A Study on the Feasibility of the Inverse Maximum Flow Problems and Flow Modification Techniques in the Case of Non-feasibility

ADRIAN DEACONU
Transilvania Univ. of Braşov
Theoretical Computer Science Dept.
Braşov, Iuliu Maniu 50, cod 500091
ROMANIA
a.deaconu@unitbv.ro

ELEONOR CIUREA
Transilvania Univ. of Braşov
Theoretical Computer Science Dept.
Braşov, Iuliu Maniu 50, cod 500091
ROMANIA
e.ciurea@unitbv.ro

CORNELIU MARINESCU
Transilvania University of Braşov
Electrical Eng. and Computer Sci.
Braşov, Eroilor 29, 500036
ROMANIA
corneliu.marinescu@unitbv.ro

Abstract: The feasibility of the inverse maximum flow problems (denoted IMFG) is studied. The feasibility can be tested in linear time. In the case of IMFG not being feasible, a new inverse combinatorial optimization problem is introduced and solved. The problem is to modify as little as possible the flow so that the problem becomes feasible for the modified flow. An example is presented.

Key–Words: inverse optimization, maximum flow.

1 Introduction

An inverse combinatorial optimization problem consists in 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. In the last years many papers were published in the field of inverse combinatorial optimization [15]. Almost every inverse problem was studied considering l_1 and l_∞ norms, resulting in different problems with completely different solution methods. Strongly polynomial time algorithms to solve the inverse maximum flow problem when l_1 norm is considered (denoted IMF) were presented by Yang, Zhang and Ma [27]. 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 nomber of nodes.

The more general case (denoted GIMF) under l_1 norm is studied in [11], where the lower and upper bounds for the flow are changed. Strongly and weakly polynomial algorithms to solve GIMF are proposed. The strongly polynomial algorithms for GIMF have the same time complexity as the algorithms 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 $R = max\{c(x,y) - f(x,y) + f(y,x) - l(y,x) | x, y \in N\}$.

The inverse maximum flow under l_∞ norm (denoted IMF∞) is considered in [10]. A very fast $O(m \cdot log(n))$ time algorithm to solve this problem is presented. The more general case of this problem (under weighted l_∞ norm) is considered in [13].

The least number of modifications to the lower or/and upper bounds is considered in [12]. 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 and Zhang [21] under the sum-type and bottleneck-type weighted Hamming distance. Strongly polynomial algorithms to solve these problems are proposed.

In this paper, a theorem on the feasibility of the inverse maximum flow problems is presented. This theorem leads to an $O(m)$ time algorithm for deciding if an inverse maximum flow problem has solution or not. If a problem is not feasible what do we do in this situation? From the practical point of view it is not acceptable to give up. The problem must be solved somehow even if we have to make a compromise. The compromise consists in modifying (as little as possible) some parameters of the problem. Of course, this leads to new inverse combinatorial optimization problems.

In this paper a new inverse combinatorial optimization problem is introduced. This problem consists in modifying as little as possible the flow in order to transform the inverse maximum problem into a feasible problem.

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.

If a network has more than a source or/and more than a sink node, it can be transformed into an *s-t* network (introducing a super-source and a super-sink node) [1].

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

$$\sum_{y\in N,(x,y)\in A} f(x,y) - \sum_{y\in N,(y,x)\in A} f(y,x) = \begin{cases} v(f), & x = s \\ -v(f), & x = t \\ 0, & x \in N - \{s,t\} \end{cases}, \quad (1)$$

where *v(f)* is the value of the flow *f* from *s* to *t*.

The capacity restrictions are:

$$l(x,y) \le f(x,y) \le c(x,y),\ \forall\ (x,y) \in A, \quad (2)$$

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

The maximum flow problem is:

$$\begin{cases} \texttt{max } v(f) \\ f \texttt{ is a feasible flow in } G \end{cases}. \quad (3)$$

We shall introduce now the definition of the minimum cut *s-t* 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 $(S,\bar{S}) = \{(x,y) \in A | x \in S \texttt{ and } y \in \bar{S}\}$ is the set of direct arcs of the cut and $(\bar{S},S) = \{(x,y) \in A | x \in \bar{S} \texttt{ and } y \in S\}$ is the set of the inverse arcs. The capacity of the *s-t* cut $[S,\bar{S}]$ in *G* is $c[S,\bar{S}] = c(S,\bar{S}) - l(\bar{S},S) = \sum_{(x,y)\in(S,\bar{S})} c(x,y) - \sum_{(x,y)\in(\bar{S},S)} l(x,y)$. 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*.

The residual network attached to the network *G* for the flow *f* is $G_f = (N, A_f, r, s, t)$, where for each pair of nodes (x,y) the value of $r(x,y)$ is defined as follows:

$$r(x,y) = \begin{cases} c(x,y) - f(x,y) + f(y,x) - l(y,x), \\ \quad \texttt{if } (x,y) \in A \texttt{ and } (y,x) \in A \\ c(x,y) - f(x,y), \\ \quad \texttt{if } (x,y) \in A \texttt{ and } (y,x) \notin A \\ f(y,x) - l(y,x), \\ \quad \texttt{if } (x,y) \notin A \texttt{ and } (y,x) \in A \\ 0, \texttt{ otherwise} \end{cases} \quad (4)$$

The set A_f contains as arcs of the residual network only the pairs of nodes $(x,y) \in N \times N$ for which the residual capacity is positive, i.e., $r(x,y) > 0$.

An inverse maximum flow problem is to change as little as possible the lower and/or upper bound vectors *l* and respectively *c* so that the given feasible flow *f* becomes a maximum flow in *G*.

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

$$\begin{cases} \texttt{min } dist((l,c),(\bar{l},\bar{c})) \\ f \texttt{ is a maximum flow in } \bar{G} = \{N, A, \bar{l}, \bar{c}, s, t\} \\ l(x,y) - \gamma(x,y) \le \bar{l}(x,y) \le \\ \quad \le \min\{\bar{c}(x,y), l(x,y) + \beta(x,y)\}, \\ \quad \forall (x,y) \in A \\ c(x,y) - \delta(x,y) \le \bar{c}(x,y) \le \\ \quad \le c(x,y) + \alpha(x,y),\ \forall\ (x,y) \in A \end{cases} \quad (5)$$

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) Hamming distance etc. So, (5) is a general model for any inverse maximum flow problem, where the lower and the upper bounds for the flow can be modified.

The values $\alpha(x,y), \beta(x,y), \gamma(x,y)$ and $\delta(x,y)$ are given non-negative integer numbers, where $\gamma(x,y) \le l(x,y)$ and $\delta(x,y) \le c(x,y)$, for each arc $(x,y) \in A$. These values show how much the bounds for the flow of the arcs can vary.

In the formula above by (l,c) is denoted 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 $\bar{c}$.

In order to make the flow f 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, the conditions $\bar{c}(x,y) \leq c(x,y) + \alpha(x,y)$ and $\bar{l}(x,y) \geq l(x,y) - \gamma(x,y)$, for each arc $(x,y) \in A$ have no effect and, instead of (5), the following mathematical model is considered:

$$\begin{cases} \texttt{min } dist((l,c),(\bar{l},\bar{c})) \\ f \texttt{ is a maximum flow in} \\ \quad \bar{G} = \{N, A, \bar{l}, \bar{c}, s, t\} \\ \bar{l}(x,y) \leq \min\{\bar{c}(x,y), l(x,y) + \beta(x,y)\}, \\ \quad \forall\,(x,y) \in A \\ c(x,y) - \delta(x,y) \leq \bar{c}(x,y),\ \forall\,(x,y) \in A \end{cases} \quad (5')$$

The inverse maximum flow problems where $l(x,y) = 0, \forall (x,y) \in A$ and only the upper bounds for the flow can be modified are particular cases of IMFG. Indeed, if the lower bounds for the flow can not be modified in order transform the given flow *f* into a maximum flow, then in $(5')$ we can consider $\beta(x,y) = 0, \forall (x,y) \in A$.

3 The Feasibility of IMFG

When solving IMFG, if the upper bound is changed on an arc (x,y), then it will be decreased with the amount of $c(x,y) - f(x,y)$. If not so, then there still is an augmenting path from *s* to *t* that contains the direct arc (x,y) and the modification of the upper bound is useless. This means that if $c(x,y) > f(x,y) + \delta(x,y)$ on an arc (x,y), then, when solving IMFG, there is no need to change the upper bound on (x,y).

Similarly, when solving IMFG, if the lower bound is changed on an arc (x,y), then it will be increased with the amount of $f(x,y) - l(x,y)$. If not so, then there still is an augmenting path from *s* to *t* that contains the arc (x,y) in inverse direction and the modification of the lower bound is useless. This means that if $f(x,y) > l(x,y) + \beta(x,y)$ on an arc (x,y), then, when solving IMFG, there is no need to change the lower bound on (x,y).

Let's determine the arcs in the network *G* on which the capacity will not be changed.

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

$$\widetilde{A}_1 = \{(x,y) \in A \,|\, f(x,y) + \delta(x,y) < c(x,y)\}. \quad (6)$$

Similarly, as it has been seen, changing the lower bound have no effect on an arc (x,y) with $f(x,y) > l(x,y) + \beta(x,y)$. So, there is no need to try changing the lower bounds of the arcs from the following set:

$$\widetilde{A}_2 = \{(x,y) \in A \,|\, l(x,y) + \beta(x,y) < f(x,y)\}. \quad (7)$$

It is easy to see that if there is a path from *s* to *t* in the network *G* that contains only direct arcs (x,y) so that $c(x,y) > f(x,y) + \delta(x,y)$ and/or inverse arcs (y,x) with $f(y,x) > l(y,x) + \beta(y,x)$, then IMFG has no solution.

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

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

We have the following theorem:

Theorem 1 *In the network G, IMFG has optimal solution for the given flow f, if and only if there is no directed path in the graph $\widetilde{G}$ from the node s to the node t.*

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

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

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

$$c''(x,y) = \begin{cases} c(x,y),\ c(x,y) > f(x,y) + \delta(x,y) \\ f(x,y),\ \texttt{otherwise} \end{cases} \tag{9}$$

and

$$l''(x,y) = \begin{cases} l(x,y),\ f(x,y) > l(x,y) + \beta(x,y) \\ f(x,y),\ \texttt{otherwise} \end{cases} \tag{10}$$

It is easy to see that $l''(x,y) \leq l(x,y) + \beta(x,y)$ and $c(x,y) - \delta(x,y) \leq c''(x,y), \forall (x,y) \in A$. In the residual network $G''_f = (N, A''_f, r'')$ attached to $G'' = (N, A, l'', c'')$ and to the flow f we have $r''(x,y) = 0$, for all $(x,y) \in (N \times N) - \widetilde{A}$. Since $\widetilde{A} \subseteq A''_f$, it means that $\widetilde{A} = A''_f$ (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''_f. This implies that the flow f 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 $l \leq \bar{l} \leq \bar{l} + \beta$ and $c - \delta \leq \bar{c} \leq c$ (from $(5')$ and because when solving IMFG there is no need to increase the upper bounds for the flow and there is no need to decrease the lower bounds for the flow), which is a compact region. So, because IMFG has a feasible solution, it results that IMFG has optimal 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 $\widetilde{m} \leq 2m$. So, this test of feasibility can be applied to any inverse maximum flow problem.

4 The Modification of Flow

In this section for a non-feasible IMFG problem in the network G for the given flow f we shall modify the flow f so that the inverse maximum flow problem in G for the modified flow becomes feasible and the distance between the value of the initial flow and the value of the modified flow is minimum.

The following inverse optimization problem (denoted TFIMF) is obtained:

$$\begin{cases} min|v(f') - v(f)| \\ f' \ \texttt{is a feasible flow in} \ G \\ \texttt{The inverse maximum flow} \\ \texttt{problem is feasible in} \ G \ \texttt{for} \ f' \end{cases} \tag{11}$$

So, the problem is to find a feasible flow f' in the network G so that the inverse maximum flow problem in G for the flow f' is feasible and $|v(f') - v(f)|$ is minimum. For a feasible flow f' the test of feasibility of IMFG can be done in $O(m)$ time (as we have seen in section three, theorem 1). The first thing we have to do is to construct all the feasible flows in G with the value equal to $v(f)$. If there is such a flow then any of these flows is solution of problem (10).

The problem of maximum flow and, consequently, the inverse minimum flow problem is with integer values (for the lower bounds, upper bounds, flow, value of the flow and the restrictions to variation of the bounds). That is why we can think to a strategy of solving the problem (10) as follows:

If there is no feasible feasible flow f_0 in G so that IMFG is feasible for f_0 in G and $v(f_0) = v(f)$ then we look for any feasible flow f_1 in G so that IMFG is feasible for f_1 in G and $|v(f_1) - v(f)| = 1$ and so on. Finally, after k iterations, $k \leq |V - v(f)| + 1 = V - v(f) + 1$ (V is the value of any maximum flow in G) we shall find a feasible flow f_k in G so that IMFG is feasible for f_k in G and $|v(f_k) - v(f)| = k - 1$. This flow is the solution of problem (10).

In order to solve the problem (10) in the manner described above we have to find a method to generate all the feasible flows in the network G for a given integer value v.

In paper [23] a method for finding all the maximum flows in a given network G is presented.

We are interested in finding all the feasible flows in the network G for a given integer value v. In order to do that we transform the network G as follows:

We introduce in G a new node denoted s' and the arc (s', s) with the upper bound equal to v, the lower bound equal to 0 and restrictions to the variation of bounds also equal to 0, i.e., $c(s', s) = v$ and $l(s', s) = \alpha(s', s) = \delta(s', s) = 0$. The node s' becomes the only source node in the modified network denoted G_v. It is easy to see that any feasible flow in G with the value equal to v is a feasible flow in G_v and any feasible flow in G_v with the value equal to v is a feasible in flow in G. Moreover, a feasible flow with the value equal to v in G_v is a maximum flow in G_v because the total flow that exits the source node s' can not exceed $v = c(s', s)$. Consequently, for a given value v we can find all the maximum flows in G_v using the algorithm

from (9) and these flows are all the feasible flows with value v in G if we ignore the arc (s', s).

We are able now to present the algorithm for solving the problem (10):

```
PROGRAM SolvingTFIMF;
BEGIN
   v := v(f);
   v' := v(f);
   Find the maximum flow F in G;
   V := v(F);
   WHILE v ≤ V DO
   BEGIN
      Construct the network G_v;
      Find all max. flows f_i (i = 1..p) in G_v;
      FOR i:=1 TO p DO
         IF IMFG is feasible in G for f_i THEN
            f' := f_i;
            STOP;
         END IF;
      END FOR;
      IF v' ≥ 0 THEN
         Find all max. flows f_i (i = 1..p) in G_v';
         FOR i:=1 TO p DO
            IF IMFG is feasible in G for f_i THEN
              f' := f_i;
              STOP;
            END IF;
         END FOR;
      END IF;
      v := v + 1;
      v' := v' − 1;
   END WHILE;
END.
```

Theorem 2 *The program "SolvingTFIMF" finds an optimum solution f' of the problem (10).*

Proof: It is easy to see that any maximum flow F in G is a feasible solution for (10). That is because for F the lower bound vector l and the upper bound vector c forms the optimal solution for IMFG in the network G (we have to make no modification to l or/and c so that F becomes a maximum flow in G).

The algorithm constructs sequentially all the flows in G with the value equal to $v(f)$, then all the flows in G equal to $v(f) + 1$, then all the flows in G equal to $v(f) - 1$, then all the flows in G equal to $v(f) + 2$, all the flows in G equal to $v(f) - 2$ and so on.

The algorithm stops (after at most $|V - v(f)| + 1 = V - v(f) + 1$ iterations of the "WHILE .... DO" loop) when a flow f' is found for which the problem IMFG is feasible in the network G. Of course, f' is a feasible solution of problem (10).

We suppose that f' (found by the algorithm) is not an optimum solution for (10). This means that there exists a flow f'' for which IMFG is feasible in G and $|v(f'') - v(f)| < |v(f') - v(f)|$.

We denote by k' the iteration of the algorithm when f' is found (and the algorithm stops). It is easy to see that $k' = |v(f') - v(f)| + 1$.

We also denote $k'' = |v(f'') - v(f)| + 1$.

Since $k'' < k'$, it results that the flow f'' was one of the feasible flows in G constructed by the algorithm in the iteration k'', previous to iteration k' and, since f'' is a feasible solution for the problem (10), the algorithm had to stop in iteration k'' not in iteration k' (contradiction).

So, f' (found by the algorithm) is an optimum solution of the problem (10).

■

5 Example

We shall take an example that illustrates how the algorithm above works. In figure **??** a network G is presented.

On each arc we have the following values (from left to right): the first one is the restriction β to the variation of the lower bound, the second one is the lower bound (l), the third one is the value of the given initial flow f, it is followed by the upper bound for the flow (c) and, finally, we have the restriction δ to the variation of the upper bound.

As we can see, in the graph $\widetilde{G}$ from figure **??** there is a directed path from s to t. This means that IMFG is not feasible (see theorem 1).

We apply the algorithm from the previous section in order to transform the problem into a feasible one.

The figures **??** - **??** present the feasible flows constructed by the algorithm till the solution is found. Each of these figures present the feasible flow (to the left) and the graph $\widetilde{G}$ (to the right). On each arc of the network to the left of each figures **??** - **??** the value of the flow is presented.

The first iteration of the algorithm constructs all 12 possible feasible flows (see figures **??** and **??** - **??**) with the value equal to v=2 (the value of the initial flow f). None of these flows is solution of our problem.

In the second iteration, the algorithm constructs the feasible flows with the value equal to $v = 3$. There are 17 feasible flows with the value equal to $v = 3$. Two of these flows are solutions for our problem (see figures **??** and **??**).

So, the algorithm stops in the second iteration after finding two solutions with the value equal to 3.

Actually, there are exactly two solutions. There are 5 different flows with the value $v' = 1$ and none of these flows is solution.

The distance between the value of the initial flow f and the value of any flow from figures **??** or **??** which is solution of our problem is $|v - v(f)| = v - v(f) = 1$.

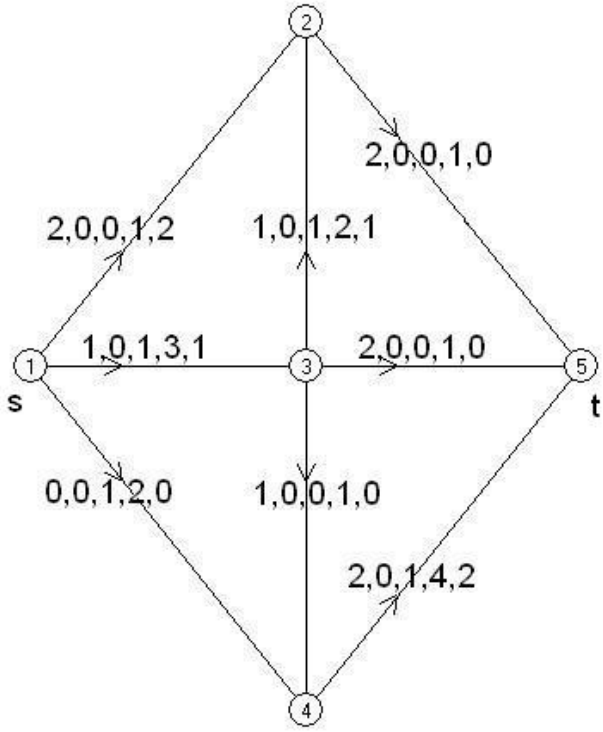

Figure 1: Initial given flow f in the network G

6 Conclusion

In this paper we have studied the feasibility of the inverse maximum flow problem. The feasibility of IMFG can be tested in linear time. If the problem is not feasible, the flow can be modified as little as possible so that the problem becomes feasible (the distance between the value of the initial flow and the value of the modified flow is minimum). A suggestive example that shows the execution of the algorithm was presented.

Acknowledgements. The research was supported by Grant PNII no. 22134/2008.

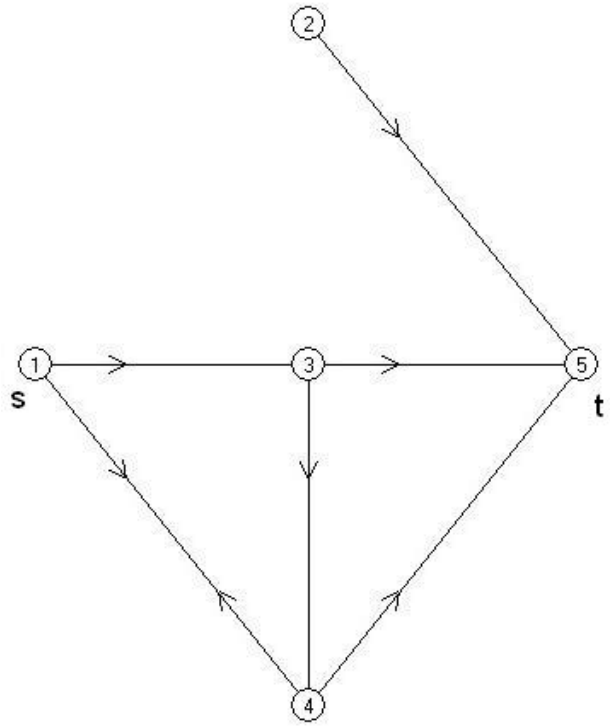

Figure 2: The initial graph $\widetilde{G}$, IMFG is not feasible

References:

[1] R. Ahuja, T. Magnanti and J. Orlin, *Network Flows. Theory, algorithms and applications*, Prentice Hall, Inc., Englewood Cliffs, NY, 1993.

[2] R.K. Ahuja, J.B. Orlin, *Combinatorial Algorithms for Inverse Network Flow Problems*, Networks, 2002.

[3] L. Ciupală and E. Ciurea, *A parallel algorithm for the minimum flow problem in bipartite networks*, Proceedings of the 12th WSEAS International Conference on Computers, Crete, Greece, 2008, pp. 2003–2007.

[4] L. Ciupală and E. Ciurea, *Sequential and parallel deficit scaling algorithms for minimum flow in bipartite networks*, WSEAS Transactions on Computers, Issue 10, Vol. 7, 2008, pp. 1545–1554.

[5] E. Ciurea, *An algorithm for minimal dynamic flow*, Korean Journal of Computational and Applied Mathematics, vol. 7(3) (2000), 259-269.

[6] E. Ciurea and O. Georgescu, *Minimum flow in unit capacity networks*, Analele Universităţii Bucureşti, XLV, 2006, pp. 11–20.

[7] E. Ciurea, O. Georgescu and D. Marinescu, *Minimum flows in bipartite networks*, Proceedings of the 10th WSEAS International Conference

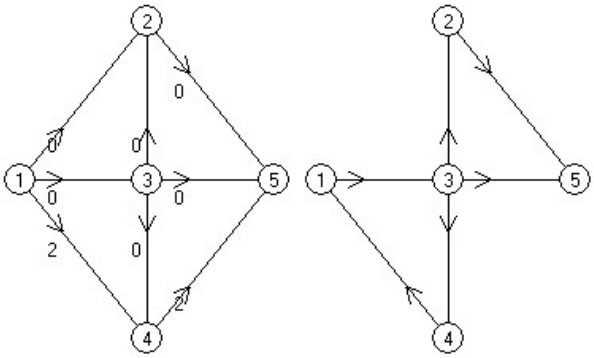

Figure 3: Iteration 1, v=2

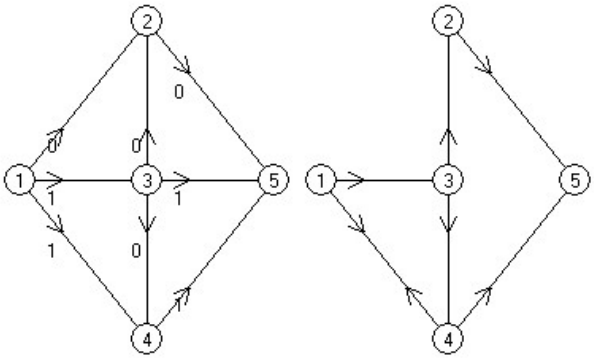

Figure 4: Iteration 1, v=2

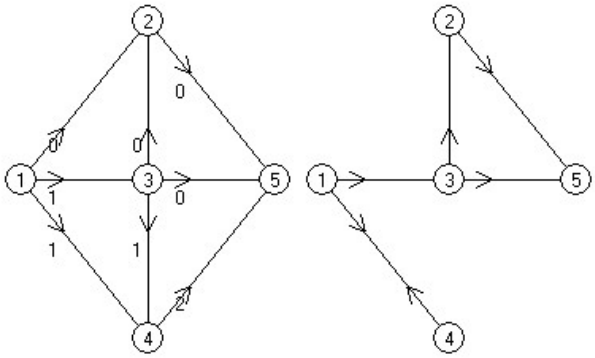

Figure 5: Iteration 1, v=2

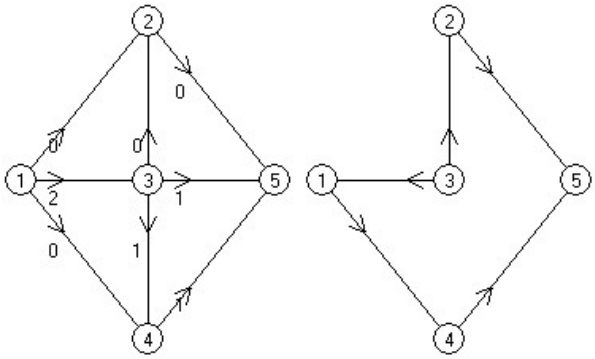

Figure 6: Iteration 1, v=2

on Mathematical and Computational Methods in Science and Engineering, Bucuresti, 2008, pp. 491–495.

[8] E. Ciurea, A. Deaconu, *Inverse Minimum Flow Problem*, Journal of Applied Mathematics and Computing, Korea 23, 2007, 193-203.

[9] A. Deaconu, E. Ciurea, Corneliu Marinescu, *Transfomation of Non-feasible Inverse Maximum Flow Problem into a Feasible one by Flow Modification*, Proc. of the 14-th WSEAS CSCC, Corfu, 2010, 250-255.

[10] A. Deaconu, *The Inverse Maximum Flow Problem Considering l_∞ Norm*, RAIRO 42, No. 3, 2008, 401-414.

[11] A. Deaconu, *The Inverse Maximum Flow Problem with Lower and Upper Bounds for the Flow*, YUJOR 18, No. 1, 2008, 13-22.

[12] A. Deaconu, *A Cardinality Inverse Maximum Flow Problem, Scientific Annals of Computer Science*, Vol. XVI, 2006, 51-62.

[13] A. Deaconu, *The Inverse Maximum Flow Problem under Weighted l_∞ Norm*, Bulletin of the *Transilvania* University of Braşov, Series III: Mathematics, Informatics, Physics, Vol 2(51), 2009, 249-254.

[14] J.R. Evans, E.Minieka, *Optimization algorithms for networks*, Marcel Dekker Inc., New York, 1992.

[15] C. Heuberger, *Inverse Combinatorial Optimization: A Survey on Problems, Methods, and Results*, Journal of Combinatorial Optimization 8, 2004, 329-361.

[16] O. Georgescu and E. Ciurea, *Decreasing path algorithm for minimum flows. Dynamic Tree Implementations*, Proceedings of the 12^{th} WSEAS International Conference on Computers, Crete, Greece, 2008, pp. 235–240.

[17] F. Glover, D. Klingman and N.V. Philips, *Network Models in Optimization and their Applications in Practice*, Wiley, New York 1992.

[18] J. Gross, J. Yellen, *Graph Theory and its Applications*, CRC Press, New York 1999.

[19] D. Jungnickel, Graphs, *Networks and Algorithms*, Springer, Berlin 1999.

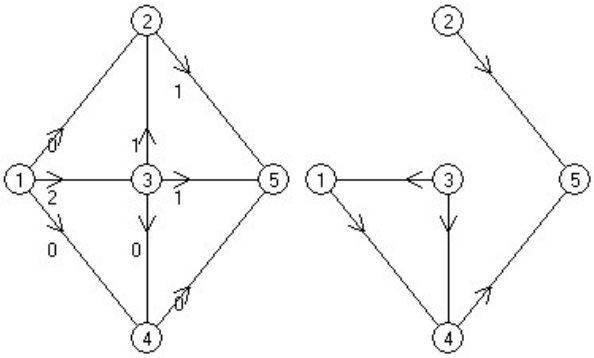

Figure 7: Iteration 1, v=2

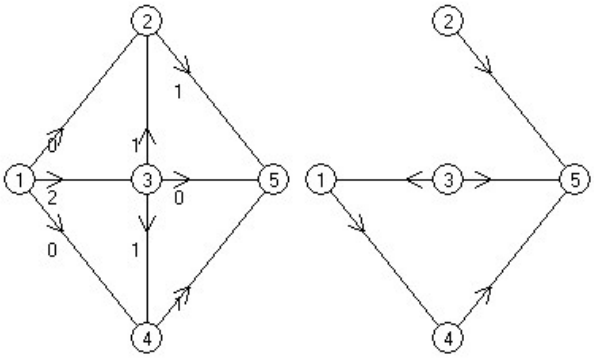

Figure 8: Iteration 1, v=2

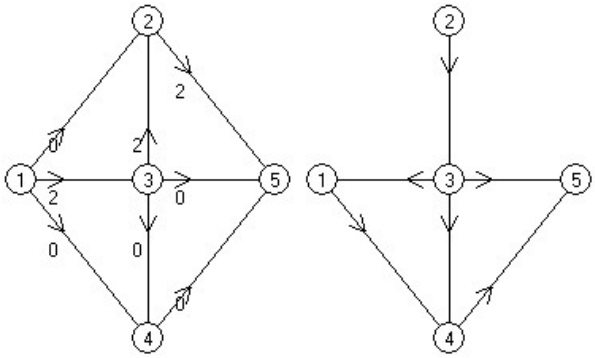

Figure 9: Iteration 1, v=2

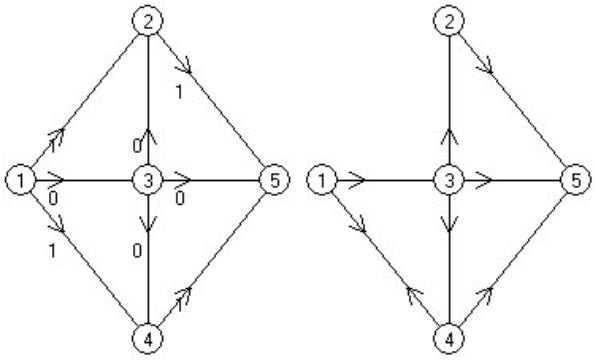

Figure 10: Iteration 1, v=2

[20] S. Lin, H. Chang and C. Kuo, *An Implementation of the Parallel Algorithm for Solving Nonlinear Multi-commodity Network Flow Problem*, WSEAS Transactions on Systems, Vol. 5(8), August 2006, pp. 1853–1860.

[21] L. Liu, J. Zhang, *Inverse Maximum Flow Problems under Weighted Hamming Distance*, Journal of Combinatorial Optimization 12(4), 2006, 395-408.

[22] E. Milkova, *Combinatorial Optimization: Mutual Relations among Graph Algorithms*, WSEAS Transactions on Mathematics, Vol. 7(5), May 2008, pp. 293–302.

[23] G. Ruhe, *Characterization of All Optimal Solutions and Parametric Maximal Flows in Networks*, Optimization 16(1), 1985, 51-61.

[24] G. Ruhe, *Algorithmic Aspects of Flows in Networks*, Kluwer Academic Publishers, Dordrecht 1991.

[25] P.T. Sokkalingam, R.K. Ahuja, J.B. Orlin, *Solving Inverse Spanning Tree Problems through Network Flow Techniques*, Oper. Res. 47, 1999, 291-298.

[26] C. Yang, X. Chen, *An Inverse Maximum Capacity Path Problem with Lower Bound Constraints*, Acta Math. Sci., Ser. B 22, 2002, 207-212.

[27] C. Yang, J. Zhang, Z. Ma, *Inverse Maximum Flow and Minimum Cut Problems*, Optimization 40, 1997, 147-170.

[28] J. Zhang, C. Cai, *Inverse Problems of Minimum Cuts*, ZOR-Math. Methods Oper. Res. 47, 1998, 51-58.

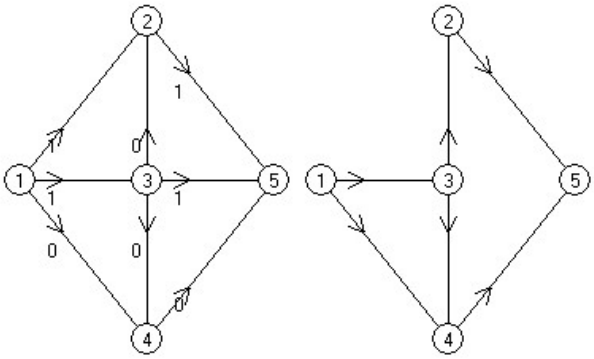

Figure 11: Iteration 1, v=2

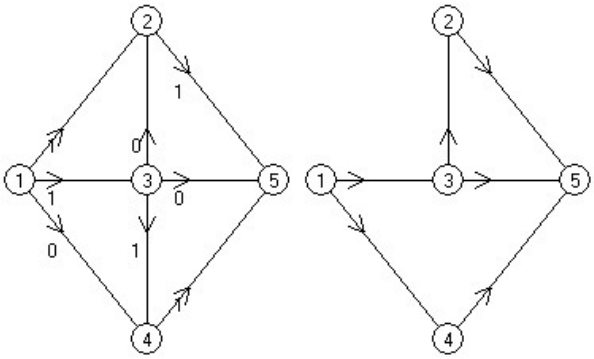

Figure 12: Iteration 1, v=2

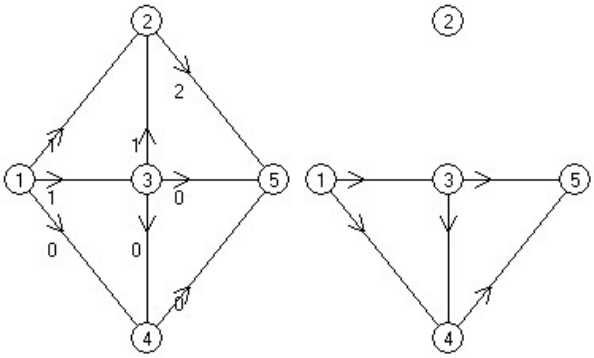

Figure 13: Iteration 1, v=2

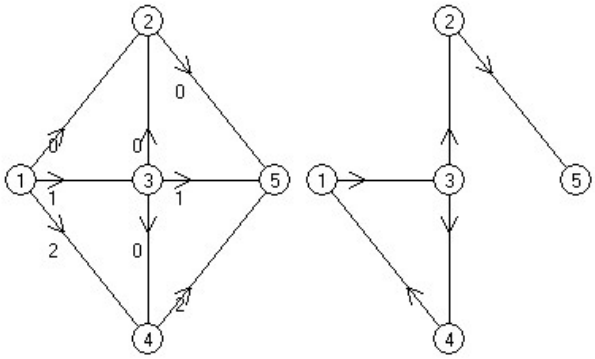

Figure 14: Iteration 2, v=3

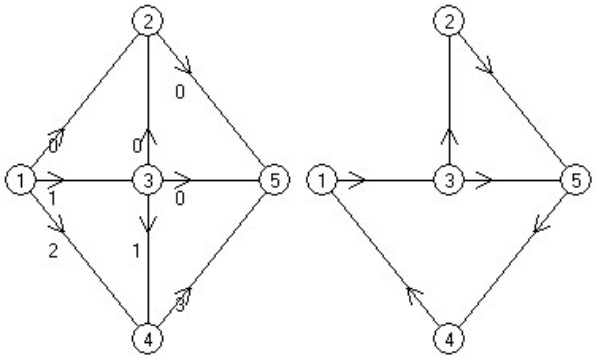

Figure 15: Iteration 2, v=3

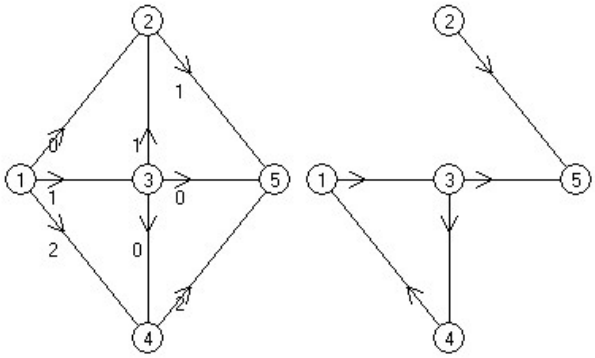

Figure 16: Iteration 2, v=3

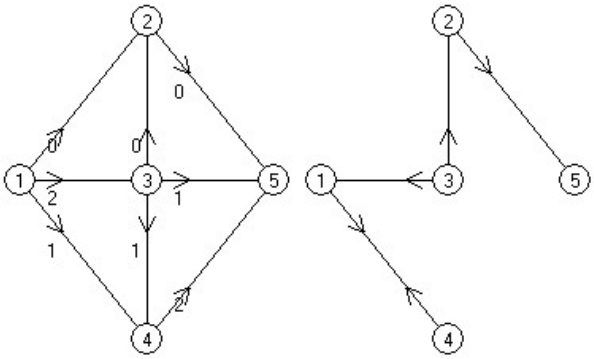

Figure 17: Iteration 2, v=3, first solution

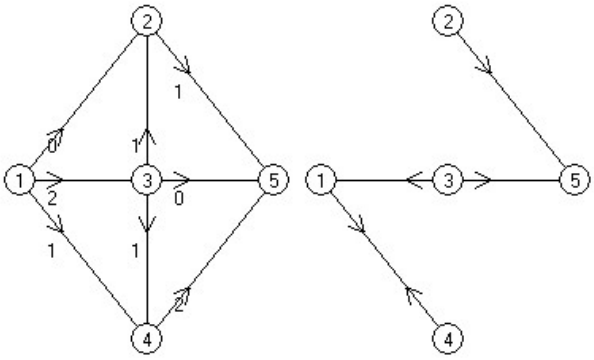

Figure 18: Iteration 2, v=3, second solution