European Journal of Operational Research 60 (1992) 273–279
North-Holland

Theory and Methodology

Approximation algorithms for three-dimensional assignment problems with triangle inequalities

Yves Crama
Department of Quantitative Economics, Faculty of Economics, University of Limburg, P.O. Box 616, 6200 MD Maastricht, Netherlands

Frits C.R. Spieksma
Department of Mathematics, Faculty of General Sciences, University of Limburg, P.O. Box 616, 6200 MD Maastricht, Netherlands

Received March 1990; revised June 1990

Abstract: The three-dimensional assignment problem (3DA) is defined as follows. Given are three disjoint n-sets of points, and nonnegative costs associated with every triangle consisting of exactly one point from each set. The problem is to find a minimum-weight collection of n triangles covering each point exactly once. We consider the special cases of 3DA where a distance (verifying the triangle inequalities) is defined on the set of points, and the cost of a triangle is either the sum of the lengths of its sides (problem $T\Delta$) or the sum of the lengths of its two shortest sides (problem $S\Delta$). We prove that $T\Delta$ and $S\Delta$ are NP-hard. For both $T\Delta$ and $S\Delta$, we present $\frac{1}{2}$- and $\frac{1}{3}$-approximate algorithms, i.e. heuristics which always deliver a feasible solution whose cost is at most $\frac{3}{2}$, resp. $\frac{4}{3}$, of the optimal cost. Computational experiments indicate that the performance of these heuristics is excellent on randomly generated instances of $T\Delta$ and $S\Delta$.

Keywords: Integer programming; three-dimensional assignment; heuristics; computational analysis

1. Introduction

Consider the following classical formulation of the (*axial*) *three-dimensional assignment problem* (3DA) (see e.g. Balas and Saltzman, 1989). Given is a complete tripartite graph $K_{n,n,n} = (I \cup J \cup K, (I \times J) \cup (I \times K) \cup (J \times K))$, where I, J, K are disjoint sets of size n, and a cost c_{ijk} for each triangle $(i, j, k) \in I \times J \times K$. The problem 3DA is to find a subset A of n triangles, $A \subseteq I \times J \times K$, such that every element of $I \cup J \cup K$ occurs in exactly one triangle of A, and the total cost $c(A) = \sum_{(i,j,k) \in A} c_{ijk}$ is minimized. Some recent references to this problem are Balas and Saltzman (1989), Frieze (1974), Frieze and Yadegar (1981), Hansen and Kaufman (1973).

When one formulates 3DA in graph-theoretic terms, as we just did, it is natural to assume that the costs c_{ijk} are not completely arbitrary, but are rather defined in terms of costs attached to the edges of the graph. More precisely, we shall restrict our attention in this paper to the special cases of 3DA where each edge $(u, v) \in (I \times J) \cup (I \times K) \cup (J \times K)$ is assigned a nonnegative *length* d_{uv}, and where the cost of a triangle $(i, j, k) \in I \times J \times K$ is defined either by its total length t_{ijk}

$$t_{ijk} = d_{ij} + d_{ik} + d_{jk} \quad (1)$$

or by s_{ijk}, the sum of the lengths of its two shortest edges:

$$s_{ijk} = \min\{d_{ij} + d_{ik},\ d_{ij} + d_{jk},\ d_{ik} + d_{jk}\}. \quad (2)$$

We refer to the problem 3DA with cost coefficients $c_{ijk} = t_{ijk}$ or $c_{ijk} = s_{ijk}$, as problem T or S, respectively.

Instances of problem T arise in the scheduling of teaching practices at colleges of education (Frieze and Yadegar, 1981).

Either T or S can also be used to model a situation encountered in the production of printed circuit boards by numerically controlled machines featuring three placement heads (see Crama et al., 1990, for details).

In the latter application (which motivated the present study), the lengths d_{uv} represent travel times of the arm of the machine between locations u and v where electronic components are to be inserted. In particular, and even though the exact definition of these travel times may be quite intricate, the lengths d_{uv} define a distance, i.e., they satisfy the *triangle inequalities*

$$d_{uv} \leqslant d_{uw} + d_{vw} \quad \text{for all } u, v, w \in I \cup J \cup K. \quad (3)$$

In the remainder of this paper, we concentrate on problems $T\Delta$ and $S\Delta$, i.e., on the special cases of T and S for which the triangle inequalities (3) hold. We show in Section 2 that $T\Delta$ and $S\Delta$ are NP-hard. In Section 3, we describe some heuristics for $T\Delta$ and $S\Delta$, and establish tight bounds on their worst-case performance. The results of computational experiments with these heuristics are presented in Section 4.

2. Complexity of $T\Delta$ and $S\Delta$

The problem 3DA is well-known to be NP-hard, even when the costs c_{ijk} can only take two distinct values (see e.g. Garey and Johnson, 1979, for a proof). We show now that its special cases $T\Delta$ and $S\Delta$ remain NP-hard too.

Theorem 1. *Problem* $T\Delta$ *is* NP-*hard.*

Proof. We use the argument presented by Garey and Johnson (1979) to establish the NP-hardness of the problem Partition into Triangles. Consider an instance $\mathcal{I}$ of 3DA, defined by three sets I_0, J_0, K_0 of size n, and $c_{ijk} \in \{0, 1\}$ for all $(i, j, k) \in I_0 \times J_0 \times K_0$.

With $\mathcal{I}$, we associate an instance of $T\Delta$, as follows. Let $M = \{(i, j, k): c_{ijk} = 0\}$, $|M| = m$, and

$$I = I_0 \cup \{i_l(e): e \in M, l = 1, 2, 3\},$$
$$J = J_0 \cup \{j_l(e): e \in M, l = 1, 2, 3\},$$
$$K = K_0 \cup \{k_l(e): e \in M, l = 1, 2, 3\}$$

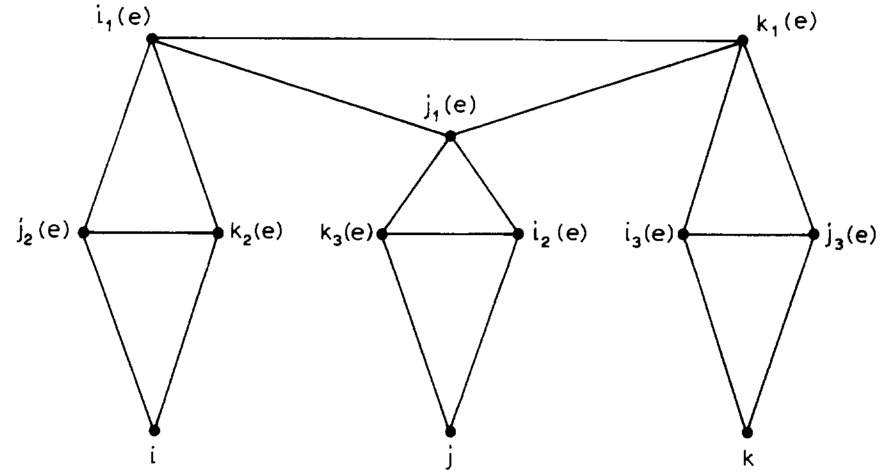

Figure 1

where $i_l(e)$, $j_l(e)$, $h_l(e)$ ($e \in M$, $l = 1, 2, 3$) are $9m$ new elements.

In order to conveniently define the lengths of the edges of the complete tripartite graph G on $I \cup J \cup K$, we first introduce m subgraphs of G. For each $e = (i, j, k) \in M$, $G(e)$ is the graph represented in Figure 1.

Now for each $(u, v) \in (I \times J) \cup (I \times K) \cup (J \times K)$, we let $d_{uv} = 1$ if (u, v) is an edge in some graph $G(e)$ ($e \in M$), and $d_{uv} = 2$ otherwise. Clearly, the triangle inequalities (3) are satisfied by this assignment, so that I, J, K and the lengths d_{uv} together define an instance $\mathcal{T}$ of $T\Delta$.

Observe that every feasible solution of $\mathcal{T}$ contains exactly $n + 3m$ triangles, each with cost at least 3. We claim that $\mathcal{T}$ has an optimal solution with value $3n + 9m$ if and only if $\mathcal{I}$ has a solution with value 0. We leave details to the reader (see Garey and Johnson, 1979, pp. 68–69). □

Theorem 2. *Problem $S\Delta$ is* NP-*hard.*

Proof. The proof is similar to the previous one: simply delete from each subgraph $G(e)$ the edges $(i_1(e), j_2(e))$, $(i, j_2(e))$, $(j_1(e), k_3(e))$, $(j, k_3(e))$, $(i_3(e), k_1(e))$, $(i_3(e), k)$ for all $e \in M$. The resulting instance of $S\Delta$ has an optimal solution with value $2n + 6m$ if and only if $\mathcal{I}$ has a solution with value 0. □

3. Approximation algorithms

In this section, we present approximation algorithms for $T\Delta$ and $S\Delta$. First, we recall a definition from Papadimitriou and Steiglitz (1982) (see also Garey and Johnson, 1979). Consider a minimization problem P, and an algorithm H which, given any instance $\mathcal{I}$ of P, returns a feasible solution $H(\mathcal{I})$ of $\mathcal{I}$. Denote by $c(H(\mathcal{I}))$ the value of this heuristic solution, and by $\mathrm{OPT}(\mathcal{I})$ the value of an optimal solution of $\mathcal{I}$. Then, H is called an *ϵ-approximate algorithm* for P, where ϵ is a nonnegative constant, if

$$c(H(\mathcal{I})) \le (1 + \epsilon)\mathrm{OPT}(\mathcal{I})$$

for all instances $\mathcal{I}$ of P.

We will show that $\frac{1}{3}$-approximate polynomial-time algorithms exist for problems $T\Delta$ and $S\Delta$. As indicated by our next theorem, the triangle inequalities (3) play an instrumental role in the proof of such results.

Theorem 3. *Unless* P = NP, *there is no ϵ-approximate polynomial algorithm for problems T and S for any $\epsilon \ge 0$.*

Proof. We establish the statement for problem T (the other case being similar). Assume that there is an ϵ-approximate algorithm for T, say H. As in the proof of Theorem 1, consider an instance $\mathcal{I}$ of 3DA with $c_{ijk} \in \{0, 1\}$ for all (i, j, k), the corresponding sets I, J, K, and the subgraphs $G(e)$ ($e \in M$). For $(u, v) \in (I \times J) \cup (I \times K) \cup (J \times K)$, let $d_{uv} = 1$ if (u, v) is an edge of $G(e)$ ($e \in M$), and $d_{uv} = (3n + 9m)\epsilon + 2$ otherwise. This defines an instance $\mathcal{T}$ of problem T, with the property that $\mathcal{T}$ has an optimal solution with value $3n + 9m$ if and only if $\mathcal{I}$ has a solution with value 0.

Now, it is easy to see that the ϵ-approximate algorithm H always returns a solution of $\mathcal{T}$ with value $3n + 9m$, if there is one (because the second best solution has value at least $(1 + \epsilon)(3n + 9m) + 1$). Hence, unless P = NP, H cannot be a polynomial-time algorithm. □

We describe now informally a polynomial-time heuristic H_{IJ} for problems T and S. This heuristic was proposed in Crama et al. (1990). The input to H_{IJ} is the set of edge-lengths d_{uv} where $(u, v) \in (I \times J) \cup (I \times K) \cup (J \times K)$ and $|I| = |J| = |K| = n$. The heuristic proceeds in two phases, first matching the elements of I and J, and next assigning the elements of K to the pairs thus formed (Frieze and Yadegar (1981) propose a similar heuristic for the general 3DA problem). More precisely

Phase 1. Find an optimal solution x^* of (P1)

$$\begin{aligned} \min \quad & \sum_{i \in I} \sum_{j \in J} d_{ij} x_{ij} \\ \text{s.t.} \quad & \sum_{i \in I} x_{ij} = 1, \quad j \in J, \\ & \sum_{j \in J} x_{ij} = 1, \quad i \in I, \\ & x_{ij} \in \{0, 1\}, \quad i \in I,\ j \in J. \end{aligned} \tag{P1}$$

Let $M = \{(i, j): x^*_{ij} = 1\}$.

Phase 2. Find an optimal solution y^* of $(P2)$

$$\begin{aligned} \min \quad & \sum_{(i,j)\in M} \sum_{k\in K} c_{ijk} y_{ijk} \\ \text{s.t.} \quad & \sum_{(i,j)\in M} y_{ijk} = 1, \quad k \in K, \\ & \sum_{k\in K} y_{ijk} = 1, \quad (i, j) \in M, \\ & y_{ijk} \in \{0, 1\}, \quad (i, j) \in M,\ k \in K, \end{aligned} \tag{P2}$$

where $c_{ijk} = t_{ijk}$ (resp. $c_{ijk} = s_{ijk}$) if the problem to be solved is an instance of T (resp. S).

The feasible solution of T (or S) returned by the heuristic H_{IJ} is $A = \{(i, j, k){:}\ y^*_{ijk} = 1\}$, and its cost is denoted by c_{IJ}.

Notice that both (P1) and (P2) are instances of the classical (two-dimensional) assignment problem, or weighted bipartite matching problem, and hence can be solved in $O(n^3)$ operations (Papadimitriou and Steiglitz, 1982). It follows that H_{IJ} also runs in time $O(n^3)$.

We leave it as an easy exercise to verify that, as suggested by Theorem 3, H_{IJ} is not an ϵ-approximate algorithm for either T or S, for any $\epsilon \geq 0$.

On the other hand, when the lengths d_{uv} satisfy the triangle inequalities, we get

Theorem 4. *H_{IJ} is a $\frac{1}{2}$-approximate algorithm for problem $T\Delta$. Moreover, there exist arbitrarily large instances $\mathcal{T}$ of $T\Delta$ such that $c_{IJ} = \frac{3}{2}\mathrm{OPT}(\mathcal{T})$.*

Proof. Let $\mathcal{T}$ be an instance of $T\Delta$. Let M be the matching of $I \cup J$ found by the first phase of H_{IJ}, and A be the assignment returned by H_{IJ}.

Consider now an optimal solution of $\mathcal{T}$, say F. With F, we associate another feasible solution $B = \{(i, j, k){:}\ (i, j) \in M \text{ and } (u, j, k) \in F \text{ for some } u \in I\}$.

We obtain successively

$$c_{IJ} = \sum_{(i,j,k)\in A} t_{ijk}$$

$$\leq \sum_{(i,j,k)\in B} t_{ijk} \tag{4}$$

$$= \sum_{(i,j,k)\in B} (d_{ij} + d_{ik} + d_{jk}) \tag{5}$$

$$\leq 2 \sum_{(i,j,k)\in B} (d_{ij} + d_{jk}) \tag{6}$$

$$= 2 \sum_{(i,j,k)\in A} d_{ij} + 2 \sum_{(i,j,k)\in F} d_{jk} \tag{7}$$

$$\leq 2 \sum_{(i,j,k)\in F} (d_{ij} + d_{jk}) \tag{8}$$

((4) holds because A is optimal for (P2), (5) is by definition of t_{ijk}, (6) uses the triangle inequality, (7) is by definition of B, and (8) follows from the optimality of M for (P1)).

By symmetry with (8), we can also derive

$$c_{IJ} \leq 2 \sum_{(i,j,k)\in F} (d_{ij} + d_{ik}). \tag{9}$$

Now, (8) and (9) together entail:

$$c_{IJ} \leq \sum_{(i,j,k)\in F} (2d_{ij} + d_{ik} + d_{jk})$$

$$= \sum_{(i,j,k)\in F} \left(\tfrac{3}{2}d_{ij} + \tfrac{1}{2}d_{ij} + d_{ik} + d_{jk}\right) \tag{10}$$

and, using the triangle inequalities to bound $\frac{1}{2}d_{ij}$

$$c_{IJ} \leq \tfrac{3}{2} \sum_{(i,j,k)\in F} t_{ijk} = \tfrac{3}{2}\mathrm{OPT}(\mathcal{T}). \tag{11}$$

To see that equality may hold in (11), consider first the graph G represented in Figure 2.

Also indicated in Figure 2 are the costs $c_{uv} \in \{1, 2\}$ of the edges of G.

Now, we define an instance $\mathcal{T}$ of $T\Delta$ as follows. We let $I = \{i_1, i_2\}$, $J = \{j_1, j_2\}$, $K = \{k_1, k_2\}$. For $(u, v) \in (I \times J) \cup (I \times K) \cup (J \times K)$, d_{uv} is the length of a shortest path from u to v in G with respect to the costs c_{uv}.

It is easy to see that an optimal solution for this instance is $F = \{(i_1, j_2, k_1), (i_2, j_1, k_2)\}$, with cost $\mathrm{OPT}(\mathcal{T}) = 8$.

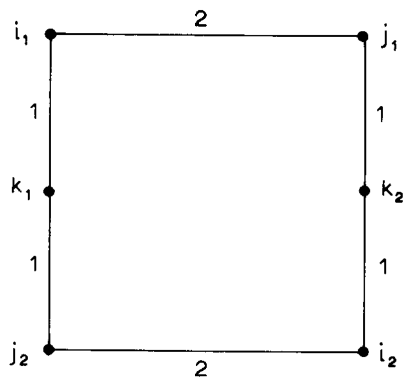

Figure 2

But H_{IJ} can pick (in Phase 1) $M = \{(i_1, j_1), (i_2, j_2)\}$, and next (in Phase 2) $A = \{(i_1, j_1, k_1), (i_2, j_2, k_2)\}$, with cost $c_{IJ} = 12 = \frac{4}{3}\text{OPT}(\mathscr{T})$. Arbitrarily large instances of $T\Delta$ can be obtained by taking several copies of G, with very large distances between points in different copies. □

The previous result also holds mutatis mutandis for problem $S\Delta$.

Theorem 5. *H_{IJ} is a $\frac{1}{2}$-approximate algorithm for problem $S\Delta$. Moreover, there exist arbitrarily large instances $\mathscr{S}$ of $S\Delta$ such that $c_{IJ} = \frac{3}{2}\text{OPT}(\mathscr{S})$.*

Proof. Let $\mathscr{S}$ be an instance of $S\Delta$. Define M, A, F and B in the same way as for the proof of Theorem 4. We derive the following inequalities:

$$c_{IJ} = \sum_{(i,j,k)\in A} s_{ijk}$$

$$\leq \sum_{(i,j,k)\in B} s_{ijk} \tag{12}$$

$$\leq \sum_{(i,j,k)\in A} d_{ij} + \sum_{(i,j,k)\in F} d_{jk} \tag{13}$$

$$\leq \sum_{(i,j,k)\in F} (d_{ij} + d_{jk}) \tag{14}$$

((12) holds because A is optimal for (P2), (13) is by definition of B and of s_{ijk}, (14) follows from the optimality of M for (P1)).

By symmetry with (14), the following inequality is also valid:

$$c_{IJ} \leq \sum_{(i,j,k)\in F} (d_{ij} + d_{ik}). \tag{15}$$

Using the triangle inequalities, one easily checks:

$$2d_{ij} + d_{ik} + d_{jk} \leq 3s_{ijk} \quad \text{for all } i, j, k. \tag{16}$$

Hence, (14), (15) and (16) together imply:

$$c_{IJ} \leq \tfrac{3}{2} \sum_{(i,j,k)\in F} s_{ijk} = \tfrac{3}{2}\text{OPT}(\mathscr{S}). \tag{17}$$

The example presented in the proof of Theorem 4 also achieves equality in (17), and can be used to build arbitrarily large instances. □

Of course, one can define in a natural way two more $\frac{1}{2}$-approximate algorithms for problems $T\Delta$ and $S\Delta$, namely the heuristics H_{IK} and H_{JK} obtained by permuting the roles of I, J and K in the description of H_{IJ}. We denote by c_{IK} and c_{JK} the values of the solutions delivered by H_{IK} and H_{JK}, respectively.

Consider now the heuristic H, which consists in applying all three heuristics H_{IJ}, H_{IK} and H_{JK} to the given instance of $T\Delta$ or $S\Delta$, and in retaining the best feasible solution thus produced. We denote by γ the value of the solution returned by H: $\gamma = \min\{c_{IJ}, c_{IK}, c_{JK}\}$.

Clearly, H can again be implemented to run in time $O(n^3)$, and H is a $\frac{1}{2}$-approximate algorithm for $T\Delta$ and $S\Delta$. But even more is true.

Theorem 6. *H is a $\frac{1}{3}$-approximate algorithm for problem $T\Delta$. Moreover, there exist arbitrarily large instances $\mathscr{T}$ of $T\Delta$ such that $\gamma = \frac{4}{3}OPT(\mathscr{T})$.*

Proof. Let $\mathscr{T}$ be an instance of $T\Delta$, and F an optimal solution of $\mathscr{T}$. As in the proof of Theorem 4, we obtain inequalities (8), (9), as well as the symmetric inequality

$$c_{IK} \leq 2 \sum_{(i,j,k)\in F} (d_{ik} + d_{jk}). \tag{18}$$

Summing up (8), (9) and (18) yields

$$3\gamma \leq 2c_{IJ} + c_{IK} \leq 4 \sum_{(i,j,k)\in F} t_{ijk} = 4\ \text{OPT}(\mathscr{T}), \tag{19}$$

which proves that H is a $\frac{1}{3}$-approximate algorithm.

Equality in (19) is achieved by the instance $\mathscr{T}$ depicted in Figure 3. Here, $I = \{i_1, i_2, i_3\}$, $J = \{j_1, j_2, j_3\}$, $K = \{k_1, k_2, k_3\}$. The lengths d_{uv} are indicated next to the edges of the 'pyramid', with

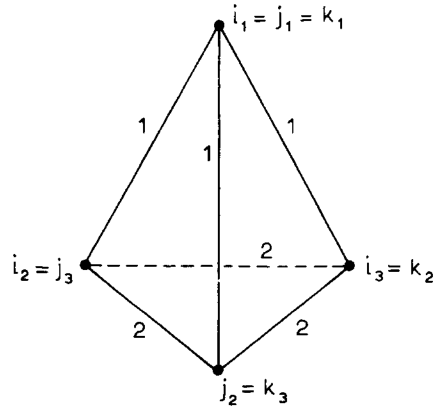

Figure 3

$d_{uv} = 0$ if $u = v$. It is easy to see that $\mathscr{T}$ is an instance of $T\Delta$. Moreover, because $\mathscr{T}$ is symmetric on I, J and K, we can assume that $\gamma = c_{IJ} = c_{IK} = c_{JK}$.

An optimal solution of $\mathscr{T}$ is given by $F = \{(i_1, j_2, k_3), (i_2, j_3, k_1), (i_3, j_1, k_2)\}$ with $\text{OPT}(\mathscr{T}) = 6$. But H_{IJ} can return a solution with cost $c_{IJ} = 8$, by picking $M = \{(i_1, j_1), (i_2, j_3), (i_3, j_2)\}$ in the first phase, and $A = \{(i_1, j_1, k_1), (i_2, j_3, k_2), (i_3, j_2, k_3)\}$ in the second phase. □

Notice that we actually proved a little bit more than announced by the statement of Theorem 6. Indeed, inequality (19) shows that the minimum of *any two* of the bounds c_{IJ}, c_{IK} and c_{JK} is already bounded by $\frac{4}{3}\text{OPT}(\mathscr{T})$. On the other hand, one can exhibit examples for which $c_{IJ} = c_{IK} = \frac{4}{3}\text{OPT}(\mathscr{T})$, and $c_{JK} = \text{OPT}(\mathscr{T})$. Thus, heuristic H is in general better than the strategy which consists in computing only two of the bounds c_{IJ}, c_{IK}, c_{JK}, and retaining the best one.

The same remarks apply to our next result:

Theorem 7. *H is a $\frac{1}{3}$-approximate algorithm for problem $S\Delta$. Moreover, there exist arbitrarily large instances $\mathscr{S}$ of $S\Delta$ such that that $\gamma = \frac{4}{3}\text{OPT}(\mathscr{S})$.*

Proof. Let $\mathscr{S}$ be an instance of $S\Delta$, and F be an optimal solution of $\mathscr{S}$. Summing up inequalities (14), (15) and

$$c_{IK} \le \sum_{(i,j,k)\in F} (d_{ik} + d_{jk}),$$

we get

$$3\gamma \le 2c_{IJ} + c_{IK} \le 2 \sum_{(i,j,k)\in F} (d_{ij} + d_{ik} + d_{jk}). \tag{20}$$

Using the triangle inequalities to bound the right-hand side of (20) yields:

$$3\gamma \le 4 \sum_{(i,j,k)\in F} s_{ijk} = 4\ \text{OPT}(\mathscr{S}). \tag{21}$$

A worst-case instance $\mathscr{S}$ is represented in Figure 4. All edges of this prism have length 1, and the distances are Euclidean.

The optimal solution $\{(i_1, j_2, k_3), (i_2, j_3, k_1), (i_3, j_1, k_2)\}$ has cost $\text{OPT}(\mathscr{S}) = 3$. The heuristic H_{IJ} may return $M = \{(i_1, j_1), (i_2, j_3), (i_3, j_2)\}$ in

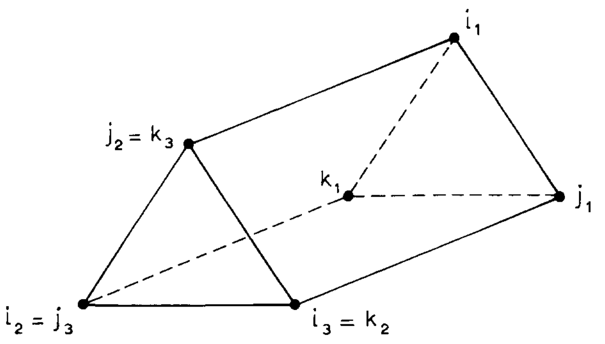

Figure 4

Phase 1, and $A = \{(i_1, j_1, k_1), (i_2, j_3, k_2), (i_3, j_2, k_3)\}$ in Phase 2, for a total cost $c_{IJ} = 4$. Hence, by symmetry, $\gamma = 4$ is possible. □

4. Computational results

Of course, the quality of a heuristic cannot only be judged by its worst-case performance. Very often, it is the case that this worst-case performance is determined by pathological instances of the problem. With this in mind, we conducted some numerical experiments to better assess the quality of the various approximation algorithms discussed in Section 3.

For fixed $n = |I| = |J| = |K|$, we considered random problems of three different types.

Type I. The elements of $I \cup J \cup K$ are generated random, uniformly in the square $[0, 1] \times [0, 1]$. For each pair of points (u, v), d_{uv} is the Euclidean distance from u to v (we also used for d_{uv} the Manhattan distance from u to v, with results similar to those displayed below).

Instances of Type I form a 'natural' class of random instances for problems $T\Delta$ or $S\Delta$. But, due to their high degree of uniformity, one may expect these instances to be easy to solve for most heuristics when n grows large. The next two types of instances are meant to be more 'irregular', and hence more difficult to solve.

Type II. The elements of I are generated to uniformly in $[0, \frac{1}{3}] \times [0, 1]$, those of J in $[\frac{1}{3}, 1] \times [0, \frac{1}{2}]$, those of K in $[\frac{1}{3}, 1] \times [\frac{1}{2}, 1]$. The distances d_{uv} are Euclidean.

Type III. We fix a parameter $p \in [0, 1]$. Then, for each pair (u, v), we let $d(u, v) = 1$ with probability p, and $d(u, v) = 2$ with probability $1 - p$.

Table 1
Problem $T\Delta$

Type	n	c_{IJ}	c_{IK}	c_{JK}	γ	LB	γ/LB
I	33	16.18	16.36	16.55	16.18	16.07	1.007
	33	14.16	14.16	14.11	14.11	13.95	1.011
	33	16.09	16.33	16.32	16.09	16.04	1.003
	66	26.92	26.87	26.68	26.68	26.54	1.005
	66	25.00	24.81	24.69	24.69	24.33	1.015
	66	28.13	27.75	27.91	27.75	27.48	1.010
II	33	48.83	48.61	48.75	48.61	47.72	1.019
	33	51.72	51.42	51.49	51.42	50.35	1.021
	33	43.52	43.83	44.01	43.52	42.60	1.022
	66	98.09	97.80	99.15	97.80	96.33	1.015
	66	91.47	91.60	91.42	91.42	88.31	1.035
	66	99.39	98.88	99.57	98.88	96.70	1.023
III	33	140	135	136	135	133	1.015
	33	141	137	139	137	130	1.054
	33	135	136	137	135	130	1.038
	66	295	293	296	293	283	1.035
	66	294	298	294	294	281	1.046
	66	295	296	293	293	280	1.046

The value of p was empirically adjusted so as to produce rather difficult problem instances for our heuristics.

For each problem type, we report in Tables 1 and 2 on the solution of three instances with $n = 33$ and three instances with $n = 66$ (more instances were actually tested, but the results displayed here are representative). The problems of Type III were generated with $p = \frac{1}{17}$ for $n = 33$ and $p = \frac{1}{50}$ for $n = 66$. Table 1 deals with problem $T\Delta$ and Table 2 with problem $S\Delta$.

Table 2
Problem $S\Delta$

Type	n	c_{IJ}	c_{IK}	c_{JK}	γ	LB	γ/LB
I	33	8.57	8.69	8.64	8.57	8.45	1.014
	33	7.61	7.54	7.53	7.53	7.39	1.019
	33	8.43	8.55	8.58	8.43	8.37	1.007
	66	14.23	14.31	14.09	14.09	13.90	1.014
	66	13.38	13.13	13.13	13.13	12.84	1.023
	66	14.70	14.53	14.46	14.46	14.20	1.018
II	33	26.54	26.65	27.32	26.54	25.96	1.022
	33	28.62	28.73	28.98	28.62	27.81	1.029
	33	23.78	23.79	24.21	23.78	23.11	1.029
	66	53.86	54.05	55.90	53.86	53.12	1.014
	66	49.29	49.47	50.43	49.29	47.88	1.029
	66	54.70	54.84	56.19	54.70	53.52	1.022
III	33	75	71	71	71	69	1.029
	33	75	72	73	72	67	1.075
	33	71	72	72	71	67	1.060
	66	163	161	165	161	151	1.066
	66	163	167	164	163	152	1.072
	66	163	165	161	161	147	1.095

For the sake of comparison, we also give in Tables 1 and 2 a lower-bound LB on the optimal value of each instance, as well as the value of the ratio $\gamma/(\mathrm{LB})$. The bound LB was computed using a Lagrangean relaxation scheme and subgradient optimization, as proposed by Frieze and Yadegar (1981).

The results exhibited in these tables indicate that, from a practical viewpoint, the heuristics presented in Section 3 perform quite satisfactorily. In particular, heuristic H solved all randomly generated instances within 10% of optimality, and often came within 3% of the optimal value (or, more precisely, of the lower-bound LB).

Acknowledgements

The authors wish to thank Koos Vrieze for his insightful suggestions, which led to the discovery of the worst-case examples presented in the proofs of Theorems 6 and 7, and Hans-Jürgen Bandelt for his comments on the paper.

References

Balas, E., and Saltzman, M.J. (1989), "Facets of the three-index assignment polytope", *Discrete Applied Mathematics* 23, 201–229.

Crama, Y., Kolen, A.W.J., Oerlemans, A.G., and Spieksma, F.C.R. (1990), "Throughput rate optimization in the automated assembly of printed circuit boards", *Annals of Operation Research* 26 (1990) 455–480.

Frieze, A.M. (1974), "A bilinear programming formulation of the 3-dimensional assignment problem", *Mathematical Programming* 7, 376–379.

Frieze, A.M., and Yadegar, J. (1981), "An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice", *Journal of the Operational Research Society* 32, 989–995.

Garey, M.R., and Johnson, D.S. (1979), *Computers and Intractability: A Guide to the Theory of NP-completeness*, W.H. Freeman, New York.

Hansen, P., and Kaufman, L. (1973), "A primal-dual algorithm for the three-dimensional assignment problem", *Cahiers du Centre d'Etudes de Recherche Opérationnelle* 15, 327–336.

Papadimitriou, C.H., and Steiglitz, K. (1982), *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ.