Sādhanā Vol. 38, Part 2, April 2013, pp. 199–209. © Indian Academy of Sciences

Inverse feasibility problems of the inverse maximum flow problems

ADRIAN DEACONU* and ELEONOR CIUREA

Department of Mathematics and Computer Science, Faculty of Mathematics and Informatics, Transilvania University of Braşov, Braşov, Iuliu Maniu st. 50, Romania
e-mail: a.deaconu@unitbv.ro; e.ciurea@unitbv.ro

MS received 19 February 2010; revised 11 March 2011; accepted 26 February 2013

Abstract. A linear time method to decide if any inverse maximum flow (denoted General Inverse Maximum Flow problems (IMFG)) problem has solution is deduced. If IMFG does not have solution, methods to transform IMFG into a feasible problem are presented. The methods consist of modifying as little as possible the restrictions to the variation of the bounds of the flow. New inverse combinatorial optimization problems are introduced and solved.

Keywords. Inverse combinatorial optimization; maximum flow; minimum cut.

1. Introduction

An inverse combinatorial 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 minimum. Different norms such as l_1, l_∞ and even l_2 are considered to measure this distance. Many papers were published in the field of inverse combinatorial optimization (Heuberger 2004). Almost every inverse problem was studied considering l_1 and l_∞ norms, resulting in different problems with completely different solution methods. A strongly polynomial time algorithm to solve the inverse maximum flow problem under l_1 norm (denoted IMF) is presented by Yang *et al* (1997). IMF is reduced to a minimum cut problem in an auxiliary network with finite and infinite arc capacities. The algorithm for IMF has an $O(n\cdot m\cdot log(n^2/m))$ time complexity, where m is the number of arcs and n is the number of nodes. IMF can not be solved using weakly polynomial algorithms (although sometimes they can be preferred) because the minimum cut is searched in a network with some arcs having infinite capacities.

The more general inverse maximum flow problem (denoted GIMF) under l_1 norm, where the lower and upper bounds for the flow are changed, is studied by Deaconu (2008b). Strongly and

*For correspondence

weakly polynomial algorithms to solve GIMF are proposed. The strongly polynomial algorithms for GIMF have the same time complexity as the algorithm for IMF, but the minimum cut is searched in a network with fewer arcs. The weakly polynomial algorithms for GIMF have an $O(min\{n^{2/3}, m^{1/2}\} \cdot m \cdot log(n^2/m) \cdot log(max\{n, R\}))$ time complexity, where $R = max\{c(i, j) - x(i, j) + x(j, i) - l(j, i) | i, j \in N\}$.

Deaconu (2008a) considered the inverse maximum flow problem under l_∞ norm. A very fast $O(m \cdot log(n))$ time algorithm to solve this problem is presented.

The least number of modifications to the lower or/and upper bounds is considered by Deaconu (2006). An $O(min\{n^{2/3}, m^{1/2}\} \cdot m)$ time algorithm for solving this problem is presented.

Four inverse maximum flow problems are also studied by Liu & Zhang (2006) under the sum-type and bottleneck-type weighted Hamming distance. Strongly polynomial algorithms to solve these problems are proposed.

We introduce a new category of inverse problems. We shall call these problems as *inverse feasibility problems*. An inverse feasibility problem is defined on a non-feasible problem. It consists of modifying some parameters as little as possible of a network so that the initial non-feasible problem with the modified parameters becomes feasible. In this paper, we study the inverse feasibility problems of the inverse maximum flow problems.

2. The inverse maximum flow problems

Let $G = (N, A, l, c, s, t)$ be an s-t network, where N is the set of nodes, A is the set of directed arcs, l and c are the lower and, respectively, the upper bound vectors for the flow, s is the source and t is the sink node.

Let x be a given feasible flow in the network G. It means that x has to satisfy the flow balance condition and the capacity restrictions. The balance condition for the flow x is:

$$\sum_{j\in N,(i,j)\in A} x(i,j) - \sum_{j\in N,(j,i)\in A} x(j,i) = \begin{cases} v(x), & i = s \\ -v(x), & i = t \\ 0, & i \in N - \{s,t\} \end{cases}, \tag{1}$$

where $v(x)$ is the value of x from s to t.

The capacity restrictions are:

$$l(i,j) \le x(i,j) \le c(i,j),\ \forall\ (i,j) \in A, \tag{2}$$

where $c(i, j) \ge l(i, j) \ge 0$, for every arc $(i, j) \in A$.

The maximum flow problem is:

$$\begin{cases} \max\ v(x) \\ x \text{ is a feasible flow in } G. \end{cases} \tag{3}$$

We shall introduce now the definition of the minimum s-t cut in the network G. The set of arcs $[S, \bar{S}] = (S, \bar{S}) \cup (\bar{S}, S)$ is called an s-t cut in G if $S \cap \bar{S} = \phi$, $S \cup \bar{S} = N$, $s \in S$ and $t \in \bar{S}$, where $(S, \bar{S}) = \{(i, j) \in A | i \in S \text{ and } j \in \bar{S}\}$ is the set of direct arcs of the cut and $(\bar{S}, S) = \{(i, j) \in A | i \in \bar{S} \text{ and } j \in S\}$ is the set of the inverse arcs. The capacity of the s-t cut $[S, \bar{S}]$ in G is $c[S, \bar{S}] = c(S, \bar{S}) - l(\bar{S}, S) = \sum_{(i,j)\in(S,\bar{S})} c(i, j) - \sum_{(i,j)\in(\bar{S},S)} l(i, j)$. An s-t cut is a minimum cut in G if its capacity is minimal in the set of s-t cuts of the network G.

In order to introduce the definition of the residual network, by convention, if $(i, j) \in A$ and $(j, i) \notin A$ then we add the arc (j, i) to the set of arcs A and we set $l(j, i) = 0$ and $c(i, j) = 0$.

We define the residual capacity attached to the network A for the flow x as follows:

$$r(i, j) = c(i, j) - x(i, j) + x(j, i) - l(j, i), \forall (i, j) \in A. \tag{4}$$

The residual network attached to the network G for the flow x is $G_x = (N, A_x, r, s, t)$, where the set A_x contains the arcs (i, j) with positive residual capacity, i.e., $r(i, j) > 0$. In the residual network the lower bounds are all equal to 0.

An inverse maximum flow problem is to change as little as possible the bound vectors l and c so that the given feasible flow x becomes a maximum flow in G.

An inverse maximum flow problem (denoted IMFG) can be formulated using the following mathematical model:

$$\begin{cases} \texttt{min } dist((l, c), (\bar{l}, \bar{c})) \\ x \texttt{ is a maximum flow in } \bar{G} = \{N, A, \bar{l}, \bar{c}, s, t\} \\ \bar{l}(i, j) \leq \min\{\bar{c}(i, j), l(i, j) + \beta(i, j)\}, \ \forall \ (i, j) \in A \\ c(i, j) - \delta(i, j) \leq \bar{c}(i, j), \ \forall \ (i, j) \in A \end{cases}. \tag{5}$$

In the model (5) different formulas to measure the distance between (l, c) and $(\bar{l}, \bar{c})$ are considered, such as (weighted) l_1 norm, (weighted) l_∞ norm, the (weighted) sum-type or bottleneck-type Hamming distance, etc.

The values $\beta(i, j)$ and $\delta(i, j)$ are given non-negative numbers, where $\delta(i, j) \leq c(i, j)$, for each arc $(i, j) \in A$. These values show how much the bounds of the flow on the arcs can vary. In order to make the feasible flow x a maximum flow in the network G, the upper bounds of some arcs from A must be decreased and/or the lower bounds of some arcs from A must be increased. So, there is no need to impose upper limitations to the variation of $\bar{c}(i, j)$ and it is also useless to have lower limitations to the variation of $\bar{l}(i, j)$.

In the formula (5) by (l, c) is denoted by the vector obtained by adding the components of the vector c at the end of l. Similarly, $(\bar{l}, \bar{c})$ is the vector obtained by putting together the components of the vectors $\bar{l}$ and $\bar{c}$.

The inverse maximum flow problems with no lower bounds is a particular cases of IMFG. Indeed, in this case the lower bounds of the flow are equal to 0 and they can not be modified in order to transform the given flow x into a maximum flow and, so, for this particular case in (5) we can consider $\beta(i, j) = 0, \forall (i, j) \in A$.

3. The feasibility of IMFG

For solving IMFG we need to determine the arcs in the network G on which the lower and upper bounds will not be changed.

First, as it has been seen, changing the upper bound have no effect on an arc (i, j) with $c(i, j) > x(i, j) + \delta(i, j)$. So, there is no need to try changing the upper bounds of the arcs from the following set:

$$\widetilde{A}_1 = \{(i, j) \in A \,|x(i, j) + \delta(i, j) < c(i, j)\}. \tag{6}$$

There is also no need to change the lower bounds of the arcs from the following set:

$$\widetilde{A}_2 = \{(i, j) \in A \,|\, l(i, j) + \beta(i, j) < x(i, j)\}. \tag{7}$$

The graph $\widetilde{G} = (N, \widetilde{A})$ can be constructed to verify the feasibility of IMFG, where:

$$\widetilde{A} = \widetilde{A}_1 \cup \{(i, j) \in N \times N \,|\, (j, i) \in A \texttt{ and } x(j, i) > l(j, i) + \beta(j, i)\}. \tag{8}$$

Theorem 1 (feasibility of IMFG): *In the network G, IMFG has optimum solution for the given flow x, if and only if there is no directed path in the graph $\widetilde{G}$ from the source node s to the sink node t.*

Proof. Let $G' = (N, A, l', c')$ be a network for which the last conditions from (5) hold: $l(i, j) \le l'(i, j) \le \min\{c'(i, j), l(i, j) + \beta(i, j)\}$ and $c(i, j) - \delta(i, j) \le c'(i, j), \forall(i, j) \in A$. Let $G'_x = (N, A'_x, r')$ be the residual network attached to the network G' for the flow x. It is easy to see that $r'(i, j) > 0, \forall(i, j) \in \widetilde{A}$ due to the restriction on the upper bound vector for the arc (i, j) of G ($c(i, j) > x(i, j) + \delta(i, j) \Rightarrow c'(i, j) > x(i, j)$ if $(i, j) \in A \cap \widetilde{A}$) or because $(i, j) \in \widetilde{A}$, $(j, i) \in A, x(j, i) > l(j, i) + \beta(j, i) \Rightarrow x(j, i) > l'(j, i)$. This means that $\widetilde{A} \subseteq A'_x$.

If IMFG is a feasible problem, then it means that there is a vector $(\bar{l}, \bar{c})$ with $\bar{l}(i, j) \le \min\{\bar{c}(i, j), l(i, j) + \beta(i, j)\}$ and $c(i, j) - \delta(i, j) \le \bar{c}(i, j), \forall(i, j) \in A$ and for which the flow x is a maximum flow in the network $\bar{G} = (N, A, \bar{l}, \bar{c})$. Since $\widetilde{A} \subseteq \bar{A}_x$, if it exists a directed path in $\widetilde{G}$ from s to t, it corresponds to a directed path in $\bar{G}_x$, which leads to an augmentation to the flow x in G (contradiction).

Now, for the inverse implication we construct the following upper and lower bound vectors for the arcs of the network G:

$$c''(i, j) = \begin{cases} c(i, j), & \texttt{if } c(i, j) > x(i, j) + \delta(i, j) \\ x(i, j), & \texttt{otherwise} \end{cases}$$

$$l''(i, j) = \begin{cases} l(i, j), & \texttt{if } x(i, j) > l(i, j) + \beta(i, j) \\ x(i, j), & \texttt{otherwise} \end{cases}$$

It is easy to see that $l''(i, j) \le l(i, j) + \beta(i, j)$ and $c(i, j) - \delta(i, j) \le c''(i, j), \forall(i, j) \in A$. In the residual network $G''_x = (N, A''_x, r'')$ attached to $G'' = (N, A, l'', c'')$ and for the flow x we have $r''(i, j) = 0$, for all $(i, j) \in (N \times N) - \widetilde{A}$. Since $\widetilde{A} \subseteq A''_x$, it means that $\widetilde{A} = A''_x$ (see (4)). Therefore, because there is no path from s to t in the graph $\widetilde{G}$, it results that there is no directed path from s to t in G''_x. This implies that the flow x is a maximum flow in the network $G'' = (N, A, l'', c'')$. It means that (l'', c'') is a feasible solution for IMFG.

In IMFG, the feasible region for the vector $(\bar{l}, \bar{c})$ can be reduced to $l \le \bar{l} \le \bar{l} + \beta$ and $c - \delta \le \bar{c} \le c$ (from (5) and because when solving IMFG there is no need to increase the upper bounds of the flow and there is no need to decrease the lower bounds of the flow), which is a compact region. So, because IMFG has a feasible solution, it results that IMFG has optimum solution. □

The verification of IMFG being feasible can be done in $O(\widetilde{m})$ time complexity, using a graph search algorithm in $\widetilde{G}$, where $\widetilde{m}$ is the number of arcs in the set $\widetilde{A}$ with $\widetilde{m} \le 2m$.

4. Inverse feasibility problems of IMFG

As we have seen in the privious section, the inverse maximum flow problems do not have always optimum solution. In this section, we shall study the problem of modifing the restrictions to the lower and upper bound (β and γ) as little as possible in order to transform the unfeasible IMFG into a feasible problem. So, we shall introduce new inverse combinatorial optimization problems (denoted TRIMF). The mathematical model for these problems is:

$$\begin{cases} \texttt{min}\ dist((\beta,\delta),(\bar{\beta},\bar{\delta})) \\ \bar{\beta} \geq 0 \\ \bar{\delta} \geq 0 \\ \texttt{IMFG with the bound restrictions}\ \bar{\beta}\ \texttt{and}\ \bar{\delta} \\ \quad \texttt{is feasible in}\ G\ \texttt{for}\ x \end{cases} \tag{9}$$

In this paper, the following formulas to measure the distance between the initial restriction vector of the bound variation (β, δ) and the modified one $(\bar{\beta}, \bar{\delta})$ will be considered: weighted l_k norm with $k \geq 1$, weighted l_∞ norm, sum-type and bottleneck-type Hamming distance.

We shall consider the network G being anti-symmetric without reducing the generality of TRIMF, i.e., $(i, j) \in A \Rightarrow (j, i) \notin A$. If a network is not antisymmetric, then it can be easily transformed into an antisymmetric network.

As we have seen in theorem 1, only the arcs of $\widetilde{A}$ from $\widetilde{G}$ are responsable for IMFG being unfeasible. These arcs correspond to arcs from the set $\widetilde{A_1} \cup \widetilde{A_2}$ in G (see (6) and (7)). So, for an arc $(i, j) \in \widetilde{A}_1$ if the upper bound restriction $\delta(i, j)$ is modified to be greater or equal to $c(i, j) - x(i, j)$, then the arc (i, j) leaves the set $\widetilde{A}_1$. Similarly, for an arc $(i, j) \in \widetilde{A}_2$ if the lower bound restriction $\beta(i, j)$ is modified to be greater or equal to $x(i, j) - l(i, j)$, then the arc (i, j) leaves the set $\widetilde{A}_2$. So, it is enough for an arc $(i, j) \in \widetilde{A}$ to increase the value of δ with $c(i, j) - x(i, j) - \delta(i, j)$ (if $(i, j) \in \widetilde{A}_1$) in order to eliminate (i, j) from $\widetilde{A}$ or, respectively, to decrease the value of β with $x(j,i) - l(j,i) - \beta(j,i)$ (if $(j,i) \in \widetilde{A}_2$) in order to eliminate (i, j) from $\widetilde{A}$. Starting from this observation we construct the following vector defined for the arcs of $\widetilde{A}$ in order to start studying the possibility of transforming IMFG into a feasible problem:

$$\lambda(i, j) = \begin{cases} c(i, j) - x(i, j) - \delta(i, j), \texttt{if}\ \ (i, j) \in \widetilde{A_1} \\ x(j, i) - l(j, i) - \beta(j, i), \texttt{if}\ \ (j, i) \in \widetilde{A_2} \end{cases}. \tag{10}$$

We shall solve now TRIMF for the weighted l_k norm, where $k \geq 1$. We shall denote TRIMF1 this problem. The mathematical model for this problem is:

$$\begin{cases} \texttt{min}\sqrt[k]{\sum_{(i,j)\in A}\{w_\beta(i, j) \cdot |\beta(i, j) - \bar{\beta}(i, j)|^k + w_\delta(i, j) \cdot |\delta(i, j) - \bar{\delta}(i, j)|^k\}} \\ \bar{\beta} \geq 0 \\ \bar{\delta} \geq 0 \\ \texttt{IMFG with the bound restrictions}\ \bar{\beta}\ \texttt{and}\ \bar{\delta}\ \texttt{is feasible in} \\ \quad G\ \texttt{for}\ x, \end{cases} \tag{11}$$

where w_β and w_δ are associated modification cost vectors for β and, respectively, δ. Of course, $w_\beta(i, j), w_\delta(i, j) \geq 0, \forall (i, j) \in A$. Due to the observation that β and δ are modified only on arcs of $\widetilde{A}$ with the quantity given by the vector λ, it results from (10) that, instead of (11), the following mathematical for TRIMF1 can be considered:

$$\begin{cases} \text{min}_{B\subseteq\widetilde{A}}\{\sum_{(i,j)\in B\cap\widetilde{A}_1} w_\beta(i,j)\cdot(\lambda(i,j))^k + \sum_{(i,j)\in B,(j,i)\in\widetilde{A}_2} w_\delta(j,i)\cdot(\lambda(i,j))^k\} \\ \bar{\beta}(i,j)=\beta(i,j), \forall(i,j)\in A-\widetilde{A}_1 \\ \bar{\delta}(i,j)=\delta(i,j), \forall(i,j)\in A-\widetilde{A}_2 \\ \bar{\beta}\geq 0 \\ \bar{\delta}\geq 0 \\ \texttt{IMFG with the bound restrictions } \bar{\beta} \texttt{ and } \bar{\delta} \texttt{ is feasible in} \\ \quad G \texttt{ for } x. \end{cases} \tag{11'}$$

We construct the network $\widetilde{G}_k=(N,\widetilde{A},\widetilde{c}_k)$, where $\widetilde{c}_k$ is defined as follows:

$$\widetilde{c}_k(i,j)=\begin{cases} w_\delta(i,j)\cdot(\lambda(i,j))^k, \texttt{if } (i,j)\in\widetilde{A}_1 \\ w_\beta(j,i)\cdot(\lambda(i,j))^k, \texttt{if } (j,i)\in\widetilde{A}_2 \end{cases}. \tag{12}$$

In order to solve TRIMF1 we have to find the set $B\subseteq\widetilde{A}$ with the propriety that if the arcs of B are eliminated from $\widetilde{G}_k$, then there is no directed path from s to t in $\widetilde{G}_k$ and the sum of c_k capacities of the arcs from B is minimum. It is not difficult to see that B is the set of direct arcs of the minimum s-t cut of $\widetilde{G}_k$, i.e., $B=(S,\bar{S})$, where $B=[S,\bar{S}]$ is the minimum s-t cut of $\widetilde{G}_k$. In order to eliminate the arc (i,j) of $B\cap\widetilde{A}_1$ from $\widetilde{A}$ it is necessary and sufficient to set $\delta(i,j)=c(i,j)-x(i,j)$ (see (10)). Similarly, in order to eliminate the arc (i,j) of B from $\widetilde{A}$ with $(j,i)\in\widetilde{A}_2$ it is necessary and sufficient to set $\beta(j,i)=x(j,i)-l(j,i)$ (see (10)). So, the optimum solution of TRIMF1 is $(\bar{\beta},\bar{\delta})$, where:

$$\bar{\beta}(i,j)=\begin{cases} x(i,j)-l(i,j), \texttt{if } (j,i)\in(S,\bar{S}) \texttt{ and } (i,j)\in\widetilde{A}_2 \\ \beta(i,j), \texttt{otherwise} \end{cases}, \forall(i,j)\in A \tag{13}$$

$$\bar{\delta}(i,j)=\begin{cases} c(i,j)-x(i,j), \texttt{if } (i,j)\in(S,\bar{S})\cap\widetilde{A}_1 \\ \delta(i,j), \texttt{otherwise} \end{cases}, \forall(i,j)\in A. \tag{14}$$

The algorithm (denoted ATRIMF1) for solving TRIMF1 is:

```
Construct the network $\widetilde{G}_k=(N,\widetilde{A},\widetilde{c}_k)$ (see (8) and (12));
Find the minimum $s$-$t$ cut $[S,\bar{S}]$ of $\widetilde{G}_k$;
Construct $\bar{\beta}$ (see (13));
Construct $\bar{\delta}$ (see (14)).
```

The time complexity of ATRIMF1 is given by the time complexity of finding the minimum s-t cut in the network $\widetilde{G}_k$.

One of the today's best strongly polynomial time complexity algorithms for maximum flow and the minimum cut is due to Goldberg & Tarjan (1986). If it is applied in the network $\widetilde{G}_k$, it has the time complexity of $O(n\cdot\widetilde{m}\cdot log(n^2/\widetilde{m}))$, where $n=|N|$ and $\widetilde{m}=|\widetilde{A}|\leq 2m$.

One of the today's best weakly polynomial algorithms for maximum flow and minimum cut is due to Goldberg & Rao (1998). If it is applied in the network $\widetilde{G}_k$, it has the time complexity of $O(min\{n^{2/3}, \widetilde{m}^{1/2}\} \cdot m \cdot log(n^2/\widetilde{m}) \cdot log(\widetilde{C}_k))$, where $\widetilde{C}_k = max\{\widetilde{c}_k(i, j)|(i, j) \in \widetilde{A}\}$.

Definition 1 (accesible node): *A node j of a graph is called accessible from the node i, if there is a directed path from i to j.*

We shall study now TRIMF under weighted l_∞ norm for measuring the distance between (δ, β) and $(\bar{\delta}, \bar{\beta})$ in (10). We shall denote TRIMF2 this problem.

The mathematical model for TRIMF2 is:

$$\begin{cases} \mathtt{min}\, \max_{(i,j)\in A}\{w_\beta(i,j)\cdot|\beta(i,j)-\bar{\beta}(i,j)|, w_\delta(i,j)\cdot|\delta(i,j)-\bar{\delta}(i,j)|\} \\ \bar{\beta} \geq 0 \\ \bar{\delta} \geq 0 \\ \mathtt{IMFG\ with\ restrictions}\ \bar{\beta}\ \mathtt{and}\ \bar{\delta}\ \mathtt{is\ feasible\ in}\ G\ \mathtt{for}\ x. \end{cases} \tag{15}$$

Due to the observation that β and δ are modified only on some arcs of $\widetilde{A}$ with the quantity given by the vector λ, it results from (10) and (12) that, instead of (15), the following mathematical for TRIMF2 can be considered:

$$\begin{cases} \mathtt{min}_{H\subseteq\widetilde{A}} \max\{\widetilde{c}_1(i,j)|(i,j)\in H\} \\ \bar{\beta}(i,j)=\beta(i,j), \forall(i,j)\in A-\widetilde{A}_1 \\ \bar{\delta}(i,j)=\delta(i,j), \forall(i,j)\in A-\widetilde{A}_2 \\ \bar{\beta} \geq 0 \\ \bar{\delta} \geq 0 \\ \mathtt{IMFG\ with\ restrictions}\ \bar{\beta}\ \mathtt{and}\ \bar{\delta}\ \mathtt{is\ feasible\ in}\ G\ \mathtt{for}\ x \end{cases} \tag{15'}$$

In order to solve TRIMF2 we have to eliminate the arcs (i, j) from $\widetilde{A}$ having the minimum value of $\widetilde{c}_1(i, j)$ and, after the elimination is done, there is no longer a directed path from s to t in the network $\widetilde{G}$. We shall sort descending the arcs of $\widetilde{G}$ by their capacities $\widetilde{c}_1$. After sorting, the arcs of H (initially $H = \widetilde{A}$) are sequentialy eliminated from H and introduced in a set denoted D (initially being null) till there is a directed path in the graph $G^D = (N, D)$ from s to t. The last arc is not eliminated from H (if it is eliminated from H, then there is a directed path from s to t in $G^H = (N, \widetilde{A} - H)$). At the end of the algorithm the set H contains the arcs of $\widetilde{A}$ that lead to modifications of the restrictions of bounds on the arcs of G.

In order to eliminate the arc (i, j) of $H \cap \widetilde{A}_1$ from $\widetilde{A}$ it is necessary and sufficient to set $\delta(i, j) = c(i, j) - x(i, j)$ (see (10)). Similarly, in order to eliminate the arc (i, j) of H from $\widetilde{A}$ with $(j, i) \in \widetilde{A}_2$ it is necessary and sufficient to set $\beta(j, i) = x(j, i) - l(j, i)$ (see (10)). So, the optimal solution of TRIMF2 is:

$$\bar{\delta}(i,j) = \begin{cases} c(i,j)-x(i,j), \mathtt{if}\ \ (i,j)\in H \\ \delta(i,j), \mathtt{otherwise} \end{cases}, \forall(i,j)\in A \tag{16}$$

$$\bar{\beta}(i,j) = \begin{cases} x(i,j)-l(i,j), \mathtt{if}\ \ (j,i)\in H \\ \beta(i,j), \mathtt{otherwise} \end{cases}, \forall(i,j)\in A. \tag{17}$$

The algorithm (denoted ATRIMF2) for solving TRIMF2 is:

```
$D := \phi$;
$H := \widetilde{A}$;
$W := \{s\}$;
Sort descending the arcs from the set $H$ by $\widetilde{c}_1$;
For $(i, j) \in H$ do
   If $j \notin W$ then
      $D := D \cup \{(i, j)\}$;
      If $i \in W$ then
         Find the nodes $W'$ accessible from the nodes of $W$ in
         the graph $G^D = (N, D)$, eliminating the visited arcs of $D$;
         $W := W \cup W'$;
      endif;
   endif;
   If $t \in W$ then
      $d := r(i, j)$;
      BREAK;
   else $H := H - \{(i, j)\}$;
   endif;
endfor;
Construct the vectors $\beta$ and $\delta$ (see (16) and (17));
$(\bar{\beta}, \bar{\delta})$ is the optimal solution of the TRIMF2 problem.
```

Theorem 2 (correctness of ATRIMF2): *The vector $(\bar{\beta}, \bar{\delta})$ found by the algorithm is the optimum solution of TRIMF2 and $\left\|(\bar{\beta}, \bar{\delta}) - (\beta, \delta)\right\|_\infty = d$, where d is the value obtained in the algorithm.*

Proof. The main idea of the algorithm is to change the capacities $\widetilde{c}_1$ (by transforming them to 0) of the arcs from $\widetilde{A}$ and to leave as many as possible arcs with the greatest capacities unchanged. The capacities $\widetilde{c}$ of the arcs that remain in the set H at the end of the algorithm will be equal to 0 after the bound restrictions will be modified in G using (16) and (17). The algorithm starts with $H = \widetilde{A}$ (all the arcs of $\widetilde{A}$ are candidates for elimination). The arcs from H are sorted descending by $\widetilde{c}_1$ and so, they are prepared for elimination from H, starting with the arc with the greatest $\widetilde{c}_1$ capacity and ending with the arc having the lowest capacity.

We shall explain now the meaning of the sets that appear in the algorithm.

The set D contains arcs with $\widetilde{c}_1$ capacity that will not be changed (until the last iteration of **for** sequence). At the beginning of the algorithm, D is null. Every time accessible nodes from s are searched, the visited arcs are eliminated from the set D in order to keep the time complexity low. The search will continue at the point where the last time it was stopped. In this proof, we shall refer to the set D as being complete (without any elimination of arcs).

At any moment, in the set W there exists only the nodes j which are accessible from s in the graph $G^D = (N, D)$.

The set W' contains the nodes which are newly found accessible from s in the graph $G^D = (N, D)$, when the current arc (i, j) enters the set D. After they are found, they are added into the set W.

During the **for** loop, one of the following 2 situations can be found for every arc (i, j) of H:

Situation 1: $j \notin W$

This means that the node j was not previously found accessible from s. So, the arc (i, j) is introduced in the set D. The next time another search will be done in the network $G^D = (N, D)$, this arc will possibly help to find other accessible nodes from s in G^D. If the node i is in the set of accessible nodes from s, then j is also accessible from s in G^D and it is possible that other nodes from N will be found accessible from s. So, the graph search is continued in G^D. All newly found accessible nodes from W' will be added into the set W.

Situation 2: $j \in W$

It means that the node j was previously found as being accessible from s in the network $G^D = (N, D)$. In this case it is no need to introduce the arc (i, j) in the set D.

The algorithm ends when the node t enters the set D, becoming accessible from s in the graph $G^D = (N, D)$. The last arc (i, j), which made the node t become accessible, will not leave the set H. The value of $\widetilde{c}_1$ capacity of the first arc $(i, j) \in H$, with the propriety that t becomes accessible from s in G^D is introduced in the variable d. So, d is the greatest value of the residual capacities of the arcs that are changed, i.e., $d = max\{\widetilde{c}_1(u, v) \mid (u, v) \in H\} = \left\|(\bar{\beta}, \bar{\delta}) - (\beta, \delta)\right\|_{\infty}$. □

Theorem 3 (complexity of ATRIMF2): *The algorithm for TRIMF2 has a time complexity of* $O(\widetilde{m} \cdot log(n))$*, where* $n = |N|$ *and* $\widetilde{m} = |\widetilde{A}|$.

Proof. Finding the set W with the nodes accessible from s and the elimination of the visited arcs from D can be done in linear time - $O(\widetilde{m})$, using a graph search algorithm applied in the graph G^D.

At every search of accessible nodes from s, arcs from D are eliminated in order to improve the time complexity of **for ... do** sequence. If they are not eliminated, then the time complexity of the algorithm would be closed to $O(\widetilde{m}^2)$.

Every time the current arc $(i, j) \in H$ has $i \in W$ and $j \notin W$, the initiated graph search is continued in G^D.

Every arc from $\widetilde{A}$ enters at most once in D and exits at most once out of D. It results that the total time needed to execute the whole graph search of G^D is $O(\widetilde{m})$.

All the operations inside **for ... do** sequence (other than graph search) can be done in $O(1)$ time.

So, **for ... do** sequence is executed in linear time complexity.

The arcs from the set H can be sorted in $O(\widetilde{m} \cdot log(\widetilde{m})) = O(\widetilde{m} \cdot log(n))$ time. Hence, the time complexity of ATRIMF2 is given by the method used to sort the arcs from H. □

We shall study now TRIMF under weighted sum-type Hamming distance. We shall denote TRIMF3 this problem. The mathematical model of this problem is:

$$\begin{cases} \texttt{min} \sum_{(i,j)\in A}\{w_\beta(i, j) \cdot H(\beta(i, j), \bar{\beta}(i, j)) + w_\delta(i, j) \cdot H(\delta(i, j), \bar{\delta}(i, j))\} \\ \bar{\beta} \geq 0 \\ \bar{\delta} \geq 0 \\ \texttt{IMFG with restrictions } \bar{\beta} \texttt{ and } \bar{\delta} \texttt{ is feasible in } G \texttt{ for } x, \end{cases} \tag{18}$$

where $H(i, j) = 0$ if $i = j$ and 1 otherwise. Of course, $w_\beta(i, j), w_\delta(i, j) \geq 0, \forall (i, j) \in A$.

In TRIMF3 $w_\beta(i,j)$ is the cost of modification of β on the arc (i,j) and $w_\delta(i,j)$ is the cost of modification of δ on (i,j). The total cost of modification of β and δ is minimized.

Due to the observation that β and δ are modified only on some arcs of $\widetilde{A}$, it results that, instead of (18), the following mathematical for TRIMF3 can be considered:

$$\begin{cases} \mathtt{min}_{B\subseteq\widetilde{A}}\{\sum_{(i,j)\in B\cap\widetilde{A}_1} w_\beta(i,j)+\sum_{(i,j)\in B,(j,i)\in\widetilde{A}_2} w_\delta(j,i)\} \\ \bar{\beta}(i,j)=\beta(i,j), \forall(i,j)\in A-\widetilde{A}_1 \\ \bar{\delta}(i,j)=\delta(i,j), \forall(i,j)\in A-\widetilde{A}_2 \\ \bar{\beta}\geq 0 \\ \bar{\delta}\geq 0 \\ \mathtt{IMFG\ with\ restrictions}\ \bar{\beta}\ \mathtt{and}\ \bar{\delta}\ \mathtt{is\ feasible\ in}\ G\ \mathtt{for}\ x. \end{cases} \tag{18'}$$

It is now clear that TRIMF3 can be treated as a particular case of TRIMF1 (if in the mathematical model (11′) of TRIMF1 k is consider equal to 0 then the model (18′) is obtained).

There are faster algorithms for this finding the maximum flow and, consequently, the minimum cut in unit capacity networks. For instance, an algorithm is presented by Ahuja *et al* (1993) which has the time complexity of $O(min\{n^{2/3}\cdot m, m^{3/2}\}) = O(min\{n^{2/3}, m^{1/2}\}\cdot m)$. More recently, the complexity of $O(n^{2/3}\cdot m)$ was also achieved by Sedeno-Noda & Gonzalez-Martin (2000).

We shall study now TRIMF under weighted bottleneck-type Hamming distance. We shall denote TRIMF4 this problem. The mathematical model of this problem is:

$$\begin{cases} \mathtt{min}\max_{(i,j)\in A}\{w_\beta(i,j)\cdot H(\beta(i,j),\bar{\beta}(i,j)), w_\delta(i,j)\cdot H(\delta(i,j),\bar{\delta}(i,j))\} \\ \bar{\beta}\geq 0 \\ \bar{\delta}\geq 0 \\ \mathtt{IMFG\ with\ restrictions}\ \bar{\beta}\ \mathtt{and}\ \bar{\delta}\ \mathtt{is\ feasible\ in}\ G\ \mathtt{for}\ x. \end{cases} \tag{19}$$

From the observation, there is no need to modify β and δ elsewhere then on some arcs of $\widetilde{A}$ and the modification has to be done with the quantity given by the vector $\boldsymbol{\lambda}$, it results from (10) and (12) that, instead of (19), the following mathematical expression for TRIMF2 can be considered:

$$\begin{cases} \mathtt{min}_{H\subseteq\widetilde{A}}\max\{\widetilde{c}_0(i,j)|(i,j)\in H\} \\ \bar{\beta}(i,j)=\beta(i,j), \forall(i,j)\in A-\widetilde{A}_1 \\ \bar{\delta}(i,j)=\delta(i,j), \forall(i,j)\in A-\widetilde{A}_2 \\ \bar{\beta}\geq 0, \bar{\delta}\geq 0 \\ \mathtt{IMFG\ with\ restrictions}\ \bar{\beta}\ \mathtt{and}\ \bar{\delta}\ \mathtt{is\ feasible\ in}\ G\ \mathtt{for}\ x. \end{cases} \tag{19'}$$

It is easy to see that we have obtained in (19′) a similar model to (15′), only the values that are maximized in the set $H\subseteq\widetilde{A}$ are different. So, we can very easily adapt ATRIMF2 to solve TRIMF4. The only difference is that the arcs of H are sorted by c_0 instead of c_1.

5. Conclusions

We have presented a fast (in linear time) method to test the feasibility of IMFG. We have studied four new inverse combinatorial problems (under different norms and distances) about transforming a non-feasible inverse maximum flow problem into a feasible one. These inverse feasibility problems consist of modifying as little as possible the restrictions to the bounds of the flow.

As a future work, we intend to study the problem of transforming a non-feasible inverse maximum flow problem into a feasible one by modifying as little as possible the given flow. This problem seems to be more difficult than the problems studied in this paper. We also intend to identify and to study new inverse feasibility problems.

References

Ahuja R, Magnanti T and Orlin J B 1993 *Network flows. Theory, algorithms and applications,* Englewood Cliffs, NJ: Prentice Hall

Deaconu A 2006 A cardinality inverse maximum flow problem. *Sci. Ann. Comput. Sci.* 16: 51–62

Deaconu A 2008a The inverse maximum flow problem considering L_∞ norm. *RAIRO-RO* 42(3): 401–414

Deaconu A 2008b The inverse maximum flow problem with lower and upper bounds for the flow. *YUJOR* 18(1): 13–22

Heuberger C 2004 Inverse combinatorial optimization: A survey on problems, methods, and results. *J. Comb. Opt.* 8: 329–361

Goldberg A V and Rao S 1998 Beyond the flow decomposition barrier. *Journal of the ACM* 45: 783–797

Goldberg A V and Tarjan R E 1986 A new approach to the maximum flow problem, *Annual ACM Symposium on Theory of Computing, Proceedings of the eighteenth annual ACM symposium on Theory of computing*, 136–146. ISBN 0-89791-193-8

Liu L and Zhang J 2006 Inverse maximum flow problems under weighted hamming distance. *J. Comb. Opt.* 12(4): 395–408

Sedeno-Noda A and Gonzalez-Martin C 2000 An O(n m log(U/n)) time maximum flow algorithm. *Naval Research Logistics* 47(6): 511–520

Yang C, Zhang J and Ma Z 1997 Inverse maximum flow and minimum cut problems. *Optimization* 40: 147–170

Zhang J and Cai C 1998 Inverse problems of minimum cuts. *ZOR-Math. Methods Oper. Res.* 47: 51–58