Combinatorial Algorithms for Inverse Network Flow Problems

by

Ravindra K. Ahuja
James B. Orlin

SWP# 4004 February 1998

Combinatorial Algorithms for Inverse Network Flow Problems

Ravindra K. Ahuja[*]
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

James B. Orlin
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

(Revised January 25, 1997)

[*] On leave from Indian Institute of Technology, Kanpur 208 016, INDIA.

Combinatorial Algorithms for Inverse Network Flow Problems

Ravindra K. Ahuja[1] and James B. Orlin[2]

ABSTRACT

An inverse optimization problems is defined as follows: Let **S** denote the set of feasible solutions of an optimization problem **P**, let c be a specified cost vector, and x^0 be a given feasible solution. We want to perturb the cost vector c to d so that x^0 is an optimal solution of **P** with respect to the cost vector d , and $||d - c||_p$ is minimum, where $||.||_p$ denotes some selected L_p norm. In this paper, we consider inverse versions of the following network flow problems: the shortest path problem, the assignment problem, the minimum cut problem, and the minimum cost flow problem. We consider inverse problems under the L_1 norm (where the objective is to minimize $\sum_{j \in J} w_j|d_j - c_j|$), and under the L_∞ norm (where the objective is to minimize $\max\{w_j|d_j - c_j| : j \in J\}$). We show that the inverse version of each of the problem considered under the L_1 norm reduces to solving a problem for the same kind; that is, an inverse shortest path problem reduces to a shortest path problem, an inverse assignment problem reduces to an assignment problem, and so on. We next show that inverse versions of the shortest path problem, the assignment problem, and the minimum cost flow problem, under the L_∞ norm reduce to solving a minimum mean cycle problem (where we wish to identify a cycle whose cost divided by the number of arcs in it is minimum). We also consider the inverse minimum cut problem under the L_∞ norm and suggest a polynomial-time binary search algorithm.

[1] Sloan School of Management, MIT, Cambridge, MA 02139, USA; On leave from Indian Institute of Technology, Kanpur 208 016, INDIA.

[2] Sloan School of Management, MIT, Cambridge, MA 02139, USA.

1. INTRODUCTION

An inverse optimization problem is defined as follows: Let **S** denote the set of feasible solutions of an optimization problem **P**, let c be a specified cost vector, and x^0 be a given feasible solution. We want to perturb the cost vector c to d so that x^0 is an optimal solution of **P** with respect to the cost vector d, and $||d - c||_p$ is minimum, where $||.||_p$ denotes some selected L_p norm. In this paper, we consider inverse optimization problems under the weighted L_1 norm, where the objective is to minimize $\sum_{j \in J} w_j|d_j - c_j|$, and under the weighted L_∞ norm, where the objective is to minimize $\max\{w_j|d_j - c_j| : j \in J\}$. Here, w_j's are specified weights.

Inverse optimization problems have been investigated rather extensively in the past few years, and inverse versions of the following problems have been studied: shortest path problem, maximum capacity path problem, spanning tree problem, sorting problem, minimum cut and maximum flow problems, minimum cost flow problem, shortest arborescence problem, matroid intersection problem, and polymatroidal flow problem. Ahuja and Orlin [1998a] provide various references in the area of inverse optimization, compile several applications, and develop algorithms to solve the general inverse problem and inverse linear programming problems. In Ahuja and Orlin [1998b], they specialize their inverse linear programming algorithms to inverse network flow problems.

We will present a brief survey of the literature devoted to inverse network flow problems. Burton and Toint [1992, 1994], and Burton, Pulleyblank and Toint [1997] have considered inverse shortest path problems (multi-source, multi-sink problems) under the L_2 norm and solved them using nonlinear programming techniques. Cai and Yang [1994], Xu and Zhang [1995], and Zhang and Yang [1995] have considered inverse shortest path problems under the weighted L_1 norm; Yang and Zhang [1996] have considered maximum capacity path problems; Huang and Liu [1995] have studied the minimum cost flow problem under the weighted L_1 norm; Yang, Zhang and Ma [1997], and Zhang and Cai [1998] have considered the minimum cut problem under the weighted L_1 norm. Each of these problems reduce to solving a minimum cost flow problem. Sokkalingam [1996] in his doctoral dissertation, under the supervision of the first author, also considered the inverse minimum cost flow problem under the weighted L_1 and weighted L_∞ norms. Almost all of this research uses linear programming duality theory to solve inverse problems.

In this paper, we present combinatorial algorithms for solving inverse problems, in contrast with the linear programming based approaches suggested in the literature. We develop algorithms for solving inverse network flow problems using combinatorial arguments and not relying on the inverse linear programming duality theory. We further restrict attention (for most problems) to the unit weight inverse problems, that is, where $w_j = 1$ for each index $j \in J$. Though our resulting algorithms are identical to those given in Ahuja and Orlin [1998b], this approach provides shorter proofs of inverse algorithms and gives an additional insight into inverse network flow problems.

We consider a directed network $G = (N, A)$ with N as the node set and A as the arc set. Let $n = |N|$ and $m = |A|$. Each arc $(i, j) \in A$ has an associated cost c_{ij}, an associated capacity $u_{ij} > 0$, and perhaps an associated weight $w_{ij} > 0$. Let $\mathbf{C} = \max\{|c_{ij}| : (i, j) \in A\}$, Let $\mathbf{U} = \max\{u_{ij} : (i, j) \in A\}$, Let $\mathbf{W} = \max\{w_{ij} : (i, j) \in A\}$. Table 1 gives a list of inverse problems considered in this paper and the problem to which the considered inverse problem reduces.

Inverse Problem Considered:	**Reduces to solving the following problem:**
Inverse single-source, single-sink shortest path problem (unit weights, L_1 norm)	A single-source, single-sink shortest path problem
Inverse assignment problem (unit weights, L_1 norm)	An assignment problem
Inverse minimum cut problem (unit weights, L_1 norm)	A minimum cut problem
Inverse minimum cost flow problem (unit weights, L_1 norm)	A unit capacity circulation problem
Inverse shortest path, assignment and minimum cost flow problems (unit weights, L_∞ norm)	A minimum mean cycle problem
Inverse shortest path, assignment and minimum cost flow problems (general weights, L_∞ norm)	A minimum cost-to-weight ratio problem (also known as the tramp steamer problem)
Inverse minimum cut problem (unit weight, L_∞ norm)	$O(\log(n\mathbf{U}))$ minimum cut problems
Inverse minimum cut problem (general weights, L_∞ norm)	$O(\log(n\mathbf{UW}))$ minimum cut problems

Table 1. Summary of problems considered in this paper and the results obtained.

2. PRELIMINARIES

This paper relies on elementary results from the network flow theory, for which we refer the reader to the textbook by Ahuja, Magnanti and Orlin [1993]. The notations used in this paper are also adapted from the same book.

Let $\mathbf{P}$ denote the following optimization problem: $\min\{cx : x \in \mathbf{S}\}$, where $\mathbf{S}$ is the set of feasible solutions. Let $x^0 \in \mathbf{S}$ denote a feasible solution of $\mathbf{P}$ which we wish to make optimal to $\mathbf{P}$ by perturbing the arc cost vector c. We call a cost vector d to be *inverse feasible* for $\mathbf{P}$ (with respect to the solution x^0) if x^0 is an optimal solution of $\mathbf{P}$ when the cost vector c is replaced by the cost vector d. The inverse problem under the L_1 norm is to find an inverse feasible cost vector d^* of $\mathbf{P}$ for which $||d^* - c||_1 = \Sigma_{j \in J}|d_j^* - c_j|$

is minimum among all inverse feasible cost vectors d. The inverse problem under the L_∞ norm is to find an inverse feasible cost vector d^* of **P** for which $||d^* - c||_\infty = \max\{| d_j^* - c_j| : j \in J\}$ is minimum. We refer to d^* an *optimal cost vector* for the inverse problem. In this paper, we call the inverse problem under the L_1 norm as the *inverse problem*, and the inverse problem under the L_∞ norm as *the minimax inverse problem.*

In this paper, we denote $||d^* - c||_1$ by $|d^* - c|$. We will first prove a result that will be used several times in deriving combinatorial proofs of algorithms. Suppose that **P** is a 0-1 integer programming problem. Let x' denote a feasible 0-1 solution to **P**. The following lemma establishes a lower bound on the objective function value of the inverse problem, $|d^* - c|$, in terms of the costs of the solutions x' and x^0.

Lemma 1. $|d^* - c| \geq cx^0 - cx'$.

Proof. Notice that

$$|d^* - c| = |c - d^*| \geq (c - d^*)\cdot(x^0 - x') = (cx^0 - cx') + (d^*x' - d^*x^0) \geq cx^0 - cx', \qquad (2.1)$$

where the first inequality in (1) follows from the fact that both x^0 and x' are 0-1 vectors, and the second inequality follows from the fact that $dx' \geq dx^0$ (because x^0 is an optimal solution of **P** with d^* as the cost vector. ♦

Let x^* denote an optimal solution of **P** with c as the cost vector. Then, Lemma 1 implies that $cx^0 - cx^*$ is a lower bound on $|d^* - c|$, the optimal objective function value of the inverse problem. In fact, it gives the tightest possible lower bound on $|d^* - c|$. We will use this result several times in the following sections to prove the correctness of our algorithms.

3. THE INVERSE SHORTEST PATH PROBLEM

In this section, we study the single-source, single-sink shortest path problem under the L_1 norm. Cai and Yang [1994], Xu and Zhang [1995], and Zhang, Ma and Yang [1995] have studied various kind of inverse shortest path problems with weighted L_1 norm and showed that they reduce to minimum cost flow problems. Dial [1997] has studied the inverse shortest path problem in acyclic networks and showed that it can be solved in O(m) time. In this section, we show that the unit weight inverse shortest path problem reduces to a shortest path problem and can be solved far more efficiently than solving a minimum cost flow problem. We assume that the network G does not contain any negative cost cycle; without this assumption, the shortest path problem is a NP-complete problem.

The single-source, single-sink shortest path problem is to determine a directed path from node s to node t in G (henceforth called an *s-t path*) whose cost, given by $c(P) = \sum_{(i,j)\in P} c_{ij}$, is minimum among all s-t paths in G. For each s-t path P in G, we can associate a 0-1 flow x in the following manner: $x_{ij} = 1$ for each $(i, j) \in P$ and $x_{ij} = 0$ for each $(i, j) \notin P$. In the inverse shortest path problem, we are given an s-t path P^0 which we wish to make a shortest path from node s to node t by modifying the arc cost vector c to d^* in a manner such that $|d^* - c|$ is minimum.

Let σ denote the shortest path distances in G from node s to all other nodes with c_{ij}'s as arc costs, and let P^* denote a shortest s-t path. Then, $\pi = -\sigma$ gives the optimal dual variables for the shortest path problem. Let $d^\pi_{ij} = c_{ij} - \pi_i + \pi_j$ denote the reduced cost of any arc $(i, j) \in A$. It follows from the shortest path optimality conditions (see, for example, Ahuja, Magnanti and Orlin [1993]) that

$$c^\pi_{ij} = 0 \qquad \text{for each arc } (i,j) \in P^*; \tag{3.1a}$$

$$c^\pi_{ij} \geq 0 \qquad \text{for each arc } (i,j) \notin P^*. \tag{3.1b}$$

These conditions are also sufficient for the optimality of a solution. Now consider the following cost vector d^*, which we will show to be an optimal cost vector of the shortest path problem:

$$d^*_{ij} = \begin{cases} c_{ij} - c^\pi_{ij} & \text{for all } (i,j) \in P^0, \\ c_{ij} & \text{for all } (i,j) \notin P^0. \end{cases} \tag{3.2}$$

Let $d^\pi_{ij} = d^*_{ij} - \pi_i + \pi_j$ for each arc $(i, j) \in A$. It follows from (3.1) and (3.2) that $d^\pi_{ij} = 0$ for each arc $(i, j) \in P^0$ and $d^\pi_{ij} \geq 0$ for each arc $(i, j) \notin P^0$. It follows from the shortest path optimality conditions that d^* is an inverse feasible cost vector with respect to the solution x^0. In words, the above result implies that for each arc in P^0 we decrease the arc cost by an amount equal to the optimal reduced cost of the arc. The cost of every other arc remains unchanged. This change decreases the cost of the path P^0 by $\sum_{(i,j)\in P^0} c_{ij}^{\pi}$ units and does not affect the cost of the path P^* (because each arc in it has zero reduced cost). After this change, the modified reduced cost of each arc (i, j) in P^0 becomes 0, and it becomes an alternate shortest s-t path in G.

We next show that d^* is an optimal cost vector. Lemma 1 implies that $c(P^0) - c(P^*)$ is a lower bound on the optimal objective function value of the inverse problem. We will show that $|d^* - c| = c(P^0) - c(P^*)$ which would imply that d^* is an optimal cost vector. Note that

$$|d^* - c| = \sum_{(i,j)\in P^0} c_{ij}^{\pi} = \sum_{(i,j)\in P^0} c_{ij} - \pi_s + \pi_t = \sum_{(i,j)\in P^0} c_{ij} - \sum_{(i,j)\in P^*} c_{ij} = c(P^0) - c(P^*), \quad (3.3)$$

where the first equality in (3.3) follows from (3.2), the second equality follows from the fact that $c_{ij}^{\pi} = c_{ij} - \pi_i + \pi_j$ and that in the summation the intermediate π_i's cancel out, and the third equality follows from the facts that $\pi_s = 0$ and that π_t is the negative of the length of the path P^*. It follows from (3.3) and Lemma 1 that d^* is an optimal cost vector of the inverse shortest path problem.

We have shown above that the inverse shortest path problem can be solved by solving a shortest path problem. When all arc costs are non-negative, we can solve the shortest path problem in $O(m + n \log n)$ time using Fredman and Tarjan's [1984] implementation of Dijkstra's algorithm. In case some arc costs are negative, we can solve the shortest path problem in O(nm) time using the FIFO label correcting algorithm (see, for example, Ahuja, Magnanti and Orlin [1993]), or in $O(\sqrt{n}\, m \log \mathbf{C})$ time using Goldberg's [1995] algorithm.

4. THE INVERSE ASSIGNMENT PROBLEM

In this section, we study the inverse assignment problem. To the best of our knowledge, no one yet has studied the inverse assignment problem, except as a special case of the inverse minimum cost flow problem.

Let $G = (N_1 \cup N_2, A)$ be a bipartite directed network with $|N_1| = |N_2|$ and $A \subseteq N_1 x N_2$. The assignment problem is to identify an assignment M such that each node $i \in N_1$ is assigned to a distinct node $j \in N_2$ so that the cost of the assignment given by $\sum_{(i,j)\in M} c_{ij}$ is minimum. For each assignment M in G, we can associate a 0-1 flow x in the following manner: $x_{ij} = 1$ for each $(i, j) \in M$ and $x_{ij} = 0$ for each $(i, j) \notin M$. Let $c(M) = \sum_{(i,j)\in M} c_{ij}$. In the inverse assignment problem, we are given an assignment M^0 which we wish to make an optimal assignment by modifying the arc cost vector c to d^* in a manner such that $|d^* - c|$ is minimum.

Let M^* denote an optimal assignment in G and π denote the optimal dual variables. Let $c_{ij}^{\pi} = c_{ij} - \pi_i + \pi_j$ denote the reduced cost of any arc $(i, j) \in A$. The optimality conditions of the assignment problem imply that

$$c_{ij}^{\pi} = 0 \qquad \text{for each arc } (i, j) \in M^*; \tag{4.1a}$$

$$c_{ij}^{\pi} \geq 0 \qquad \text{for each arc } (i, j) \notin M^*. \tag{4.1b}$$

These conditions are also sufficient for the optimality of a solution. Now consider the following cost vector d^*, which we will show to be an optimal cost vector of the assignment problem:

$$d_{ij}^* = \begin{cases} c_{ij} - c_{ij}^{\pi} & \text{for all } (i,j) \in M^0, \\ c_{ij} & \text{for all } (i,j) \notin M^0. \end{cases} \tag{4.2}$$

Let $d_{ij}^{\pi} = d_{ij}^* - \pi_i + \pi_j$ for each arc $(i, j) \in A$. It follows from (4.1) and (4.2) that $d_{ij}^{\pi} = 0$ for each arc $(i, j) \in M^0$ and $d_{ij}^{\pi} \geq 0$ for each arc $(i, j) \notin M^0$, implying that d^* is an inverse feasible cost vector with respect to the solution M^0. In words, the above result implies that for each arc in M^0 we decrease the arc cost by an amount equal to the optimal reduced cost of the arc. The cost of every other arc remains unchanged. This change decreases the cost of the assignment M^0 by $\sum_{(i,j)\in M^0} c_{ij}^{\pi}$ units and does not affect the cost of the assignment M^* (because each arc in it has zero reduced cost). After this change, the modified reduced cost of each arc (i, j) in M^0 becomes 0, and it becomes an alternate optimal assignment in G.

Now note that

$$|d^* - c| = \sum_{(i,j)\in M^0} c^{\pi}_{ij} = c(M^0) - \sum_{(i,j)\in M^0} (\pi_i - \pi_j) = c(M^0) - \sum_{(i,j)\in M^*} (\pi_i - \pi_j)$$
$$= c(M^0) - c(M^*), \qquad (4.3)$$

where the first equality in (4.3) follows from (4.2), the second equality follows from the fact that $c^{\pi}_{ij} = c_{ij} - \pi_i + \pi_j$, the third equality follows from the fact both M^0 and M^* are perfect matchings, and the fourth equality follows from (4.1a) which implies that $c_{ij} = \pi_i - \pi_j$ for each $(i, j) \in M^*$. Using Lemma 1 in (4.3) yields that d^* is an optimal cost vector for the inverse assignment problem.

Currently, the best available strongly polynomial time bound to solve the assignment problem is $O(nm + n^2 \log n)$ and is attained by several algorithms (see, for example, Goldfarb [1985]). The best available weakly polynomial algorithms are due to Gabow and Tarjan [1989] and Orlin and Ahuja [1992], and they both runs in $O(\sqrt{n}\, m \log(n\mathbf{C}))$ time.

5. The Inverse Minimum Cut Problem

In this section, we study the inverse minimum cut problem. The inverse minimum cut problem has earlier been studied by Yang, Zhang, and Ma [1997], and Zhang and Cai [1998], and they show that the weighted version of the inverse minimum cut problem can be reduced to a minimum cost flow problem. We show that the unit weight version of the inverse problem reduces to solving a minimum cut problem.

Consider a network G = (N, A) where u_{ij}'s denote arc capacities and s and t are two specified nodes, called the *source* and *sink* nodes, respectively. In the network G, we define a *cut* as a set of arcs whose deletion disconnects the network into two or more components, and such that no subset of arcs in it has this property. This minimality property of a cut implies that a cut disconnects the network into exactly two components. We call a cut an *s-t cut* if the nodes s and t belong to different components. An alternate method to represent a cut is by using the sets of nodes in the two components it forms. Let S and $\overline{S}$ (with $\overline{S} = N - S$) denote the sets of nodes in the components defined by a cut containing the nodes s and t respectively. Then, we can represent the s-t cut as $[S, \overline{S}]$. Let $(S, \overline{S})$ denote the set of *forward arcs* in the cut, that is, $(S, \overline{S}) = \{(i, j) \in A : i \in S$ and

$j \in \bar{S}\}$, and $(\bar{S}, S)$ denote the set of *backward arcs* in the cut, that is, $(\bar{S}, S) = \{(i, j) \in A : i \in \bar{S}$ and $j \in S\}$. We define the *capacity* of the s-t cut $[S, \bar{S}]$ as the sum of the capacities of the forward arcs in the cut. We denote it by $u[S, \bar{S}]$, that is, $u[S, \bar{S}] = \sum_{\{(i,j) \in (S,\bar{S})\}} u_{ij}$. The *minimum cut problem* is to determine an s-t cut of minimum capacity. The inverse minimum cut problem is to modify the arc capacity vector u to d^* so that the cut $[S^0, \bar{S}^0]$ is a minimum cut with respect to the capacity vector d^* and $|d^* - u|$ is minimum.

We can associate the 0-1 solution x with each cut $[S, \bar{S}]$ in G in the following manner: $x_{ij} = 1$ for each $(i, j) \in (S, \bar{S})$ and $x_{ij} = 0$ for each $(i, j) \notin (S, \bar{S})$. It is easy to see that $ux = u[S, \bar{S}]$. We can hence use the result in Lemma 1. We will also use the following variant of the max-flow min-cut theorem in our analysis:

Property 1. *An s-t cut $[S^0, \bar{S}^0]$ is a minimum cut in G if and only if there exists a feasible flow y from node s to node t in G that saturates the cut $[S^0, \bar{S}^0]$, that is, $y_{ij} = u_{ij}$ for each arc $(i, j) \in (S^0, \bar{S}^0)$ and $y_{ij} = 0$ for each arc $(i, j) \in (\bar{S}^0, S^0)$.*

Let the network $G' = (N, A')$ be obtained from $G = (N, A)$ by deleting the backward arcs in the cut $[S^0, \bar{S}^0]$. In other words, $A' = A \backslash (\bar{S}^0, S^0)$. Observe that every flow y that saturates the cut $[S^0, \bar{S}^0]$ in G, we can define a corresponding flow y′ in G′ by setting $y'_{ij} = y_{ij}$ for each arc $(i, j) \in A \backslash (\bar{S}^0, S^0)$. Similarly, for every flow y′ saturating the cut $[S^0, \bar{S}^0]$ in G′, we can define a flow y saturating the same cut in G by setting $y_{ij} = y'_{ij}$ for each $(i, j) \in A \backslash (\bar{S}^0, S^0)$ and $y_{ij} = 0$ for each $(i, j) \in (\bar{S}^0, S^0)$. These observations together with Property 1 implies the following lemma:

Lemma 2. *The cut $[S^0, \bar{S}^0]$ is a minimum cut in G if and only if $[S^0, \bar{S}^0]$ is a minimum cut in G′.*

Lemma 2 allows us to solve the inverse minimum cut problem on the network G′ instead of the network G. It will be clear in the subsequent arguments why solving the inverse minimum cut problem in G′ is more straightforward than solving the inverse minimum cut problem in G. We first solve a maximum flow problem in G′. Let y^* denote the maximum flow and $[S^*, \bar{S}^*]$ be a minimum cut in G′. Let $v^* = u[S^*, \bar{S}^*]$

denote the maximum flow value. We know that the "net" flow across the cut $[S^0, \bar{S}^0]$ (as a matter of fact, across any s-t cut) equals v^*, that is,

$$\sum_{(i,j)\in(S^0,\bar{S}^0)} y^*_{ij} - \sum_{(i,j)\in(\bar{S}^0,S^0)} y^*_{ij} = v^* = u[S^*, \bar{S}^*]. \qquad (5.1)$$

We know that the cut $[S^0, \bar{S}^0]$ has no backward arcs in G′, that is, $(\bar{S}^0, S^0) = \phi$. Hence, (5.1) reduces to $\sum_{(i,j)\in(S^0,\bar{S}^0)} y^*_{ij} = u[S^*, \bar{S}^*]$. Further, by definition, $\sum_{(i,j)\in(S^0,\bar{S}^0)} u_{ij} = u[S^0, \bar{S}^0]$. Subtracting the preceding two equations yields:

$$\sum_{(i,j)\in(S^0,\bar{S}^0)} (u_{ij} - y^*_{ij}) = u[S^0, \bar{S}^0] - u[S^*, \bar{S}^*]. \qquad (5.2)$$

Let the capacity vector d^* be defined as follows:

$$d^*_{ij} = \begin{cases} y^*_{ij} & \text{for each arc } (i,j) \in (S^0, \bar{S}^0), \\ u_{ij} & \text{for each arc } (i,j) \notin (S^0, \bar{S}^0). \end{cases} \qquad (5.3)$$

Then the capacity of the cut $[S^0, \bar{S}^0]$ with respect to d^* is the same as the capacity of the cut $[S^*, \bar{S}^*]$, and the flow y^* saturates both the cuts. Therefore, the cut $[S^0, \bar{S}^0]$ is a minimum cut in G′ with respect to d^* as the arc capacity vector, implying that d^* is an inverse feasible cost vector for the inverse problem.

We next show that d^* is an optimal cost vector. Lemma 1 in the context of the minimum cut problem implies that $u[S^0, \bar{S}^0] - u[S^*, \bar{S}^*]$ is a lower bound on the optimal objective function value of the inverse problem. Now notice that

$$|d^* - u| = |u - d^*| = \sum_{(i,j)\in(S^0,\bar{S}^0)} (u_{ij} - y^*_{ij}) = u[S^0, \bar{S}^0] - u[S^*, \bar{S}^*], \qquad (5.4)$$

where the second equality follows from (5.3) and the third inequality follows from (5.2). Hence, d^* is an inverse feasible cost vector that attains the lower bound of $u[S^0, \bar{S}^0]$- $u[S^*, \bar{S}^*]$ on the objective function value of the inverse problem; therefore, it must be an optimal cost vector.

To summarize, we have shown that the inverse minimum cut problem reduces to solving a minimum cut problem. Currently, the fastest strongly polynomial bound to solve the minimum cut problem (and the maximum flow problem) is $O(nm \log(n^2/m))$ and is due to Goldberg and Tarjan [1986]. The best weakly polynomial bound to solve the maximum flow problem is $O(\min\{n^{2/3}, m^{1/2}\}m \log(n^2/m) \log \mathbf{U})$ and is due to Goldberg and Rao [1997].

6. The Inverse Minimum Cost Flow Problem

In this section, we study the inverse minimum cost flow problem. The weighted inverse minimum cost flow problem has earlier been studied by Huang and Liu [1995] and Sokkalingam [1996], and they show that it reduces to a minimum cost flow problem. We show that the unit weight inverse minimum cost flow problem reduces to a unit capacity minimum cost flow problem and thus can be solved more efficiently.

In the minimum cost flow problem on a network $G = (N, A)$, each arc $(i, j) \in A$ has an associated cost c_{ij} and an associated capacity u_{ij}, and each node i has an associated supply/demand $b(i)$. If $b(i) \geq 0$, then node i is a supply node; otherwise it is a demand node. The problem concerns determining the least cost shipment that meets the demands at demand nodes of the network by the available supplies at the supply nodes by sending a flow that honors arc capacities. We will assume in this section that for any node pair (i, j) both (i, j) and (j, i) do not belong to A. This condition is not necessary for our algorithm but simplifies our notation. Further, this condition can easily be imposed by inserting an artificial node on one of the arcs (i, j) or (j, i).

Our proof requires the construction of the residual network $G(x^0)$ defined with respect to the flow x^0. To construct it, we consider each arc $(i, j) \in A$ one by one, and add arcs to $G(x^0)$ in the following manner: (i) if $x^0_{ij} < u_{ij}$, then we add the arc (i, j) of cost c_{ij} to $A(x^0)$; (ii) if $x^0_{ij} > 0$, then we add the arc (j, i) with cost $-c_{ij}$; and (iii) $0 < x^0_{ij} < u_{ij}$, then we add the arcs (i, j) and (j, i) with costs c_{ij} and $-c_{ij}$ respectively. Using our assumption that for any pair of nodes i and j, both (i, j) and (j, i) do not belong to A, it is easy to observe one-to-one correspondence between arc costs in G and arc costs in $G(x^0)$. We represent the arc costs in $G(x^0)$ by placing a bar over the arc costs in G. For example, the arc cost vectors c and d in G are denoted by the cost vectors $\bar{c}$ and $\bar{d}$ in $G(x^0)$. We call an arc $(i, j) \in A(x^0)$ a *forward arc* if $(i, j) \in A$, and an arc $(i, j) \in A(x^0)$ a *reverse arc* if (j, i) $\in$ A. Notice that if an arc (i, j) is a forward arc in $G(x^0)$ then $\bar{c}_{ij} = c_{ij}$, and if the arc (i, j) is a reverse arc in $G(x^0)$ then $\bar{c}_{ij} = -c_{ji}$.

We define $|\bar{d} - \bar{c}|$ in the residual network somewhat differently than in G. For any arc (i, j) $\in$ A, the residual network $G(x^0)$ may contain both the arcs (i, j) and (j, i). Their costs have the same magnitudes but opposite sign, and changing the cost of one arc changes the cost of the other arc automatically by the same amount. We will assume that whenever both (i, j) and (j, i) are present, only one of these arcs will contribute to $|\bar{d} - \bar{c}|$. The following property follows directly from this definition.

Property 2. *If c and d are two arc cost vectors in G, and $\bar{c}$ and $\bar{d}$ are the two corresponding arc cost vectors in $G(x^0)$, then $|d - c| = |\bar{d} - \bar{c}|$.*

A byproduct of Property 2 is that d^* is an optimal cost vector in G if and only if $\bar{d}^*$ is an optimal cost vector in $G(x^0)$. Our algorithm uses the following well known results from the network flow theory.

Property 3. *The flow x^0 is an optimal flow of the minimum cost flow problem if and only if $G(x^0)$ contains no negative cost directed cycles (called negative cycles).*

Property 4. *Let π be any vector of size n and $\bar{c}^{\pi}_{ij}$ denote the arc reduced costs defined as $\bar{c}^{\pi}_{ij} = \bar{c}_{ij} - \pi_i + \pi_j$. Then, for any directed cycle W in $G(x^0)$, $\Sigma_{(i,j)\in W}\bar{c}^{\pi}_{ij} = \Sigma_{(i,j)\in W}\bar{c}_{ij}$. Consequently, if $\bar{c}^{\pi}_{ij} \geq 0$ for each arc (i, j) $\in A(x^0)$, then $G(x^0)$ does not contain any negative cycle.*

We will first determine an optimal cost vector $\bar{d}^*$ for the residual network $G(x^0)$ and then use it to obtain an optimal cost vector d^* for the network G. To prove the optimality of the cost vector $\bar{d}^*$, we will use Property 3. Let $\mathcal{W} = \{W_1, W_2, \dots, W_K\}$ denote any collection of arc-disjoint negative cycles in the residual network $G(x^0)$. We henceforth assume that an arc-disjoint collection of cycles does not contain both the arcs (i, j) and (j, i) for any node pair i and j. There is no loss of generality in this assumption because if a directed cycle W_p contains the arc (i, j) and another directed cycle W_q contains the arc (j, i), then the directed walk $\{W_p \cup W_q\} \backslash \{(i, j) \cup (j, i)\}$ can be decomposed into directed cycles satisfying the assumption and both have the same cost. Let $\bar{c}(W_k)$ denote the cost of the cycle W_k, that is, $\bar{c}(W_k) = \Sigma_{(i,j)\in W_k} \bar{c}_{ij}$, and $\bar{c}(\mathcal{W})$ denote the cost of the collection, that is, $\bar{c}(\mathcal{W}) = \sum_{k=1}^{K} \bar{c}(W_k)$. Observe that in an optimal cost vector $\bar{d}^*$, costs of arcs in any cycle W_k must increase by at least $-\bar{c}(W_k)$ units in order to eliminate this negative cycle. Hence

$$\Sigma_{(i,j)\in W_k} |\bar{d}_{ij}^* - \bar{c}_{ij}| \geq -\bar{c}(W_k). \tag{6.1}$$

Since all the cycles $W_1, W_2, \ldots, W_K$ are arc-disjoint, it follows from (6.1) that

$$|\bar{d}^* - \bar{c}| = \Sigma_{(i,j)\in A(x^0)} |\bar{d}_{ij}^* - \bar{c}_{ij}| \geq \sum_{k=1}^{K} \Sigma_{(i,j)\in W_k} |\bar{d}_{ij}^* - \bar{c}_{ij}| \geq -\sum_{k=1}^{K} \bar{c}(W_k) = -\bar{c}(\mathcal{W}), \tag{6.2}$$

establishing the following lemma:

Lemma 3. *Let $\mathcal{W}$ be any collection of arc-disjoint negative cycles in $G(x^0)$. Then, $-\bar{c}(\mathcal{W})$ is a lower bound on $|\bar{d}^* - \bar{c}|$.*

Now notice that any collection $\mathcal{W}$ of arc-disjoint cycles in $G(x^0)$ defines a 0-1 circulation in $G(x^0)$ obtained by sending unit flow along each of the cycles $W_1, W_2, \ldots, W_K$. (A *circulation* is a flow with zero supply/demand vector.) Also notice that any 0-1 circulation defines a collection of arc-disjoint cycles in $G(x^0)$. These observations imply that we can find a minimum cost collection of arc-disjoint cycles in $G(x^0)$ by solving a minimum cost circulation in $G(x^0)$ subject to the additional restriction that each arc capacity is 1. Let y^* denote the optimal 0-1 flow in $G(x^0)$ obtained by setting each arc capacity to 1. Let π denote the optimal dual variables, and $\bar{c}_{ij}^{\pi}$'s denote the optimal reduced costs. In the circulation y^*, arcs with $y_{ij}^* = 0$ are at their lower bounds and arcs with $y_{ij}^* = 1$ are at their upper bounds. This fact together with the minimum cost flow optimality conditions imply that

$$\bar{c}_{ij}^{\pi} \geq 0 \qquad \text{for all } (i, j) \text{ with } y_{ij}^* = 0, \tag{6.3a}$$

$$\bar{c}_{ij}^{\pi} \leq 0 \qquad \text{for all } (i, j) \text{ with } y_{ij}^* = 1, \tag{6.3b}$$

where $\bar{c}_{ij}^{\pi} = \bar{c}_{ij} - \pi_i + \pi_j$. We will show that the following cost vector $\bar{d}^*$ is an optimal cost vector for the residual network $G(x^0)$:

$$\bar{d}_{ij}^* = \begin{cases} \bar{c}_{ij} + |\bar{c}_{ij}^{\pi}| & \text{for all } (i, j) \text{ with } \bar{c}_{ij}^{\pi} < 0, \\ \bar{c}_{ij} & \text{otherwise.} \end{cases} \tag{6.4}$$

We will show that if we modify arc costs as given by (6.4), then the modified reduced cost of each arc $(i, j) \in A(x^0)$ becomes nonnegative. First consider any arc $(i, j) \in A(x^0)$ with $\bar{c}_{ij}^{\pi} < 0$. In this case, the cost of the arc increases by the amount $|\bar{c}_{ij}^{\pi}|$ and after the increase the modified reduced cost of the arc (i, j) becomes zero. Notice that increasing the cost of arc (i, j) affects the reduced cost of the arc (j, i) in case it is also present in the residual network $G(x^0)$. But since $\bar{c}_{ji}^{\pi} = -\bar{c}_{ij}^{\pi}$, the modified reduced cost of the arc (j, i) also becomes zero after the change. In the other case when $\bar{c}_{ij}^{\pi} \geq 0$ and the arc $(j, i) \notin A(x^0)$, cost of the arc (i, j) does not change and hence its modified reduced cost remains non-negative. Since all modified reduced costs are non-negative, it follows from Property 3 that the residual network $G(x^0)$ does not contain any negative cycle, implying that $\bar{d}^*$ is an inverse feasible cost vector.

We next prove that $\bar{d}^*$ is an optimal cost vector. Let $\mathcal{W}^* = \{W_1^*, W_2^*, \ldots, W_K^*\}$ denote the arc-disjoint collection of negative cycles with respect to the flow y*. Observe that

$$\begin{aligned} |\bar{d}^* - \bar{c}| &= \sum_{\{(i,j) \in A(x^0): \bar{c}_{ij}^{\pi} < 0\}} |\bar{c}_{ij}^{\pi}| = -\sum_{\{(i,j) \in A(x^0): y_{ij}^* = 1\}} \bar{c}_{ij}^{\pi} = -\sum_{k=1}^{K} \sum_{(i,j) \in W_k^*} \bar{c}_{ij}^{\pi} \\ &= -\sum_{k=1}^{K} \sum_{(i,j) \in W_k^*} \bar{c}_{ij} = -\bar{c}(\mathcal{W}^*), \end{aligned} \tag{6.5}$$

where the first equality follows from (6.4), the second equality follows from (6.3b), the third equality follows from the definition of $\mathcal{W}^*$, and the fourth equality follows from Property 4. The fact $|\bar{d}^* - \bar{c}| = -\bar{c}(\mathcal{W}^*)$ in view Lemma 3 implies that $\bar{d}^*$ is an optimal cost vector for $G(x^0)$.

We next convert the optimal cost vector $\bar{d}^*$ for $G(x^0)$ into an optimal cost vector d^* for G. It follows from the definition of the residual network that increasing the cost of a forward arc (i, j) $\in$ A(x^0) increases the cost of the arc (i, j) in G and increasing the cost of a reverse arc (j, i) decreases the cost of the arc (i, j) in G. These observations allow us to write an equivalent from of (6.5) as follows:

$$d^*_{ij} = \begin{cases} c_{ij} + |\bar{c}^{\pi}_{ij}| & \text{if } \bar{c}^{\pi}_{ij} < 0 \text{ and } (i,j) \text{ is a forward arc in } G(x^0), \\ c_{ij} - |\bar{c}^{\pi}_{ji}| & \text{if } \bar{c}^{\pi}_{ji} < 0 \text{ and } (j,i) \text{ is a reverse arc in } G(x^0), \\ c_{ij} & \text{otherwise.} \end{cases} \tag{6.6}$$

To summarize, we have reduced the inverse minimum cost flow problem into a minimum cost flow problem in a unit capacity network. The minimum cost flow problem in a unit capacity network is in general easier to solve than the general minimum cost flow problem. Using the successive shortest path algorithm, this minimum cost circulation problem can be solved in O(m(m + n log n)) time (see, for example, Ahuja, Magnanti, and Orlin [1993]). Using the cost scaling algorithm, this minimum cost circulation problem can be solved in O(O(min{$n^{5/3}$, $m^{3/2}$}log(n**C**)) time, using the algorithm due to Gabow and Tarjan [1989].

7. The Minimax Inverse Minimum Cost Flow Problem

In the minimax inverse minimum cost flow problem, our objective is to modify the cost vector c to d^* so that the given solution x^0 becomes a minimum cost flow in G and max{| d^*_{ij} − c_{ij}|: (i, j) $\in$ A} is minimum. We will subsequently refer to the objective function of this inverse problem by $||d^* - c||_\infty$. We will show that the minimax inverse minimum cost flow problem reduces to a minimum mean cycle problem in the residual network G(x^0) = (N, A(x^0)).

Similar to what we did in the previous section, we shall first determine an optimal cost vector $\bar{d}^*$ for the residual network G(x^0) and then use it to obtain an optimal cost vector d^* for the network G. Clearly, if G(x^0) does not contain any negative cycle with $\bar{c}$ as the arc cost vector, then $\bar{d}^* = \bar{c}$; otherwise, arc costs must be modified to eliminate such negative cycles. We will henceforth assume that G(x^0) contains a negative cost cycle. Let W^* be the minimum mean cycle in G(x^0), that is, a directed cycle in G(x^0) for which the mean cost given by $\bar{c}(W)/|W|$ is minimum among all directed cycles W in the network. Let $\mu^* = \bar{c}(W^*)/|W^*|$. By our assumption, $\mu^* < 0$.

Consider any negative cycle W in $G(x^0)$. Let $\bar{c}(W)$ denote the cost of this cycle, that is, $\bar{c}(W) = \sum_{(i,j)\in W} \bar{c}_{ij}$. Then in the optimal cost vector $\bar{d}^*$, costs of the arcs in the cycle W must increase by at least $-\bar{c}(W)$ units in order to eliminate this negative cycle. It is easy to see that if we wish to minimize the maximum increase in the cost of any arc cost, then each arc cost in the cycle W must increase by at least $-\bar{c}(W^*)/|W^*|$ units. Hence, every negative cycle W in $G(x^0)$ gives a lower bound of $-\bar{c}(W^*)/|W^*|$ on $\| \bar{d}^* - \bar{c} \|_\infty$, the objective function of the inverse problem. Clearly, the tightest possible bound on $\| \bar{d}^* - \bar{c} \|_\infty$ is $-\bar{c}(W^*)/|W^*| = -\mu^*$, provided by the minimum mean cycle W^*. We will show that this lower bound is achievable, that is, there exists a feasible cost vector $\bar{d}^*$ which satisfies $\| \bar{d}^* - \bar{c} \|_\infty = -\mu^*$.

We solve a minimum mean cycle problem in $G(x^0)$ whose solution gives the minimum mean cost μ^* and the dual variables π such that $\bar{c}^\pi_{ij} \geq \mu^*$ for each arc $(i, j) \in A(x^0)$. The preceding inequalities can be stated as

$$-\bar{c}^\pi_{ij} \leq -\mu^* \qquad \text{for every arc } (i, j) \in A(x^0). \tag{7.1}$$

Now consider the cost vector $\bar{d}^*$ defined as follows:

$$\bar{d}^*_{ij} = \begin{cases} \bar{c}_{ij} + |\bar{c}^\pi_{ij}| & \text{for all } (i, j) \text{ with } \bar{c}^\pi_{ij} < 0, \\ \bar{c}_{ij} & \text{otherwise.} \end{cases} \tag{7.2}$$

If follows from our discussion in the previous section that if we modify arc costs as in (7.2), then all modified reduced costs in $G(x^0)$ become nonnegative, and from Property 4, $G(x^0)$ contains no negative cost cycle. This establishes that $\bar{d}^*$ is an inverse feasible cost vector. Next notice that

$$\| \bar{d}^* - \bar{c} \|_\infty = \max\{|\bar{c}^\pi_{ij}| : \bar{c}^\pi_{ij} < 0\} = \max\{-\bar{c}^\pi_{ij} : \bar{c}^\pi_{ij} < 0\} \leq -\mu^*, \tag{7.3}$$

where the first equality follows from (7.2), the second equality follows from the fact that $\bar{c}^\pi_{ij} < 0$, and the third equality follows from (7.1). It follows from (7.3) that $-\mu^*$ is an upper bound on $\| \bar{d}^* - \bar{c} \|_\infty$. We showed earlier that $-\mu^*$ is a lower bound on the optimal objective function value of the inverse problem. Consequently, $\bar{d}^*$ is an optimal cost vector for the minimax inverse minimum cost flow problem in the residual network $G(x^0)$. The corresponding optimal cost vector d^* in the network G can be determined using (6.6).

To summarize, we have shown above that the minimax inverse minimum cost flow problem reduces to solving a minimum mean cycle problem. Since the shortest path problem and the assignment problem are special cases of the minimum cost flow problem, the minimax versions of these cases also reduce to a minimum mean cycle problem. Currently, the best available strongly polynomial time algorithm to solve the minimum mean cycle problem is an O(nm) algorithm due to Karp [1978], and the best available weakly polynomial time algorithm is an $O(\sqrt{n}\ m \log(n\mathbf{C}))$ algorithm due to Orlin and Ahuja [1992].

__Weighted Version__

We next study the weighted version of the minimax inverse minimum cost flow problem. In this problem, the objective function of the inverse problem is to minimize $\max\{w_{ij}|d^*_{ij} - c_{ij}|: (i, j) \in A\}$, where $w_{ij} > 0$ for each $(i, j) \in A$. As earlier, we will assume that the residual graph $G(x^0)$ contains a negative cycle.

Consider any negative cycle W in $G(x^0)$. Let $\bar{c}(W)$ denote the cost of this cycle. Then, in an optimal cost vector $\bar{d}^*$, costs of the arcs in the cycle W must increase by at least $-\bar{c}(W)$ units in order to eliminate this negative cycle. Suppose that we increase the cost of an arc $(i, j) \in W$ by α_{ij} units. Then, $\sum_{(i,j)\in W} \alpha_{ij} \geq -\bar{c}(W)$. The impact of this change on the objective function value will be $\max\{w_{ij}\alpha_{ij}: (i, j) \in W\}$. This impact will be minimum when each $w_{ij}\alpha_{ij}$ is the same for every arc $(i, j) \in W$, say, $w_{ij}\alpha_{ij} = T$. Substituting this result in $\sum_{(i,j)\in W} \alpha_{ij} \geq -\bar{c}(W)$ yields $\sum_{(i,j)\in W} T/w_{ij} \geq -\bar{c}(W)$, which can be restated as $T \geq -\bar{c}(W)/\tau(W)$, where $\tau(W) = \sum_{(i,j)\in W} \tau_{ij}$ with $\tau_{ij} = 1/w_{ij}$. We have thus shown that each negative cycle W provides a lower bound of $-\bar{c}(W)/\tau(W)$ on the optimal solution value of the inverse problem. Let W^* denote a directed cycle in $G(x^0)$ with the smallest value of $\bar{c}(W)/\tau(W)$; we call such a cycle *the minimum cost-to-weight ratio cycle*. Let $\mu^* = \bar{c}(W^*)/\tau(W^*)$. Then, $-\mu^*$ gives the greatest lower bound on the optimal objective function value of the inverse problem. While solving the minimum cost-to-time ratio cycle in $G(x^0)$ with c_{ij}'s as arc costs and τ_{ij}'s as arc times, we obtain both μ^* and a vector π so that $-\bar{c}^{\pi}_{ij}/\tau_{ij} \le -\mu^*$. We define the arc cost vector $\bar{d}^*$ as in (7.2). Using similar arguments as in the unit weight case, it can be shown that $\bar{d}^*$ is an optimal cost vector for the weighted minimax inverse minimum cost flow problem in $G(x^0)$. After obtaining the optimal cost vector $\bar{d}^*$ for $G(x^0)$, we convert it into an optimal cost vector d^* for G using (6.6).

We have thus shown that the weighted minimax inverse minimum cost flow problem can be solved by solving a minimum cost-to-weight ratio cycle problem. The minimum cost-to-weight ratio problem can be solved in $O(nm \log(\mathbf{CW}))$ time using Lawler's algorithm, or in $O(n^4 \log n)$ time using Meggido's [1979] algorithm, where $\mathbf{C} = \max\{c_{ij} : (i, j) \in A\}$ and $\mathbf{W} = \{w_{ij} : (i, j) \in A\}$. It can also be solved in $O(\sqrt{n}\, m \log^2(\mathbf{CW}))$ time using Goldberg's [1995] shortest path algorithm.

8. The Minimax Inverse minimum Cut Problem

We shall now study the minimax inverse minimum cut problem. This is the minimax version of the inverse minimum cut problem studied in Section 5, where the objective is to modify the capacity vector u to d^* so that the s-t cut $[S^0, \bar{S}^0]$ becomes a minimum cut in the network G and $\max\{d^*_{ij} - u_{ij}: (i, j) \in A\}$ is minimum. We will show that if all arc capacities are integer, then we can solve the minimax inverse minimum cut problem as a sequence of $O(\log(n\mathbf{U}))$ minimum cut problems. We will use the same notation in this section as used in Section 5.

To solve the inverse problem, we first delete the backward arcs in the cut $[S^0, \bar{S}^0]$ and denote the resulting network by G'. Lemma 2 implies that $[S^0, \bar{S}^0]$ is a minimum cut in G if and only if $[S^0, \bar{S}^0]$ is a minimum cut in G'. In view of this result, we can restrict attention to solving the inverse problem in G'. We next obtain a minimum cut $[S^*, \bar{S}^*]$ in G' by solving a maximum flow problem. If $u[S^0, \bar{S}^0] = u[S^*, \bar{S}^*]$, then $[S^0, \bar{S}^0]$ is also a minimum cut in G' and $d^* = c$; otherwise arc capacities must be modified.

Suppose that the optimal objective function value of the minimax inverse minimum cut problem is δ^*, that is, $\|d^* - u\|_\infty = \delta^*$. Observe that there exists an optimal solution of the inverse problem where the modified capacity d^*_{ij} of each arc $(i, j) \in$

$(S^0, \overline{S}^0)$ satisfies $d^*_{ij} = \max\{0, u_{ij} - \delta^*\}$ and the modified capacity d^*_{ij} of each arc $(i, j) \notin (S^0, \overline{S}^0)$ satisfies $d^*_{ij} = u_{ij} + \delta^*$; for (i) if $d^*_{ij} > \max\{0, u_{ij} - \delta^*\}$ for some arc $(i, j) \in (S^0, \overline{S}^0)$, then we can decrease d^*_{ij} to $(u_{ij} - \delta^*)$ and $[S^0, \overline{S}^0]$ remains a minimum cut in G', and (ii) if $d^*_{ij} < u_{ij} + \delta^*$ for some arc $(i, j) \notin (S^0, \overline{S}^0)$, then we can increase d^*_{ij} to $(u_{ij} + \delta^*)$ and $(S^0, \overline{S}^0)$ remains a minimum cut in G'; in either case the objective function value of the inverse problem, $\|d^* - u\|_\infty$, does not increase. Let $G'(\delta)$ denote the network obtained from G' by defining the arc capacities in the following manner:

$$d_{ij}(\delta) = \begin{cases} \max\{0, u_{ij} - \delta\} & \text{for each arc } (i, j) \in (S^0, \overline{S}^0), \\ u_{ij} + \delta & \text{for each arc } (i, j) \notin (S^0, \overline{S}^0). \end{cases} \tag{8.1}$$

Each value of δ for which $[S^0, \overline{S}^0]$ is a minimum cut in $G'(\delta)$ gives an inverse feasible cost vector $d(\delta)$. The minimax inverse problem is to find the minimum value of δ for which $d(\delta)$ is an inverse feasible cost vector. Let $\lambda[\delta]$ denote the capacity of a minimum cut in $G'(\delta)$. Now consider $\lambda[\delta]$ as a function of δ. As we increase the value of δ from zero, capacities of some s-t cuts decrease, but the capacity of the cut $[S^0, \overline{S}^0]$ decreases at the fastest rate. Eventually, $[S^0, \overline{S}^0]$ becomes a minimum cut and then remains a minimum cut until its capacity becomes zero. It is easy to see that $\lambda[\delta]$ is a piecewise linear convex function of δ and the minimax inverse problem is to find the minimum value of δ, say δ_{opt}, when $[S^0, \overline{S}^0]$ becomes a minimum cut in $G'(\delta)$. We refer to the points where the slope of the function $\lambda[\delta]$ changes as *breakpoints* of $\lambda[\delta]$. Clearly, δ_{opt} will be a breakpoint of the function $G'(\delta)$. We can determine this value by using a binary search algorithm and performing search on the possible values of δ and solving a minimum cut problem in $G'(\delta)$ at each search point. In the binary search algorithm, we start with the search interval $[0, \mathbf{U}]$, and at each iteration we reduce the search interval by a factor of 2 until eventually the search interval becomes so small that it contains exactly one breakpoint.

We now discuss how small the search interval should be before it is guaranteed to contain a single breakpoint. Observe that each linear segment in $G'(\delta)$ represents a line of the form $a + s\delta$, where both a and s are integer numbers and the slope of the line segment s satisfies $0 \le |s| \le m$ (recall our assumption that all arc capacities are integer). Consequently, a breakpoint, which is formed by the intersection of two line segments, say, $a_1 + s_1\delta$ and $a_2 + s_2\delta$, is a rational number of the form p/q, where both p and q are integer numbers and $q \le 2m$. Further, two distinct rational numbers with integer denominators bounded by 2m must differ by at least $1/4m^2$. This observation implies that we can terminate the binary search algorithm when the size of the search interval becomes less than $1/4m^2$. Hence we obtain a bound of $\log(4m^2\mathbf{U}) = O(\log(n\mathbf{U}))$ on the number of iterations performed by the binary search algorithm, and an overall bound of $O(O(T(n, m, \mathbf{U}) \log(n\mathbf{U}))$ on the running time of the algorithm, where $T(n, m, \mathbf{U})$ is the time needed to solve a minimum cut problem on a network with n nodes, m arcs, and maximum arc capacity **U**. Currently, $T(n, m, \mathbf{U}) = O(nm \log(n^2/m))$ due to Goldberg and Tarjan [1986] is the best strongly polynomial bound, and $T(n, m, \mathbf{U}) = O(\min\{n^{2/3}, m^{1/2}\}m \log(n^2/m) \log \mathbf{U})$ due to Goldberg and Rao [1997] is the best available weakly polynomial bound.

Weighted Version

We next study the weighted version of the minimax inverse minimum cut problem. In this problem, the objective function of the inverse problem is to minimize $\max\{w_{ij}| d^*_{ij} - c_{ij}| : (i, j) \in A\}$ where $w_{ij} > 0$ for each arc $(i, j) \in A$. As in the unit weight case, we delete the backward arcs in $[S^0, \overline{S}^0]$ to obtain the network G′. Let δ^* denote the optimal objective function value of the inverse problem. Using arguments similar to the unit weight case, it can be shown that there exists an optimal solution of the inverse problem where $d^*_{ij} = \max\{0, u_{ij} - \delta^*/w_{ij}\}$ for every arc $(i, j) \in (S^0, \overline{S}^0)$ and $d^*_{ij} = u_{ij} + \delta^*/w_{ij}$ for every arc $(i, j) \notin (S^0, \overline{S}^0)$. We define the network $G'(\delta^*)$ with arc capacities defined in the following manner:

$$d_{ij}(\delta) = \begin{cases} \max\{0, u_{ij} - \delta / w_{ij}\} & \text{for each arc } (i, j) \in (S^0, \overline{S}^0), \\ u_{ij} + \delta / w_{ij} & \text{for each arc } (i, j) \notin (S^0, \overline{S}^0). \end{cases} \tag{8.2}$$

We next use the binary search for δ in the interval $[0, \mathbf{UW}]$ to determine the minimum value δ^* of δ, for which $[S^0, \overline{S}^0]$ is a minimum cut in G′ and terminate the binary search when the length of the search interval is less than $1/4m^2\mathbf{W}^2$. At termination, $d^* = d(\delta^*)$. The running time of this method is $O(T(n, m, \mathbf{U}) \log(n\mathbf{UW}))$, where $T(n, m, \mathbf{U})$ is the time needed to solve a minimum cut problem with n nodes, m arcs, and maximum arc capacity **C**.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support from the Office of Naval Research under contract ONR N00014-96-1-0051 as well as a grant from the United Parcel Service. We also acknowledge the help of Don Wagner who raised some perceptive and fundamental questions that led to the pursuit of the research reported in this paper.

REFERENCES

Ahuja, R. K., T. L. Magnanti, and J. B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, NJ.

Ahuja, R. K., and J. B. Orlin. 1998a. Inverse optimization, Part I: Linear programming and general problem. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Ahuja, R. K., and J. B. Orlin. 1998b. Inverse optimization, Part 2: Network flow problems. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Burton, D., B. Pulleyblank, and Ph. L. Toint. 1997. The inverse shortest paths problem with upper bounds on shortest paths costs. In *Network Optimization*, edited by P. Pardalos, D. W. Hearn, and W. H. Hager, *Lecture Notes in Economics and Mathematical Systems*, Volume 450, pp. 156-171.

Burton, D., and Ph. L. Toint. 1992. On an instance of the inverse shortest paths problem. *Mathematical Programming* **53**, 45-61.

Burton, D., and Ph. L. Toint. 1994. On the use of an inverse shortest paths algorithm for recovering linearly correlated costs. *Mathematical Programming* **63**, 1-22.

Cai, M. and X. Yang. 1994. Inverse shortest path problems. Technical Report, Institute of Systems Sciences, Academia Sinica, Beijing, China.

Dial, B. 1997. Minimum-revenue congestion pricing, Part 1: A fast algorithm for the single-origin case. Technical Report, The Volpe National Transportation Systems Center, Kendall Square, Cambridge, MA 02142.

Ford. L. R., Jr., and D. R. Fulkerson. 1962. *Flows in Networks*. Princeton University Press, Princeton, NJ.

Fredman, M. L., and R. E. Tarjan. 1984. Fibonacci heaps and their uses in improved network optimization algorithms. *Proceedings of the 25th Annual IEEE Symposium on Foundations of Computer Science*, pp. 338-346.

Gabow, H. N., and R. E. Tarjan. 1989. Faster scaling algorithms for network problems. *SIAM Journal on Computing* **18**, 1013-1036.

Goldberg, A. V. 1995. Scaling algorithms for the shortest path problem. *SIAM Journal on Computing* **24**, 494-504.

Goldberg, A. V., and S. Rao. 1997. Length function for flow computation. Technical Report # 97-055, NEC Research Institute, 4 Independence Way, Princeton, NJ.

Goldberg, A. V., and R. E. Tarjan. 1986. A new approach to the maximum flow problem. *Proceedings of the 18th ACM Symposium on the theory of Computing*, pp. 136-146. Full paper in *Journal of ACM* **35**(1990), 873-886.

Goldfarb, D. 1985. Efficient dual simplex algorithms for the assignment problem. *Mathematical Programming* 33, 187-203.

Huang, S., and Z. Liu. 1995. On the inverse version of the minimum cost flow problem. Working Paper, Dept. of ISMT, School of Business and Management, Hong Kong University of Science and Technology, Hong Kong.

Karp, R. M. 1978. A characterization of the minimum cycle mean in a diagraph. *Discrete Mathematics* **23**, 309-311.

Lawler, E. L. 1966. Optimal cycles in doubly weighted linear graphs. *In Theory of Graphs: International Symposium*, Dunod, Paris, and Gordon and Breach, New York, pp. 209-213.

Meggido, N. 1979. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research* **4**, 414-424.

Orlin, J. B., and R. K. Ahuja. 1992. New scaling algorithms for the assignment and minimum cycle mean problems. *Mathematical Programming* **54**, 41-56.

Sokkalingam, P.T., 1996. *The Minimum Cost Flow Problem : Primal Algorithms and Cost Perturbations.* Unpublished Dissertation, Department of Mathematics, Indian Institute of Technology, Kanpur, INDIA.

Xu, S., and J. Zhang. 1995. An inverse problem of the weighted shortest path problem. *Japanese Journal of Industrial and Applied Mathematics* **12**, 47-59.

Yang, C., and J. Zhang. 1996. Inverse maximum capacity path with upper bound contraints. To appear in *OR Spektrum*.

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

Zhang, J., and M. C. Cai. 1998. Inverse problem of minimum cuts. *Mathematical Methods of Operations Research* **47**, No. 1.

Zhang, J., Z. Ma, and C. Yang. 1995. A column generation method for inverse shortest path problems, *ZOR-Mathematical Methods for Operations Research* **41**, 347-358.