

# Maximizing Residual Flow under an Arc Destruction

Y.P. Aneja\*, R. Chandrasekaran<sup>†</sup>, K.P.K. Nair<sup>‡</sup>

## Abstract

In this paper we consider two problems related to single commodity flows on a directed network. In the first problem, for a given  $s - t$  flow, if an arc is destroyed, all the flow that is passing through that arc is destroyed. What is left flowing from  $s$  to  $t$  is the residual flow. The objective is to determine a flow pattern such that the residual flow is maximized. We provide a strongly polynomial algorithm for this problem, and consider various extensions of this basic model. In the second problem, known as the “most vital arc” problem, the objective is to remove an arc so that the maximal flow on the residual network is as small as possible. Results are derived which help implement efficient scheme for solving this problem also.

## 1 Introduction

It is not uncommon that in a communication or a transportation network, some nodes or arcs fail due to some random natural causes. One can also conceive of such failures of certain components of the network due to vandalism, sabotage or some action of an adversary. Any flow that was going through the destroyed nodes or arcs is lost. Generally, it is not easy to reorient flows quickly to take advantage of alternate routes for sending flows. Hence assuming that the flows can not be reoriented, the *residual flow* going from a source  $s$  to a sink  $t$  is the flow that was not using any of the destroyed nodes and/or arcs. The objective is, therefore, to determine an  $s - t$  flow such that the residual flow value is maximized. To the best of our knowledge this model has not been studied in the literature. In the first part of the paper we consider this problem, restricting to the case where only one arc is destroyed.

The second problem that relates to the above problem is known in network literature as the “most vital arc” problem [1], where as mentioned in the abstract, the objective is to determine an arc whose removal minimizes the maximum  $s - t$  flow on the residual network. This is different from the first problem and the difference lies in the order in which the flow and failure take place. In the vital arc problem the failure is announced first and the flow router knows this information. In Problem 1, the flows occur first and the failure mechanism knows this information. The following example in Figure 1 illustrates the difference between the two problems.

---

\*Faculty of Business Administration, University of Windsor, Windsor, Ontario, Canada N9B 3P4

<sup>†</sup>School of Management, University of Texas at Dallas, Richardson, Texas, U.S.A..

<sup>‡</sup>Faculty of Administration, University of New Brunswick, Fredericton, N.B., Canada E3B5A3

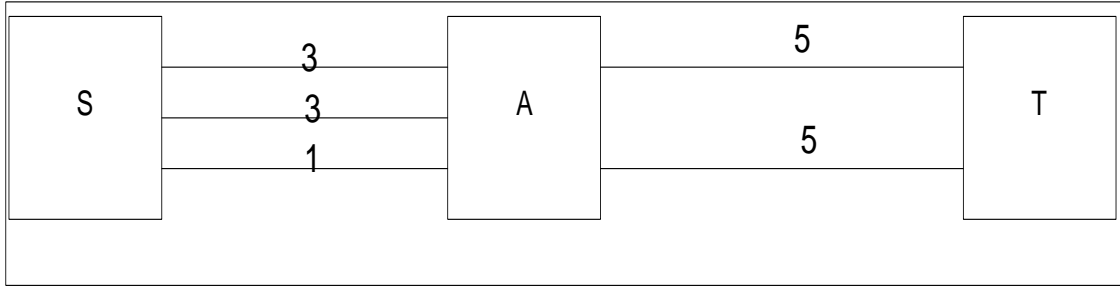


Figure 1: Examples of the Two Problems

In the vital arc problem we remove one of the two  $(S, A)$  arcs with capacity 3 and the surviving flow is 4. In problem 1, in the optimal flow pattern, arcs with capacities 3, 3, and 1 will be saturated and the two arcs of the  $(A, T)$  will each carry a flow of 3.5. The failure that maximizes the loss is one of these last two arcs. The net surviving flow is 3.5. Thus the two problems are quite different. Further, this example shows that the optimal flow in Problem 1 need not be integral.

In the next section, we define Problem 1 formally restricting our analysis to removal of only one arc. We then show that for an optimal solution, we can restrict ourselves to flows that are acyclic (containing no directed flow cycle) and maximal (in terms of input flow value). We then formulate the problem as a parametric flow problem and provide a strongly polynomial algorithm for the problem. Section 3 considers various extensions of this model.

In section 4 we consider Problem 2, the “most vital arc” problem. In order to improve on the existing approach, we show various properties of the optimal solution to help improve the search. We first show conditions under which an optimal flow for the Problem 1 readily provides an optimal solution for this problem. We then provide a simple procedure for finding a good lower bound on the capacity of an optimal arc. Finally, we show that if the problem is defined over an undirected network, further characterization of an optimal arc reduces the search considerably.

## 2 Problem 1: Formulation and Solution Method

Let  $G = [N, A, c]$  be a directed network with capacities on arcs  $c : A \mapsto R$ . Let  $s, t \in N$ . Let  $f : A \mapsto R$  be a flow from  $s$  to  $t$ , with value  $v(f)$  satisfying the flow conservation conditions for nodes other than  $s$  and  $t$ , and capacity constraints. That is,

$$\sum_{j:(i,j) \in A} f_{i,j} - \sum_{j:(j,i) \in A} f_{j,i} = \begin{cases} v(f) & i = s \\ 0 & i \neq s, t \\ -v(f) & i = t \end{cases}$$

$$0 \leq f_{i,j} \leq c_{i,j} \quad \forall (i, j) \in A$$

Now suppose an arc  $(i, j)$  fails and all the flow passing through this arc is lost and results in a net loss of  $\mu_{i,j}(f)$ . The surviving flow is then  $\omega_{i,j}(f) = v(f) - \mu_{i,j}(f)$ . Let  $\mu^0(f)$  be the

maximum flow loss due to an arc failure. This maximum is over all possible choices of an arc failure. We want to find  $f$  that maximizes  $\omega^0(f) (= \nu(f) - \mu^0(f))$ . This is the problem that we consider in this section.

The first question that we need to address is: given a flow  $f$ , what is the loss  $\mu_{i,j}(f)$  if arc  $(i, j)$  fails? If the flow is originally described in the arc-chain form (a chain in our way of thinking does not have an arc repeated), and then converted to  $f$ , then the loss due to the failure of an arc is the flow on it. If this  $f$  contains a cycle with positive flow, then removing this cycle creates a flow  $\hat{f}$  which is vectorially smaller than  $f$ . but corresponds to the same input flow. Hence  $\hat{f}$  can not increase the loss due to an arc failure in comparison to  $f$ . Thus we can assume that there exists an optimal chain flow which gets converted to an **acyclic** node-arc flow  $f$ . In an acyclic flow the flow loss due to an arc failure is, of course, the flow on the failed arc.

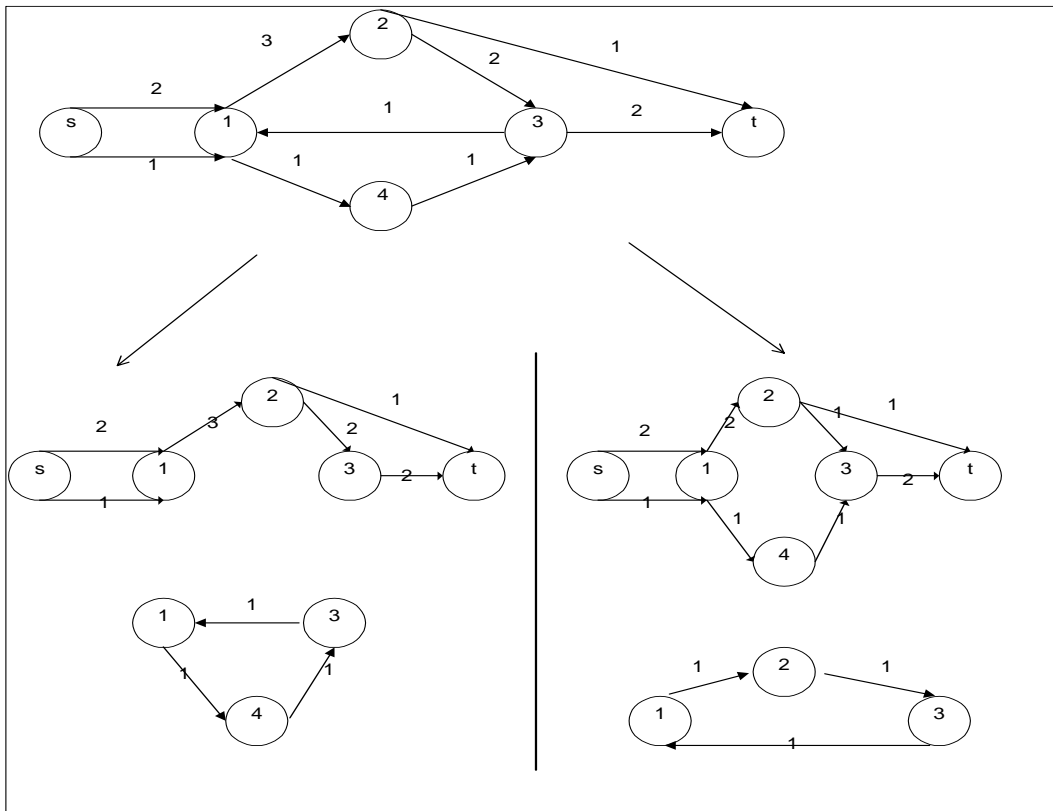


Figure 2: Decomposition of a Cyclic Flow into Two Acyclic Flows

Suppose the flow  $f$  is provided in the node-arc form. If  $f$  is not acyclic, then the loss  $\mu_{i,j}(f)$  due to failure of an arc  $(i, j)$  can not be determined unless we know how the flow is being routed, as there are several ways in which the flow  $f$  can be decomposed to an acyclic flow and flow cycles. Assuming that the objective of the flow router is to minimize the maximum loss, the appropriate decomposition would be the one that minimizes this maximum loss. This is equivalent to sending the corresponding acyclic flow.

The example in Figure 2 illustrates the argument. The numbers on the arcs indicate the flows. The cyclic flow can be decomposed in two ways. In the decomposition on the left, failure of arc  $(1, 2)$  would result in a loss of 3 units, whereas in the decomposition on the right, the maximum loss would be 2 units. Thus the flow sender would choose the second decomposition. We have thus proved the following.

**Lemma 1** *There exists an optimal flow which is acyclic.*

It is not clear at this stage that the a flow  $f$  that maximizes  $v(f)$ , the value of input flow, would also maximize the surviving or residual flow  $\omega^0(f)$ . The following lemma answers the question in the affirmative.

**Lemma 2** *There exists an optimal solution that simultaneously maximizes both  $v(f)$  and  $\omega^0(f)$ .*

**Proof.** If  $f$  does not maximize  $v$ , then there exists an augmenting path. This augmentation will increase the flow on no arc by more than the augmentation itself. Hence, the additional loss due to failure of any arc can not be more than the augmentation itself and the surviving flow will not be decreased. ■

We can thus restrict ourselves to maximal flows. In the worst case analysis that we are considering, the loss will be the flow on an arc carrying the maximum flow. Thus the objective is to obtain a maximal flow in which the largest arc flow as small as possible. This allows us to formulate the problem as follows.

$$\begin{aligned} \sum_j f_{i,j} - \sum_j f_{j,i} &= \begin{cases} F & i = s \\ 0 & i \neq s, t \\ -F & i = t \end{cases} \\ 0 &\leq f_{i,j} \leq \min[c_{i,j}, \lambda] \quad \forall (i, j) \in A \end{aligned}$$

If the maximum flow in  $G(\lambda) = [N, A, c(\lambda)]$ , where  $c_{i,j}(\lambda) = \min[c_{i,j}, \lambda] \quad \forall (i, j) \in A$ , is denote by  $f(\lambda)$  with value  $F(\lambda)$ , then  $F(\lambda)$  is a piecewise linear concave function of  $\lambda$ , with each linear piece having an integral slope. This can be seen as follows. Take any  $s - t$  cut  $D$ . Let  $D(\lambda) = \sum_{(i,j) \in D} c_{i,j}(\lambda)$ . Since each  $c_{i,j}(\lambda)$  is piece-wise linear concave in  $\lambda$ , so is  $D(\lambda)$  and represents the capacity of the cut  $D$  in  $G(\lambda)$ . Slope of any linear piece in  $D(\lambda)$ , for a given  $\lambda$ , is an integer with value equal to the number of arcs with capacity  $\lambda$ . Since  $F(\lambda) = \min\{D(\lambda) : D \in \Omega\}$ , where  $\Omega$  is the set of all  $s - t$  cuts,  $F(\lambda)$  is piece-wise linear concave with integral slopes. The largest slope gives the cardinality of an  $s - t$  cut with minimal cardinality. Consider now the function  $F(\lambda) - \lambda$ . If this function has a unique maximum at  $\lambda = \lambda^*$ , then this  $\lambda^*$  provides the desired maximal flow with the largest arc flow of  $\lambda^*$ . Otherwise,  $F(\lambda) - \lambda$  has a linear piece with zero slope defined over some interval, and for any  $\lambda$  in this interval  $f(\lambda)$  provides an optimal flow to the problem.

The following algorithm provides a strongly polynomial algorithm for finding  $\lambda^*$ , provided we use a strongly polynomial algorithm to find a max flow. The algorithm is based on the

Newton's method specialized for combinatorial optimization [2]. Analogous specializations have been used in several other problems [6],[3],[4]. Theoretical efficiency of Newton's method for combinatorial optimization problems has been established in [7].

For a given  $\lambda$ , let  $D(\lambda)$  represent a min-cut, separating  $s$  and  $t$  in  $G(\lambda)$ , and  $q(\lambda)$  the number of arcs in  $D(\lambda)$  with capacity  $\lambda$ .

**algorithm** *Newton*;

find a max flow  $f^*$  and min-cut  $D^*$  with value  $F^*$  in  $G$ .

$\lambda := \max \{c_{i,j} : (i,j) \in D^*\}$ ;

find  $f(\lambda), F(\lambda), D(\lambda), q(\lambda)$ ;

**while**  $F(\lambda) < F^*$  **do**

$\lambda := \lambda + (F^* - F(\lambda)) / q(\lambda)$ ;

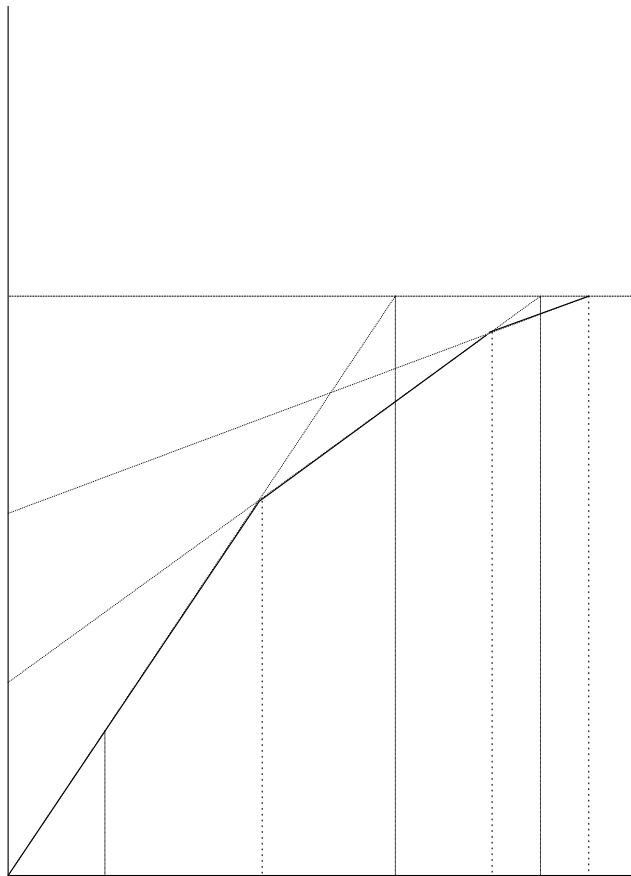
find  $f(\lambda), F(\lambda), D(\lambda), q(\lambda)$ ;

**end\_while** ;

terminate with  $\lambda^* := \lambda, f(\lambda^*)$  and  $D(\lambda^*)$ ;

The algorithm is strongly polynomial since at any iteration, either the algorithm stops or the value of  $q(\lambda)$  decreases by at least one. Since the starting value of  $q(\lambda)$  is bounded by  $|V|$ , the time complexity of the algorithm is, therefore,  $O(|V|\theta)$ , where  $O(\theta)$  is the complexity

of the max-flow algorithm. The figure below illustrates the algorithm.



Newton's Method

The method here starts with  $\lambda = \lambda_1$ , and terminates with  $\lambda = \lambda_3$ .

### 3 Extensions

#### 3.1 Integral Flows

As we observed in the introduction, the optimal flow need not be integral. This happens as the  $\lambda$ -values generated by the Newton's Algorithm need not be integral, rendering the capacity of certain arcs non-integral. Suppose we are interested in an integral optimal solution. That is, among all integral flows, we want a flow pattern that optimizes the residual flow. Suppose the Newton's method terminates with  $\lambda^*$ . That is, the concave function  $F(\lambda) - \lambda$  is maximized at  $\lambda^*$ . If  $\lambda^*$  is integral, then of course  $f(\lambda^*)$  is integral. Otherwise, let  $\lambda_1 = \lfloor \lambda^* \rfloor$ , and  $\lambda_2 = \lceil \lambda^* \rceil$ . If  $F(\lambda_1) - \lambda_1 < F(\lambda_2) - \lambda_2$ , then  $f(\lambda_1)$  is an optimal integral flow pattern, otherwise  $f(\lambda_2)$  provides an optimal solution.

## 3.2 Undirected and Mixed Networks

Each edge  $[i, j]$  can be replaced by two directed arcs  $(i, j)$  and  $(j, i)$ . In this manner the problem on undirected or mixed networks can be converted to an equivalent problem on a directed network. Since the optimal flow found will be acyclic, removal of a directed arc is equivalent to removing the corresponding undirected arc from the network.

## 3.3 Node Failure

Suppose, instead of an arc, a node (other than  $s$  and  $t$ ) fails. In this case all the flow passing through that node is lost. Here we define  $w^0(f)$  to be the maximum loss due to a node failure. The objective, as before, is to maximize  $w^0(f)$ . As established in Lemmas 1 and 2, using similar arguments, one can show that in this case also, there exists an optimal flow which is acyclic and maximizes both  $v(f)$  and  $w^0(f)$ . To solve this problem, we modify our network from  $G(N, A)$  to  $G'(N', A')$  whereby, we replace each node  $i \in N$  with two nodes  $i', i'' \in N'$ ; if  $(i, j) \in A$ , then  $(i'', j') \in A'$ . Also  $(i', i'') \in A' \quad \forall i \in N$ . The flow problem on  $G'$  is formulated as:

$$\begin{aligned} \sum_j f_{i,j} - \sum_j f_{j,i} &= \begin{cases} F & i = s \\ 0 & i \neq s, t \\ -F & i = t \end{cases} \\ 0 &\leq f_{i'',j'} \leq c_{i,j} \quad \forall (i, j) \in A \\ 0 &\leq f_{i',i''} \leq \lambda \quad \forall i \in N. \end{aligned}$$

Here we restrict the capacity of only node generated arcs to  $\lambda$ . As before, however,  $F(\lambda)$ , a max-flow in  $G'(\lambda)$  is a piece-wise linear concave function in  $\lambda$  with integral slope for each linear piece. Hence, the Newton's method described in Section 2 can also be used to solve this problem and provides a strongly polynomial algorithm for this problem as well. Again, an integral solution if needed can be obtained easily as before.

## 3.4 Probabilistic Arc Failure

Suppose, instead of the worst case analysis dealt with earlier, we are interested in maximizing the average residual flow. Let  $q$  be the probability that no arc fails, and  $(1 - q)$  be the probability that one arc fails. Let  $p_{i,j}$  be the conditional probability that arc  $(i, j)$  fails given that an arc fails. Then, for a feasible flow  $f$  with value  $F$ , the expected flow equals  $F - \sum_{(i,j) \in A} (1 - q)p_{i,j}f_{i,j}$ . It can be shown that an optimal solution is obtained at  $F = F^*$ . Thus, maximizing the expected residual flow is equivalent to the problem of minimum cost flows which can be solved by any well known strongly polynomial algorithms [1].

## 4 Problem 2: Most Vital Arc

An obvious approach to solve this problem is to determine, for each arc, the max-flow value over the reduced network where the arc under consideration is removed; and then choose an arc whose removal results in the smallest max-flow value. The complexity of this obvious approach is  $O(|A|\theta)$ . Hence polynomiality is not an issue in this problem. There are known ways to reduce the amount of work by eliminating certain arcs from consideration. For example, all arcs whose capacities are lower than that of the maximum capacity arc on a min-cut can be removed from consideration. In this method we have a current lower bound on the optimal capacity and we remove all arcs with capacity lower than this value. Then we consider the arc with the largest capacity among the arcs still under consideration and see if its removal reduces the flow by a larger amount. If so, our lower bound is increased to the amount by which the flow is reduced. Otherwise, this arc is removed from further consideration.. While this method may reduce the search in practice, the worst case complexity has not been shown to improve.

We want to characterize certain properties of an optimal arc which would help improve the above mentioned approach. For example, if conditions of lemma 3 are satisfied, the complexity of the search procedure reduces from  $O(|A|\theta)$  to  $O(|V|\theta)$ . If the network is undirected and all arc capacities are distinct then again the complexity of the search process reduces to  $O(|V|\theta)$ , as shown by lemma 5.

Following lemma establishes criteria under which an optimal arc is readily available from the optimal flow solution to our first problem.

**Lemma 3** *Consider the min-cut  $D(\lambda^*)$  in  $G(\lambda^*)$  in the algorithm Newton described in section 2. If this cut, in  $G$ , has at most one arc with capacity  $> \lambda^*$ , then an arc with the largest capacity in this cut is an optimal arc.*

**Proof.** Consider the max-flow  $f(\lambda^*)$  with value  $F^*$ . No arc in this flow pattern carries a flow of more than  $\lambda^*$ . Hence removal of any arc would result in a flow loss of at most the flow on that arc. Thus the residual network would have a max-flow value of at least  $F^* - \lambda^*$ . Consider the cut  $D(\lambda^*)$  In  $G(\lambda^*)$  all forward arcs of this cut are saturated, and all reverse arcs flowless. In  $G$ , by the assumptions of lemma, at most one arc has capacity  $> \lambda^*$ , and all other arcs have capacity  $\leq \lambda^*$ . Hence, in  $G$ , the flow  $f(\lambda^*)$  saturates all arcs of the cut  $D(\lambda^*)$ , except at most one arc with the largest capacity. The capacity of this cut is, therefore,  $(F^* - \lambda^*) + c_{\max}$ , where  $c_{\max}$  is the capacity of the largest capacity arc in this cut. Removal of this largest capacity arc results in a cut with capacity  $(F^* - \lambda^*)$  for the residual network. This must be a min-cut for the residual network, and an optimal arc is the largest capacity arc in this cut. ■

If conditions of this lemma are not satisfied, then we need other ways of cutting down the search. Let  $F'$  be the max-flow value over the residual network with an optimal arc removed, and  $v^* = F^* - F'$ . Thus  $v^*$  represents the maximum reduction possible in the max-flow by destroying an arc. Consider any min-cut (separating  $s$  and  $t$ ) in  $G$ . Removal of the largest capacity arc results in flow reduction equal to the capacity of the removed arc. Hence, if  $\bar{c}$

is the capacity of the largest capacity arc among all  $(s - t)$  min-cuts, then  $\bar{c} \leq v^*$ . If  $\bar{c}$  can be found easily, then we can reduce our search to only arc with capacity  $> \bar{c}$ . To find  $\bar{c}$ , note that given any min-cut and any max flow, all the forward (reverse) arcs of the min-cut are saturated (flowless) [5]. Thus, given any max flow and any min-cut, any arc that appears in some min-cut is saturated in the current flow. However, there may be saturated arcs in the current flow which do not belong to any min-cut. To determine if a saturated arc  $(i, j)$  belongs to some min-cut, we ask: is there is a flow-augmenting path from node  $i$  to node  $j$  with the current max-flow? If yes, then using this path along with arc  $(i, j)$  in the reverse direction, we have a cycle along which a positive amount of flow can be sent. This results in another max-flow with arc  $(i, j)$  unsaturated implying that  $(i, j)$  is not a min-cut arc. The following is a systematic procedure to search for  $\bar{c}$ .

**algorithm** *MaxCapMinCut*;

```

find a max-flow  $f^*$  and a min-cut  $D^*$  and label all saturated arcs;
while the maximum capacity labeled arc  $(i, j) \notin D^*$  do
    if  $\exists$  a flow augmenting path from  $i$  to  $j$ , then unlabel  $(i, j)$ 
    else stop with  $\bar{c} := c_{i,j}$ .

stop with  $\bar{c} := c_{i,j}$ ;

```

Following lemmas are useful for narrowing the search further for undirected networks.

**Lemma 4** *Let  $(i, j)$  be an optimal arc. Then  $c_{i,j}$  is the capacity of the largest capacity arc of some cut (separating  $s$  and  $t$ ).*

**Proof.** Proof: Let  $F^*$  and  $F'$  be the max-flow values in the original and residual network (obtained by removing  $(i, j)$ ), respectively. Then obviously  $F^* > F'$ . Let  $D'$  be a set of min-cut arcs in the residual network. Then it is easy to check that  $D = D' \cup \{(i, j)\}$  is a cut with capacity  $F' + c_{i,j}$ . Removing an arc of capacity  $> c_{i,j}$  from this cut results in a cut for the residual network with capacity  $< F'$ , a contradiction. Hence every arc in this cut has capacity  $\leq c_{i,j}$ . ■

**Lemma 5** *Let  $(i, j)$  be an optimal (undirected) arc. If  $T$  is a maximal spanning tree of undirected network  $G$ , then  $T$  contains an arc with capacity  $c_{i,j}$ .*

**Proof.** Consider the cut  $D$  of the previous lemma which showed that  $c_{i,j}$  is the capacity of the largest capacity arc in  $D$ . Removal of arcs of  $D$  breaks  $T$  into two parts. Thus  $T$  contains exactly one arc  $(k, \ell)$  of  $D$ . This arc must have capacity  $c_{i,j}$ , for if  $c_{k,\ell} < c_{i,j}$ , then replacing arc  $(k, \ell)$  with arc  $(i, j)$  in  $T$ , results in another spanning tree with larger value, a contradiction. ■

Thus the search for an optimal arc can be restricted further to arcs with capacities equal to one of the  $n - 1$  capacities of the arcs in a maximal spanning tree. If all the arcs in the network have distinct capacities, then this result improves the complexity of the search procedure to  $O(|V| \theta)$  from  $O(|A| \theta)$ .

## References

- [1] Ahuja, R.K., T.L. Magnanti, and J.B. Orlin. *Network Flows*, Prentice Hall, NJ. 1993.
- [2] Aneja, Y.P. and S.N. Kabadi, "Ratio Combinatorial Programs," *Euro. J. Oper. Res.* 81, 629-633 (1995).
- [3] Aneja, Y.P. and K.P.K. Nair, "Bicriteria Transportation Problem," *Management Science*, 25, 73-78 (1979).
- [4] Eisner, M.J. and D.G. Severence, "Mathematical Techniques for Efficient Record Segmentation for Large Shared Databases," *J.A.C.M.* 23, 619-635 (1976).
- [5] Ford, L.R., and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ. 1962.
- [6] Kabadi, S.N. and Y.P. Aneja, " $\epsilon$ -Approximation Minimization of Convex Functions in Fixed Dimension," *Operations Research Letters*, 18, 171-176 (1996).
- [7] Radzik, T. "Newton's Method for Fractional Combinatorial Optimization, " *Proc. 33rd IEEE Symposium on FOCS*, 659-669 (1992).