Yugoslav Journal of Operations Research
26 (2016), Number 2, 159–171
DOI: 10.2298/YJOR141217015G

WEAKLY AND STRONGLY POLYNOMIAL ALGORITHMS FOR COMPUTING THE MAXIMUM DECREASE IN UNIFORM ARC CAPACITIES

Mehdi GHIYASVAND
Department of Mathematics, Faculty of Science, Bu-Ali Sina University, Hamedan, Iran
mghiyasvand@basu.ac.ir

Received: December 2014 / Accepted: May 2015

Abstract: In this paper, a new problem on a directed network is presented. Let D be a feasible network such that all arc capacities are equal to U. Given a $\tau > 0$, the network D with arc capacities $U - \tau$ is called the τ-network. The goal of the problem is to compute the largest τ such that the τ-network is feasible. First, we present a weakly polynomial time algorithm to solve this problem, which runs in $O(\log(nU))$ maximum flow computations, where n is the number of nodes. Then, an $O(m^2n)$ time approach is presented, where m is the number of arcs. Both weakly and strongly polynomial algorithms are inspired by McCormick and Ervolina(1994).

Keywords: Optimal Witness, Feasible Flow, Maximum Flow.

MSC: 90B18.

1. INTRODUCTION

A directed network $D = (N, A)$ is given, where N is a set of *nodes* and A is a set of ordered pairs of nodes, called *arcs*. We denote an arc from node i to node j by (i, j) and define the *flow* on arc (i, j) by x_{ij}. Let d_i be the *demand* at node i (if $d_i < 0$, then $-d_i$ is a *supply*). In this paper, we suppose that all arc capacities are equal to U. A flow x is called a *feasible flow* if it satisfies the following constraints:

$$\sum_{j\in N} x_{ji} - \sum_{j\in N} x_{ij} = d_i, \quad \forall\, i \in N, \tag{1}$$

$$0 \leq x_{ij} \leq U, \quad \forall\, (i,j) \in A. \tag{2}$$

Network D is called *feasible* if there exists a flow x satisfying the conditions (1) and (2). For a given $\tau > 0$, we define the network D with arc capacities $U - \tau$ as *the τ-network*. In this paper, we compute the value τ^*, which is the largest value of τ, such that the τ-network is feasible. Thus, τ^* is the maximum decrease in all arc capacities such that the new network is still feasible (we call this problem *the MDUAC-problem*). In this paper, we present weakly and strongly polynomial time algorithms to solve this problem. These algorithms are inspired by McCormick and Ervolina[10].

The most vital arc problem[2,3,4,5,6,8,9,11,12,14,15] is defined for the shortest path and maximum flow problems. In the shortest path problem, the most vital arc is an arc whose removal yields the greatest increase in the shortest distance between two given nodes. In the maximum flow problem, the most vital arc is an arc whose deletion causes the largest decrease in the maximum flow value. Hence, the main difference between the most vital arc and the MDUAC-problems is as follows: in the most vital arc problem, the capacity of one arc(which is the most vital arc) is decreased to zero, but, in the MDUAC-problem, the capacities of all arcs is decreased by τ^*.

This paper consists of four sections in addition to Introduction section. Section 2 reviews some results used in the subsequent sections. In Section 3, the smallest average cut is defined and the relationship between these cuts and the value of the maximum decrease in all arc capacities is proved. A weakly polynomial and a strongly polynomial time algorithm to solve the problem are presented in Sections 4 and 5, respectively.

The MDUAC-problem arises in numerous applications: transportation networks, electrical and power networks, telephone networks, national highway systems, rail networks, airline service networks, manufacturing and distribution networks, computer networks, airline reservation systems, etc In all of these networks, we wish to move some entity (electricity, a consumer product, a person or a vehicle, a message) from one point to another in an underlying network. Networks may have equal capacities where having a big capacity produce a big cost. In MDUAC-problem, the minimum capacity is computed so that the given network can move the given entities among nodes. Thus, our result can be useful for economic reasons. For example, a production and distribution company sends its productions from warehouses (as supply nodes) to retailers (as demand nodes). Each warehouse i has a certain supply, and each retailer has a certain demand. For doing this, the company hires U units of space of the lorries which are traveling between two nodes (i.e. an arc). For economic reasons, the manager of this company wants to decrease the hired space of the lorries. He/she asks to have maximum decrease in the hired space U such that the company still can send all productions from warehouses to retailers.

2. PRELIMINARIES

2.1. *Optimal witness*

If $S, T \subset N$ form a nontrivial partition of N (i.e., $S, T \neq \phi$, $S \cap T = \phi$, and $S \cup T = N$), then we define the *cut* (S, T) as $(S, T) = \{(i, j) \in A \mid i \in S \text{ and } j \in T\}$. Let $|T \to S|$ be the number of arcs from set T into set S. *The value of* (S, T) is defined as :

$$V(S,T) = \sum_{i \in S} d_i - U\,|(T,S)|\,. \tag{3}$$

Theorem 1 (Hoffman Theorem [7]). A network with constraints (1) and (2) is feasible if and only if for every cut (S, T), we have $V(S, T) \leq 0$.

Theorem 1 says that a network is feasible if it does not have any cut (S, T) with $V(S, T) > 0$, which is called a *witness*. A cut (S^*, T^*) is called an *optimal witness* if it maximizes $V(S, T)$ over all cuts. Thus $V(S^*, T^*) \leq 0$ if and only if the network is feasible.

2.2. *The feasible flow procedure*

The feasible flow procedure (see [1], Page 169) first chooses an initial flow $\widehat{x}$ satisfying (2) and computes $e_i = \sum_j \widehat{x}_{ji} - \sum_j \widehat{x}_{ij} - d_i$. The auxiliary network D^I is constructed as follow: A new source node s and arcs (s, k) for all nodes k that $e_k > 0$ are added to the network. The capacity of such arcs will be e_k. Also, a new sink node t and arcs (l, t) for all nodes l such that $e_l < 0$ (with capacity $-e_l$) are added. For each arc (i, j) in the original network, two arcs (i, j) and (j, i) with upper bounds $r_{ij} = u_{ij} - \widehat{x}_{ij}$ and $r_{ji} = \widehat{x}_{ij} - l_{ij}$ are introduced, where l_{ij}, u_{ij} are the lower and upper bounds on arc (i, j). Then, the maximum flow from s to t in the auxiliary network saturates all the source and sink arcs if and only if the original network is feasible. Therefore, the feasible flow procedure takes one maximum flow computation.

Theorem 2 [7]. If D is not feasible and $(s \cup S_0, t \cup T_0)$ is a min cut in D^I, then (T_0, S_0) is an optimal witness in D.

2.3. *McCormick and Ervolina's idea*

In this section, we briefly explain the idea of McCormick and Ervolina[10]. Let l_{ij} and u_{ij} be the lower and upper bounds on arc $(i, j) \in A$. Define the parametric network $D(\delta)$ as D with bounds

$$l_{ij} - \delta \leq x_{ij} \leq u_{ij} + \delta, \quad \textit{for each } (i,j) \in A.$$

A flow x is a circulation if only (1) is required. For a given circulation x, let

$$\delta(x) = \max\{0, \max_{(i,j)\in A}\{l_{ij} - x_{ij}, x_{ij} - u_{ij}\}\},$$

which is the smallest δ such that x is a feasible solution of $D(\delta)$. Define $\delta^* = \min\{\delta(x) \mid x \text{ is a circulation}\}$, it is the smallest δ such that $D(\delta)$ is feasible. For each cut (S,T), let $(S,T) = \{(i,j) \in A \mid i \in S, j \in T\}$, $\{S,T\} = (S,T) \cup (T,S)$, and

$$\overline{V}(S,T) = \frac{\sum_{(i,j)\in(S,T)} l_{ij} - \sum_{(i,j)\in(T,S)} u_{ij} + \sum_{(i,j)\in S} d_i}{|\{S,T\}|}.$$

A maximum mean cut is a cut (S^*, T^*) such that

$$\overline{V}^* = \overline{V}(S^*, T^*) = \max_{(S,T)} \overline{V}(S,T).$$

McCormick and Ervolina[10] showed (1) that cut (S,T) is a maximum mean cut if and only if for $\delta = \overline{V}(S,T)$, and (2) $\delta^* = \overline{V}^*$. Then, using these properties, they computed a maximum mean cut (S^*, T^*) and δ^*. In the next sections, we extend McCormick and Ervolina's idea to present weakly and strongly polynomial time algorithms to compute τ^*.

3. THE MAXIMUM DECREASE IN ALL ARC CAPACITIES

3.1. *Smallest average cuts*

Given a cut (S,T), define *the average of cut* (S,T) as follows:

$$\psi(S,T) = \frac{U\,|T \to S| - \sum_{i\in S} d_i}{|T \to S|} = U - \frac{\sum_{i\in S} d_i}{|T \to S|}. \tag{4}$$

A smallest average cut (S^*, T^*) is defined by the following:

$$\psi^* = \psi(S^*, T^*) = \min_{(S,T)} \psi(S,T).$$

3.2. *A relationship between τ^* and a smallest average cut*

Given a $\tau \geq 0$, let $V_\tau(S,T)$ be the value of cut (S,T) in τ-network, the following lemma shows a relationship between $V_\tau(S,T)$ and $V(S,T)$.

Lemma 3. For each cut (S,T), we have $V_\tau(S,T) = V(S,T) + \tau\,|T \to S|$.

Proof. By (3) and the definition of τ-network, we have

$$V_\tau(S,T) = \sum_{i \in S} d_i - (U - \tau)\,|T \to S| = V(S,T) + \tau\,|T \to S|. \quad \square$$

Theorem 4. $\tau^* = \psi^*$.

Proof. By (3) and the definition of ψ^*, for each cut (S,T), we have

$$\psi^* \leq \frac{U\,|T \to S| - \sum_{i \in S} d_i}{|T \to S|} = \frac{-V(S,T)}{|T \to S|},$$

which means, by Lemma 3,

$$V_{\psi^*}(S,T) \leq V(S,T) + \frac{-V(S,T)}{|T \to S|}\,|T \to S| = 0, \quad \textit{for each cut}\ (S,T).$$

Thus, by Theorem 1, ψ^*-network is feasible, but τ^* is the largest τ such that the τ-network is feasible. Hence, $\psi^* \leq \tau^*$.

Now, it is enough that we prove $\psi^* \geq \tau^*$. Define

$$\beta(S,T) = U\,|T \to S| - \sum_{i \in S} d_i = \sum_{(i,j) \in (T,S)} U - \sum_{i \in S} d_i. \tag{5}$$

Consider an arbitrary τ such that the τ-network is feasible. Let x be a feasible flow in it. Thus, by (1), for each cut (S,T), we get $\sum_{(i,j) \in (T,S)} x_{ij} - \sum_{(i,j) \in (S,T)} x_{ij} = \sum_{i \in S} d_i$, or

$$\sum_{(i,j) \in (S,T)} x_{ij} - \sum_{(i,j) \in (T,S)} x_{ij} + \sum_{i \in S} d_i = 0.$$

Hence, by (5), we get

$$\beta(S,T) = \sum_{(i,j) \in (S,T)} x_{ij} + \sum_{(i,j) \in (T,S)} (U - x_{ij}). \tag{6}$$

x is a feasible flow in the τ-network, so, $0 \leq x_{ij} \leq U - \tau$, for each $(i,j) \in A$. Thus, by (6),

$$\beta(S,T) \geq \sum_{(i,j) \in (S,T)} x_{ij} + \tau|T \to S| \geq \tau|T \to S|,$$

which means $\frac{\beta(S,T)}{|T \to S|} \geq \tau$. Hence, by (4), we get $\psi(S,T) \geq \tau$. Since (S,T) is an arbitrary cut, we get $\psi^* \geq \tau$. On the other hand, τ is arbitrary, which means $\psi^* \geq \tau^*$. $\square$

By Theorem 4, in order to compute τ^*, it is enough that we compute the value of a smallest average cut ψ^*.

Theorem 5. A cut (S, T) is a smallest average cut if and only if the $\psi(S, T)$-network is feasible.

Proof. Let (S, T) be a smallest average cut (i.e. $\psi^* = \psi(S, T)$). By Theorem 4, we get that the $\psi(S, T)$-network is feasible. Now, supposing that the $\psi(S, T)$-network is feasible. By definition of τ^*, it is the largest τ such that the the τ-network is feasible. Hence, by Theorem 4, cut (S, T) should be a smallest average cut. □

4. A WEAKLY POLYNOMIAL TIME ALGORITHM

If τ^* is an integer, then, it can be computed using a binary search in $[0, U]$. In this case, τ^* is computed using $O(\log U)$ maximum flow computations. But it is possible that τ^* is in range [0,1], so, in binary search, a finishing condition is necessary. In this section, we compute τ^* for general case that it can be any number in range $[0, U]$.

In Section 3, computing the value of the maximum decrease in arc capacities is reduced to computing a smallest average cut. In this section, we present a weakly polynomial time algorithm to find a smallest average cut.

Lemma 6. Let (S_1, T_1) and (S_2, T_2) be two cuts with different averages. If U and d_i's are integer, then $|\psi(S_1, T_1) - \psi(S_2, T_2)| \geq \frac{1}{m^2}$.

Proof. The cuts (S_1, T_1) and (S_2, T_2) have two different averages, so, $\psi(S_1, T_1) \neq \psi(S_2, T_2)$, which means

$$(U\,|T_1 \to S_1| - \sum_{i \in S_1} d_i)\,|T_2 \to S_2| \neq (U\,|T_2 \to S_2| - \sum_{i \in S_2} d_i)\,|T_1 \to S_1|.$$

Since U and d_i's are integers, we get

$$|\,(U\,|T_1 \to S_1| - \sum_{i \in S_1} d_i)\,|T_2 \to S_2| - (U\,|T_2 \to S_2| - \sum_{i \in S_2} d_i)\,|T_1 \to S_1|\,| \geq 1.$$

Thus, we have

$$|\psi(S_1, T_1) - \psi(S_2, T_2)| = \frac{|\,(U\,|T_1 \to S_1| - \sum_{i \in S_1} d_i)\,|T_2 \to S_2| - (U\,|T_2 \to S_2| - \sum_{i \in S_2} d_i)\,|T_1 \to S_1|\,|}{|T_1 \to S_1|\,|T_2 \to S_2|}$$

$$\geq \frac{1}{m^2}. \quad \square$$

Our algorithm to compute τ^* is presented in Algorithm 1. The algorithm maintains two bounds ∇ and Δ, such that the ∇-network is feasible, but Δ-network is infeasible. Since 0-network is feasible, but U-network is infeasible, initialization is $\nabla = 0$ and $\Delta = U$. In each iteration, the interval $[\nabla, \Delta]$ is reduced by half using $\tau := \frac{\nabla+\Delta}{2}$. If τ-network is feasible, then the algorithm lets $\nabla = \tau$, but if it is infeasible, the algorithm lets $\Delta = \tau$. This continues until $\Delta - \nabla \le \frac{1}{m^2}$. In the next lemmas, we show that Algorithm 1 computes τ^*.

```
Computing $\tau^*$:
   Begin
     Let $\nabla := 0$ and $\Delta := U$;
     Do until $(\Delta - \nabla \le \frac{1}{m^2})$;
     Begin
       Let $\tau := \frac{\nabla+\Delta}{2}$;
       If $\tau$-network is feasible then $\nabla = \tau$;
       Else $\Delta = \tau$;
     End;
     Let $\tau_0 = \Delta$;
     Let $(s \cup S_0, t \cup T_0)$ be a min cut in the feasible flow procedure w.r.t. $\tau_0$-network;
     Let $\tau_1 = \psi(T_0, S_0)$;
   End.
```

Algorithm 1. Computing τ^*.

Lemma 7. Let τ_0-network be infeasible and $\tau_1 = \psi(T_0, S_0)$, where $(s \cup S_0, t \cup T_0)$ is a minimum cut in the feasible flow procedure corresponding to τ_0-network, then $\tau_1 < \tau_0$.

Proof. Since $(s \cup S_0, t \cup T_0)$ is a minimum cut in the feasible flow procedure corresponding to τ_0-network, we get, by Theorem 2, (T_0, S_0) as an optimal witness in τ_0-network. $V_{\tau_0}(T_0, S_0)$ is the value of the cut (T_0, S_0) in τ_0-network and τ_0-network is infeasible, so, by Theorem 1, we have

$$V_{\tau_0}(T_0, S_0) > 0. \tag{7}$$

By Lemma 3, for cut (T_0, S_0) in τ_0-network, we have $V_{\tau_0}(T_0, S_0) = V(T_0, S_0) + \tau_0\,|S_0 \to T_0|$, or $\frac{V_{\tau_0}(T_0,S_0)}{|S_0 \to T_0|} = \frac{V(T_0,S_0)}{|S_0 \to T_0)|} + \tau_0$, which means, by (3) and (4),

$$\frac{V_{\tau_0}(T_0, S_0)}{|S_0 \to T_0|} = -\psi(T_0, S_0) + \tau_0.$$

Hence, by (7), we get $-\psi(T_0, S_0) + \tau_0 > 0$, or $\tau_1 < \tau_0$. □

Lemma 8. At the end of Algorithm 1, we have $\tau^* = \tau_1$.

Proof. Since ∇-network is feasible, but Δ-network is infeasible, by the definition of τ^*, we have

$$\nabla \leq \tau^* < \Delta. \tag{8}$$

At the end of Algorithm 1, $\tau_0 = \Delta$ and $\tau_1 = \psi(S_0, T_0)$, where $(s \cup S_0, t \cup T_0)$ is a minimum cut in the feasible flow procedure corresponding to τ_0-network, so, by Lemma 7,

$$\tau_1 < \tau_0 = \Delta. \tag{9}$$

Supposing that (for the sake of contradiction), τ_1-network is infeasible, which means $\tau^* < \tau_1$. Thus, by (8) and (9), we get

$$\nabla \leq \tau^* < \tau_1 < \Delta. \tag{10}$$

At the end of Algorithm 1, $\Delta - \nabla < \frac{1}{m^2}$. Hence, by (10), $\tau_1 - \tau^* < \frac{1}{m^2}$, contradicting Lemma 6 (note, $\tau_1 = \psi(T_0, S_0)$ and $\tau^* = \psi^*$). Consequently, τ_1-network is feasible, which means, by Theorem 5, (T_0, S_0) is a smallest average cut. Thus, by Theorem 4, $\tau^* = \psi(T_0, S_0)$. □

The next theorem presents the running time of Algorithm 1.

Theorem 9. Algorithm 1 runs in $O(\log(nU))$ maximum flow computations.

Proof. Starting values are $\nabla = 0$ and $\Delta = U$. The algorithm finishes when $\Delta - \nabla < \frac{1}{m^2}$, so, the number of iterations is $O(\log(m^2U)) = O(\log(nU))$. In each iteration, the algorithm computes a min cut, which needs a maximum flow computation. □

5. A STRONGLY POLYNOMIAL TIME ALGORITHM

In this section, we extend the idea of McCormick, and Ervolina[10] to present a strongly polynomial time algorithm to compute τ^*. Algorithm 2 shows the method. It computes cuts (S_0, T_0), (S_1, T_1),..., (S_k, T_k),... and values τ_0, τ_1,...,τ_k,... so that (S_k, T_k) is an optimal witness in τ_k-network. It is obvious that if $\psi(S_i, T_i)$-network is feasible, then, by Theorems 4 and 5, we have $\tau^* = \psi(S_i, T_i)$.

By definitions, $V_x(S, T)$ is the value of cut (S, T) in x-network. For notational convenience, for each $i = 1, 2, ...,$ define

$$v_k(x) = V_x(S_k, T_k),$$
$$n_k = |T_k \to S_k|, \text{ and}$$
$$v_k = -V(S_k, T_k).$$

Thus, by Algorithm 2,

$$\tau_{k+1} = \frac{v_k}{n_k}, \tag{11}$$

and, by Lemma 3,

$$v_k(x) = -v_k + x\, n_k. \tag{12}$$

Also, define

$$\theta_k = \frac{v_k(\tau_k)}{v_{k-1}(\tau_{k-1})}. \tag{13}$$

Some properties of (11), (12), and (13) are presented in the next lemma, which plays an important role to prove that Algorithm 2 is indeed a strongly polynomial-time algorithm.

Lemma 10.
a. τ_i is strictly decreasing during Algorithm-2.
b. $v_k(\tau_{k-1}) > v_k(\tau_k)$.
c. $v_{k-1}(\tau_k) = 0$.
d. $v_{k-1}(\tau_{k-1}) \geq v_k(\tau_{k-1})$.
e. $0 < \theta_k < 1$.

Proof.
a. For each i, τ_i-network is infeasible and $\tau_{i+1} = \psi(T_0, S_0)$, where $(s \cup S_0, t \cup T_0)$ is a minimum cut in the feasible flow procedure corresponding to τ_i-network, so, by Lemma 7, $\tau_{i+1} < \tau_i$.

b. By (12), $v_k(x)$ decreases linearly as x decreases, so, by (a), $v_k(\tau_{k-1}) > v_k(\tau_k)$.

c. By (12), $v_{k-1}(\tau_k) = -v_{k-1} + \tau_k\, n_{k-1}$, so, by (11), $v_{k-1}(\tau_k) = 0$.

d. By Algorithm 2 and Theorem 2, (S_{k-1}, T_{k-1}) is an optimal witness in τ_{k-1}-network. On the other hand, $v_{k-1}(\tau_{k-1}) = V_{\tau_{k-1}}(S_{k-1}, T_{k-1})$ and $v_k(\tau_{k-1}) = V_{\tau_{k-1}}(S_k, T_k)$ are the values of cuts (S_{k-1}, T_{k-1}) and (S_k, T_k) in τ_{k-1}-network. Thus, $v_{k-1}(\tau_{k-1}) \geq v_k(\tau_{k-1})$.

e. If $v_k(\tau_k) = -v_k + \tau_k\, n_k \le 0$, then $\frac{v_k}{n_k} \ge \tau_k$, so, by (11), we get $\tau_{k+1} \ge \tau_k$, contradicting (a). Thus, $v_k(\tau_k) > 0$. Similarly, $v_{k-1}(\tau_{k-1}) > 0$, so, by (14), $\theta_k > 0$.

If $\theta_k \ge 1$, then $v_{k-1}(\tau_{k-1}) \le v_k(\tau_k)$ contradicting (b) and (d). □

The next theorem presents an upper bound for the number of iterations in Algorithm 2.

Theorem 11. Algorithm 2 terminates after at most m iterations.

Proof. By (12), we get:

$$v_{k-1}(\tau_{k-1}) = -v_{k-1} + \tau_{k-1}\, n_{k-1},$$
$$v_{k-1}(\tau_k) = -v_{k-1} + \tau_k\, n_{k-1}.$$

Thus, $v_{k-1}(\tau_{k-1}) - v_{k-1}(\tau_k) = (\tau_{k-1} - \tau_k)n_{k-1}$, which means

$$n_{k-1} = \frac{v_{k-1}(\tau_{k-1}) - v_{k-1}(\tau_k)}{\tau_{k-1} - \tau_k}. \tag{14}$$

By Lemma 10(a), the denominator of (14) is not zero. Similarly,

$$n_k = \frac{v_k(\tau_{k-1}) - v_k(\tau_k)}{\tau_{k-1} - \tau_k}.$$

Hence, by (14), we have

$$n_{k-1} = n_k \frac{v_{k-1}(\tau_{k-1}) - v_{k-1}(\tau_k)}{v_k(\tau_{k-1}) - v_k(\tau_k)}.$$

By Lemma 10(b), the denominator here can not be zero. Thus, by Lemma 10(c,d),

$$n_{k-1} \ge n_k \frac{v_{k-1}(\tau_{k-1})}{v_{k-1}(\tau_{k-1}) - v_k(\tau_k)}.$$

Now, by (13), we get

$$n_{k-1} \ge n_k \frac{v_{k-1}(\tau_{k-1})}{v_{k-1}(\tau_{k-1})(1-\theta_k)} = \frac{n_k}{1-\theta_k},$$

which means $n_k \le n_{k-1}(1-\theta_k)$. Therefore, by Lemma 10(e), we get $n_k < n_{k-1}$. Since the n_k's are integers and n_0 is at most m, Algorithm 2 terminates after at most m iterations. □

Algorithm 2:

```
Begin
  Let $\tau_0 = U$ and $i = 0$;
```

```
    Construct $\tau_i$-network;
    **Do until** (the $\tau_i$-network is feasible);
    **Begin**
        Let $(s \cup S_0, t \cup T_0)$ be a min cut in the feasible flow procedure w.r.t. $\tau_i$-network;
        Let $(S_i, T_i) = (T_0, S_0)$ and $\tau_{i+1} = \psi(S_i, T_i)$;
        Let $i = i + 1$;
    **End**
    $\tau^* = \tau_i$;
**End**.
```

Algorithm 2. Strongly polynomial time algorithm to compute τ^*.

By Theorem 11, Algorithm 2 terminates after at most m maximum flow computations. Recently, Orlin[14] presented an $O(mn)$ time algorithm to solve the maximum flow problem. Therefore, our algorithm runs in $O(m^2n)$ time.

Example 1. In this example, an implementation of Algorithm 1 for the network in Fig. 1 is presented. We have $\Delta = U = 8$, $\nabla = 0$, $\frac{1}{m^2} = 1/64 = 0.015625$, and $\Delta - \nabla = 8 > \frac{1}{m^2}$. Let $\tau = \frac{\nabla+\Delta}{2} = 4$. The 4-network is infeasible, so, we get $\Delta = 4$ and $\tau = \frac{\nabla+\Delta}{2} = \frac{0+4}{2} = 2$. In 2-network (see Fig. 2), the following is a feasible flow:

$$x_{23} = x_{31} = x_{25} = x_{54} = 5,\ x_{35} = 2,\ x_{12} = x_{42} = x_{34} = 0, \tag{15}$$

which means 2-network is feasible. Hence, $\nabla = 2$ and $\tau = \frac{\nabla+\Delta}{2} = \frac{2+4}{2} = 3$. Fig. 3 shows 3-network, which is feasible (since (15) is also a feasible flow in 3-network), which means $\nabla = 3$ and $\tau = \frac{\nabla+\Delta}{2} = \frac{3+4}{2} = 3.5$. The 3.5-network is infeasible. In a similar way, the 3.125-network, the 3.125-network, the 3.0625-network, the 3.03125-network, and the 3.015626-network are infeasible. Thus, $\Delta = 3.015625$ and $\tau = \frac{\nabla+\Delta}{2} = \frac{3+3.015625}{2} = 3.0078125$, which means $\Delta = 3.0078125$. Current values are

$$\nabla = 3 \ \textit{and} \ \Delta = 3.0078125,$$

which means $\Delta - \nabla = 0.0078125 < \frac{1}{m^2} = 0.015625$. Hence, by Algorithm 1, current iteration is the final one. By the feasible flow procedure, a minimum cut is

$$(s \cup S_0, t \cup T_0) = (s \cup \{2\},\ t \cup \{1, 3, 4, 5\}).$$

Therefore,

$$\tau^* = \psi(T_0, S_0) = U - \frac{\sum_{i \in T_0} d_i}{|S_0 \to T_0|} = 8 - \frac{10}{2} = 3.$$

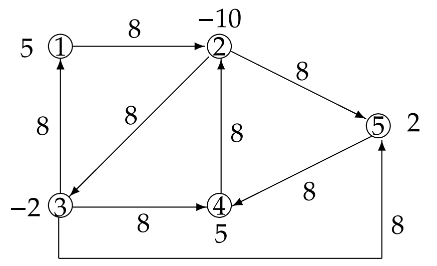

Figure 1: An example network

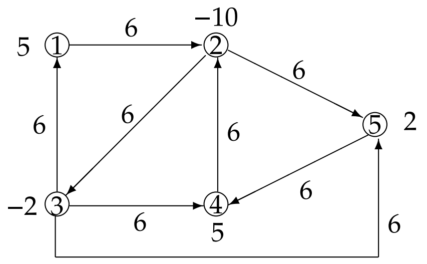

Figure 2: The 2-network.

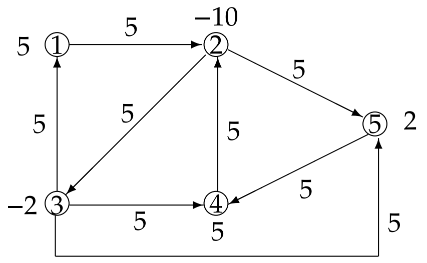

Figure 3: The 3-network.

Acknowledgements: I would like to express great appreciation to Reviewer-B for his/her valuable comments and suggestions, which have helped to improve the quality and presentation of this paper.

REFERENCES

[1] Ahuja,R.K., Magnanti, T.L., and Orlin, J.B., *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.

[2] Aneja, Y.P., Chandrasekaran, R., and Nair, K.P.K., "Maximizing residual flow under an arc destruction", *Networks*, 38 (2001) 194-198.

[3] Ball, M.O., Golden, B.L., and Vohra, R.V., "Finding the most vital arcs in a network", *Operations Research Letters*, 8 (1989) 73-76.

[4] Bar-Noy, A., Khuller, S., and Schieber, B., "The complexity of finding most vital arcs and nodes", TR CS-TR-3539, Institute for Advanced Studies, University of Maryland, College Park, MD, 1995.

[5] Barton, A.J., “Addressing the problem of finding a single vital edge in a maximum flow graph”, NRC/ERB-1129, 2005.
[6] Corley, H.W., and Sha, D.Y., “Most vital links and nodes in weighted networks”, *Operations Research Letters*, 1 (1982), 157-160.
[7] Hoffman, A.J., “Some recent applications of the theory of linear inequalities to extremal combinatorial analysis”, *American Mathematical Society*, 10 (1960) 113-127.
[8] Ivanchev,D., “Finding the k most vital elements of an s-t planar directed network”, *Yugoslav Journal of Operations Research*, 10 (2000) 13-26.
[9] Malik, K., Mittal, A.K., and Gupta, S.K., “The K most vital arcs in the shortest path problem”, *Operations Research Letters*, 8 (1989) 223-227.
[10] McCormick, S.T., and Ervolina, T.R., “Computing Maximum Mean Cuts”, *Discrete Applied Math*, 52 (1994) 53-70.
[11] Nardelli, E., Proietti, G., and Widmayer, P., “Finding the detour-critical edge of a shortest path between two nodes”, *Information Processing Letters*, 67 (1998) 51-54.
[12] Nardelli, E., Proietti, G., and Widmayer, P., “A faster computation of the most vital edge of a shortest path between two nodes”, *Information Processing Letters*, 79 (2001) 81-85.
[13] Nardelli, E., Proietti, G., and Widmayer, P., “Finding the most vital node of a shortest path”, *Theoretical Computer Science*, 296 (2003) 167-177.
[14] Orlin, J.B., “Max flows in O(mn) time, or better”, *STOC* (2013) 765-774.
[15] Ratliff, D., Sicilia, T., and Lubore, H., “Finding the n most vital links in flow networks”, *Management Science*, 21 (1975) 531-539.