Bull. Math. Soc. Sci. Math. Roumanie
Tome 53(101) No. 4, 2010, 305–313

Minimum flows in directed s-t planar networks

by
Oana Georgescu and Eleonor Ciurea

Abstract

We present an algorithm for finding maximum cut and an algorithm for minimum flow in directed s-t planar networks. Finally, we present an example for these two algorithms.

Key Words: Network flow, network algorithms, minimum flow problem, planar graphs
2010 Mathematics Subject Classification: Primary 90B10; Secondary 90C35, 05C35, 68R10.

1 Introduction

The computation of a maximum flow in a graph has been an important and well studied problem, both in the field of computer science and operations research. Many efficient algorithms have been developed to solve this problem, see, e.g., [1]. The computation of a maximum flow in a planar graph has been extensively investigated by many researchers starting from the work of Ford and Fulkerson [3] who developed an $O(n^2)$ time algorithm for s-t graphs when the source s and sink t are on the same face. This algorithm was later improved to $O(n \log n)$ time by Itai and Shiloach [6]. By introducing the concept of potentials, Hassin [4] gave an elegant algorithm that run in $O(n\sqrt{\log n})$ time using Frederickson's shortest path algorithm. Itai and Shiloach [6] also developed an algorithm to find a maximum flow in an undirected planar graph when the source and sink are not on the same face. For faster maximum flow algorithms in planar (but not necessarily s-t planar) undirected and directed networks, see Hassin and Johnson [5] and Johnson and Venkatesan [8]. Khuller and Naor [7] presents the flow in planar graph with vertex capacities.

The computation of a minimum flow in a directed network has been investigated by Ciurea and Ciupală [2]. The minimum flow problem in planar graph was not treated. In this paper we present the minimum flow problem in directed s-t planar network.

2 Terminology and preliminaries

We consider a capacitated network $G = (N, A, l, c, s, t)$ with a nonnegative capacity $c(i, j)$ and with a nonnegative lower bounds $l(i, j)$ associated with each arc $(i, j) \in A$. We distinguish two special nodes in the network G: a source node s and a sink node t.

A *flow* is a function $f : A \to \mathrm{R}_+$ satisfying the following conditions:

$$f(i,N) - f(N,i) = \begin{cases} -v, & \text{if } i = s \\ 0, & \text{if } i \neq s, t \\ v, & \text{if } i = t \end{cases} \tag{2.1.a}$$

$$l(i,j) \leq f(i,j) \leq c(i,j), \forall (i,j) \in A, \tag{2.1.b}$$

for some $v \geq 0$, where

$$f(i,N) = \sum_{j|(i,j)\in A} f(i,j)$$

and

$$f(N,i) = \sum_{j|(j,i)\in A} f(j,i)$$

We refer to v as the *value of the flow f*.

The minimum flow problem is to determine a flow f for which v is minimized.

A *cut* is a partition of the nodes set N into two subsets S and $\bar{S} = N$-S; we represent this cut using notation $[S, \bar{S}]$. We refer to an arc (i, j) with $i \in S$ and $j \in \bar{S}$ as a *forward arc* of the cut and an arc (i, j) with $i \in \bar{S}$ and $j \in S$ as a *backward arc* of the cut. Let $(S, \bar{S})$ denote the set of forward arcs in the cut and let $(\bar{S}, S)$ denote the set of backward arcs. We refer to a cut $[S, \bar{S}]$ as an *s-t cut* if $s \in S$ and $t \in \bar{S}$.

For the minimum flow problem, we define the *capacity* $c[S, \bar{S}]$ of the s-t cut $[S, \bar{S}]$ as the sum of the lower bounds of the forward arcs minus the sum of the capacities of the backward arcs. That is,

$$c[S, \bar{S}] = l(S, \bar{S}) - c(\bar{S}, S). \tag{2.2}$$

We refer to an s-t cut whose capacity is the maximum among all s-t cuts as a *maximum cut*.

In [2] the next theorem is proved.

Theorem 2.1. *If there exists a feasible flow in the network, the value of the minimum flow from a source node s to a sink node t in a capacitated network with nonnegative lower bounds equals the capacity of the maximum s-t cut.*

In this paper we present the minimum flow problem in directed s-t planar network. Research on planar flow is motivated by the fact that more efficient algorithms can be developed by exploiting the planar structure of the digraph.

Definition 2.1. *A digraph $G = (N, A)$ is said to be planar if we can draw it in a two-dimensional (Euclidian) plane so that no two arcs cross (or intersect) each other.*

That is, we allow the arcs to touch one another only at the nodes. For some digraphs, however, no matter how we draw them, some arcs will always cross. We refer to such digraphs as *nonplanar*. Researchers have developed very efficient algorithms (in fact, linear time algorithms) for testing the planarity of a digraph.

Definition 2.2. *Let $G = (N, A)$ be a planar digraph. A face x of G is a region of the (two-dimensional) plane bounded by arcs that satisfies the condition that any two points in the region can be connected by a continuous curve that meets no nodes and arcs. The boundary of a face x is the set of all arcs that enclose it. Faces x and y are said to be adjacent if their boundaries contain a common arc.*

The planar digraph G has an unbounded face.

Next we present two well known properties of planar digraphs.

Theorem 2.2. *If a connected planar digraph has n nodes, m arcs and f faces, then $f = m - n + 2$.*

Theorem 2.3. *If a planar digraph has n nodes and m arcs, then $m < 3n$.*

Planar digraphs have many special properties. In particular, every connected planar digraph $G = (N, A)$ has an associated "twin" planar digraph $G' = (N', A')$ which we refer to as the *dual* of G. Our discussion in this paper applies to a special class of planar digraphs known as directed s-t planar networks.

Definition 2.3. *A directed planar network with a source node s and a sink node t is called s-t planar if nodes s and t both lie on the boundary of unbounded face.*

For notational convenience, we refer to the original directed network G as the primal directed network.

3 Finding maximum cuts using longest directed path

We define the dual directed network of a directed s-t planar network $G = (N, A, l, c, s, t)$ as follows: we first draw an arc (t, s) with $l(t, s) = 0$, $c(t, s) = 0$, which divides the unbounded face into two faces: a new unbounded face and a new bounded face. Then we place a node x' inside each face x of the primal network G. Let s' and t', respectively, denote the nodes in the dual network corresponding to the new bounded face and the new unbounded face. Each arc $(i, j) \in A$ lies on the boundary of the two faces x and y; corresponding to this arc, the dual graph contains two oppositely directed arcs (x', y') and (y', x'). If arc (i, j) is a clockwise arc in the face x, we define the cost of arc (x', y') as $l(i, j)$ and the cost of arc (y', x') as $-c(i, j)$. We define arc costs in the opposite manner if arc (i, j) is a counterclockwise arc in the face x. The dual directed network contains the arc (s', t') which we delete from the network. There is an one-to-one correspondence between s-t cuts in the primal network and directed paths from node s' to node t' in the dual network; moreover, the capacity of the cut equals the cost of the corresponding directed path. Consequently, we can obtain a maximum s-t cut in the primal network by determining a longest directed path from node s' to node t' in the dual network.

Since in the previous discussion we showed that by solving a longest directed path problem in the dual network, we can identify a maximum cut in a primal directed s-t planar network, we suppose that dual network $G' = (N', A', b', s', t')$ does not contain a directed cycle of positive cost. Since for all arcs $(i, j) \in A$ with $l(i, j)$, $c(i, j)$, for any directed cycle $\overset{\circ}{C'} = (x', y', x')$ from $G' = (N', A', b', s', t')$ it has $b'(\overset{\circ}{C'}) = b'(x', y') + b'(y', x') = l(i, j) - c(i, j) \leq 0$. We can solve the longest directed path problem in the dual network $G' = (N', A', b', s', t')$ using Bellman-Ford algorithm.

Theorem 3.1. *It is possible to determine a maximum s-t cut in a directed s-t planar network in $O(n^2)$ time.*

Proof. We showed that can obtain a maximum s-t cut in the primal network by determining a longest directed path from node s' to node t' in the dual network. The Bellman-Ford algorithm has complexity $O(m'n')$. The planarity implies that $m' = O(n), n' = O(n)$ and the algorithm for determine a maximum s-t cut has complexity $O(n^2)$. □

4 Finding minimum flow

We present an algorithm for finding a minimum flow in a directed s-t planar network.

Let $d'(y')$ denote the longest directed path distance from node s' to node y' in the directed dual network $G' = (N', A', b', s', t')$. The longest directed path distances satisfy the following conditions:

$$d'(y') \geq d'(x') + l'(x', y') \text{ for each } (x', y') \in A' \tag{4.1}$$

Each arc (i, j) in the directed s-t planar network $G = (N, A, l, c, s, t)$ corresponds to two arcs (x', y') and (y', x') in the directed dual network $G' = (N', A', b', s', t')$.

The minimum flow problem in a general directed s-t network can be solved in two phases:

(1) establish a feasible flow, if it exists;

(2) from a given feasible flow, establish the minimum flow.

For finding minimum flow in a directed s-t planar network we determine the function $f : A \to \mathrm{R}$ with the next algorithm:

```
(1)PROGRAM MF;
(2)BEGIN
(3)    BELLMAN-FORD(G');
(4)    FOR (i, j) ∈ A DO
(5)        f(i, j) = d'(y') − d'(x')
(6)END.
```

Theorem 4.1. *The algorithm MF determines a flow f in network G.*

Proof. We show that f satisfies the mass balance constraints. Each node k in G, except nodes s and t, define a cut $E_k = [\{k\}, N\text{-}\{k\}]$ consisting of all of the arcs incident to that node. The arcs in G' corresponding to arcs in E_k define a direct cycle, say $\overset{\circ}{C}'$ which has the arcs (x', y') with $b'(x', y') = l(k, l)$. Obviously,

$$\sum_{\overset{\circ}{C}'_k} (d'(y') - d'(x')) = 0 \tag{4.2}$$

because the terms cancel each other. Using $f(i, j) = d'(y) - d'(x)$ and (4.2) it results that

$$\sum_{E_k} f(i, j) = 0$$

which implies that inflow equals outflow at node k. $\square$

Theorem 4.2. *If the flow f determined with the algorithm MF satisfies the condition $f \le c$ then the flow f is a minimum feasible flow.*

Proof. The expressions (4.1) imply that $f(i, j) = d'(y) - d'(x) \ge l'(x', y') = l(i, j)$ for each $(i, j) \in A$. Therefore $l \le f \le c$ and f is a feasible flow. We show that the flow f is a minimum flow. Let P' be a longest path from node s' to node t' in G'. The definition of P' implies that $d'(y') - d'(x') = b'(x', y')$ for each $(x', y') \in P'$ and corresponding $[S, \bar{S}] = (S, \bar{S})$ in G. We obtain from (2.2) that $c[S, \bar{S}] = l(S, \bar{S})$. We have $b(P') = l(S, \bar{S})$, because $b'(x', y') = l(i, j)$, $(x', y') \in P'$, $(i, j) \in (S, \bar{S})$. Therefore, $v = f[S, \bar{S}] = f(S, \bar{S}) = b'(P') = l(S, \bar{S}) = c[S, \bar{S}]$ and Theorem 2.1 implies that f is a minimum feasible flow. $\square$

Theorem 4.3. *If there exists a feasible flow in a directed s-t planar network, then it is possible to determine a minimum flow in $O(n^2)$ time.*

Proof. The procedure from line (3) has the complexity $O(n^2)$. The lines (4) and (5) have complexity $O(m)$=$O(n)$. Therefore, it is possible to determine a minimum flow in $O(n^2)$ time. $\square$

5 Example

We determine minimum flow in the network $G = (N, A, l, c, s, t)$ with $N = \{1,2,3,4,5,6\}$ and $A = \{(1,2),(1,3),(2,3),(2,4),(2,5),(3,5),(4,6),(5,4),(6,4)\}$. The network G is a directed s-t planar network with $s = 1$ and $t = 6$ and it is shown in Fig. 5.1. The values for lower bounds and capacities are indicated on the arcs.

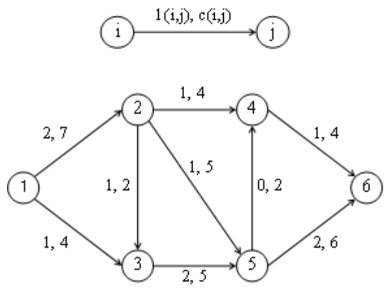

Figure 5.1: Directed s-t planar network

Fig. 5.2 shows the directed s'-t' dual network G' corresponding to primal network G; the dashed lines are the arcs in the dual network and $s' = 1'$, $t' = 6'$.

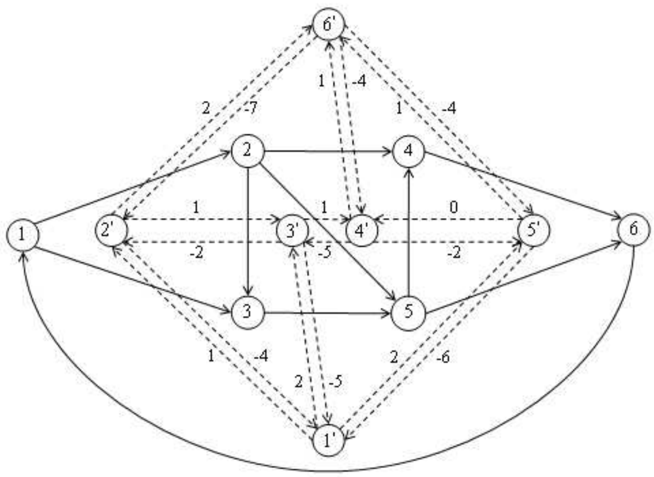

Figure 5.2: Directed s'-t' dual network

Arc $(1,3)$ is a clockwise arc in the face $1'$, so we define the cost of arc $(1', 2')$

as

$$b'(1',2') = l(1,3) = 1$$

and the cost of arc $(2',1')$ as

$$b'(2',1') = -c(1,3) = -4.$$

We determine the other values for the costs of arcs in dual network in the similar manner for all the clockwise arcs.

It is easy to verify that if the arc $(5,4)$ is the counterclockwise arc in the face $4'$ than it is a clockwise arc in the face $5'$; therefore,

$$b'(5',4') = l(5,4) = 0$$

and

$$b'(4',5') = -c(5,4) = -2.$$

Distance vector is d'=$(0,1,2,3,2,4)$ and predecessor vector is $p' = (0,1',1',3',1',4')$.

A longest directed path from node $s' = 1'$ to node $t' = 6'$ is $D' = (1',3',4',6')$ and corresponding maximum s-t cut in G is

$$[S,\bar{S}] = (S,\bar{S}) = \{(2,4),(2,5),(3,5)\}.$$

We have

$$c[S,\bar{S}] = l(S,\bar{S}) = l(2,4) + l(2,5) + l(3,5) = 1 + 1 + 2 = 4$$

and

$$b'(D') = b'(1',3') + b'(3',4') + b'(4',6') = 2 + 1 + 1 = 4.$$

Therefore,

$$c[S,\bar{S}] = l(S,\bar{S}) = b'(D').$$

Let k=3 a node in G and the cut is

$$E_3 = [\{3\},\{1,2,4,5,6\}] = (\{3\},\{1,2,4,5,6\}) \cup (\{1,2,4,5,6\},\{3\}) =$$

$$= \{(3,5)\} \cup \{(1,3),(2,3)\}.$$

The arcs in G' corresponding to arcs in E_3 define the directed cycle

$$\overset{\circ}{C'_3} = (1',3',2',1').$$

We obtain

$$c[S_3,\bar{S}_3] = l(S_3,\bar{S}_3) - c(\bar{S}_3,S_3) = l(3,5) - c(1,3) - c(2,3) = 2 - 2 - 4 = -4$$

and

$$b'(\overset{\circ}{C'_3}) = b'(1',3') + b'(3',2') + b'(2',1') = 2 - 2 - 4 = -4.$$

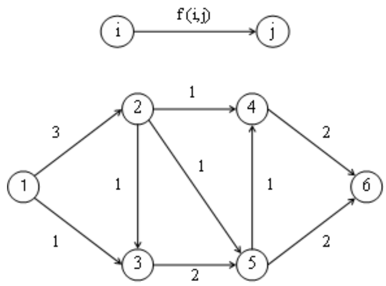

Figure 5.3: Minimum flow

Therefore,

$$c[S_3, \bar{S}_3] = b'(\overset{\circ}{C}{}'_3).$$

Fig. 5.3 shows a minimum flow f with $f(i,j) = d'(y') - d'(x')$.

The minimum flow f has value $v = f[S, \bar{S}] = f(S, \bar{S}) = f(2,4) + f(2,5) + f(3,5) = 1 + 1 + 2 = 4$. Therefore, $v = c[S, \bar{S}]$. According to the previous Theorem 2.1 we have that this flow f is the minimum flow.

References

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

[2] Ciurea, E., Ciupală, L., Sequential and parallel algorithms for minimum flows, J Appl Math Comput, Vol. 15, No. 1-2, 53-75 (2004)

[3] Ford, L.R., Fulkerson, D.R., Maximal flow through a network, Can J Math, Vol. 8, 399-404 (1956)

[4] Hassin, R., Maximum flows in s-t-planar networks, Inf Process Lett, Vol. 13, 107 (1981)

[5] Hassin, R., Johnson, D.B., An $O(n \log^2 n)$ algorithm for maximum flow in undirected planar networks, SIAM J Comput, Vol. 14, 612-624 (1985)

[6] Itai, A., Shiloach, Y., Maximum flow in planar networks, SIAM J Comput, Vol. 8, 135-150 (1979)

[7] Khuller, S., Naor, J., Flows in planar graphs with vertex capacities, Algorithmica, 11, 200-225 (1994)

[8] Johnson, D.B., Venkatesan, S., Using divide and conquer to find flows in directed planar networks in $O(n^{1.5} \log n)$ time, Proceedings of the 20th Annual

Allerton Conference on Communication, Control and Computing, University of Illinois Urbana-Champaign, IL, 898-905 (1982)

Received: 30.06.2008
Revised: 09.07.2008
Accepted: 30.09.2008

Transilvania University of Braşov,
Department of Theoretical Computer Science,
Bd. Iuliu Maniu 50, 500091,
Braşov ROMANIA,
E-mail: o.georgescu@unitbv.ro
e.ciurea@unitbv.ro