdot xy + (n - a - c) \cdot y + (n - b) \cdot z.$$

(ii) $b + c \geqslant n$ and $a + c > n$: The elements z and y are matched as in (i), the x's are matched to the y's. But since this time $a + c > n$ holds, we must give the remaining elements $x \in A$ to elements z because $z \leqslant yz$. The optimal assignment is shown in Fig. 3, and has the value

$$v := (b + c - n) \cdot yz + (n - c) \cdot xy + (a + c - n) \cdot xz + (2n - a - b - c) \cdot z.$$

(iii) $b + c < n$ and $a + c \leqslant n$: Since $b + c < n$, terms like $x \cdot y \cdot z$ or $1 \cdot y \cdot z$ are not optimal. Therefore, no element z is matched with an y. The elements $x \in A$ fill up the free positions in the assignment and are then combined with y, since $y \leqslant z$. The optimal solution is illustrated in Fig. 4 and yields

$$v := (n - b - c) \cdot x + c \cdot z + (a + b + c - n) \cdot xy + (n - a - c) \cdot y.$$

(iv) $b + c < n$ and $a + c > n$: This case is similar to (iii), the additional x's have to be matched with elements $z \in C$. Hence, we get the optimal assignment, shown in Fig. 5, with value

$$v := (n - b - c) \cdot x + b \cdot xy + (n - a) \cdot z + (a + c - n) \cdot xz.$$

Case III: $a + b + c \geqslant 2n + 1$. This case is treated similarly to Case I, with the only difference that here the blocks of 1's (instead of the blocks containing x, y or z) have

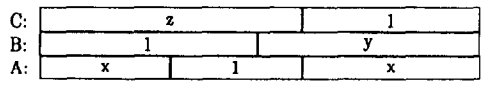

Fig. 3. Illustration of Case II(ii).

Fig. 4. Optimal assignment of Case II(iii).

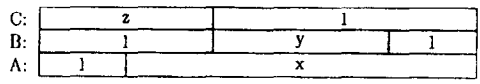

Fig. 5. Solution of Case II(iv).

to be arranged. By this we derive the optimal assignment value

$$v := (n-c)\cdot xy + (n-a)\cdot yz + (n-b)\cdot xz + (a+b+c-2n)\cdot xyz.$$

Summarizing, we formulate the following theorem.

Theorem 4.2. *The minimization problem for an input obeying Restriction* 1 *can be solved in* $O(n)$ *time.*

Now we are prepared to formulate a more general case in the subsequent restriction :

Restriction 2. The sequences are of the form $A = (1,\ldots,1,x,\ldots,x)$, $B = (1,\ldots,1,y,\ldots,y)$ and $C = (c_1,c_2,\ldots,c_n)$ with $1 < x \leqslant y$ and $1 = c_1 \leqslant c_2 \leqslant \cdots \leqslant c_n$. Let a, b denote the number of x's occurring in the sequence A and of y's in B.

Lemma 4.3. *Let* $c_i < c_j$ *and* S *denote the set of terms occurring in the objective function in an optimum solution of a problem fulfilling Restriction* 2. *Then* S *does not contain*

(i) $\{1\cdot 1\cdot c_i\}$ *and an element of* $\{x\cdot 1\cdot c_j, 1\cdot y\cdot c_j, x\cdot y\cdot c_j\}$ *at the same time,*
(ii) $\{x\cdot 1\cdot c_i\}$ *and an element of* $\{1\cdot y\cdot c_j, x\cdot y\cdot c_j\}$ *at the same time,*
(iii) $\{1\cdot y\cdot c_i\}$ *and* $\{x\cdot y\cdot c_j\}$ *at the same time.*

Proof. Assume that $1\cdot 1\cdot c_i$ and $x\cdot 1\cdot c_j$ occur in an optimal solution. Since $x\cdot 1\cdot c_i + 1\cdot 1\cdot c_j < 1\cdot 1\cdot c_i + x\cdot 1\cdot c_j$ we get a better solution by exchanging 1 and x. The other cases are treated in the same way. □

The following algorithm solves our minimization problem fulfilling Restriction 2.

Algorithm 1.

(0) Start with following initialization:
$a_i := 1 \quad \forall i = 1,\dots,n$
$b_i := y \quad \forall i = 1,\dots,b$
$b_i := 1 \quad \forall i = b+1,\dots,n$
$i := 1$, $j := b+1$ and $k := a$.

(1) While $k > 0$ do
if $j > n$ then $a_i := x$, $i := i+1$
else
if $c_i y \leqslant c_j$ then $a_i := x$, $i := i+1$
else $a_j := x$, $j := j+1$
$k := k-1$.

(2) Compute $v := \sum_{i=1}^{n} a_i b_i c_i$.

Theorem 4.4. *Algorithm* 1 *solves the minimization problem fulfilling Restriction* 2 *in* $O(n)$ *time.*

Proof. Using Lemma 4.3 we see that all elements $y \in B$ have to be paired with the b smallest values of sequence C. By this, we get two increasing sequences $S_1 := (yc_1, yc_2, \dots, yc_b)$ and $S_2 := (c_{b+1}, \dots, c_n)$. Referring to Proposition 2.1 the remaining elements x must be assigned to the a smallest values of S_1 and S_2. Therefore, we only have to compare the first elements of S_1 and S_2. The minimum is matched with x and deleted from its sequence. Repeating this step till all x are assigned delivers an optimal solution to our problem.

Since C is sorted and at most n iterations have to be computed while matching the x's, we get the claimed time complexity of $O(n)$. □

If we use the result of Blum et al. [5] that the kth largest number of an unsorted n-element sequence can be found in $O(n)$, we can relax our assumption that sequence C has to be sorted. Searching for the bth largest number in C, splitting it into two sequences with smaller and larger values than the b largest element – which can also be done in $O(n)$ time – and assigning the y's to the sequence with smaller values yield the same result as the initialization step in Algorithm 1. Again applying [5] we split the resulting sequence into two parts and assign the elements x to the a smallest values. So we can also derive the optimal assignment in $O(n)$, if C is arbitrary.

If we look at the above special case in more detail, we see that the basic main idea is to apply twice Proposition 2.1. Therefore, we consider the following algorithm SIMPLE which is a generalization of Algorithm 1.

Let three sequences R, S and T be given which are sorted increasingly. Then SIMPLE (R,S,T) first constructs a sequence U with $u_i := r_i \cdot s_{n-i+1}$. Note that according to Proposition 2.1 $\sum_{i=1}^{n} u_i$ is an optimal value for $\min_\phi \sum_{i=1}^{n} r_i s_{\phi(i)}$. Then we arrange the

sequences U and T according to Proposition 2.1 by sorting U in increasing and T in decreasing order.

In the following we derive two different conditions on the three sequences A, B and C such that algorithm SIMPLE yields an optimal solution.

Theorem 4.5. *Let three sequences* $A = (1,\dots,1,x,\dots,x)$, $B = (y_1, y_2,\dots,y_n)$ *and* $C = (c_1, c_2,\dots,c_n)$ *with* $1 \leqslant y_1 \leqslant \cdots \leqslant y_n$ *and* $1 \leqslant c_1 \leqslant \cdots \leqslant c_n$ *be given. Then the optimal assignment can be found in* $O(n \log n)$ *time using* SIMPLE, *if either*

(i) $x \geqslant y_i \quad \forall i = 1,\dots,n$ *or*

(ii) $x \leqslant y_{i+1}/y_i \quad \forall i = 1,\dots,n-1$.

Proof. (i) We claim that we get an optimal solution by applying SIMPLE (A, C, B). Given an arbitrary solution $X := a_{i_1} b_{j_1} c_{k_1} + \cdots + a_{i_n} b_{j_n} c_{k_n}$ we define a function $h(X) := \sum_{\ell=1}^{n} i_\ell k_\ell$. Note that $h(X)$ is minimal if and only if the sequences A and C are arranged according to Proposition 2.1. So assume we have given an arbitrary optimal solution Y, in which the sequences A and C are not arranged as in the solution of SIMPLE (A, C, B). But then Y contains either the triples (a) $1 \cdot y_j \cdot c_k$ and $x \cdot y_i \cdot c_\ell$ or (b) $1 \cdot y_i \cdot c_k$ and $x \cdot y_j \cdot c_\ell$ where $i < j$ and $k < \ell$. In the following we show that in both cases we get another optimal solution Y' with $h(Y') < h(Y)$ by exchanging either c_k with c_ℓ or x with 1. The inequality $x \geqslant y_p$ for $p = 1,\dots,n$ implies that $x \cdot y_i \geqslant 1 \cdot y_j$ for all $i < j$. Thus we immediately get that $1 \cdot y_j \cdot c_k + x \cdot y_i \cdot c_\ell \geqslant 1 \cdot y_j \cdot c_\ell + x \cdot y_i \cdot c_k$. Also from $y_j \cdot c_\ell \geqslant y_i \cdot c_k$ we obtain that $1 \cdot y_i \cdot c_k + x \cdot y_j \cdot c_\ell \geqslant x \cdot y_i \cdot c_k + 1 \cdot y_j \cdot c_\ell$. Thus in both cases the exchange step yields another optimal solution Y' with $h(Y') < h(Y)$. Thus we reach after a finite number of exchanges an optimal solution with minimum h-value. But this means that the sequences A and C are arranged as in SIMPLE (A, C, B). Now we fix the order of the sequences A and C, their product yields a new sequence $D = (d_i)$ with $d_i := a_i c_{n+1-i}$. Applying Proposition 2.1 to the new sequence D and to B shows that the solution generated by SIMPLE (A, C, B) is optimal.

(ii) The proof can be done in a similar way to (i) by showing that SIMPLE (B, C, A) yields an optimal solution. □

Other possibilities of restricting our problem are summarized in the following.

Restriction 3. Sequences A and B contain together at most k distinct values, where k is some fixed integer $k \geqslant 1$.

Theorem 4.6. *The minimization problem for an input obeying Restriction* 3 *can be solved in* $O(n^{k^2+1} \log n)$ *time.*

Proof. Let $v_1,\dots,v_k$ denote the k values appearing in $A \cup B$, and let a_i and b_i, $1 \leqslant i \leqslant k$, count the occurrences of v_i in the sequences A and B, respectively. We

introduce k^2 integer variables x_{ij}, $1 \leqslant i, j \leqslant k$, fulfilling $0 \leqslant x_{ij} \leqslant a_i$ and $\sum_{m=1}^{k} x_{im} = a_i$. Intuitively, in an assignment the value of x_{ij} determines how many of the v_i in A are paired with v_j's in B. Observe that as soon as the values of all x_{ij} are fixed, we may apply Proposition 2.1 to compute the optimum assignment under these x_{ij}.

As $0 \leqslant x_{ij} \leqslant a_i \leqslant n$ holds, there are at most n^{k^2} different ways to assign values to the x_{ij}. We check all these possibilities and compute each time the optimum value according to Proposition 2.1. This yields the claimed time complexity. □

Restriction 4. The sequence A contains $(n - k)$ times the value x, where k is some fixed integer $k \geqslant 1$.

Theorem 4.7. *The minimization problem for an input obeying Restriction* 4 *can be solved in* $O(n^{2k+1} \log n)$ *time.*

Proof. Consider the terms in the optimum sum to which sequence A contributes a factor of x. By Proposition 2.1, the structure of the contributions of sequences B and C is uniquely determined.

Thus, we simply generate $O(n^{2k})$ potential candidates for the optimum solution in the following way. We choose for each of the k numbers in A that are not equal to x, two arbitrary partners from B and C. Clearly, this can be done in n^{2k} different ways. The remaining numbers in B and C are paired according to Proposition 2.1. By the above paragraph, one of these potential candidates leads to the optimum solution. □

5. Structural properties of cost coefficients

It turns out that many cost coefficients will never occur in an optimal assignment due to the structure of the problem. Those cost coefficients which may occur in an optimal solution are called *relevant*. The subsequent theorem specifies some *irrelevant* cost coefficients, which will never occur in an optimal solution. By fixing them to infinity known heuristics and exact solution methods can be improved.

Theorem 5.1. *Let A, B and C be sorted increasingly. Then there exists always an optimal assignment which does not contain any coefficient* $a_i b_j c_k$ *with either*

(i) $i + j + k < n + 2$ *or*

(ii) $i + j + k > 2n + 1$.

Proof. To see (i), let $a_i b_j c_k$ be a triple of indices in an optimal solution of the 3AP with $i + j + k < n + 2$. Then this assignment also contains a coefficient with $a_{i^\star} b_{j^\star} c_{k^\star}$ with $i < i^\star, j < j^\star$ and $k < k^\star$.

Table 1
Number of feasible solutions before and after deletion of irrelevant cost coefficients

n	Before	After deletion
2	4	3
3	36	17
4	576	151
5	14400	1899
⋮	⋮	⋮

Suppose, $a_{i^\star}b_{j^\star}c_{k^\star}$ with $i < i^\star, j < j^\star$ and $k < k^\star$ does not occur. But then for $n - i$ triples of indices in this assignment, namely

$$\{(i+1, x_1, y_1), (i+2, x_2, y_2), \ldots, (n, x_{n-i}, y_{n-i})\},$$

we must either have $x_r < j$ or $y_r < k \; \forall r = 1, \ldots, n-i$. On the other hand

$$|\{x_r : x_r < j\}| = j - 1 \quad \text{and} \quad |\{y_r : y_r < k\}| = k - 1.$$

So to fulfill the above requirement the three indices i, j and k have to satisfy $j - 1 + k - 1 \geqslant n - i$. But this leads to a contradiction. Therefore this solution also contains at least one triple $a_{i^\star}b_{j^\star}c_{k^\star}$ with $i < i^\star$, $j < j^\star$ and $k < k^\star$. But then, referring to Proposition 2.1

$$a_i b_j c_{k^\star} + a_{i^\star} b_{j^\star} c_k \leqslant a_i b_j c_k + a_{i^\star} b_{j^\star} c_{k^\star},$$

exchanging e.g. c_k and $c_{k^\star}$ leads to another optimal solution without cost coefficient $a_i b_j c_k$.

The proof of (ii) is done analogously to (i). □

The immediate consequence of Theorem 5.1 is that by fixing those cost coefficients to infinity which fulfill above inequalities the number of the remaining relevant cost coefficients can be reduced. Instead of n^3 only $n(n^2 - 1)/3$ relevant cost coefficients remain. Therefore also those feasible assignments containing at least one irrelevant cost coefficient need not to be taken into account any more. Table 1 shows the number of feasible solutions before and after "deleting" all irrelevant cost coefficients.

6. Computational results

Since the minimization problem is NP-hard, we investigate several heuristics to obtain "good" solutions.

The straightforward Greedy heuristic which selects in every step the smallest cost coefficient is not suited for our case. It is easy to see that it yields the solution

Table 2
Comparison of the average objective values of MAXREGRET and its improved version MAXREGRET_2

n	MAXREGRET ⌀ value	MAXREGRET_2 ⌀ value
4	583.99	621.79
6	954.14	946.27
8	1367.77	1289.20
10	1741.42	1569.90
12	2351.02	2071.26
14	2967.13	2630.59
16	3264.38	2788.21

$\sum_{i=1}^{n} a_i b_{\phi(i)} c_{\psi(i)}$ where (a_i), $(b_{\phi(i)})$ and $(c_{\psi(i)})$ are ordered increasingly. Thus the Greedy solution is an optimum solution of the maximization problem and thus the worst solution for the problem under investigation.

In the following we shall compare the heuristic MAXREGRET proposed by Balas and Saltzman [3] for the general 3AP with two special heuristics SIMPLE_3 and LSH tailored for the decomposable case.

The heuristic MAXREGRET works as follows: For each of the $3n$ possible two-dimensional submatrices contained in the cost coefficient cube the *regret* is calculated. The regret is defined as difference between the two smallest cost coefficients in that two-dimensional submatrix. Then MAXREGRET selects the minimal cost coefficient in that two-dimensional matrix for which the regret is maximal. After forcing this coefficient into the assignment the problem size is reduced by one. Again a new regret is calculated and another coefficient is selected until all n triples of a three-dimensional assignment are fixed.

Note that MAXREGRET can also choose irrelevant cost coefficients. To avoid this, we improve the quality of MAXREGRET by exploiting the special structure of the problem and therefore consider a modified version of MAXREGRET, say MAXREGRET_2. This heuristic works as follows: First all irrelevant coefficients described by Theorem 5.1 are set to infinity and then MAXREGRET is applied to the modified cost array. Table 2 shows the differences in the optimal objective values of applying MAXREGRET to the cost array with and without irrelevant cost coefficients, i.e. the differences between MAXREGRET and MAXREGRET_2.

To exploit the structure of the coefficients even more we propose two other heuristics. The heuristic SIMPLE_3 is based on the algorithm SIMPLE and works as follows: Given three sequences A, B and C then SIMPLE_3 executes SIMPLE (A,B,C), SIMPLE (B,C,A) and SIMPLE (C,A,B) – the three possibilities of applying Proposition 2.1 twice – and reports the best solution value of these three different constructed thee-dimensional assignments.

To illustrate SIMPLE_3 consider the following example: Let three sequences, say $A = (1,2,3,4)$, $B = (1,2,4,5)$ and $C = (2,3,3,6)$, be given. Then all three different

solutions generated by SIMPLE_3 are given by:

$$\begin{pmatrix} 1\cdot 5\cdot 3 \\ 2\cdot 4\cdot 2 \\ 3\cdot 2\cdot 3 \\ 4\cdot 1\cdot 6 \end{pmatrix}, \quad \begin{pmatrix} 1\cdot 5\cdot 6 \\ 2\cdot 4\cdot 3 \\ 3\cdot 1\cdot 3 \\ 4\cdot 2\cdot 2 \end{pmatrix}, \quad \begin{pmatrix} 4\cdot 1\cdot 6 \\ 3\cdot 2\cdot 3 \\ 1\cdot 4\cdot 3 \\ 2\cdot 5\cdot 2 \end{pmatrix}$$

with the objective values: 73, 79 and 74, SIMPLE_3 delivers 73.

A further improvement is obtained by the following local search heuristic, LSH (see also [15]), which again is based on the special structure of the problem and Proposition 2.1. It runs as follows:

Heuristic LSH

0. Start with arbitrary permutations ϕ and ψ.
1. Apply Proposition 2.1 and set

$$\tilde{\pi} := \operatorname{argmin}\left\{ \min_{\pi} \sum_{i=1}^{n} a_i \cdot w_{\pi(i)} \right\} \quad \text{with} \quad w_i := b_{\phi(i)} \cdot c_{\psi(i)}.$$

   Redefine $\phi := \tilde{\pi} \circ \phi$ and $\psi := \tilde{\pi} \circ \psi$.
2. Apply Proposition 2.1 and set

$$\tilde{\pi} := \operatorname{argmin}\left\{ \min_{\pi} \sum_{i=1}^{n} b_{\pi(\phi(i))} \cdot w_i \right\} \quad \text{with} \quad w_i := a_i \cdot c_{\psi(i)}.$$

   Redefine $\phi := \tilde{\pi} \circ \phi$.
3. Apply Proposition 2.1 and set

$$\tilde{\pi} := \operatorname{argmin}\left\{ \min_{\pi} \sum_{i=1}^{n} c_{\pi(\psi(i))} \cdot w_i \right\} \quad \text{with} \quad w_i := a_i \cdot b_{\phi(i)}.$$

   Redefine $\psi := \tilde{\pi} \circ \psi$.
4. Goto 1. as long as improvements are reached by Steps 1–3.

To illustrate heuristic LSH consider again the above example with $A = (1,2,3,4)$, $B = (1,2,4,5)$ and $C = (2,3,3,6)$. Let us start with $\phi = \langle 3,2,1,4\rangle$ and $\psi = \langle 4,1,2,3\rangle$. This solution has the objective value 82.

Performing Step 1 we determine $\tilde{\pi} = \langle 1,3,4,2\rangle$. Thus $\phi = \langle 4,3,1,2\rangle$ and $\psi = \langle 2,1,3,4\rangle$. The objective value decreases to 74.

Step 2 yields $\tilde{\pi} = \langle 2,1,3,4\rangle$ and therefore the new permutation $\phi = \langle 4,3,2,1\rangle$. The value of the objective function is 73.

Computing Step 3 leads to no improvement as well as Steps 1 and 2 do not change the current solution. So LSH stops with $\phi = \langle 4,3,2,1\rangle$ and $\psi = \langle 2,1,3,4\rangle$, the value 73 and the rearrangement of $A = (1,2,3,4)$, $B = (5,4,2,1)$ and $C = (3,2,3,6)$.

Additionally we consider the heuristic SIMPLE_LSH where the starting permutations in LSH are chosen from the result of SIMPLE_3.

Table 3
Comparison of LSH with SIMPLE_3, SIMPLE_LSH and the improved heuristic MAXREGRET; the cost coefficients a_i, b_j and c_k are uniformly drawn from [1, 10]

n	MAXREGRET_2		SIMPLE_3	LSH		SIMPLE_LSH	
	∅ value	∅ time	∅ value	∅ value	∅ time	∅ value	∅ time
4	621.79	0.007	468.17	443.69	0.003	443.70	0.007
6	946.27	0.036	661.97	634.46	0.011	634.20	0.009
8	1289.20	0.086	871.02	820.26	0.019	819.94	0.016
10	1569.90	0.199	1015.79	960.48	0.025	960.55	0.024
12	2071.26	0.394	1256.68	1188.28	0.031	1188.02	0.024
14	2630.59	0.697	1565.55	1469.19	0.033	1469.27	0.031
16	2788.21	1.162	1576.04	1476.80	0.041	1476.99	0.036

Table 4
Comparison of LSH with SIMPLE_3, SIMPLE_LSH and the improved heuristic MAXREGRET; the cost coefficients a_i, b_j and c_k are uniformly distributed in [1, 50].

n	MAXREGRET_2		SIMPLE_3	LSH		SIMPLE_LSH	
	∅ value	∅ time	∅ value	∅ value	∅ time	∅ value	∅ time
4	64428	0.007	50089	47443	0.008	47414	0.005
6	92177	0.033	62912	58295	0.014	58306	0.012
8	107096	0.092	75935	70581	0.016	70562	0.017
10	151016	0.203	98618	90151	0.030	90147	0.018
12	186680	0.406	120798	110330	0.033	110293	0.027
14	213976	0.728	136931	124753	0.041	124746	0.035
16	229851	1.200	142953	129064	0.053	129086	0.046

To test the heuristics we generated for each n between 4 and 16 one hundred different problems with integer cost coefficients a_i, b_j and c_k uniformly distributed in [1, 10] and [1, 50], respectively. All test runs were performed on a PC 386/387 with 20 MHz clock. In Tables 3 and 4 average running times in CPU-seconds and average values of the objective function of the considered heuristics are compared. Table 3 shows the results of problems with values in [1, 10] whereas Table 4 of those problems with values in [1, 50]. One can see the quite good behaviour of LSH and SIMPLE_LSH in comparison with the improved version of MAXREGRET and SIMPLE_3. (Since for SIMPLE_3 the running time is negligible, only the objective values are listed.)

7. Conclusion

In this paper a special case of an axial three-dimensional assignment problem was investigated, in which the cost coefficients d_{ijk} can be decomposed into the product of three values a_i, b_j and c_k. We have shown that the maximum version is easy to solve just by sorting the sequences a_i, b_i and c_i in increasing order, whereas the minimization problem remains NP-hard. So several special cases were considered and

general types of problems which can be solved in polynomial time are summarized in Theorems 4.5–4.7. Besides these a structural property of the cost coefficients which occur in an optimal solution is given by Theorem 5.1. With the help of this theorem one can improve the performance of heuristics as well as of exact solution methods. For the general minimization case the heuristics LSH and SIMPLE_LSH were presented which showed a quite good behaviour in comparison with other heuristics.

Acknowledgements

Rüdiger Rudolf acknowledges financial support by the Fonds zur Förderung der wissenschaftlichen Forschung, Project P8971-PHY. Gerhard J. Woeginger acknowledges the support by the Christian Doppler Laboratorium für Diskrete Optimierung.

We wish to thank two anonymous referees for their careful reading of a first version of this paper and in particular for contributing Theorem 4.5.

References

[1] A. Aggarwal and J.K. Park, Parallel searching in multidimensional monotone arrays, Research Report RC 14826, IBM T.J. Watson Research Center, Yorktown Heights, NY (1989), submitted to J. Algorithms. Parts of this paper appeared in: Notes on Searching in Multidimensional Monotone Arrays, Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (1988) 497–512.

[2] A. Aggarwal and J.K. Park, Sequential searching in multidimensional monotone arrays, Research Report RC 15128, IBM T.J. Watson Research Center, Yorktown Heights, NY (1989), submitted to J. Algorithms. Parts of this paper appeared in: Notes on Searching in Multidimensional Monotone Arrays, Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (1988) 497–512.

[3] E. Balas and M.J. Saltzman, An algorithm for the three-index assignment problem, Oper. Res. 39 (1991) 150–161.

[4] W. Bein, P. Brucker, J.K. Park and P.K. Pathak, A Monge property for the d–dimensional transportation problem, Discrete Appl., Math. 58 (1995) 97–109.

[5] M. Blum, R.W. Floyd, V.R. Pratt, R.L. Rivest and R.E. Tarjan, Time bounds for selection, J. Comput. System Sci. 7 (1972) 448–461.

[6] R.E. Burkard and K. Fröhlich, Some remarks on 3-dimensional assignment problems, Methods Oper. Res. 36 (1980) 31–36.

[7] R.E. Burkard and R. Rudolf, Computational investigations on 3-dimensional axial assignment problems, Belgian J. Oper. Res. Statist. Comput. Sci. 32 (1–2) (1993) 85–98.

[8] Y. Crama and F.C.R. Spieksma, Approximation algorithms for three-dimensional assignment problems with triangle inequalities, European J. Oper. Res. 60 (1992) 273–379.

[9] K. Fröhlich, Dreidimensionale Zuordnungsprobleme, Masters Thesis, Math. Institut, Universität Köln (1979).

[10] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness (Freeman, San Francisco, CA, 1979).

[11] P. Hansen and L. Kaufman, A primal-dual algorithm for the three-dimensional assignment problem, Cahiers CERO 15 (1973) 327–336.

[12] G.H. Hardy, J.E. Littlewood and G. Pòlya, Inequalities (Cambridge University Press, Cambridge, 1967) 260ff.

[13] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller and J.W. Thatcher, eds., Proceedings of the Complexity of Computer Computations (Plenum Press, New York, 1972) 85–104.
[14] W.P. Pierskalla, The multidimensional assignment problem, Oper. Res. 16 (1968) 422–431.
[15] R. Rudolf, Dreidimensionale axiale Zuordnungsprobleme, Masters Thesis, Technical University Graz, Graz (1991).
[16] M. Vlach, Branch and bound method for the three-index assignment problem, Ekonom.-Mat. Obzor 2 (1967) 181–191.