Journal of the Operational Research Society (1999) 50, 177–182
http://www.stockton-press.co.uk/jor

An alternate linear algorithm for the minimum flow problem

VG Adlakha
University of Baltimore, USA

We present an alternate linear algorithm for finding the minimum flow in (*s, t*)-planar networks using a new concept of *minimal removable sets* developed here. The iterative nature of the algorithm facilitates the adjustment of solutions for systems in developmental stages. The minimum flow algorithm presented here requires $O(|V|)$ time, where V denotes the set of vertices. The minimum flow problem arises in many transportation and communication systems.

Keywords: communication; minimum flow; transportation

Introduction

The minimum flow problem (MFP) in a network requires finding a flow with the smallest value so that the flow on each arc is bounded below by the weight of the arc. This problem arises in many transportation and communication systems. Here we describe a simple application. Consider a railroad network with stations as nodes and rail links as arcs. The freight volume per day on each link is known and requires a fixed minimum number of coaches per link to move the freight. We want to satisfy this per-day coach requirement on each link by using the minimum number of coaches in the system. This problem can be modeled as an MFP. For a similar example using airplanes see Lawler.[1] To the best of our knowledge, the first polynomial time algorithm for finding a minimum flow (MF) in a network was provided by Voitishin.[2] The algorithm runs in $O(|V|^3)$ time ($|V|$ = number of vertices) and is a modification of the maximum flow algorithm of Karazanov.[3] Provan and Kulkarni[4] used the MFP to find a maximum cardinality *uniformly-directed cut* that is useful in the simulation of networks with random arc weights. They also solve the problem in $O(|V|^3)$ time. Adlakha *et al*[5] presented an $O(|V|)$ algorithm for finding the minimum flow (MF) in (*s, t*)-planar directed networks using the concept of dualities in planar networks.

In this paper we develop an alternate linear algorithm for the MFP in an (*s, t*)-planar directed network. Even though the complexity of this algorithm is not different from the algorithm developed by Adlakha *et al*,[5] a major advantage of this algorithm lies in computation and implementation process. The algorithm does not require the use of dual network; hence, the user is spared the steps of transforming the primal network to construct the dual. In that sense, we provide a much simpler direct algorithm for the MFP. The iterative nature of the algorithm facilitates adjusting solutions for systems in developmental stages. The MF algorithm developed here uses the concepts of topmost paths and minimal removable sets. The first concept was introduced by Itai and Shiloach,[6] while the second is developed in this paper.

Correspondence: Dr VG Adlakha, Department of Management, University of Baltimore, 1420 N. Charles Street, Baltimore, MD 21201, USA.
E-mail: vadlakha@ubmail.ubalt.edu

Minimal removable sets

Let $G = (V, A)$ be a directed planar network with the set of nodes $V = \{v_0, v_1, \ldots, v_m\}$, the set of arcs $A = \{a_1, a_2, \ldots, a_n\}$, origin s, and destination t. For $a \in A$, let $\alpha(a)$ be the starting node of arc a, $\beta(a)$ be the ending node of arc a, and $w(a)$ be a non-negative weight associated with arc a. $I(v)$ is the set of arcs ending at node v and $O(v)$ is the set of arcs starting at node v. We assume that G is (*s, t*)-planar, that is, G has a planar representation such that a directed arc can be drawn from the sink node t to the source node s without violating planarity. We assume that one such representation is fixed in advance. The MFP is defined in detail by Adlakha *et al.*[5] We follow and maintain the same notation.

For $X \subset V$, define $\bar{X} = V - X$ and $(X, \bar{X}) = \{a \in A : \alpha(a) \in X, \beta(a) \in \bar{X}\}$. The set of edges $(X, \bar{X})$ is called an (*s, t*)-cut if $s \in X$ and $t \in \bar{X}$. An (*s, t*)-cut $(X, \bar{X})$ is called a uniformly-directed cut (UDC) if $(\bar{X}, X)$ is empty. Sigal[7] studied several properties of UDCs, and Provan and Kulkarni[4] showed that the value of minimum flow in G is equal to the weight of the maximum weight UDC in G. They solved the problem in $O(|V|^3)$ time.

A directed (*s, t*)-path in G is a sequence of arcs $(a_1, a_2, \ldots, a_k)$ such that $a_i \in A$ for $i = 1, \ldots, k$, $\alpha(a_1) = s$, $\beta(a_k) = t$ and $\alpha(a_{i+1}) = \beta(a_i)$ for $i = 1, \ldots, k - 1$. Henceforth, any path will always be a directed (*s, t*)-path. A path is called simple if it does not visit any node more than once. Let $P_1 = (a_1, a_2, \ldots, a_k)$ and $P_2 = (b_1, b_2, \ldots, b_m)$ be two distinct simple paths in G. As the paths P_1 and P_2 are distinct, there exists an r such that $a_i = b_i$ for $i = 1, \ldots, r$ and $a_{r+1} \neq b_{r+1} (0 \leqslant r < k \leqslant m)$. If $r = 0$, a_0 is assumed to be the imaginary arc from t to s that can be drawn without violating planarity. The path P_1 is said to *lie above* P_2 (or, equivalently, P_2 *lies below* P_1) if the arc a_{r+1} is before b_{r+1} in a clockwise sweep around the node $\beta(a_r) = \beta(b_r)$, starting from arc a_r. Note that the relation *lies above* defines a complete order on the set of all simple (s, t)-paths in G. Let L be the set of all simple (s, t)-paths in G ordered according to this *lies-above* ordering. The paths in L are said to be in topmost path first (TPF) order. The first path in L is called the topmost path and the last one is called the bottommost path.

Again consider two distinct simple paths P_1 and P_2. A path P_1 is said to be *completely above* P_2 (or, equivalently, P_2 *completely below* P_1) if $\beta(a_i) = \beta(b_j)$ and $a_{i+1} \neq b_{j+1}$ implies that the arc a_{i+1} is before b_{j+1} in the clockwise sweep around $\beta(a_i)$ starting from a_i for $i, j \geqslant 0$. An algorithm to list paths in the TPF order is provided in Kulkarni and Adlakha.[8]

We provide an example: consider the planar network of seven nodes and ten arcs displayed in Figure 1. Node 0 is the source and node 4 is the sink. Table 1 lists the paths of this network in TPF order. Path (a_6, a_7, a_4) lies below path (a_1, a_5, a_9, a_{10}) but does not lie completely below. Path (a_6, a_7, a_4) lies completely below path (a_1, a_5, a_7, a_4).

Let P be the topmost path of the network G, let D be a non-empty set of arcs contained in P. Construct network $G_D = (V_D, A_D)$ with $A_D = A - D$ and $V_D = V - \{$the isolated nodes in the graph $G' = (V, A_D)\}$. A non-empty set of arcs D is said to be a removable set if the network $G_D = (V_D, A_D)$ is empty or has the same single source s and single sink t as the network G. Note that if D is a removable set, a non-empty network G_D is also an (s, t)-planar PERT network. A removable set D in the topmost path P is said to be a minimal removable (MR) set if no non-empty proper subset of D is a removable set. For an MR set D, define its adjacent set $A(D) = \{a \in P_D - (P_D \cap P)\}$, the set of arcs in topmost path P_D of the reduced network A_D that are not in P. For example, consider the network G of Figure 1 and let $D = \{a_2, a_3\}$. The network G_D is shown in Figure 2. Since G_D has a single source and a single sink, identical to those in G, D is a removable set. It is also an MR set since $\{a_2\}$ and $\{a_3\}$ are not removable. The topmost path in G_D is $P_D = \{a_1, a_5, a_7, a_4\}$ and $A(D) = \{a_5, a_7\}$.

Table 1 Paths of network of Figure 1 in TPF order

i	*Arcs in Path i*				
1	a_1	a_2	a_3	a_4	
2	a_1	a_2	a_3	a_8	a_{10}
3	a_1	a_5	a_7	a_4	
4	a_1	a_5	a_7	a_8	a_{10}
5	a_1	a_5	a_9	a_{10}	
6	a_6	a_7	a_4		
7	a_6	a_7	a_8	a_{10}	
8	a_6	a_9	a_{10}		

Theorem 1 *There exists at least one removable set in the topmost path of the network G.*

Proof Let P be the topmost path of a non-empty network G. Consider two cases:

(i) *P is the only path in G:* $D = \{a : a \in P\}$ is a removable set, since in this case G_D is empty.
(ii) *G has more than one path:* Let $L = \{$all paths in G in TPF order$\}$. Let P' be the first path in $L - \{P\}$ that lies completely below P. Define D to be all arcs in P that are not in P'. Obviously G_D has the same single source s and single sink t as the network G. Hence D is a removable set.

The following algorithm lists all MR sets of the topmost path.

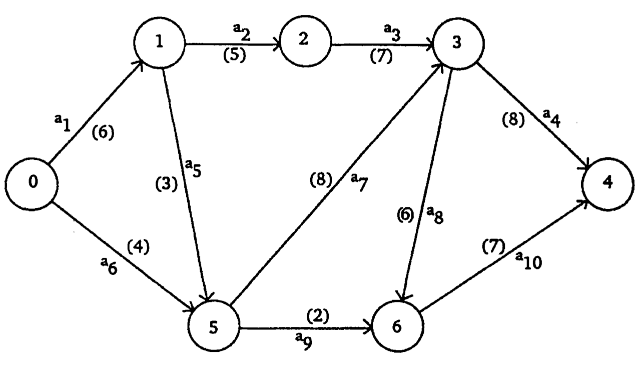

Figure 1 The example network 1 : G. (The numbers in brackets represent arc weights).

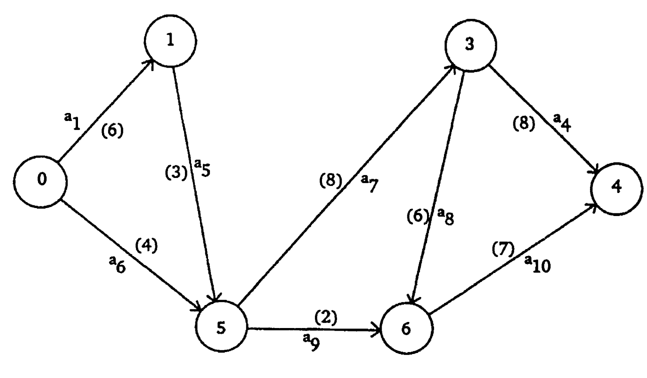

Figure 2 The reduced network: G_D. $D = \{a_2, a_3\}$. (The numbers in brackets represent arc weights).

Algorithm A

Input: An (*s*, *t*)-planar network $G = (V, A, s, t)$ with topmost path $P = (a_1, a_2, \ldots, a_k)$. Let $\alpha(a_{i+1}) = v_i, i = 0, 1, \ldots, k-1$ and $\beta(a_k) = v_k$. Hence $v_0 = s$ and $v_k = t$.

Output: A list of all MR sets of *P.*

Step 0: Set $O'(v_i) = |O(v_0)| + 1$ for $i = 0$
$= |O(v_i)|$ for $i = 1, 2, \ldots, k$

and $I'(v_i) = |I(v_i)|$ for $i = 0, 1, \ldots, k-1$
$= |I(v_k)| + 1$ for $i = k$.

Step 1: Set $x = 0$.

Step 2: Set $y = \min\{i : x < i \leqslant k, I'(v_i) > 1\}$,
$z = \max\{i : x \leqslant i < y, O'(v_i) > 1\}$,
List $\{a_{j+1} : z \leqslant j < y\}$ as an MR set.

Step 3: Construct $S = \{i : y \leqslant i \leqslant k, O'(v_i) > 1\}$.
If the set S is empty, go to Step 4.
Else, set $x = \min\{i : i \in S\}$, go to Step 2.

Step 4: STOP. All MR sets of *P* have been listed.

Note that the initialisation in Step 0 is equivalent to adding an imaginary arc from *s* to *t*. It also guarantees that $O'(v_0) = O'(s) > 1$ and $I'(v_k) = I'(t) > 1$. Steps 1 and 3 ensure that $O'(v_x) > 1$ whenever Step 2 is being carried out. Hence *y* and *z* in Step 2 are well defined.

Example Consider the network in Figure 1. The topmost path is $P = \{a_1, a_2, a_3, a_4\}$. The following two tables present the input parameters along with the calculations of Step 0 and the results of the iterations.

Thus the path P has two MR sets, $\{a_2, a_3\}$ and $\{a_4\}$.

i	*0*	*1*	*2*	*3*	*4*
v_i	0	1	2	3	4
$O'(v_i)$	3	2	1	2	0
$I'(v_i)$	0	1	1	2	3

Iteration no.	*x*	*y*	*z*	*MR set*	*S*
1	0	3	1	$\{a_2, a_3\}$	{3}
2	3	4	3	$\{a_4\}$	∅

Now consider the network G_D in Figure 2 obtained by removing the MR set $D = \{a_2, a_3\}$ of network *G* in Figure 1. The topmost path *P* in G_D is $\{a_1, a_5, a_7, a_4\}$. The following two tables give the results of the algorithm.

i	*0*	*1*	*2*	*3*	*4*
v_i	0	1	5	3	4
$O'(v_i)$	3	1	2	2	0
$I'(v_i)$	0	1	2	1	3

Iteration no.	*x*	*y*	*z*	*MR set*	*S*
1	0	2	0	$\{a_1, a_5\}$	{2, 3, 4}
2	2	4	3	$\{a_4\}$	∅

Theorem 2 *Algorithm A lists a set D if and only if D is an MR set of the topmost path.*

Proof Consider a set $D = \{a_{j+1} : z \leqslant j < y\}$ listed in Step 2 of *Algorithm A*. From the construction of *y* and *z* it is clear that $|I(v_i)| = |O(v_i)| = 1$ for $z < i < y$. Hence the reduced network $G_D = (V_D, A_D)$ has $A_D = A - D$ and $V_D = V - \{v_i : z < i < y\}$. Next consider the node v_z. If $z = 0, v_z$ is the source node in G_D. If $v_z \neq v_0$ then $|I(v_z)| \geqslant 1$ and $|O(v_z)| \geqslant 1$. Hence v_z is not a source or sink. Similarly, v_y is a sink node if and only if $v_y = v_k$. Therefore G_D has the single source node $v_0 = s$ and single sink node $v_k = t$. Hence, by definition, *D* is a removable set.

If *D* consists of a single arc (that is, $y = z + 1$), *D* is obviously an MR set. Suppose *D* has more than one arc. Let D' be a non-empty proper subset of *D*. Then there exists an arc $a_r \in D - D'$ such that either $a_{r-1} \in D'$ or $a_{r+1} \in D'$. Suppose $a_{r-1} \in D'$. Now consider the reduced network $G_{D'} = (V_{D'}, A_{D'})$. It is clear that $v_r \in V_{D'}$ and that $v_r (\neq v_0)$ is a source node in $G_{D'}$. Therefore $G_{D'}$ has more than one source node and D' is not a removable set. Therefore no non-empty proper subset of *D* is removable, and *D* is an MR set. Therefore every set listed by the *Algorithm A* in Step 2 is an MR set.

Next we show that if *D* is an MR set of the topmost path, then *Algorithm A* lists it. Define $z = \min\{i : a_i \in D\}$. If $D = \{a_i : z \leqslant i < k\}$, set $y = k$; otherwise define $y = \min\{i : a_{i-1} \in D$ and $a_i \notin D\}$. Now define $D' = \{a_i : z \leqslant i < y\}$. Obviously $D' \subseteq D$. Since *D* is an MR set, $O'(v_z) > 1, I'(v_y) > 1$, and $O'(v_i) = I'(v_i) = 1$ for $z < i < y$. Hence D' itself is MR set. But *D* is an MR set. Hence $D = D'$. Therefore *D* is an MR set implies that there is a *y* and *z* such that $D = \{a_i : z \leqslant i < y\}$ with $I'(v_r) = O'(v_r) = 1$ for $z < r < y$ and $O'(v_z) > 1$ and $I'(v_y) > 1$. Hence *D* will be listed in Step 2 of *Algorithm A*. ■

Minimum flow algorithm

The following algorithm computes the minimum flow (MF) through a single source, single sink, directed, acyclic (*s*, *t*)-planar network with non-negative arc weights.

Algorithm B.

Input: An (*s*, *t*)-planar network $G = (V, A)$, with source node *s* and sink node *t*, and a set of non-negative arc weights *w*.

Output: A minimum feasible flow f^*.

Step 0: For each $a \in A$, let $f_1(a), f_2(a)$ be dummy flow

variables.
Set $f_1(a) = 0, f_2(a) = 0$ for all $a \in A$, and $v(f^*) = 0$.
Let P be the topmost (s, t)-path in G.

Step 1: Find $D_1, D_2, \ldots, D_r$, the MR sets of P.
Set $\alpha_i = \max w(a), a \in D_i$.

Step 2: For each i, $1 \leqslant i \leqslant r$,

Set $f_1(a) = \alpha_i$, for $a \in D_i$,
$f_2(a) = \alpha_i$, for $a \in A(D_i)$,
and $w(a) = w(a) + \alpha_i$ for $a \in A(D_i)$.

Step 3: Let $D = \cup_{i=1}^{r} D_i$,
Construct the reduced network $G_D = (V_D, A_D)$. If $G_D = \emptyset$, go to Step 4.
If $G_D \neq \emptyset$, set $G = G_D$ and $P =$ topmost (s, t)-path in G. Go to Step 1.

Step 4: STOP. Set $v(f*) = \max \alpha_i$, $1 \leqslant i \leqslant r$,
and $f*(a) = f_1(a) - f_2(a), a \in A$.
f^* is the minimum feasible flow with value $v(f^*)$.

Note: It is worthwhile to observe that the flow on each arc $f^*(a), a \in A$, can be calculated for all $a \in D_i$ at each iteration during Step 2, also.

Example: We use the network of Figure 1 again to illustrate *Algorithm **B***. The following Table 2 provides all relevant quantities for each iteration. Note that the network in Figure 3 is precisely the G_D at the end of the first iteration.

The flow on each arc, in order of computation by iteration, is as follows:

a	$f_1(a)$	$f_2(a)$	$f^*(a)$
a_2	7	0	7
a_3	7	0	7
a_4	8	0	8
a_1	10	0	10
a_5	10	7	3
a_7	15	7	8
a_8	15	8	7
a_6	17	10	7
a_9	17	15	2
a_{10}	17	8	9

Hence the MF through the network is 17. This is easily verified by calculating the weights of all UDCs of the network G.

Remark 1 A network G must have a non-empty topmost path P and, from Theorem 1, P has a non-empty MR set D. Hence the reduced network G_D constructed in Step 3 will have strictly fewer arcs than G has; therefore the algorithm terminates in less than $|A| = n$ steps.

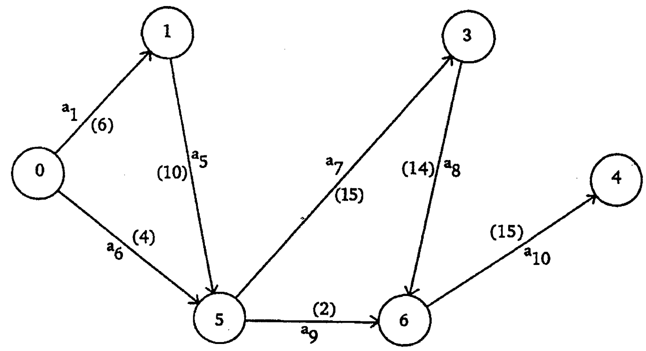

Figure 3 The reduced network* : $G_D.D = \{a_2, a_3\} \cup \{a_4\}$. (The numbers in brackets represent arc weights after the first iteration of Algorithm B).

Table 2

Iteration no.	$a \in P$ $w(a) \in P$	D_i	α_i	$A(D_i)$	*w(a) for* $a \in A(D_i)$
1	$\{a_1, a_2, a_3, a_4\}$ 6 5 7 8	$\{a_2, a_3\}$	7	$\{a_5, a_7\}$	{10, 15}
		$\{a_4\}$	8	$\{a_8, a_{10}\}$	{14, 15}
2	$\{a_1, a_5, a_7, a_8, a_{10}\}$ 6 10 15 14 15	$\{a_1, a_5\}$	10	$\{a_6\}$	14
		$\{a_7, a_8\}$	15	$\{a_9\}$	17
3	$\{a_6, a_9, a_{10}\}$ 14 17 15	$\{a_6, a_9, a_{10}\}$	17	—	—

Remark 2 Since we start with non-negative weights in G and at each iteration the weights $w(a)$ increase for some $a \in A$, the constant α_i will always be non-negative; therefore the value of the flow generated in each iteration is a non-decreasing function.

Theorem 3 *Value* $v(f^*)$ provided by *Algorithm **B*** is the value of the feasible MF for the network G.

Proof The proof follows from the following observations:

(i) For any intermediate node v of an MR set D_i, we have $I(v) = O(v) = 1$. Hence, the set D_i can be equivalently replaced by an artificial arc with weight α_i of Step 1.
(ii) The capacity of each UDC would not be changed by removing the artificial arc and adding its weight α_i to those of the arcs of the adjacent set $A(D_i)$ in Step 2. ■

The *Algorithm **B*** described here can be seen to be of $O(|V|)$, following arguments similar to the ones in Itai and Shiloach.[6]

A Comparison and sensitivity analysis

Adlakha *et al*[5] represented an $O(|V|)$ *Algorithm **DU*** for determining MF. The *Algorithm **DU*** is based on finding the dual of the network and then solving the PERT problem. The *Algorithm* ***B*** developed in this paper is also of $O(|V|)$. A comparison of these two algorithms is necessary in order to discuss the desirability of *Algorithm* ***B*** presented in this paper.

Firstly, *Algorithm* ***B*** is simple, easy to understand, and easy to program, whereas a heavier investment of time is required to initially implement *Algorithm* ***DU***. Secondly, in any practical application, conditions do not necessarily remain stable. Routes are eliminated or added. Consider an example of a new bus service in an existing town or in a town which is currently in a developmental phase. A network can be selected for initial service and the optimum solution can be found using an MF algorithm. Then as time goes by, new weights and new routes may be introduced to accommodate new demands and developments in the area. We discuss methodological details below. In an evolving MFP network, we may encounter changes of two kinds:

C1: There is a change in the weight of an arc or the weights of multiple arcs.
C2: A path has been added or deleted.

Let M denote the maximum cardinality UDC which has been determined by a previous minimum flow solution, either by the *Algorithm* ***DU*** or *Algorithm* ***B***. We consider change C1 first. Let $A(i)$ be the set of arcs a_i, $i \in I$, that are affected by the change C1 and let $w'(a_i), i \in I$, be the corresponding new weights. Then three possibilities arise:

$$\begin{aligned} X: &\quad A(i) \subset M, \\ Y: &\quad A(i) \cap M \neq \emptyset, A(i) \not\subset M, \end{aligned}$$

and

$$Z: \quad A(i) \cap M = \emptyset.$$

Furthermore, under each possibility, there may be different situations as follows:

(a) $X1: w'(a_i) \geqslant w(a_i)$ for every $i \in I$,

$X2: w'(a_i) < w(a_i)$ for some $i \in I$;

(b) $Y1: w'(a_i) \geqslant w(a_i)$ for every $a_i \in M$,

$w'(a_i) \leqslant w(a_i)$ for every $a_i \in A(i) - M$,

$Y2: w'(a_i) < w(a_i)$ for some $a_i \in M$;

$w'(a_i) \leqslant w(a_i)$ for every $a_i \in A(i) - M$,

$Y3: w'(a_i) > w(a_i)$ for some $a_i \in A(i) - M$;

(c) $Z1: w'(a_i) \leqslant w(a_i)$ for every $a_i \in A(i)$,

$Z2: w'(a_i) > w(a_i)$ for some $a_i \in A(i)$.

It can be observed that under situations X1 and Y1, a new MF is easily obtained by increasing the flow on arcs $a_i \in A(i) \cap M$ to $w'(a_i)$. In situation Z1, the MF remains unchanged. Under the remaining situations, X2, Y2, Y3, and Z2, the MFP would have to be solved again.

Under change C2, that is, whenever a path is deleted, added, or changed, the MFP would also have to be solved all over again.

Whenever a MFP has to be solved again, the procedure has to be repeated completely if using *Algorithm* ***DU***. With *Algorithm* ***B***, however, this is not necessary, at least, in situations X2, Y2, Y3, Z2, and change C2. Note that *Algorithm* ***B*** first ranks paths in TPF order and then builds onto the MF in the network in sequential steps. Assume that the paths in the original network $G = (V, A)$ are arranged in TPF order as $P_1, P_2, \ldots$, and let $P'_1, P'_2, \ldots$ be a similar arrangement after introduction of situation X2, Y2, Y3, Z2, or change C2. Let

$$\begin{aligned} P_i = P'_i &\quad \text{for } i < j \\ P_j \neq P'_j &\quad \text{for } i = j. \end{aligned}$$

Then one has to solve the MFP starting from and beyond path P'_j only while recalling the earlier iterations from memory storage. This could be a considerable saving in time and effort.

Conclusions

In this paper we have presented an $O(|V|)$ iterative algorithm to construct a minimum flow (MF) on an (s, t)-planar, directed, acyclic network with a single source s and a single sink t, and with non-negative weights. In order to develop the algorithm for planar networks we have introduced the concept of minimal removable (MR) sets and have provided an algorithm to list them. The MF algorithm can be enhanced by using series-parallel reduction on G and G_D. For example, parallel arcs can be replaced by a single arc whose weight is the sum of the weights of these arcs; series arcs can be replaced by a single arc whose weight is the maximum of the weight of these arcs. These reductions will speed up the algorithm, but will leave its complexity unchanged. Future research should look into using the concept of MR sets in stochastic MF and other networks problems.

Acknowledgements—The author is indebted to Professor Vidyadhar Kulkarni of the University of North Carolina at Chapel Hill, who provided considerable insight into this research.

References

1 Lawler EL (1976). *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston: New York.
2 Voitishin YV (1980). Algorithms for solving the minimal flow in a network. *Cybernetics* **16**: 131–134.
3 Karazanov AV (1974). Determining the maximal flow in network with the method of preflows. *Soviet Math Dokl* **LS**: 434–437.

4 Provan JS and Kulkarni VG (1989). Exact cuts in networks. *Networks* **19**: 281–289.
5 Adlakha V, Gladysz B and Kamburowski J (1991). Minimum flow in (*s*, *t*) planar networks. *Networks* **21**: 767–773.
6 Itai A and Shiloach Y (1979). Maximum flow in planar networks. *SIAM J Comput* **8**: 135–151.
7 Sigal CE (1977). The stochastic shortest route problem. *Ph.D. Dissertation,* Purdue University, USA.
8 Kulkarni VG and Adlakha V (1985). Maximum flow in planar networks with exponentially distributed arc capacities. *Stochastic Models* **1**: 263–290.

Received January 1997;
accepted June 1998 after one revision