

Received November 21, 2020, accepted December 13, 2020, date of publication December 16, 2020, date of current version December 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3045288

Inverse Maximum Capacity Path Problems Under Sum-Type and Max-Type Distances and Their Practical Application to Transportation Networks

ADRIAN M. DEACONU¹ AND JAVAD TAYYEBI²

¹Department of Mathematics and Computer Science, University Transilvania, 500036 Braşov, Romania

²Department of Industrial Engineering, Birjand University of Technology, Birjand 9719866981, Iran

Corresponding author: Adrian M. Deaconu (a.deaconu@unitbv.ro)

This work was supported by the University Transilvania, Brasov.

ABSTRACT The maximum capacity path problem is to find a path connecting two given nodes in a network such that the minimum arc capacity on this path is maximized. The inverse maximum capacity path problem (IMCP) is to modify the capacities of the arcs as little as possible so that a given path becomes maximum capacity path in the modified network. Two cases of IMCP are considered: the capacity of the given path is preserved or not. IMCP is studied and solved both, under any sum-type (e.g., weighted l_k norms and sum-type Hamming distance) and max-type distance (e.g., weighted l_∞ norm or bottleneck Hamming distance). The obtained algorithms for IMCP are applied to solve a real road transportation network optimization problem.

INDEX TERMS Maximum capacity path, inverse optimization, minimum cut, Hamming distance.

I. INTRODUCTION

The maximum capacity path problem (MCP) is an earliest combinatorial optimization problem. It is to find a path connecting two given nodes in a network, such that the minimum arc capacity on this path is maximized. There are many practical applications of MCP. For instance, in a network that represents connections between routers in the Internet, the capacity of an arc represents the bandwidth of the corresponding connection between two routers, the maximum capacity path problem is to find the path between two Internet nodes (source and sink) that has the maximum possible bandwidth [1]. Besides this well-known network routing problem, MCP is also an important component of the Schulze method for deciding the winner of a multiway election [2]. It is applied to digital compositing [3], metabolic pathway analysis [4], the computation of maximum flow [5], etc. Most shortest path algorithms can be adapted to compute the maximum capacity path, by modifying them to use the bottleneck distance, instead of path length. There is also a very fast algorithm proposed in [6] to solve MCP in linear time.

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

Inverse optimization is a relatively new research domain and it has been intensively studied [7]–[13]. Many papers have recently been published in this domain and are still being published nowadays. Inverse problems have lately found a lot of applications in modern areas, such as mathematical biology, materials science, remote sensing, medical imaging, seismology, geophysics, oceanography, mathematical finance, etc. An inverse network optimization problem consists of modifying some parameters of a network, such as capacities or costs, so that a given feasible solution of the direct optimization problem becomes an optimal solution and the distance between the initial vector and the modified vector of parameters is minimized. Different norms, such as l_1 , l_∞ , l_2 or even l_k , or Hamming distances are considered to measure this distance.

The inverse maximum capacity path problem (IMCP) is to modify the capacities of the arcs as little as possible so that a given path becomes maximum capacity path in the modified network. IMCP was for the first time studied under l_1 norm in [14]. In this paper, a $O(mQ)$ time algorithm was obtained, where m is the number of arcs and Q is the complexity of finding the minimum weight cut set. Later, IMCP was studied and solved under weighted l_1 and l_∞ norms [15]. Similar time complexity under weighted l_1 norm

was obtained in [15]. In the case of weighted l_∞ norm, a strongly polynomial algorithm was obtained in [15]. In [16], IMCP under weighted l_1 norm and sum-type Hamming distance was reduced to $O(m)$ minimum cut problems, while IMCP under bottleneck-Hamming distance was reduced to $O(\log(m))$ cut feasibility problems.

In previous papers, IMCP was only studied under very particular distances: weighted l_1 , weighted l_∞ , and Hamming distances (Please see references cited in the paper). Our paper considers IMCP in the most general case: under any sum-type or max-type distances. Specially, IMCP under weighted l_k norms ($k > 0$) is also a particular case of our work and it was not studied before because this case is much more complicated than the others. In general, there are very few papers concerning inverse problems under l_k norms in the literature [17], [18]. For the first time in literature, our paper focuses on a special case of IMCP in which the capacity of the given path is preserved. This case has two major advantages:

1) It permits some real-world applications. For instance in Section VII, we showed how a transportation optimization problem can be modeled and solved using this newly introduced case.

2) It can be solved by polynomial-time algorithms whose worst-case complexity is better than the algorithms of classical IMCPs (see Table 1).

Let us state an application of IMCP in transportation networks. A convenient criterion used in selecting routes is the distance between public rest areas containing gas pumps, restaurants, and service centers. Due to the safety and comfort of trip, a driver may choose a route, which its public rest areas are near together. Therefore, we can assign any route to a distance, which is the maximum distance of rest areas within the route. Now suppose that a path P is so crowded, and we would like to shunt traffic flow toward parallel paths. To achieve this goal, we may vacate some rest centers to maximize the distance of P , and on the other hand, we can construct new rest centers and equip available rest areas to minimize the distance in other parallel paths. This is a typical example of IMCPs. It is remarkable that this application has a special characteristic: it does not seem to be reasonable that we vacate some rest centers in P . This motivates us to think the problem in the special case that the capacity of P is preserved. For this problem, there is a cost that must be minimized. The total cost of modification or the maximum cost of modification on a portion of road could be considered for minimization. Also, the cost per unit of modification or the cost calculated per each portion of road is involved. Depending on the chosen objective minimization, different IMCP problem is obtained. This is the reason why IMCP is studied in this paper under different sum-type and max-type distances.

This paper is organized as follows. In Section 2, MCP is briefly introduced. In the next four sections the inverse counterpart of MCP (IMCP) is studied in two cases: the capacity of the given path is preserved or not. In each of these cases we show how IMCP can be solved under both, sum-type

and max-type distances. In Chapter 7, a road transportation network optimization in a region of Iran is performed using an algorithm for special case of IMCP. Conclusions are drawn in the end of the paper.

II. MAXIMUM CAPACITY PATH PROBLEM

Let $G = (N, A, c, s, t)$ be a directed and connected network, where N is the set of nodes, A is the set of arcs which is a subset of $N \times N$ (an arc $a = (i, j)$ starts from node i and terminates at node j), $c : A \rightarrow R_+$ is the capacity function, s and t are two special nodes of N , s is called the source node and t is referred to as the sink node. We denote by n the number of nodes and by m the number of arcs, i.e.,

$$n = |N| \quad \text{and} \quad m = |A|. \tag{1}$$

A path P from a node $u \in N$ to a node $v \in N$ in the network G is given by a sequence of nodes: $P = (u = i_1, i_2, \dots, i_l = v)$, where $l \geq 1$, and $(i_k, i_{k+1}) \in A, k = 1, 2, \dots, l - 1$. For simplifying, hereafter, a path from s to t is called an $s - t$ path.

The capacity of an $s - t$ path P is denoted by $c(P)$ and is given by the minimum of its capacities, that is,

$$c(P) = \min_{a \in P} c(a). \tag{2}$$

The maximum capacity path problem (MCP) in the network G is to find an $s - t$ path \tilde{P} having the maximum capacity among all $s - t$ paths:

$$c(\tilde{P}) = \max \{c(P) \mid P \text{ is an } s - t \text{ path}\}. \tag{3}$$

III. INVERSE MAX. CAPACITY PATH PROBLEM

In this section, we introduce the inverse version of maximum capacity path problem, and we provide some initial results.

Let \tilde{P} be a given $s - t$ path in the network $G = (N, A, c, s, t)$. The inverse maximum capacity path problem (IMCP) is to modify the arc capacities as little as possible so that \tilde{P} becomes a maximum capacity path in the modified network.

First, we study the special case that the capacity of the given path must be preserved. In this case, in order to solve IMCP, it is no need to increase any arc capacity. So, IMCP is to find a new capacity vector $\bar{c} : A \rightarrow R_+$ as a solution of the following optimization problem:

$$\left\{ \begin{array}{l} \min \{dist(\bar{c}, c)\} \\ \bar{c}(\tilde{P}) = c(\tilde{P}) \\ \bar{c}(a) \geq c(a) - b(a), a \in A \\ \tilde{P} \text{ is a maximum capacity path in } \bar{G} = (N, A, \bar{c}, s, t) \end{array} \right. \tag{4}$$

In (4), $b(a)$ is a given upper bound on the modification of the capacity of a (namely, the capacity of a cannot be decreased more than $b(a)$ units), where $c(a) \geq b(a) \geq 0$ and $dist$ is a distance function to measure the distance between \bar{c} and c .

Due to the existence of bound constraints in (4), we construct the following set of arcs:

$$\tilde{A} = \{a \in A \mid c(a) - b(a) > c(\tilde{P})\}. \tag{5}$$

The following result justifies the reason of constructing \tilde{A} .

Theorem 1: IMCP is feasible if and only if there is no $s - t$ path in the directed graph $\tilde{G} = (N, \tilde{A})$.

Proof: To prove the necessity part, we suppose that IMCP has a feasible solution $\bar{c} : A \rightarrow R_+$. By assumption, the modified capacity \bar{c} of each arc is not greater than the initial capacity c .

We suppose that there is a path \tilde{P} in \tilde{G} from s to t . In $\tilde{G} = (N, \tilde{A}, \bar{c}, s, t)$ we have:

$$\begin{aligned} \bar{c}(\tilde{P}) &= \min \{ \bar{c}(a) \mid a \in \tilde{P} \} \\ &\geq \min \{ c(a) - b(a) \mid a \in \tilde{P} \} > c(\tilde{P}) \geq \bar{c}(\tilde{P}) \end{aligned} \quad (6)$$

From (6) it is clear that \tilde{P} is not a maximum capacity path in \tilde{G} . This is a contradiction with the last condition from (4).

Now, to prove the sufficiency, we suppose that there is no $s - t$ path in the directed graph $\tilde{G} = (N, \tilde{A})$. We construct $c' : A \rightarrow R_+$ as follows:

$$c'(a) = \begin{cases} c(a), & \text{if } a \in \tilde{A}, \\ c(\tilde{P}), & \text{otherwise.} \end{cases} \quad (7)$$

We have $c'(a) = c(a) \geq c(a) - b(a)$ if $a \in \tilde{A}$ and $c'(a) = c(\tilde{P}) \geq c(a) - b(a)$ if $a \in A - \tilde{A}$ and so, the second constraint of (4) is satisfied by c' . From the fact that there is no $s - t$ path in $\tilde{G} = (N, \tilde{A})$ and all the arcs of $A - \tilde{A}$ have the same capacities $c(\tilde{P})$, it results that any $s - t$ path in $G' = (N, A, c', s, t)$ has the capacity equal to $c(\tilde{P})$ and so, the last condition in (4) is fulfilled. So, the last two constraints in (4) are satisfied by c' . It means that the set of feasible solutions is not empty. \square

To solve IMCP, we now introduce the following notation:

$$\tilde{A}_2 = \{a \in A \mid c(a) \leq c(\tilde{P})\} \quad (8)$$

The following result clarifies the reason of defining \tilde{A}_2 .

Theorem 2: If IMCP has an optimal solution, then there exists an optimal solution \tilde{c} to (4) so that

1. the capacities remain unchanged on the arcs of \tilde{A} and \tilde{A}_2 , i.e.,

$$\tilde{c}(a) = c(a), \quad a \in \tilde{A} \cup \tilde{A}_2,$$

2. if the capacities of some arcs are changed, then they are equal to the fixed value $c(\tilde{P})$, i.e.,

$$\tilde{c}(a) = \begin{cases} c(\tilde{P}), & \text{if } \tilde{c}(a) < c(a) \\ c(a), & \text{otherwise} \end{cases}$$

Proof: Consider an optimal solution \bar{c} of IMCP.

Proof of part 1: Let us take an arc $\alpha \in \tilde{A} \cup \tilde{A}_2$ so that $\bar{c}(\alpha) < c(\alpha)$. Of course, there are two different situations, either $\alpha \in \tilde{A}$ or $\alpha \in \tilde{A}_2$:

Case 1 ($\alpha \in \tilde{A}$): We have $\bar{c}(\alpha) \geq c(\alpha) - b(\alpha) > c(\tilde{P}) = \bar{c}(\tilde{P})$. On the other hand, we know that any $s - t$ path P has the capacity $\bar{c}(P) \leq \bar{c}(\tilde{P})$. So, it is obvious that the modification of

the capacity of α is useless since $\bar{c}(\alpha) > \bar{c}(P)$. The capacity vector \bar{c} defined as

$$\bar{c}(a) = \begin{cases} c(\alpha), & \text{if } a = \alpha \\ \bar{c}(a), & \text{otherwise} \end{cases}$$

is also an optimal solution to IMCP.

Case 2 ($\alpha \in \tilde{A}_2$): If there is a $s - t$ path P in \tilde{G} containing α , then $\bar{c}(P) \leq \bar{c}(\alpha) \leq c(\alpha) \leq c(\tilde{P}) = \bar{c}(\tilde{P})$. So, when the solution \bar{c} of IMCP is built, it is no need to change the capacity of α .

Proof of part 2: Let us take an arc $\alpha \in A$ of which capacity is modified, i.e., $\bar{c}(\alpha) < c(\alpha)$ and let us suppose that $\bar{c}(\alpha) \neq c(\tilde{P})$. There are 2 cases:

- 1) If $\bar{c}(\alpha) > c(\tilde{P}) = \bar{c}(\tilde{P})$, then we have a similar situation to case 1 in the first part of this proof.
- 2) If $\bar{c}(\alpha) < c(\tilde{P}) = \bar{c}(\tilde{P})$, then we have a similar situation to case 2 in the first part of this proof.

So, in both cases, modifying the capacity of α is useless. \square

From theorem 2, it results that only the arcs belonging to the set

$$\hat{A} = A - (\tilde{A} \cup \tilde{A}_2), \quad (9)$$

are the only candidates for modification to solve IMCP. Based on theorem 2, it results that if the capacity of an arc $a \in \hat{A}$ has to be modified, then its capacity becomes $\bar{c}(a) = c(\tilde{P})$.

Using the last two observations, the problem (4) is rewritten as follows:

$$\begin{cases} \min \{ \text{dist}(\bar{c}, c) \}, \\ \bar{c}(\tilde{P}) = c(\tilde{P}), \\ \bar{c}(a) = c(a), a \in A - \hat{A}, \\ \bar{c}(a) \in \{c(a), c(\tilde{P})\}, a \in \hat{A}, \\ \tilde{P} \text{ is a maximum capacity path in } \tilde{G} = (N, A, \bar{c}, s, t) \end{cases} \quad (10)$$

Depending on the formula used in (10) to measure the distance between the two capacity vectors, it yields two different problems with different approaches. We distinguish between two categories of distance formulas commonly applied to two m -dimensional vectors: sum-type and max-type. The first one is a sum of distances applied to each component and the second one is given by the maximum of the distances on components. We use a distance function d_i satisfying

$$\begin{aligned} d_i : R \times R &\rightarrow R_+, \\ d_i(u, u) &= 0, \forall u \in R \quad \text{and} \\ d_i(u, v) &= d(v, u), \quad \forall u, v \in R \end{aligned} \quad (11)$$

to measure the distance between i -th components of the two capacity vectors.

In the case of sum-type distance, the objective function of (10) is

$$\text{dist}(\bar{c}, c) = \sum_{a \in A} d_a(\bar{c}(a), c(a)). \quad (12)$$

In the case of max-type distance, we have:

$$\text{dist}(\bar{c}, c) = \max_{a \in A} d_a(\bar{c}(a), c(a)). \quad (13)$$

Hereafter, we associate the cost $\hat{c}(a)$ to every arc $a \in \hat{A}$:

$$\hat{c}(a) = d_a(c(a), c(\bar{P})) \tag{14}$$

Let us now introduce an auxiliary network $\hat{G} = (N, \hat{A}, \hat{c}, s, t)$. This network is the key idea to solve IMCP since it contains only the arcs of G for which the capacities can be modified. Moreover, if the capacity of an arc $a \in \hat{A}$ is modified then the cost of modifying the capacity is $\hat{c}(a)$. In order to solve IMCP, a set $B \subseteq \hat{A}$ has to be found so that if the arcs belonging to B are eliminated from \hat{A} , s no longer communicates with t through a directed path and the distance $dist$, obtained from (12) or (13), is minimized. So, the following theorem is immediate.

Theorem 3: The solution of IMCP is obtained as follows:

$$\bar{c}(a) = \begin{cases} c(\bar{P}), & \text{if } a \in B \\ c(a), & \text{if } a \in A - B \end{cases} \tag{15}$$

where B is the set of arcs eliminated from \hat{A} to minimize the objective function.

IV. IMCP UNDER SUM-TYPE DISTANCES

We denote IMCP under sum-type distance (12) by IMCPS. The corresponding optimization problem (10) under this distance is as follows:

$$\begin{cases} \min \left\{ \sum_{a \in A} d_a(\bar{c}(a), c(a)) \right\}, \\ \bar{c}(\bar{P}) = c(\bar{P}), \\ \bar{c}(a) = c(a), a \in A - \hat{A}, \\ \bar{c}(a) \in \{c(a), c(\bar{P})\}, a \in \hat{A}, \\ \bar{P} \text{ is a maximum capacity path in } \bar{G} = (N, A, \bar{c}, s, t). \end{cases} \tag{16}$$

For instance, in (16) we can consider the weighted l_k norm which is defined for two vectors x and y as follows:

$$l_k(x, y) = \sqrt[k]{\sum_{i=1}^m w_i |x_i - y_i|^k}. \tag{17}$$

Here, the value $w_i > 0$ is the given per unit cost of modification on the i -th component. Notice that the root of order $k > 0$ in (17) will be ignored since it has no influence in optimizing the objective function of (16). It is obvious that we have $d_a(\bar{c}(a), c(a)) = w_a |\bar{c}(a) - c(a)|^k$ in this case.

The sum-type Hamming distance is another well-known sum-type distance in the literature that can be used in (16):

$$H_{sum}(x, y) = \sum_{i=1}^m w_i H(x_i, y_i). \tag{18}$$

In formula (18), the value $w_i > 0$ is the fixed cost of modification of the i -th component and $H(u, v)$ is the Hamming distance between u and v which is defined as

$$H : R \times R \rightarrow \{0, 1\}, H(u, v) = \begin{cases} 1, & \text{if } u \neq v \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

When considering problem (16) under the sum-type Hamming distance, we have $d_a(\bar{c}(a), c(a)) = w_a \cdot H(\bar{c}(a), c(a))$.

Now let us focus on solving problem (16). For this purpose, we need to define the notion of $s - t$ cut [5]. Before doing that, we first introduce some notations. For two non-empty node sets V_1 and V_2 , the set (V_1, V_2) contains the arcs that connects nodes from V_1 to the nodes from V_2 , i.e., $(V_1, V_2) = \{a = (i, j) \in A | i \in V_1, j \in V_2\}$. We define $h(V_1, V_2) = \sum_{(i,j) \in (V_1, V_2)} h(i, j)$ for an arbitrary function $h: A \rightarrow R_+$.

For a non-empty set $X \subset V$, we denote $V - X$ by \bar{X} . The set of arcs $[X, \bar{X}] = (X, \bar{X}) \cup (\bar{X}, X)$ is called a *cut* in the network G . (X, \bar{X}) is called the set of forward arcs of the cut and (\bar{X}, X) is called the set of its backward arcs. In the special case that $s \in X$ and $t \in \bar{X}$, $[X, \bar{X}]$ is called an $s - t$ cut. The *capacity or cost* of a cut $[X, \bar{X}]$ is denoted by $c[X, \bar{X}]$ and is defined as follows:

$$h[X, \bar{X}] = h(X, \bar{X}). \tag{20}$$

An $s - t$ cut is called *minimum $s - t$ cut* if its cost is minimum among all $s - t$ cuts.

In IMCPS, the sum of costs $\hat{c}(a)$ on components must be minimized (see Theorem 3). Noting the notion of minimum $s - t$ cuts, it is easy to see that the set B in Theorem 3 is a minimum $s - t$ cut in \hat{G} . So, we have the following result:

Theorem 4: The capacity vector defined as

$$\bar{c}(a) = \begin{cases} c(\bar{P}), & \text{if } a \in (X, \bar{X}), \\ c(a), & \text{if } a \in A - (X, \bar{X}), \end{cases} \tag{21}$$

is an optimal solution of (16), where (X, \bar{X}) is a minimum $s - t$ cut in \hat{G} .

Now we are able to present our proposed algorithm to solve IMCPS.

AIMCPS:

Compute \tilde{A} (see (5));

if there is a path from s to t in $\tilde{G} = (N, \tilde{A})$ **then**

Stop. IMCPS is infeasible.

end if;

Compute \tilde{A}_2 (see (8));

Compute \hat{A} (see (9));

Compute \hat{c} using (14);

Find a minimum $s - t$ cut $[X, \bar{X}]$ in $\hat{G} = (N, \hat{A}, \hat{c}, s, t)$;

Compute \bar{c} using (21);

\bar{c} is an optimal solution of IMCPS.

Let us study now the time complexity of AIMCPS.

Theorem 5: AIMCPS has a strongly polynomial time complexity of $O(m \cdot n)$ in which m and n are defined as in (1).

Proof: Verifying the existence of a directed $s - t$ path can be done in $O(m)$ time using a depth search algorithm in \tilde{G} [5]. The minimum $s - t$ cut can be found in $O(m)$ time after a maximum flow is computed in \hat{G} [5]. The best-known algorithm for determining the maximum flow has a time complexity of $O(m \cdot n)$ [19]. So, the time complexity of AIMCPS is $O(m \cdot n)$. \square

V. IMCP UNDER MAX-TYPE DISTANCES

In this section, we present a polynomial-time algorithm to solve IMCP under max-type distance. This problem is denoted by IMCPM. In this case, problem (10) becomes

$$\left\{ \begin{array}{l} \min \{ \max_{a \in A} d_a(\bar{c}(a), c(a)) \}, \\ \bar{c}(\bar{P}) = c(\bar{P}), \\ \bar{c}(a) = c(a), a \in A - \hat{A}, \\ \bar{c}(a) \in \{c(a), c(\bar{P})\}, a \in \hat{A}, \\ \bar{P} \text{ is a maximum capacity path in } \bar{G} = (N, A, \bar{c}, s, t). \end{array} \right. \quad (22)$$

In (22), we can consider the weighted l_∞ norm which measures the distance between vectors x and y as

$$l_\infty(x, y) = \max_{i=1}^m w_i |x_i - y_i|. \quad (23)$$

Here, the value $w_i > 0$ is the given per unit cost of modification on the i -th component. So, we have $d_a(\bar{c}(a), c(a)) = w_a |\bar{c}(a) - c(a)|$.

As another max-type distance, we can also consider the bottleneck Hamming distance

$$H_{max}(x, y) = \max_{i=1}^m w_i H(x_i, y_i), \quad (24)$$

where the value $w_i > 0$ is the given fixed cost of modification on the i -th component and $H(u, v)$ is defined as in (19). In this case, we have $d_a(\bar{c}(a), c(a)) = w_a \cdot H(\bar{c}(a), c(a))$.

We propose the following algorithm to solve IMCPM.

AIMCPM:

Compute \tilde{A} (see (5));

if there is a path from s to t in $\tilde{G} = (N, \tilde{A})$ **then**

Stop. IMCPM has not solution.

end if;

Compute \hat{A} (see (9));

if there is no path from s to t in $\hat{G} = (N, \hat{A})$ **then**

Stop. $\bar{c} = c$ is the optimum solution of IMCPM.

end if;

Compute \hat{c} using (14);

Compute a maximum capacity path P in $\hat{G} = (N, \hat{A}, \hat{c}, s, t)$;

Compute $B = \{a \in \hat{A} | \hat{c}(a) \leq \hat{c}(P)\}$;

Compute \bar{c} using (15);

\bar{c} is an optimal solution of IMCPM.

Theorem 6: The capacity vector \bar{c} constructed by AIMCPM is an optimal solution of (22).

Proof: If IMCPM is feasible and there is no path from s to t in \hat{G} , then it is easy to see that there is no path from s to t in G with the capacity greater than $c(\bar{P})$. It means that \bar{P} is the maximum capacity path in G . So, $\bar{c} = c$ is an optimal solution to IMCPM.

Now, let us concentrate on the situation when IMCPM is feasible and there is at least one path from s to t in \hat{G} . In this case, all $s - t$ paths in \hat{G} must be interrupted by eliminating arcs from \hat{G} . We recall that the set of eliminated arcs was denoted in Theorem 3 by B . To solve IMCPM, the maximum

of costs $\hat{c}(a)$ on the arcs of \hat{A} must be minimized. This means that the minimum capacity arcs on the maximum capacity path P in \hat{G} must be eliminated. Fortunately, the elimination of all arcs a with $\hat{c}(a) \leq \hat{c}(P)$ interrupts all possible $s - t$ paths in \hat{G} and it does not change the value of the max-type distance. So, for solving IMCPM, we can set $B = \{a \in \hat{A} | \hat{c}(a) \leq \hat{c}(P)\}$. Based on Theorem 3, it follows that \bar{c} calculated using (15) is an optimal solution of IMCPM. \square

Let us study the time complexity of AIMCPM.

Theorem 7: AIMCPM has a linear time complexity of $O(m)$ in which m is defined in (1).

Proof: Verifying the existence of a directed path from s to t in \tilde{G} and \hat{G} can be done in $O(m)$ time using a depth search algorithm [5]. The maximum capacity path in \hat{G} can be found in $O(m)$ time [7]. So, the time complexity of AIMCPM is $O(m)$. \square

VI. GENERAL CASE

In this section, we consider the case when the capacity of the given path \bar{P} can be modified. We denote this problem by GIMCP. It is to find a new capacity vector $\bar{c} : A \rightarrow R_+$ in a way that

$$\left\{ \begin{array}{l} \min \{ \text{dist}(\bar{c}, c) \}, \\ c(a) + h(a) \geq \bar{c}(a) \geq c(a) - b(a), a \in A, \\ \bar{P} \text{ is a maximum capacity path in } \bar{G} = (N, A, \bar{c}, s, t). \end{array} \right. \quad (25)$$

In (25), the capacity of every arc a cannot be decreased more than $b(a)$ units, where $c(a) \geq b(a) \geq 0$, and the capacity of every arc a cannot be increased more than $h(a)$ units, where $h(a) \geq 0$. It is obvious that the capacities of arcs belonging to \bar{P} may be either fixed or increased, while the capacities of the other arcs may be either fixed or decreased. Of course, at the end, all the modified capacities are the same.

To solve GIMCP, we successively increase the capacity of \bar{P} and apply our proposed algorithms for every new capacity of \bar{P} .

If the value of the capacity of the path \bar{P} is increased to p the network $G' = (N, A, c', s, t)$ is obtained, where

$$c'(a) = \begin{cases} p, & \text{if } a \in \bar{P} \text{ and } c(a) < p, \\ c(a), & \text{otherwise.} \end{cases} \quad (26)$$

Absolutely, the above increasing is possible only if the modified capacity vector c' satisfies the constraints of (25):

$$c(a) + h(a) \geq p, \text{ if } a \in \bar{P} \text{ and } c(a) < p. \quad (27)$$

So, it results that the upper bound for increasing the capacity of \bar{P} is

$$h(\bar{P}) = \min(c(P_{max}), \min\{c(a) + h(a) | a \in \bar{P}\}), \quad (28)$$

where P_{max} is the maximum capacity path in the initial network G .

The idea of our proposed algorithm is to find a value $p \in [c(\bar{P}), h(\bar{P})]$ so that if we increase the capacity of \bar{P} to p and decrease the capacity of other paths to p , then the cost of these modifications is minimized. The following algorithm is

presented to solve GIMCP under any general distance when the capacities are required to be integer:

AGIMCP:

Compute $h(\bar{P})$ using (28);

for $p = c(\bar{P})$ **to** $h(\bar{P})$ **do**

Increase the capacity on \bar{P} to p (see (26));

Apply AIMCPS or AIMCPM in the network

$G' = (N, A, c', s, t)$;

end for;

The optimum solution of GIMCP is given by the minimum cost solution constructed in the above “for” loop.

Theorem 8: AGIMCP has a time complexity of $O(m \cdot n \cdot d)$ under sum-type distances and has a complexity of $O(md)$ under the max-type distances, where $d = h(\bar{P}) - c(\bar{P})$.

Proof: The result is immediate since AIMCPS and AIMCPM run respectively in $O(m \cdot n)$ and $O(m)$ times (see theorems 5 and 7). \square

Although AGIMCP has a semi-polynomial time due to the existence of d , we can restrict the search region to convert it into a polynomial-time algorithm for solving GIMCPS under some well-known distances. Let us first focus on the case that $dist$ is either the weighted l_1 norm or one of bottleneck-type and sum-type Hamming distances. The following result determines that the region search is reduced to a set of $O(m)$ cardinality.

Theorem 9: There is an optimal solution c^* of GIMCP under the weighted l_1 norm and the bottleneck-type and sum-type Hamming distances so that the $c^*(\bar{P})$ belongs to the finite set

$$S = [c(\bar{P}), h(\bar{P})] \cap \left(\bigcup_{a \in A} \{c(a)\} \bigcup_{a \in A \setminus \bar{P}} \{c(a) - b(a)\} \cup \{h(\bar{P})\} \right). \quad (29)$$

Proof: Let c^* be an optimal solution so that $c^*(\bar{P}) \notin S$. Assume that we have sorted the elements of S in a non-decreasing order, and the sorted list is as

$$c(\bar{P}) = p_1 < p_2 < \dots < p_l = h(\bar{P}). \quad (30)$$

So there is an index $k \in \{1, 2, \dots, l-1\}$ so that $p_k \leq c^*(\bar{P}) < p_{k+1}$. Let us split the proof into two cases:

- 1) GIMCP under the bottleneck-type and sum-type Hamming distances;
- 2) GIMCP under the weighted l_1 norm.

Case 1: Define the new capacity vector c^{**} as follows:

$$c^{**}(a) = \begin{cases} c(a), & c^*(a) = c(a), \\ p_k, & c^*(a) \neq c(a). \end{cases} \quad (31)$$

It is easy to see that \bar{P} is a feasible solution of (25). On the other hand, by definition, we have $c^{**}(a) \neq c(a) \Rightarrow c^*(a) \neq c(a)$. So the objective value of c^{**} is less than or equal to c^* . These results show that c^{**} is also an optimal solution satisfying the desired result.

Case 2: Now we have to continue the proof in two distinct situations:

- 1) $\sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a > \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a$;
- 2) $\sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a \leq \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a$.

In the first situation, it is easy to see that c^{**} defined in (31) is also feasible and has a cost less than or equal to that of c^* because

$$\begin{aligned} dist(c^*, c) &= \sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a (c^*(\bar{P}) - c(a)) \\ &\quad + \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a (c(a) - c^*(\bar{P})) \\ &= c^*(\bar{P}) \left(\sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a - \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a \right) \\ &\quad + \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a c(a) - \sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a c(a) \\ &\geq p_k \left(\sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a - \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a \right) \\ &\quad + \sum_{a \in A \setminus \bar{P}: c(a) > c(\bar{P})} w_a c(a) - \sum_{a \in \bar{P}: c(a) < c(\bar{P})} w_a c(a) \\ &= \sum_{a \in \bar{P}: c(a) < p_k} w_a (p_k - c(a)) \\ &\quad + \sum_{a \in A \setminus \bar{P}: c(a) > p_k} w_a (c(a) - p_k) = dist(c^{**}, c). \end{aligned}$$

Consequently, c^{**} satisfies the desired result.

In the second situation, we define the new capacity vector \bar{c}^{**} as follows:

$$\bar{c}^{**}(a) = \begin{cases} c(a), & c^*(a) = c(a), \\ p_{k+1}, & c^*(a) \neq c(a). \end{cases} \quad (32)$$

Similarly, one can prove that the cost of \bar{c}^{**} is less than or equal to that of c^* . This completes the proof because \bar{c}^{**} is the desired optimal solution. \square

Based on Theorem 9, it suffices that the search region is restricted to the set S . So we have the following algorithm.

AGIMCP2:

Compute $h(\bar{P})$ using (28);

$S = \{h(\bar{P})\}$;

for each arc $a \in A$ **do**

if $c(\bar{P}) \leq c(a) \leq h(\bar{P})$ **then**

$S = S \cup c(a)$;

end if;

if $c(\bar{P}) \leq c(a) - b(a) \leq h(\bar{P})$ **then**

$S = S \cup c(a) - b(a)$;

end if;

end for;

for all values p in S **do**

Increase the capacity on \bar{P} to p (see (26));

Apply AIMCPS or AIMCPM in the network

$G' = (N, A, c', s, t)$;

end for;

The optimal solution of GIMCP is given by the minimum cost solution constructed in the above “for” loop.

Theorem 10: AGIMCP2 solves GIMCP under the Hamming distances as well as the weighted l_1 norm in strongly polynomial time.

Proof: Since $|S| \leq 2m + 1$, it follows that the algorithm has at most $O(m)$ iterations. Based on the fact that the complexity of AIMCPS is $O(m \cdot n)$ (see Theorem 5), it results that the time complexity of AGIMCP2 is $O(m^2 \cdot n)$ under the sum-type Hamming distance and the weighted l_1 norm. Moreover, the complexity of AIMCPM is $O(m)$ which implies that AGIMCP2 solves the problem under the bottleneck-type Hamming distance in $O(m^2)$ time. \square

We now concentrate on GIMCP under the weighted l_∞ norm.

Theorem 11: GIMCP under the weighted l_∞ norm has an optimal solution c^* so that $c^*(\bar{P})$ belongs to the set

$$\bar{S} = [c(\bar{P}), h(\bar{P})] \cap \left(\bigcup_{a \in A \setminus \bar{P}} \bigcup_{b \in \bar{P}} \left\{ \frac{w_a c(a) + w_b c(b)}{w_a + w_b} \right\} \cup \bigcup_{a \in A \setminus \bar{P}} \{c(a) - b(a)\} \cup \{c(\bar{P}), h(\bar{P})\} \right). \quad (33)$$

Proof: Let c^* be an optimal solution of GIMCP under the weighted l_∞ norm and $p^* = c^*(\bar{P})$. Assume that:

$$\bar{a} = \operatorname{argmax}_{a \in A \setminus \bar{P}} w_a (c(a) - p^*)$$

and

$$\bar{b} = \operatorname{argmax}_{a \in \bar{P}} \{w_a (p^* - c(a))\}.$$

If p^* is an element of $\bigcup_{a \in A \setminus \bar{P}} \{c(a) - b(a)\} \cup \{c(\bar{P}), h(\bar{P})\}$, then the result is immediate. So, we assume now that $p^* \notin \bigcup_{a \in A \setminus \bar{P}} \{c(a) - b(a)\} \cup \{c(\bar{P}), h(\bar{P})\}$. By definition, $\operatorname{dist}(c^*, c) = \max \{w_{\bar{a}} (c(\bar{a}) - p^*), w_{\bar{b}} (p^* - c(\bar{b}))\}$. If $w_{\bar{a}} (c(\bar{a}) - p^*)$ and $w_{\bar{b}} (p^* - c(\bar{b}))$ are not equal, then we can decrease their maximum to reduce the objective value. This causes that the other is increased. So, the optimal state

is when both the values are equal:

$$\begin{aligned} w_{\bar{a}} (c(\bar{a}) - p^*) &= w_{\bar{b}} (p^* - c(\bar{b})) \\ \Rightarrow p^* &= \frac{w_{\bar{a}} c(\bar{a}) + w_{\bar{b}} c(\bar{b})}{w_{\bar{a}} + w_{\bar{b}}} \\ \Rightarrow p^* &\in \bigcup_{a \in A \setminus \bar{P}} \bigcup_{b \in \bar{P}} \left\{ \frac{w_a c(a) + w_b c(b)}{w_a + w_b} \right\}. \end{aligned}$$

This completes the proof. \square

Based on Theorem 11, we can propose the following algorithm to solve GIMCP under the weighted l_∞ norm.

AGIMCP3:

Compute $h(\bar{P})$ using (28);

$\bar{S} = \{c(\bar{P}), h(\bar{P})\}$;

foreach arc $a \in A \setminus \bar{P}$ **do**

for each arc b in \bar{P} **do**

if $c(\bar{P}) \leq \frac{w_a c(a) + w_b c(b)}{w_a + w_b} \leq h(\bar{P})$ **then**

$\bar{S} = \bar{S} \cup \frac{w_a c(a) + w_b c(b)}{w_a + w_b}$

if $c(\bar{P}) \leq c(a) \leq h(\bar{P})$ **then**

$\bar{S} = \bar{S} \cup c(a)$;

end if;

if $c(\bar{P}) \leq c(a) - b(a) \leq h(\bar{P})$ **then**

$\bar{S} = \bar{S} \cup c(a) - b(a)$;

end if;

end for;

for all values p in \bar{S} **do**

Increase the capacity on \bar{P} to p (see (26));

Apply AIMCPM in $G' = (N, A, c', s, t)$;

end for;

The optimum solution of GIMCP is given by the minimum cost solution constructed in the above “for” loop

Theorem 12: AGIMCP3 has a time complexity of $O(m^2n)$.

Proof: Since $|\bar{S}| \leq O(mn)$ and AIMCPM runs in $O(m)$ time (see Theorem 7), it follows that AGIMCP3 solves the problem under the weighted l_∞ norm in $O(m^2n)$ time. \square

In table 1, a comparative list by time complexities of algorithms for IMCP from this paper and from the known literature is presented.

VII. PRACTICAL APPLICATION

Figure 1 depicts the roadmap of a seaside zone in Iran. Two major cities in the zone, Mahmood Abad and Behshahr, are marked with red dots on the map.

The shortest possible route between the two cities is the coastline route that crosses the cities of Fereydoon Kenar – Babolsar – Khazar Abad, and Kenar Darya. Because this route is welcomed by many passengers, transportation engineers want to impose traffic solutions on the routes of this zone in order to reduce the volume of traffic on this route. Figure 2 shows the network structure of the routes between Mahmood Abad and Behshahr. For this purpose, each road is

TABLE 1. Algorithms efficiencies comparison.

IMCP case	Norm / distance	Current paper time complexity	Known literature time complexity	Remarks
Special case when capacity of the given path is preserved	any sum-type norm	$O(mn)$	-	This special case was not studied before, has practical application to transportation network optimization, and algorithms are more efficient than from the classical case
	any max-type norm	$O(m)$	-	
	weighted l_k norm	$O(mn)$	-	
	weighted l_1 norm	$O(mn)$	-	
	weighted l_∞ norm	$O(m)$	-	
	sum-type Hamming distance	$O(mn)$	-	
	bottleneck Hamming distance	$O(m)$	-	
Classical case when capacity of the given path is not preserved	any sum-type norm	$O(mnd)$	-	These more general and complex cases were not previously studied
	any max-type norm	$O(md)$	-	
	weighted l_k norm	$O(mnd)$	-	
	weighted l_1 norm	$O(m^2n)$	$O(m^2n)$	The algorithms in current paper and from previously known literature have similar efficiency
	weighted l_∞ norm	$O(m^2n)$	$O(m^2n)$	
	sum-type Hamming distance	$O(m^2n)$	$O(m^2n)$	
	bottleneck Hamming distance	$O(m^2)$	$O(m^2)$	

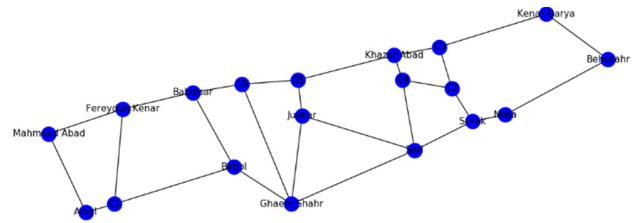









FIGURE 2. The network structure corresponding to Figure 1.

TABLE 2. Capacity of vehicles.

Type	Capacity
Motorcycles 	1
Passenger cars 	2
Pickups, Panels, Vans 	3
Buses 	4
Single unit trucks 	5
Single trailer trucks 	6
Multi Trailer Trucks 	7

capacity of a route if they wish to reduce the traffic load on that road, but they are not able to increase the capacity of a route without improving the infrastructure and widening the road. Therefore, reducing the capacity does not cost much, while increasing the capacity is accompanied by exorbitant road construction costs. So, it is natural that we consider only reducing capacity in this example.

To reduce traffic on the first route, transport engineers have proposed an alternative route for transportation between the two cities, which passes through the cities of Amol, Babol, Sari and Neka. The strategy is that road traffic load must be transferred to the second route. For this purpose, the capacity of all routes between these two cities can be reduced in such a way that the second route has more capacity than any other route. Table 3 shows the initial capacity of roads. For simplicity, we consider the cost of reducing the capacity of each road by one unit in proportion to the Euclidean distance of its endpoints. Notice that this is the same IMCP with the integer capacities and in a situation where increasing the capacity of the given path is not allowed. Although the underlying network is undirected, it is easy to see that our proposed algorithms work well in undirected networks. If we use AIMCPS to solve this instance of IMCP under the weighted l_1 norm, the minimum cut C is obtained which contains edges ('Behshahr', 'Neka'), ('Khazar Abad', 'C3'), and ('C3', 'C2'). So we have to reduce the capacities of two



FIGURE 1. Roadmap of a seaside zone in Iran (<https://www.mazrec.co.ir/en-maz-intro>).

assigned a capacity, which indicates the allowed passage of vehicles according.

Table 2 shows the types of capacity. For example, if we allocate a capacity of 4 to a road, the passage of all types of trucks is prohibited and the passage of the other vehicles is allowed. It is noteworthy that if a route includes several roads, it is natural to consider the minimum capacity of its roads as the capacity of that route. The capacity of a road depends on various factors such as road width, and road infrastructure. Of course, to control traffic, traffic experts can reduce the

TABLE 3. Data for IMCP.

Road		Capacity	Cost	Lower bound	
From	To			$c(a) - b(a)$	
Surak	C2	5	4	1	
Surak	Sari	3	7	1	
Surak	Neka	4	4	1	
Babolsar	C6	7	6	1	
Ghaem Shahr	C6	4	15	1	
	C5	4	6	1	
Behshahr	Kenar Darya	7	9	1	
Behshahr	Neka	6	14	1	
Khazar Abad	C3	6	5	1	
Khazar Abad	C5	6	12	1	
Khazar Abad	C4	5	3	1	
Fereydun Kenar	Babolsar	7	8	1	
Fereydun Kenar	C1	5	11	1	
Fereydun Kenar	Mahmood Abad	6	9	1	
Juybar	Ghaem Shahr	6	10	1	
Juybar	Sari	4	14	1	
Juybar	C5	7	4	1	
Ghaem Shahr	Sari	6	16	1	
Ghaem Shahr	Babol	3	8	1	
Amol	C1	3	3	1	
Amol	Mahmood Abad	4	10	1	
Kenar Darya	C3	5	13	1	
Babolsar	Babol	6	10	1	
	C2	C3	4	5	1
	Sari	C4	5	8	1
	C2	C4	7	6	1
	Babol	C1	5	15	1

edges ('Khazar Abad', 'C3'), and ('C3', 'C2') respectively from 6 and 4 to 3 and 3 with total cost 20.

VIII. CONCLUSION

In this paper, two cases of IMCP were considered: when the capacity of the given path is preserved or not. In the first case, an $O(mn)$ time algorithm under sum-type distance was proposed and then, an $O(m)$ time algorithm under max-type distance was presented. In the former, the problem is reduced to a minimum cut problem in an auxiliary network and in the later, the problem is reduced to computing a maximum

capacity path. In the case of capacity preservation, a practical application of road transportation network optimization in a region of Iran was performed using an algorithm for IMCP.

In the case when the value of the maximum capacity path is not preserved, the time complexity is worse. The first two algorithms are adapted resulting into an $O(mnd)$ time algorithm under sum-type distance and, respectively, an $O(md)$ time algorithm under max-type distance. In the particular cases of weighted l_1 norm, sum-type Hamming distance and weighted l_∞ norm the time complexity of the algorithm is $O(m^2n)$ which is strongly polynomial and in the particular case of bottleneck Hamming distance, a better $O(mn)$ time complexity is obtained. In the particular cases of weighted l_1 and l_∞ norm, and Hamming distances, algorithms from current paper and from previously known literature have similar efficiencies (see Table 1).

REFERENCES

- [1] N. Shacham, "Multicast routing of hierarchical data," in *Proc. SUPERCOMM/ICC 92 Discovering New World Commun.*, 1992, pp. 1217–1221.
- [2] M. Schulze, "A new monotonic, clone-independent, reversal symmetric, and condorcet-consistent single-winner election method," *Social Choice Welfare*, vol. 36, no. 2, pp. 267–303, Feb. 2011.
- [3] E. Fernandez, R. Garfinkel, and R. Arbiol, "Mosaicking of aerial photographic maps via seams defined by bottleneck shortest paths," *Oper. Res.*, vol. 46, no. 3, pp. 293–304, Jun. 1998.
- [4] E. Ullah, K. Lee, and S. Hassoun, "An algorithm for identifying dominant-edge metabolic pathways," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Dec. 2009, pp. 144–150.
- [5] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993, pp. 210–212.
- [6] A. P. Punnen, "A linear time algorithm for the maximum capacity path problem," *Eur. J. Oper. Res.*, vol. 53, no. 3, pp. 402–404, Aug. 1991.
- [7] R. K. Ahuja and J. B. Orlin, "Combinatorial algorithms for inverse network flow problems," *Networks*, vol. 40, no. 4, pp. 181–187, Dec. 2002.
- [8] R. K. Ahuja and J. B. Orlin, "Inverse Optimization," *Oper. Res.*, vol. 49, pp. 771–783, Oct. 2001.
- [9] M. Demange and J. Monnot, "An introduction to inverse combinatorial problems," in *Paradigms of Combinatorial Optimization* (Problems and New approaches). London, U.K.: Wiley, 2010.
- [10] C. Heuberger, "Inverse combinatorial optimization: A survey on problems, methods, and results," *J. Combinat. Optim.*, vol. 8, no. 3, pp. 329–361, Sep. 2004.
- [11] X. Guan, P. M. Pardalos, and B. Zhang, "Inverse max+sum spanning tree problem under weighted l_1 norm by modifying the sum-cost vector," *Optim. Lett.*, vol. 12, no. 5, pp. 1065–1077, Jul. 2018.
- [12] X. Guan, X. He, P. M. Pardalos, and B. Zhang, "Inverse Max+Sum spanning tree problem under Hamming distance by modifying the sum-cost vector," *J. Glob. Optim.*, vol. 2017, vol. 69, no. 4, pp. 911–925.
- [13] X. Guan, P. M. Pardalos, and X. Zuo, "Inverse Max+Sum spanning tree problem by modifying the sum-cost vector under weighted l_1 Norm," *J. Glob. Optim.*, vol. 61, no. 1, pp. 165–182, 2015.
- [14] C. Yang and J. Zhang, "Inverse maximum capacity problems," *OR Spektrum*, vol. 20, 1997, pp. 97–110.
- [15] X. Yang and J. Zhang, "Some inverse min-max network problems under weighted l_1 and l_∞ norms with bound constraints on changes," *J. Combinat. Optim.*, vol. 13, no. 2, pp. 123–135, Dec. 2006.
- [16] X. Guan and J. Zhang, "Inverse bottleneck optimization problems on networks," in *Proc. AAIM*, 2006, pp. 220–230.
- [17] A. Deaconu and E. Ciurea, "The inverse maximum flow problem under Lk norms," *Carpathian J. Math.*, vol. 28, no. 1, pp. 59–66, 2012.
- [18] L. H. M. Van, K. T. Nguyen, and N. T. Hung, "Computational aspects of the inverse single facility location problem on trees under 1-norm," *Theor. Comput. Sci.*, vol. 844, pp. 133–141, Dec. 2020.
- [19] J. B. Orlin, "Max flows in $O(nm)$ time, or better," in *Proc. 45th Annu. ACM Symp. Symp. Theory Comput. (STOC)*, 2013, pp. 765–774.



ADRIAN M. DEACONU was born in Braşov, Romania, in 1974. He received the B.S. and M.S. degrees in computer science and the Ph.D. degree in inverse network optimization from the University Transilvania, Braşov, in 1998 and 2008, respectively. He is a member of the Romanian Mathematical Society and the Mathematical Modeling and Software Products Research Centre, Research Development Institute, Braşov.

From 2000 to 2008, he was an Assistant Professor with the Department of Theoretical Computer Science, University Transilvania. From 2019 to 2020, he was a Visiting Professor with the University College Cork, Ireland. Since 2008, he has been an Associate Professor with the Department of Mathematics and Computer Science, University Transilvania. He is the author of four books and more than 25 articles. His research interests include algorithms, graphs, optimization, inverse combinatorial optimization, image processing, computational geometry, sustainable energy, and smart grids.

Dr. Deaconu's awards include four papers on the list of Awarding Research Results–Articles (Romanian Ministry of National Education), in 2009, 2019, and 2020. He is currently a Guest Editor of the *International Journal of Photoenergy*.



JAVAD TAYYEBI was born in Birjand, Iran, in 1985. He received the B.S. degree in mathematics from the University of Birjand, in 2007, the M.S. degree in applied mathematics from the Sharif University of Technology, Iran, in 2009, and the Ph.D. degree in operations research from the University of Birjand, in 2014. The subject of his thesis was “inverse optimization.”

Since 2015, he has been an Assistant Professor with the Department of Industrial Engineering, Birjand University of Technology. He is the author of more than 15 articles. His research interests include network optimization, game theory, and fuzzy theory. He is a member of the Iranian Operations Research Society.

...