Iranian Journal of Operations Research
Vol. 5, No. 2, 2014, pp. 12-25

Capacity Inverse Minimum Cost Flow Problem under the Weighted Hamming Distances

Massoud Aman[1], Javad Tayyebi[2*]

Given an instance of the minimum cost flow problem, a version of the corresponding inverse problem, called the capacity inverse problem, is to modify the upper and lower bounds on arc flows as little as possible so that a given feasible flow $\boldsymbol{x}^0$ becomes optimal to the modified minimum cost flow problem. The modifications can be measured by different distances. Here, we consider the capacity inverse problem under the bottleneck-type and the sum-type weighted Hamming distances. In the bottleneck-type case, the binary search technique is applied to present an algorithm for solving the problem in $O(nm\,log\,n)$ time. In the sum-type case, it is shown that the inverse problem is strongly NP-hard even on bipartite networks.

Keywords: *Combinatorial optimization, minimum cost flow problem, inverse problem, Hamming distance, complexity*

Manuscript was received on 15/04/2015, revised on 13/05/2015 and accepted for publication on 17/06/2015.

1. Introduction

For a particular optimization problem, a corresponding inverse problem is to modify some parameters of the problem as little as possible such that a given feasible solution $\boldsymbol{x}^0$ becomes optimal to the new problem. The modifications can be measured by different distances such as l_1, l_2 and l_∞ norms and also the weighted Hamming distances. Inverse optimization problems have many applications in traffic modeling, seismic tomography and the design of computer networks [4, 5, 6, 9].

The concept of inverse problems was first proposed by Tarantola [14] in geophysical sciences. Subsequently, Burton and Toint [4, 5] made use of this concept in the context of combinatorial optimization and considered the inverse shortest path problem. Since then, inverse optimization problems have been studied by many authors; see [7, 11], for a survey.

Here, we consider a particular type of inverse minimum cost flow problems. First, let us review the well-known minimum cost flow problem. The problem is to minimize the cost of sending a flow x in a given directed graph $G(V, A)$, where $V = \{1, 2, \dots, n\}$ is the set of nodes and A is the set of m arcs under some balancing constraints over the nodes and capacity constraints over the arcs. This problem can be formulated as follows:

* Corresponding Author.
[1] Department of Mathematics, Faculty of Mathematical Sciences and Statistics, Birjand University, Birjand, Iran, Email: mamann@birjand.ac.ir.
[2] Department of Mathematics, Faculty of Mathematical Sciences and Statistics, Birjand University, Birjand, Iran, Email: javadtayyebi@birjand.ac.ir.

$$\begin{aligned} \min \ & \sum_{(i,j)\in A} c_{ij}x_{ij} \\ \text{s.t.} \ & \sum_{(i,j)\in A} c_{ij}x_{ij} - \sum_{(j,i)\in A} c_{ij}x_{ij} = b_i, \quad \forall i \in V, \\ & l_{ij} \le x_{ij} \le u_{ij}, \quad \forall (i,j) \in A, \end{aligned} \tag{1}$$

where c_{ij} denotes the cost of sending one unit of flow on arc (i,j), $l_{ij} \geq 0$ and $u_{ij} \geq 0$ are the minimum and maximum amount that can flow on arc (i,j), respectively, b_i indicates the supply or the demand of each node i. We denote the network corresponding to the problem (1) by $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$. The problem (1) can be solved by several strongly polynomial-time algorithms [1].

Two versions of the inverse minimum cost flow problems are considered in the literature: the cost inverse minimum cost flow (CoIMCF) problem and the capacity inverse minimum cost flow (CaIMCF) problem. Suppose that $\boldsymbol{x}^0$ is a given feasible flow for (1). In the CoIMCF problem, the cost vector $\boldsymbol{c}$ is adjusted as little as possible to make $\boldsymbol{x}^0$ form a minimum cost flow to the problem (1). While in the CaIMCF problem, the vectors $\boldsymbol{l}$ and $\boldsymbol{u}$ are modified minimally so that $\boldsymbol{x}^0$ becomes optimal to the problem (1). The CoIMCF problem is studied by several authors [3, 12, 15, 16, 17]. Zhang and Liu [17] considered the inverse linear programming problems under the l_1 norm and proposed a method for solving the problem based on the reduced cost optimality conditions. They showed that the method yielded a strongly polynomial-time algorithm for the CoIMCF problem. Ahuja and Orlin [2, 3] considered the inverse linear programming problem under the l_1 and l_∞ norms and showed that the inverse problem of a linear programming problem was also a new linear programming problem and analyzed the inverse minimum cost flow problem as a special case. They showed that for the l_1 norm, the CoIMCF problem reduced to a unit capacity minimum cost flow problem and for the l_∞ norm, the CoIMCF problem turned out to be solvable as a minimum cost-to-time ratio cycle problem. In [8], the linear search and the binary search techniques were applied to solve some special types of the CoIMCF problems such as the inverse shortest path problem and the inverse assignment problem. Jiang et al. [12] considered the CoIMCF problem under the sum-type and the bottleneck-type weighted Hamming distances. In the sum-type case, they showed that the problem was APX-hard. In the bottleneck-type case, an $O(nm^2)$ algorithm was presented to solve the problem. In [15], Tayyebi and Aman showed that the proposed algorithm in [12] did not solve the inverse problem in the general case. Then, they proposed two new algorithms to solve the problem in $O(nm^2) and\ O(mn\log n)$ times [15, 16]. To the best of our knowledge, the CaIMCF problem was studied only by Güler and Hamacher [10]. They showed that the problem was NP-hard under the l_1 norm due to the minimum weighted feedback arc set problem. Furthermore, they presented a greedy algorithm for solving the problem under the l_∞ norm in $O(nm^2)$ time.

Here, we study the CaIMCF problem under the weighted Hamming distances. We considered both the bottleneck-type and the sum-type cases. In the bottleneck-type case, an $O(nm\log n)$ algorithm based on the binary search technique is proposed to solve the inverse problem. In spite of the fact that the Hamming distance is nonconvex and discontinuous, the CaIMCF problem under the l_∞ norm has a behavior similar to the bottleneck-type case. Hence, our proposed algorithm can also solve the CaIMCF problem under the l_∞ norm with a better complexity than the one presented in [10]. In the sum-type case, it is shown that the minimum node cover problem is reduced to the inverse problem and consequently, the inverse problem is strongly NP-hard even

on bipartite networks. It is remarkable that the reduction presented in [10] is also valid for the sum-type case but it only proves that the problem is weakly NP-hard on general networks.

The rest of our work is organized as follows. Section 2 describes the CaIMCF problems in detail and gives combinatorial formulations of the problems. Section 3 considers the CaIMCF problem under the bottleneck-type Hamming distance and presents a strongly polynomial-time algorithm. Section 4 analyzes the CaIMCF problem and its complexity under the sum-type Hamming distance. Section 5 compares our results with ones obtained in [10]. Finally, our final remarks are given in Section 6.

2. Preliminaries

In this section, we state some notions used throughout the manuscript and formulate the inverse problems.

Let $\boldsymbol{x}^0$ be a feasible flow to problem (1) defined on the network $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$. We denote the residual network of $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$ with respect to $\boldsymbol{x}^0$ by $G'(V, A', \boldsymbol{u}', \boldsymbol{c}')$ [1]. We also denote the arc sets $A' \cap A$ and $A' \backslash A$ by A'_U and A'_L, respectively.

The following lemma gives the negative cycle optimality conditions for a feasible flow $\boldsymbol{x}^0$ of problem (1) [1].

Lemma 2.1. A feasible flow $\boldsymbol{x}^0$ is optimal to problem (1) if and only if the corresponding residual network does not contain any negative (cost) directed cycle.

Suppose that $\boldsymbol{x}^0$ is a feasible flow to problem (1). In the CaIMCF problem under the bottleneck-type weighted Hamming distance, we look for vectors $\hat{\boldsymbol{l}}$ and $\hat{\boldsymbol{u}}$ such that the following conditions are satisfied:

I. $\boldsymbol{x}^0$ is a minimum cost flow in the network $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$;
II. $-p_{ij}^l + l_{ij} \le \hat{l}_{ij} \le \min\{x_{ij}^0, q_{ij}^l + l_{ij}\}$, for every $(i,j) \in A$, where $p_{ij}^l \ge 0$ and $q_{ij}^l \ge 0$ are respectively the given bounds for decreasing and increasing l_{ij};
III. $\max\{x_{ij}^0, -p_{ij}^u + u_{ij}\} \le \hat{u}_{ij} \le q_{ij}^u + u_{ij}$, for every $(i,j) \in A$, where $p_{ij}^u \ge 0$ and $q_{ij}^u \ge 0$ are respectively the given bounds for decreasing and increasing u_{ij};
IV. the value

$$\max\{\max_{(i,j)\in A} w_{ij}^l H(l_{ij}, \hat{l}_{ij}), \max_{(i,j)\in A} w_{ij}^u H(u_{ij}, \hat{u}_{ij})\}$$

is minimized where $w_{ij}^l \ge 0$ and $w_{ij}^u \ge 0$, $(i,j) \in A$, are respectively penalties for modifying l_{ij} and u_{ij}. For each $r, \hat{r} \in \mathbb{R}$, $H(r, \hat{r})$ is the Hamming distance between r and $\hat{r}$, i.e., $H(r, \hat{r}) = 1$, if $r \ne \hat{r}$ and $H(r, \hat{r}) = 0$, otherwise.

It is notable that the constraints (II) and (III) guarantee that $\hat{l}_{ij} \le x_{ij}^0 \le \hat{u}_{ij}$ for every $(i,j) \in A$. Using Lemma 2.1, the CaIMCF problem under the bottleneck-type Hamming distance can be formulated as follows:

$$\min z = \max\left\{\max_{(i,j)\in A} w_{ij}^l H(l_{ij}, \hat{l}_{ij}), \max_{(i,j)\in A} w_{ij}^u H(u_{ij}, \hat{u}_{ij})\right\} \quad (2a)$$

$$\text{s.t. } G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}') \text{ contains no negative directed cycle,} \quad (2b)$$

$$-p_{ij}^l + l_{ij} \le \hat{l}_{ij} \le \min\{x_{ij}^0, q_{ij}^l + l_{ij}\}, \quad \forall (i,j) \in A, \quad (2c)$$

$$\max\{x_{ij}^0, -p_{ij}^u + u_{ij}\} \le \hat{u}_{ij} \le q_{ij}^u + u_{ij}, \quad \forall (i,j) \in A, \tag{2d}$$

where $G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}')$ is the residual network of $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ with respect to $\boldsymbol{x}^0$. Here, all the residual networks considered are with respect to $\boldsymbol{x}^0$. For this reason, henceforth we introduce a residual network only by determining its associated network.

In the CaIMCF problem under the sum-type Hamming distance, we look for vectors $\hat{\boldsymbol{l}}$ and $\hat{\boldsymbol{u}}$ such that the constraints (I), (II) and (III) hold and

V. the value

$$\sum_{(i,j)\in A} w_{ij}^l H(l_{ij}, \hat{l}_{ij}) + \sum_{(i,j)\in A} w_{ij}^u H(u_{ij}, \hat{u}_{ij})$$

is minimized, where $w_{ij}^l \ge 0$ and $w_{ij}^u \ge 0$ are defined as in (IV).

Therefore, the CaIMCF problem under the sum-type Hamming distance can be formulated as follows:

$$\min z = \sum_{(i,j)\in A} w_{ij}^l H(l_{ij}, \hat{l}_{ij}) + \sum_{(i,j)\in A} w_{ij}^u H(u_{ij}, \hat{u}_{ij}), \tag{3a}$$

$$\text{s.t. } G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}') \text{ contains no negative directed cycle,} \tag{3b}$$

$$-p_{ij}^l + l_{ij} \le \hat{l}_{ij} \le \min\{x_{ij}^0, q_{ij}^l + l_{ij}\}, \quad \forall (i,j) \in A, \tag{3c}$$

$$\max\{x_{ij}^0, -p_{ij}^u + u_{ij}\} \le \hat{u}_{ij} \le q_{ij}^u + u_{ij}, \quad \forall (i,j) \in A, \tag{3d}$$

where $G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}')$ is the residual network of $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$.

3. The Bottleneck-type Case

In this section, we consider problem (2) and propose an efficient algorithm for solving it.

Lemma 3.1. If problem (2) is feasible, then it has an optimal solution.

Proof. Due to the fact that the objective value, z, of problem (2) belongs to the finite set $\{w_{ij}^l: (i,j) \in A\} \cup \{w_{ij}^u: (i,j) \in A\} \cup \{0\}$, the result is immediate. □

Proposition 3.2. Problem (2) is infeasible if and only if there is a negative cycle C in the residual network of $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$ so that the following conditions hold:

(a) $u_{ij} - p_{ij}^u > x_{ij}^0$, for each $(i,j) \in C \cap A'_U$.

(b) $l_{ij} + q_{ij}^l < x_{ij}^0$, for each $(j,i) \in C \cap A'_L$.

Proof. We first prove the necessity by contradiction. Suppose that each negative cycle C in the residual network does not satisfy at least one of the two conditions (a) and (b). We show that C can be removed from the residual network by adjusting the bound vectors. If there exists an arc $(i_0, j_0) \in C \cap A'_U$ with $u_{i_0 j_0} - p_{i_0 j_0}^u \le x_{i_0 j_0}^0$, then we set $\hat{u}_{i_0 j_0} = x_{i_0 j_0}^0$. Thus, the bound constraint $\max\{x_{i_0 j_0}^0, u_{i_0 j_0} - p_{i_0 j_0}^u\} \le \hat{u}_{i_0 j_0} \le u_{i_0 j_0} + q_{i_0 j_0}^u$ holds and the negative cycle is removed from the residual network. In the case that C contains an arc $(j_0, i_0) \in C \cap A'_L$ with $l_{i_0 j_0} + q_{i_0 j_0}^l \ge x_{i_0 j_0}^0$, we set $\hat{l}_{i_0 j_0} = x_{i_0 j_0}^0$ to remove the cycle C from the residual network while honoring the bound constraint

$-p^l_{i_0j_0} + l_{i_0j_0} \le \hat{l}_{i_0j_0} \le \min\{x^0_{i_0j_0}, q^l_{i_0j_0} + l_{i_0j_0}\}$. By repeating this process, each negative cycle can be removed from the residual network and consequently, a feasible solution $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is obtained.

We now prove the sufficiency. Suppose that there is a negative cycle C satisfying the conditions (a) and (b). Since $\hat{u}_{ij} \ge u_{ij} - p^u_{ij} > x^0_{ij}$, for each $(i,j) \in C \cap A'_U$ and $\hat{l}_{ij} \le l_{ij} + q^l_{ij} < x^0_{ij}$, for each $(j,i) \in C \cap A'_L$, we cannot remove C by modifying the initial bound vectors. Therefore, problem (2) is infeasible. □

The proof of Proposition 3.2 is constructive and can be used as an algorithm for solving problem (2). Let us describe this algorithm in more details. The algorithm first sets $\hat{\boldsymbol{l}} = \boldsymbol{l}$ and $\hat{\boldsymbol{u}} = \boldsymbol{u}$. In each iteration, the algorithm identifies a negative cycle C in the residual network of $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ and either removes it from the residual network or determines that the problem is infeasible. Suppose that the arc set S is defined as follows:

$$S = C \cap (A''_U \cup \{(i,j) \in A : (j,i) \in A''_L\})$$

where

$$A''_U = \{(i,j) \in A'_U : u_{ij} - p^u_{ij} \le x^0_{ij}\}, \tag{4a}$$

$$A''_L = \{(j,i) \in A'_L : x^0_{ij} \le l_{ij} + q^l_{ij}\}. \tag{4b}$$

The set S contains the arcs which can be modified for removing C. From Proposition 3.2, if $S = \emptyset$, then problem (2) is infeasible and the algorithm terminates. Otherwise, an arc $(i_0, j_0) \in S$ is selected with the minimum penalty,

$$(i_0, j_0) = \operatorname{argmin}\{\min_{(i,j)\in S\cap A''_U} w^u_{ij}, \min_{(i,j)\in S \text{ with } (j,i)\in A''_L} w^l_{ij}\}. \tag{5}$$

To remove the cycle C from the residual network, the algorithm sets $\hat{u}_{i_0j_0} = x^0_{i_0j_0}$ if $(i_0, j_0) \in A''_U$ and $\hat{l}_{i_0j_0} = x^0_{i_0j_0}$ if $(j_0, i_0) \in A''_L$. This process repeats until one of the following cases occurs: (I) the algorithm identifies a negative cycle C with $S = \emptyset$. In this case, the problem is infeasible, and (II) the current residual network contains no negative cycle. In this case, the solution $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is optimal for problem (2) due to (5).

We now discuss about the complexity of this algorithm. In each iteration, the algorithm adjusts the lower bound or the upper bound of one arc. This observation gives a bound of $2m$ on the number of iterations. In each iteration, we can use the FIFO label-correcting algorithm as a subroutine to detect the presence of a negative cycle in $O(mn)$ time [1]. Therefore, problem (2) can be solved in $O(m^2 n)$ time. In the next subsection, we present a faster algorithm for solving problem (2).

Now, we introduce a special form of the optimal solutions and focus on finding such an optimal solution.

Proposition 3.3. If problem (2) is feasible, then it has an optimal solution $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ having the following properties:

(a) *If* $\hat{u}_{ij} \ne u_{ij}$, for some $(i,j) \in A''_U$, then $\hat{u}_{ij} = x^0_{ij}$.
(b) $\hat{u}_{ij} = u_{ij}$, for each $(i,j) \in A'_U \backslash A''_U$.
(c) *If* $\hat{l}_{ij} \ne l_{ij}$, for some $(j,i) \in A''_L$, then $\hat{l}_{ij} = x^0_{ij}$.
(d) $\hat{l}_{ij} = l_{ij}$, for each $(j,i) \in A'_L \backslash A''_L$.

Proof. First, we prove the part (a) by contradiction. Since problem (2) is feasible, based on Lemma 3.1, there exists an optimal solution. Suppose that $(\bar{\boldsymbol{l}}, \bar{\boldsymbol{u}})$ is an optimal solution for problem (2) so that $\bar{u}_{i_0 j_0} \neq x^0_{i_0 j_0}$ and $\bar{u}_{i_0 j_0} \neq u_{i_0 j_0}$, for some arc $(i_0, j_0) \in A''_U$. Now, we define the upper bound vector $\hat{\boldsymbol{u}}$ as follows:

$$\hat{u}_{ij} = \begin{cases} x^0_{ij}, & (i,j) = (i_0, j_0), \\ \bar{u}_{ij}, & (i,j) \neq (i_0, j_0), \end{cases} \quad \forall (i,j) \in A.$$

It is obvious that the residual network of $G(V, A, \bar{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ contains exactly the arcs of the residual network of $G(V, A, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \boldsymbol{c})$, except the arc (i_0, j_0). This together with the fact that the residual network of $G(V, A, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \boldsymbol{c})$ does not contain any negative cycle implies that the residual network of $G(V, A, \bar{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ also contains no negative cycle. Therefore, the solution $(\bar{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is feasible to problem (2). On the other hand, the objective values of both solutions are the same, because $\bar{u}_{ij} \neq u_{ij}$ if and only if $\hat{u}_{ij} \neq u_{ij}$, for each $(i,j) \in A$. Consequently, the solution $(\bar{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is also optimal to problem (2). If we repeat this process for each $(i,j) \in A''_U$ with $\bar{u}_{ij} \neq u_{ij}$, then we can obtain an optimal solution that satisfies part (a).

Now, we prove part (b) by contradiction. Suppose that $(\bar{\boldsymbol{l}}, \bar{\boldsymbol{u}})$ is an optimal solution for problem (2) so that $\bar{u}_{i_0 j_0} \neq u_{i_0 j_0}$, for some arc $(i_0, j_0) \in A'_U \backslash A''_U$. Define the upper bound vector $\hat{\boldsymbol{u}}$ as follows:

$$\hat{u}_{ij} = \begin{cases} u_{ij}\,, & (i,j) = (i_0, j_0), \\ \bar{u}_{ij}\,, & (i,j) \neq (i_0, j_0), \end{cases} \quad \forall (i,j) \in A.$$

Since both the residual networks of $G(V, A, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \boldsymbol{c})$ and $G(V, A, \bar{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ are the same, it is easy to verify that $(\bar{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is a feasible solution with the objective value less than or equal to that of $(\bar{\boldsymbol{l}}, \bar{\boldsymbol{u}})$. Hence, $(\bar{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is optimal for problem (2). By repeating this process, we can obtain an optimal solution satisfying part (b). Similarly, parts (c) and (d) can be proved. □

Using Proposition 3.3, we can reduce problem (2) to the following combinatorial optimization problem:

$$\begin{aligned} &\min z = \max\left\{ \max_{(i,j) \in A'_L} w^l_{ij} H(l_{ij}, \hat{l}_{ij}), \max_{(i,j) \in A''_U} w^u_{ij} H(u_{ij}, \hat{u}_{ij}) \right\}, \\ &\text{s.t. The residual network of } G(V, A, \hat{\boldsymbol{l}}, \hat{\mathbf{u}}, \boldsymbol{c}) \text{ contains no negative directed cycle,} \\ &\hat{l}_{ij} = \begin{cases} l_{ij}, & (j,i) \in A'_L \backslash A''_L, \\ l_{ij} \text{ or } x^0_{ij}, & (j,i) \in A''_L, \end{cases} \quad \forall (i,j) \in A, \\ &\hat{u}_{ij} = \begin{cases} u_{ij}, & (i,j) \in A'_U \backslash A''_U, \\ u_{ij} \text{ or } x^0_{ij}, & (i,j) \in A''_U, \end{cases} \quad \forall (i,j) \in A. \end{aligned} \tag{6}$$

Remark 1. In the CaIMCF problem under an arbitrary distance, we have to either increase a lower bound from l_{ij} to x^0_{ij} or decrease an upper bound from u_{ij} to x^0_{ij} to remove negative cycles from the residual network. For this reason, the results of propositions 3.2 and 3.3 are also valid for any other distances, especially for l_∞ norm. Therefore, the CaIMCF problem under the l_∞ norm is reduced to problem (6) with $w^l_{ij} = x^0_{ij} - l_{ij}$ and $w^u_{ij} = u_{ij} - x^0_{ij}$, for each $(i,j) \in A$.

3.1. An efficient algorithm

Here, we propose an algorithm based on binary search technique to solve problem (6). Assume that F is the set of objective functions of problem (6), i.e., $F = \{0\} \cup \{(w_{ij}^u : (i,j) \in A\} \cup \{w_{ij}^l : (i,j) \in A\}$. Suppose that we have sorted the elements of F in nondecreasing order: let $0 = w_0 \le w_1 \le w_2 \le \cdots \le w_f$ be the sorted list where $f = |F|$. For each fixed index $k \in \{0,1,\dots,f\}$, the solution $(\hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)})$ of problem (6) is defined as follows:

$$\hat{l}_{ij}^{(k)} = \begin{cases} x_{ij}^0, & (j,i) \in A_L'' \text{ with } w_{ij}^l \le w_k, \\ l_{ij}, & \text{otherwise}, \end{cases} \quad \forall (i,j) \in A, \tag{7a}$$

$$\hat{u}_{ij}^{(k)} = \begin{cases} x_{ij}^0, & (i,j) \in A_U'' \text{ with } w_{ij}^U \le w_k, \\ u_{ij}, & \text{otherwise}, \end{cases} \quad \forall (i,j) \in A, \tag{7b}$$

where A_L'' and A_U'' are defined by (4). The following result is on the feasibility of such solutions.

Theorem 3.4. For $k \in \{0,1,\dots,f\}$, let the solution $(\hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)})$ be defined by (7). The residual network of $G(V, A, \hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)}, \boldsymbol{c})$, denoted by $G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}')$, satisfies the following two properties:

(a) If $G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}')$ contains no negative cycle, then the solution $(\hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)})$ is feasible for problem (6) with an objective value less than or equal to w_k.
(b) If $G'(V, A', \hat{\boldsymbol{u}}, \boldsymbol{c}')$ contains a negative cycle, then problem (6) has no feasible solution with an objective value less than or equal to w_k.

Proof. The proof of the part (a) is obvious. We only prove part (b) by contradiction. Assume that problem (6) has a feasible solution $(\bar{\boldsymbol{l}}, \bar{\boldsymbol{u}})$, whose objective value is less than or equal to w_k. Let the residual network of $G(V, A, \bar{l}, \bar{\boldsymbol{u}}, \boldsymbol{c})$ be denoted by $G(V, \bar{A}', \bar{\boldsymbol{u}}', \boldsymbol{c}')$. Suppose $\hat{A}_U' = A' \cap A$ and $\hat{A}_L' = A' \backslash A$. Let $\hat{A}_L''$ and $\hat{A}_U''$ be defined by (4) corresponding to $\hat{A}_L'$ and $\hat{A}_U'$, respectively. Similarly, suppose $\bar{A}_U' = \bar{A}' \cap A$ and $\bar{A}_L' = \bar{A}' \backslash A$ and also, $\bar{A}_L''$ and $\bar{A}_U''$ are defined by (4) corresponding to $\bar{A}_L'$ and $\bar{A}_U'$, respectively.
Let (i,j) be an arbitrary element of $\hat{A}_U'$. Then, $\hat{u}_{ij}^{(k)} \ne x_{ij}^0$. Consequently, either $w_{ij}^u > w_k$ or $u_{ij} - p_{ij}^u > x_{ij}^0$. If $w_{ij}^u > w_k$, then $\bar{u}_{ij} = u_{ij} \ne x_{ij}^0$, because $(\bar{\boldsymbol{l}}, \bar{\boldsymbol{u}})$ is a feasible solution with objective value less than or equal to w_k. If $u_{ij} - p_{ij}^u > x_{ij}^0$, then $\bar{u}_{ij} \ne x_{ij}^0$, because of honoring the bound constraint $\max\{x_{ij}^0, -p_{ij}^u + u_{ij}\} \le \bar{u}_{ij}$. Hence, (i,j) belongs to $\bar{A}_U'$ in both cases. This shows that $\hat{A}_U' \subseteq \bar{A}_U'$. Similarly, it can be verified that $\hat{A}_L' \subseteq \bar{A}_L'$. Obviously, these relations imply

$$\hat{A}_L'' \subseteq \bar{A}_L'', \quad \hat{A}_U'' \subseteq \bar{A}_U''. \tag{8}$$

Due to the fact that the residual network of $G(V, A, \bar{\boldsymbol{l}}, \bar{\boldsymbol{u}}, \boldsymbol{c})$ does not contains any negative cycle, the relations (8) imply that $G'(V, A', \hat{\boldsymbol{u}}', \boldsymbol{c}')$ also contains no negative cycle, which is a contradiction. □

Using Theorem 3.4, the following results are immediate.

Corollary 3.5. Problem (6) is infeasible if and only if the residual network of $G(V, A, \hat{\boldsymbol{l}}^{(f)}, \hat{\boldsymbol{u}}^{(f)}, \boldsymbol{c})$ contains at least one negative cycle, where $(\hat{\boldsymbol{l}}^{(f)}, \hat{\boldsymbol{u}}^{(f)})$ is defined by (7) for $k = f$.

Corollary 3.6. If $k \in \{0,1,\dots,f\}$ be the least index for which the residual network of $G(V,A,\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)},\boldsymbol{c})$ contains no negative cycle, then the solution of $(\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)})$ defined by (7) is optimal for problem (6).

Based on Corollary 3.6, solving problem (6) reduces to finding the least index k for which the residual network of $G(V,A,\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)},\boldsymbol{c})$ does not contain any negative cycle. We use the binary search technique to find such an index $k \in \{0,1,\dots,f\}$.

Our proposed algorithm for solving the problem (6) contains two phases. The first phase determines whether or not the problem (6) is feasible by checking whether the residual network of $G(V,A,\hat{\boldsymbol{l}}^{(f)},\hat{\boldsymbol{u}}^{(f)},\boldsymbol{c})$ contains at least a negative cycle (see Corollary 3.5). If the residual network contains no negative cycle, then problem (6) has the feasible solution $(\hat{\boldsymbol{l}}^{(f)},\hat{\boldsymbol{u}}^{(f)})$ and the second phase starts. Otherwise, problem (6) is infeasible and the algorithm terminates. The second phase has a repetitive process based on the binary search technique. At each iteration of the second phase, the algorithm looks for feasible solutions whose objective value is better than those of feasible solutions found in previous iterations.

Suppose that the second phase has started. The goal in the second phase is to find the least index $k \in \{0,1,\dots,f\}$ by using the binary search technique so that $(\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)})$ is a feasible solution for problem (6). Since the algorithm has determined that $(\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)})$ is feasible for $k = f$ in the first phase, the algorithm checks whether or not the residual network of $G(V,A,\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)},\boldsymbol{c})$ has a negative cycle for $k = [\frac{f}{2}]$ and $s = [\frac{f}{4}]$ in the first iteration of the second phase. Note that s is a half-step of k to decrease or to increase k in each iteration. If the residual network does not contain any negative cycle, then $(\hat{\boldsymbol{l}}^{(k)},\hat{\boldsymbol{u}}^{(k)})$ is feasible for problem (6) with the objective value w_k and the algorithm decreases the value of k by s units for finding a feasible solution with the objective value less than w_k. Otherwise, the value of k increases by s units, because problem (1) has no feasible solution with the objective value less than or equal to w_k (see Theorem 3.4). The algorithm repeats this process and updates $s = [\frac{s}{2}]$ at each iteration until $s = 0$.

Now, we are ready to state the proposed algorithm formally.

Algorithm 1.

Input: A network $G(V,A,\boldsymbol{l},\boldsymbol{u},\boldsymbol{c})$, a feasible solution $\boldsymbol{x}^0$, penalty vectors $\boldsymbol{w}^l$ and $\boldsymbol{w}^u$ and bound vectors $\boldsymbol{p}^l,\boldsymbol{q}^l,\boldsymbol{p}^u$ and $\boldsymbol{q}^u$.

Step 1: Sort the objective values of problem (6). Suppose that $0 = w_0 \le w_1 \le \cdots \le w_f$ is the sorted list of these values.

Step 2: Using (7), obtain the vectors $\hat{\boldsymbol{l}}^{(f)}$ and $\hat{\boldsymbol{u}}^{(f)}$. Construct the residual network of $G(V,A,\hat{\boldsymbol{l}}^{(f)},\hat{\boldsymbol{u}}^{(f)},\boldsymbol{c})$ with respect to $\boldsymbol{x}^0$. If the residual network contains at least one negative cycle, then stop because problem (6) is infeasible; otherwise, go to Step 3 (see Corollary 3.5).

Step 3: Set $s = [\frac{f}{4}]$, $k = [\frac{f}{2}]$, $z^* = w_f$ and $(\boldsymbol{l}^*,\boldsymbol{u}^*) = (\hat{\boldsymbol{l}}^{(f)},\hat{\boldsymbol{u}}^{(f)})$.

Step 4: Using (7), set the vectors $\hat{\boldsymbol{l}}^{(k)}$ and $\hat{\boldsymbol{u}}^{(k)}$ and construct the residual network of $G(V, A, \hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)}, \boldsymbol{c})$ with respect to $\boldsymbol{x}^0$. If the residual network contains no negative cycle, then set $z^* = w_k$, $(\boldsymbol{l}^*, \boldsymbol{u}^*) = (\hat{\boldsymbol{l}}^{(k)}, \hat{\boldsymbol{u}}^{(k)})$ and $k = k - s$; otherwise, set $k = k + s$ (see Corollary 3.6).

Step 5: If $s = 0$ then go to Step 6, otherwise update $s = [\frac{s}{2}]$ and go to Step 3.

Step 6 (output): Stop. $(\boldsymbol{l}^*, \boldsymbol{u}^*)$ is an optimal solution to problem (6) with the objective value z^*.

In Algorithm 1, $(\boldsymbol{l}^*, \boldsymbol{u}^*)$ maintains the last feasible solution found by the algorithm and z^* is its objective value.

Proposition 3.7. Algorithm 1 solves problem (6) in $O(mn \log n)$ time.

Proof. Since the number of iterations is $O(\log f) = O(\log n)$ and a negative cycle can be identified in $O(mn)$ time by using the FIFO label-correcting algorithm [1], the result is immediate. □

Remark 2. Based on Remark 1, Algorithm 1 can solve the CaIMCF problem under the l_∞ norm if we initialize $w_{ij}^l = x_{ij}^0 - l_{ij}$ and $w_{ij}^u = u_{ij} - x_{ij}^0$, for each $(i, j) \in A$.

Remark 3. As the vectors $\boldsymbol{p}^l$ and $\boldsymbol{q}^u$ are not being used in definition (4), we can remove them from the inputs of Algorithm 1.

4. The Sum-type Case

In this section, we consider problem (3) and show that it is strongly NP-hard even on bipartite networks by a reduction from the minimum node cover problem. Since the feasible sets of both problems (2) and (3) are the same, Proposition 3.3 is also valid for problem (3). Hence, we can reduce problem (3) to the following problem:

$$\begin{aligned} &\min z = \sum_{(j,i)\in A_L''} w_{ij}^l H\big(l_{ij}, \hat{l}_{ij}\big) + \sum_{(i,j)\in A_U''} w_{ij}^u H\big(u_{ij}, \hat{u}_{ij}\big) \\ &\text{s.t. the residual network of } G\big(V, A, \hat{\boldsymbol{l}}, \hat{\mathbf{u}}, \boldsymbol{c}\big) \text{ contains no negative directed cycle,} \\ &\hat{l}_{ij} = \begin{cases} l_{ij}, & (j,i) \in A_L' \backslash A_L'', \\ l_{ij} \text{ or } x_{ij}^0, & (j,i) \in A_L'', \end{cases} \quad \forall (i,j) \in A, \\ &\hat{u}_{ij} = \begin{cases} u_{ij}, & (i,j) \in A_U' \backslash A_U'', \\ u_{ij} \text{ or } x_{ij}^0, & (i,j) \in A_U'', \end{cases} \quad \forall (i,j) \in A, \end{aligned} \tag{9}$$

where A_L'' and A_U'' are defined as in (4).

The minimum node cover problem: Given an undirected graph $\bar{G}(\bar{V}, \bar{A})$, the minimum node cover problem is to find a set of nodes such that each arc of the graph is incident to at least one node of the set. This problem can be formulated as follows:

$$\begin{aligned} \min z = & \sum_{i \in \bar{V}} x_i \\ \text{s.t. } & x_i + x_j \geq 1, \quad \forall (i,j) \in \bar{A}, \\ & x_i \in \{0,1\}, \quad \forall i \in \bar{V}. \end{aligned} \tag{10}$$

The decision version of problem (10) is one of the 21 NP-complete problems introduced by Karp [13]. Let us state formally the decision versions of problems (9) and (10).

The Minimum Node Cover Decision (MNCD) problem:
Instance: An undirected graph $\bar{G}(\bar{V}, \bar{A})$ and a positive number k.
Question: Is there a set $S \subseteq \bar{V}$ so that $|S| \leq k$ and each arc of the graph is incident to at least one node of *S*?

The Capacity Inverse Minimum Cost Flow Decision (CIMCFD) problem:
Instance: An instance of problem (1) defined on the directed network $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$, a feasible solution $\boldsymbol{x}^0$ for problem (1), penalty vectors $\boldsymbol{w}^l$ and $\boldsymbol{w}^u$, bound vectors $\boldsymbol{p}^l, \boldsymbol{q}^l, \boldsymbol{p}^u$ and $\boldsymbol{q}^u$ and a number $k' > 0$.
Question: Is there a feasible solution $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ to problem (9) so that

$$\sum_{(j,i) \in A_L''} w_{ij}^l H\left(l_{ij}, \hat{l}_{ij}\right) + \sum_{(i,j) \in A_U''} w_{ij}^u H\left(u_{ij}, \hat{u}_{ij}\right) \leq k'? \tag{11}$$

Theorem 4.1. Problem (9) is strongly NP-hard even for bipartite networks.

Proof. The result is proved by a polynomial-time many-one reduction from the MNCD problem to the CIMCFD problem. Suppose that an instance of the MNCD problem defined on an undirected graph $\bar{G}(\bar{V}, \bar{A})$ is given, where $V = \{1,2,\dots,n\}$ is the node set and $\bar{A}$ is the arc set. We introduce a bipartite directed network $G(V, A, \boldsymbol{l}, \boldsymbol{u}, \boldsymbol{c})$ as follows:

- The node set V contains two nodes i and i', for each $i \in \bar{V}$. Using the notation $\bar{V}' = \{1', 2', \dots, n'\}$, $V = \bar{V} \cup \bar{V}'$.

- For each $(i,j) \in \bar{A}$, we add two arcs (i,j') and (j,i') to G which are called the natural arcs. We associate with each $i \in \bar{V}$, one arc $(i,i') \in A$. Such arcs are referred to as the artificial arcs. Thus, A is the union of the artificial arc set $\{(i,i') : i \in \bar{V}\}$ and the natural arc set $\{(i,j'), (j,i') : (i,j) \in \bar{A}\}$. Note that the network G is bipartite and all its arcs are oriented from $\bar{V}$ to $\bar{V}'$.

- The cost of each artificial arc (i,i') is equal to 1 and the cost of each natural arc is equal to zero.

- The lower and the upper bounds of all arcs are respectively zero and one.
- Each node $i \in \bar{V}$ has a supply equal to 1 and each node $i' \in \bar{V}'$ has a demand equal to -1.

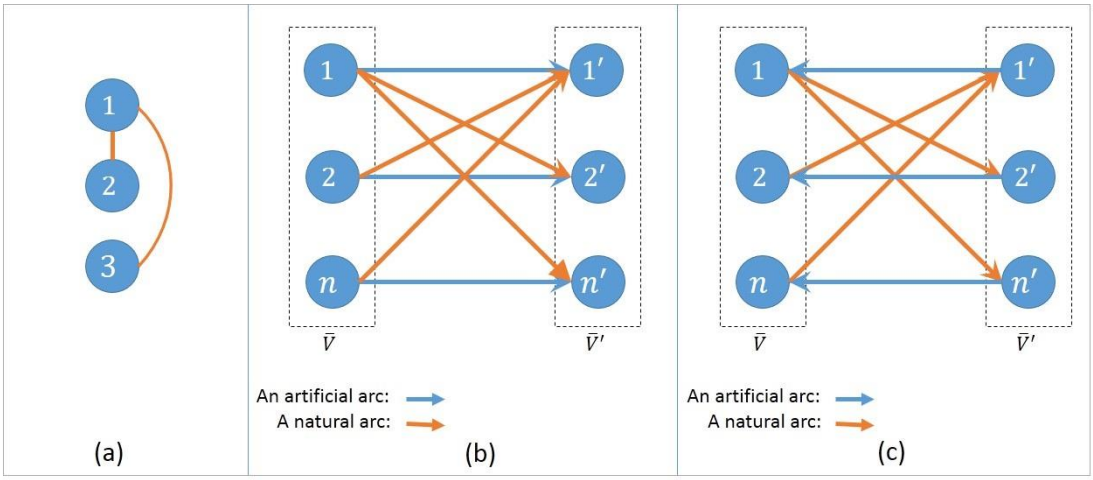

Figure 1: (a) An undirected graph $\bar{G}(\bar{V}, \bar{A})$, (b) the bipartite network G constructed from $\bar{G}$, (c) the residual network of G with respect to $\boldsymbol{x}^0$

Consider the feasible solution $\boldsymbol{x}^0$ defined as follows: $x^0_{ii'} = 1$ for each artificial arc (i, i'), and $x^0_{ij'} = 0$, for each natural arc (i, j'). The residual network of G with respect to $\boldsymbol{x}^0$ has the same nodes of G and the same arcs of G except that the artificial arcs are oriented from $\bar{V}'$ to $\bar{V}$. Then, A'_U is the same natural arc set and $A'_L = \{(i', i): i \in \bar{V}\}$. It is obvious that $c'_{i'i} = -1$, for each $(i', i) \in A'_L$, and $c'_{ij'} = 0$, for each $(i, j') \in A'_U$. Hence, each directed cycle of the residual network has a negative cost because it contains at least two arcs belonging to A'_L. A simple example is presented in Fig. 1 to illustrate how to construct the bipartite network G from $\bar{G}$.

Now, we introduce the parameters of the CIMCFD problem defined on G with respect to the initial feasible solution $\boldsymbol{x}^0$. All the penalties are equal to 1, i.e., $w^l_{ij'} = w^u_{ij'} = 1$, for each $(i, j') \in A$. We set $p^u_{ij'} = q^u_{ij'} = p^l_{ij'} = 0$ and $q^l_{ij'} = 1$, for each $(i, j') \in A$ and also, $k' = k$. Therefore, $A''_L = A'_L$ and $A''_U = \emptyset$.

To establish the result, it is sufficient to prove the following claim, because the introduced instance of the CIMCFD problem satisfies the similarity assumption, i.e., the data of the instance are polynomially bounded with respect to the problem size [1].

Claim. An instance of the MNCD problem is a yes instance if and only if the corresponding instance of the CIMCFD problem is a yes instance.

Necessity: Suppose that S is a solution to a given yes instance of the MNCD problem. Consider the following solution $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ to the corresponding instance of the CIMCFD problem:

$$\begin{aligned}
&\hat{u}_{ij'} = 1 \ (= u_{ij'}), && \text{for each arc } (i, j') \in A,\\
&\hat{l}_{ij'} = 0 \ (= l_{ij'}), && \text{for each natural arc } (i, j'),\\
&\hat{l}_{ii'} = 0 \ (= l_{ii'}), && \text{for each artificial arc } (i, i') \text{ with } i \notin S,\\
&\hat{l}_{ii'} = 1 \ (\neq l_{ii'}), && \text{for each artificial arc } (i, i') \text{ with } i \in S.
\end{aligned}$$

We prove that the residual network of $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ with respect to $\boldsymbol{x}^0$ contains no (negative) directed cycle and also, the relation (11) is satisfied. By contradiction, assume that C is a directed

cycle of the corresponding residual network. C contains at least one natural arc (i,j') together with two artificial arcs (i',i) and (j',j). Since S is a node cover and the arc (i,j') corresponds to the arc $(i,j) \in \bar{A}$, it follows that either $i \in S$ or $j \in S$. Without loss of generality, we assume that $i \in S$. Consequently, $\hat{l}_{ii'} = 1 (= x^0_{ii'})$ and the arc (i,i') cannot belong to the residual network which leads to a contradiction. On the other hand, we have

$$\sum_{(j,i)\in A''_L} w^l_{ij} H(l_{ij}, \hat{l}_{ij}) + \sum_{(i,j)\in A''_U} w^u_{ij} H(u_{ij}, \hat{u}_{ij}) = \sum_{(j,i)\in A''_L} H(0, \hat{l}_{ij}) + \sum_{(i,j)\in A''_U} H(1,1) = \sum_{(i',i)\in A'_L \text{ with } i\in S} H(0,1) = |S|, \tag{12}$$

which guarantees (11). This completes the proof for necessity.

Sufficiency: Suppose that $(\hat{\boldsymbol{l}}, \hat{\boldsymbol{u}})$ is a solution to a given yes instance of the CIMCFD problem. The relation $A''_U = \emptyset$ implies that $\hat{\boldsymbol{u}} = \boldsymbol{u}$. Assume that S is a set of nodes so that $i \in S$ if and only if $\hat{l}_{ii'} = 1$, for each $(i',i) \in A''_L$. We show that S is a node cover of $\bar{G}$. Each arc $(i,j) \in \bar{A}$ corresponds to the cycle $i - j' - j - i' - i$ of G. The fact that the residual network of $G(V, A, \hat{\boldsymbol{l}}, \hat{\boldsymbol{u}}, \boldsymbol{c})$ with respect to $\boldsymbol{x}^0$ contains no cycle implies that either $\hat{l}_{ii'} = 1$ or $\hat{l}_{jj'} = 1$. Equivalently, either $i \in S$ or $j \in S$. This shows that S is a node cover of $\bar{G}$. By the definition of S, (12) holds. Thus, $|S| \leq k$. This completes the proof of sufficiency. □

5. Comparison of results

Güler and Hamacher [10] studied the capacity inverse minimum cost flow problem. Here, we compare our considered problems and our results with the ones in [10].

The main difference between our studied problems and the ones in [10] is in the objective functions. In [10], authors have used the l_1 and l_∞ norms, while we apply the Hamming distances. The Hamming distances, unlike to the l_1 and l_∞ norms, are nonconvex and discontinuous. Therefore, the known methods for the l_1 and l_∞ norms cannot usually apply to solve the inverse problems under the Hamming distance. For example, the inverse minimum cost flow problem under the l_1 or l_∞ norms is pollynomially solvable while problem under the sum-type hamming distance is APX-hard [3,12]. The second difference is that modifying the lower bound vector is not allowed in [10] while it is allowed for our problems.

In spite of these differences, it is remarkable that Algorithm 1 can solve the inverse problem under the l_∞ norm (see remarks 1 and 2). This improves the running time of the algorithm presented in [10] from $O(nm^2)$ to $O(nm \log n)$. The main difference between these algorithms is that ours is based on the binary search technique but the one in [10] uses a greedy algorithm.

In the sum-type case, we showed that the problem is NP-hard due to the minimum node cover problem. In [10], the authors showed that the problem under the l_1 norm is also NP-hard using a reduction from the weighted feedback arc set problem. It is remarkable that their reduction is also valid for the problem under the sum-type Hamming distance and it proves that the problem is

weakly NP-hard in a general network, while our presented reduction shows that the problem is strongly NP-hard even on bipartite networks.

6. Conclusion and Future Research

We studied the capacity inverse minimum cost flow problem under the weighted Hamming distances. Both the bottleneck-type and the sum-type cases were considered. In the bottleneck-type case, an algorithm based on the binary search technique was proposed to solve the problem in $O(mn \log n)$ time. This algorithm can be applied to solve the capacity inverse minimum cost flow problem under the l_∞ norm. Furthermore, it has a better complexity than one presented in [10]. In the sum-type case, we showed that the problem is strongly NP-hard even on bipartite networks. As the capacity inverse minimum cost flow problem under the sum-type Hamming distance is NP hard, it is worthwhile to design efficient algorithms for some special cases of the problem and propose heuristic (approximation) algorithms for obtaining satisfactory solutions of the problem.

Acknowledgements

The authors thank the anonymous referees whose valuable comments lead to an improved presentation of our work.

References

[1] Ahuja, R.K., Magnanti, T.L. and Orlin, J.B. (1993), Network Flows: Theory, Algorithms, and Applications, Englewood Cliffs, Prentice-Hall International, New Jersey.
[2] Ahuja, R.K. and Orlin, J.B. (2001), Inverse optimization, *Operation Research*, 92, 771-783.
[3] Ahuja, R.K. and Orlin J.B. (2002), Combinatorial algorithms for inverse network flow problems, *Networks*, 40, 181-187.
[4] Burton, D. and Toint, Ph.L. (1992), On an instance of the inverse shortest paths problem, *Mathematical Programming*, 53, 45-61.
[5] Burton, D. and Toint, Ph.L. (1994), On the use of an inverse shortest paths algorithm for recovering linearly correlated costs, *Mathematical Programming*, 63, 1-22.
[6] Call, M. (2010), Inverse Shortest Path Routing Problems in the Design of IP Networks, Ph.D. Thesis, Linkping University, Sweden.
[7] Demange, M. and Monnot, J. (2010), An introduction to inverse combinatorial problems, In: Paschos V. Paradigms of Combinatorial Optimization (Problems and New Approaches), London-Hoboken (UK-USA), Wiley, pp. 547-586.
[8] Duin, C.W. and Volgenant, A. (2006), Some inverse optimization problems under the Hamming distance, *European Journal of Operational Research*, 170, 887-899.
[9] Gódor, I., Harmatos, J. and Jüttner, A. (2005), Inverse shortest path algorithms in protected UMTS access networks, *Computer Communications*, 28, 765-772.
[10] Güler, C. and Hamacher, H.W. (2010), Capacity inverse minimum cost flow problem, *Journal of Combinatorial Optimization*, 19, 43-59.
[11] Heuberger, C. (2004), Inverse optimization: a survey on problems, methods, and results, *Journal of Combinatorial Optimization*, 8, 329-361.
[12] Jiang, Y., Liu, L., Wuc, B. and Yao, E. (2010), Inverse minimum cost flow problems under the weighted Hamming distance, *European Journal of Operational Research*, 207, 50-54.

[13] Karp, R.M. (1972), Reducibility Among Combinatorial Problems, Complexity of Computer Computations, New York, Plenum Press, 85-103.
[14] Tarantola, A. (1987), Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation, Amsterdam, Elsevier.
[15] Tayyebi, J. and Aman, M. (2014), Note on Inverse minimum cost flow problems under the weighted Hamming distance, *European Journal of Operational Research*, 234, 916-920.
[16] Tayyebi, J. and Aman, M. (2015), On inverse linear programming problems under the bottleneck-type weighted Hamming distance, *Discrete Applied Mathematics*, doi:10.1016/j.dam.2015.12.017.
[17] Zhang, J. and Liu, Z. (1996), Calculating some inverse linear programming problems, *Journal of Computational and Applied Mathematics*, 72, 261-273.