"Vasile Alecsandri" University of Bacău
Faculty of Sciences
Scientific Studies and Research
Series Mathematics and Informatics
Vol. 19 (2009), No. 1, 151 - 162

THE MAXMIN ALGORITHM FOR THE MAXIMUM FLOW

ELEONOR CIUREA AND MIHAI - STEFAN IOLU

Abstract. In this paper we study maximum flow algorithms by using a new approach.

1. Introduction

The theory of flows is one of the most important parts of Combinatorial Optimization.

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 [1], [9], [10], [11], [12], [13] and [15].

The computation of a minimum flow in a network has been investigated by the authors in [2], [3], [4], [5], [6], [7] and [8].

The brief outline of the paper is as follows: in Section 2 we discuss some basic notions and results used in the rest of the paper. Section 3 deals with a new approach for solving of the maximum flow problem in networks. In Section 4 we present an example for this approach.

Keywords and phrases: network flow, network algorithms, maximum flow problem, minimum flow problem.
(2000)Mathematics Subject Classification: 05C85, 90B10

In the next presentation we assume familiarity with preflow algorithms and we omit many details, since they are straightforward modifications of known results. The reader interested in further details is urged to consult the book [1] for maximum flow problem and the paper [7] for minimum flow problem.

2. Terminology and Preliminaries

In this section we discuss some basic notions and results used in the rest of the paper.

We consider a capacitated network $G = (N, A, l, c, s, t)$ with a nonnegative capacity $c(x, y)$ and with a nonnegative lower bounds $l(x, y)$ associated with each arc $(x, y) \in A$. We distinguish two special nodes in the network G: a source node s and a sink node t. We further assume, without loss of generality, that the network contains no parallel edges.

For a given pair of not necessarily disjoint subsets X, Y of the nodes set N of a network G we use the notation:

$$(X, Y) = \{(x, y) | (x, y) \in A, x \in X, y \in Y\}$$

and for a given function f on arcs set A we use the notation:

$$f(X, Y) = \sum_{(X,Y)} f(x, y)$$

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

$$f(x, N) - f(N, x) = \begin{cases} v, & x = s \\ 0, & x \neq s, t \\ -v, & x = t \end{cases} \tag{1a}$$

$$l(x, y) \leq f(x, y) \leq c(x, y), \forall (x, y) \in A, \tag{1b}$$

for some $v \geq 0$. We refer to v as the *value of the flow f.*

The maximum (minimum) flow problem is to determine a flow f for which v is maximized (minimized).

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

denote the set of forward arcs in the cut and let (T, S) denote the set of backward arcs.

For the maximum flow problem, we define the *capacity* $\widetilde{c}[S, T]$ of a $s - t$ cut $[S, T]$ as:

$$\widetilde{c}[S, T] = c(S, T) - l(T, S) \tag{2}$$

and for the minimum flow problem, we define the *capacity* $\widehat{c}[S, T]$ of a $s - t$ cut $[S, T]$ as

$$\widehat{c}[S, T] = l(S, T) - c(T, S) \tag{3}$$

We refer to an $s - t$ cut whose capacity $\widetilde{c}[S, T]$ is the minimum ($\widehat{c}[S, T]$ is the maximum) among all $s - t$ cuts as a *minimum (maximum) cut*.

The maximum (minimum) flow problem in a network $G = (N, A, l, c, s, t)$ can be solved in two phases:

(P1) establish a feasible flow f, if it exists;

(P2) from a given feasible flow f, establish the maximum (minimum) flow $\widetilde{f}$ ($\widehat{f}$).

Theorem 1. *Let G=(N,A,l,c,s,t) be a network, [S,T] a s-t cut and f a feasible flow with value v. Then*

$$v = f[S, T] = f(S, T) - f(T, S) \tag{4a}$$

and therefore, in particular,

$$\widehat{c}[S, T] \leq v \leq \widetilde{c}[S, T] \tag{4b}$$

Theorem 2. *Let $G = (N, A, l, c, s, t)$ be a network, $[\widetilde{S}, \widetilde{T}]$ a minimum $s - t$ cut and $[\widehat{S}, \widehat{T}]$ a maximum $s - t$ cut. We denote the values of a maximum flow $\widetilde{f}$ and minimum flow $\widehat{f}$ by $\widetilde{v}$ and $\widehat{v}$ respectively. Then*

$$\widetilde{v} = \widetilde{c}[\widetilde{S}, \widetilde{T}] \tag{5a}$$

and

$$\widehat{v} = \widehat{c}[\widehat{S}, \widehat{T}] \tag{5b}$$

A *preflow* is a function $f : A \to \mathbb{R}_+$ that satisfies (1b) and

$$f(N, x) - f(x, N) \geq 0, \forall x \in N - s, t \tag{6a}$$

for maximum flow problem and

$$f(x, N) - f(N, x) \leq 0, \forall x \in N - s, t \tag{6b}$$

for minimum flow problem.

For any preflow f we define the *excess*, respectively *deficit* of node x as

$$\widetilde{e}(x) = f(N, x) - f(x, N), \forall x \in N \tag{7a}$$

respectively

$$\widehat{e}(x) = f(x, N) - f(N, x), \forall x \in N \tag{7b}$$

for maximum respectively minimum flow problem.

We refer to a node x with $\widetilde{e}(x) = 0 (\widehat{e}(x) = 0)$ as *balanced.* A preflow f satisfying the condition $\widetilde{e}(x) = 0 (\widehat{e}(x) = 0), \forall x \in N - s, t$ is a *flow.* Thus, a flow is a particular case of preflow.

For the maximum (minimum) flow problem, the *residual capacity* $\widetilde{r}(x, y)$ $(\widehat{r}(x, y))$ of any arc $(x, y) \in A$, with respect to a given flow/preflow f, is given by $\widetilde{r}(x, y) = c(x, y) - f(x, y) + f(y, x) - l(y, x)$ $(\widehat{r}(x, y) = c(y, x) - f(y, x) + f(x, y) - l(x, y))$. By convention, if $(x, y) \in A$ and $(y, x) \notin A$ then we add arc (y, x) to the set of arcs A and we set $l(y, x) = 0$ and $c(y, x) = 0$.

The network $\widetilde{G} = (N, A)$ $(\widehat{G} = (N, \widehat{A}))$ consisting only of the arcs with positive residual capacity is referred to as the *residual network* (with respect to the flow/preflow f).

In the residual network $\widehat{G} = (N, \widehat{A})$, the distance function is a function $\widehat{d} : N \to \mathbb{N}$ and we say that a *distance function* is *valid* if it satisfies the following conditions:

$$\widehat{d}(s) = 0 \tag{8a}$$

and

$$\widehat{d}(y) \leq \widehat{d}(x) + 1, \forall (x, y) \in \widehat{A} \tag{8b}$$

We refer to $\widehat{d}(x)$ as the *distance label of node x* and to the arc $(x, y) \in \widehat{A}$ as an *admissible* arc if $\widehat{d}(y) = \widehat{d}(x) + 1$; otherwise it is *inadmissible.* We refer to a directe path from node s to node t consisting entirely of admissible arcs as an *admissible path.* We say that the distance labels are *exact* if for each node x, $\widehat{d}(x)$ equals the length of the shortest directed path from node s to node x in the residual network $\widehat{G}$. We refer to a path in G from the source node s to the sink node t as a *decreasing path* if it consists only of arcs with positive residual capacity. Clearly, there is an one to one correspondence between set of decreasing paths in G and the set of decreasing directed paths from s to t in $\widehat{G}$.

We define the *layered network* $\widehat{G}' = (N, \widehat{A}', \widehat{r})$ as follows: the nodes set N is partitioned into layers N_0, ..., N_k, where layer N_i contains the nodes x whose exact distance labels equals i, so that $\widehat{d}(x) = i$. Furthermore, for each arc (x, y) in the layered network, $x \in N_i$ and $y \in N_{i+1}$ for some i. We say that $\widehat{f}'$ is a *blocking flow* if the layered network $\widehat{G}'$ contains no decreasing directed path.

Any decreasing directed path algorithm terminates with optimal residual capacities. From these residual capacities $\widehat{r}(x, y)$ we can determine a minimum flow by the following expression

$$\widehat{f}(x,y) = l(x,y) + max\{\widehat{r}(x,y) - c(y,x) + l(y,x), 0\} \tag{9}$$

We can determine the maximum flow by using the residual capacities $\widetilde{r}(x, y)$ in the following manner:

$$\widetilde{f}(x,y) = l(x,y) + max\{0, c(x,y) - \widetilde{r}(x,y) - l(x,y)\} \tag{10}$$

In the literature, for the maximum flow problem there are two approaches:

(1) using increasing path algorithms (see table 1);
(2) using preflow algorithms (see table 2).

	Algorithm	**Running time**
1	Generic augmenting path	$O(nmC)$
2	Ford-Fulkerson labelling algorithm	$O(nmC)$
3	Gabow capacity scaling	$O(nmlog\bar{c})$
4	Ahuja-Orlin maximum capacity scaling	$O(nmlog\bar{c})$
5	Edmons-Karp of the shortest path	$O(nm^2)$
6	Ahuja-Orlin of the shortest path	$O(n^2m)$
7	Dinic algorithm of layered network	$O(n^2m)$
8	Ahuja-Orlin of the layered network	$O(n^2m)$

TABLE 1. The running time for increasing paths algorithms

	Algorithm	**Running time**
1	Generic preflow	$O(n^2m)$
2	FIFO preflow	$O(n^3)$
3	Highest label preflow	$O(n^2m^{1/2})$
4	Deficit scaling	$O(nm + n^2log\bar{c})$

TABLE 2. The running time for preflow algorithms

In the literature there are three approaches for solving the minimum flow problem:

(1) using decreasing path algorithms;
(2) using preflow algorithms;
(3) by establishing a maximum flow from node t to node s in network $\widetilde{G}$ (see [7]).

The same transformation, that we made to generic augmenting path algorithm in order to obtain generic decreasing path algorithm, can be made to any of the other algorithm from table 1. The same ideea can be applied to the algorithms from table 2. The complexities of these algorithms are the same as those presented in table 1 and table 2.

In section 3 we present a new approach for solving the maximum flow problem.

3. The maxmin algorithm

We can solve the maximum flow problem by determining a minimum flow from the sink node to the source node in the residual network $\widehat{G}$. This approach is based on the following idea: the aim of the maximum flow problem is to send as much flow as possible from the source node to the sink node, that is opposite to the aim of the minimum flow problem. The algorithm that we will present computes a maximum flow in the following manner: knowing a feasible flow, we determine a maximum flow from the source node to the sink node by establishing a minimum flow in the residual network $\widehat{G}$ from the sink node t to the source node s. For determining a minimum flow from the sink to the source, we can use any minimum flow algorithm, including preflow algorithms. The maxmin algorithm is the following:

```
Algorithm MAXMIN;
BEGIN
  let f be a feasible flow in network G;
  determine the residual network $\widehat{G}$;
  establish a minimum flow $\widehat{f}$ from $t$ to $s$
                 in $\widehat{G}$;
  $\widetilde{f} = \widehat{f}$ is a maximum flow from the source
                 node $s$ to the sink node $t$;
END.
```

Theorem 3. *The maxmin algorithm computes correctly a maximum flow.*

Let $\widehat{f}$ be the flow at the end of the algorithm. Therefore, $\widehat{f}$ is a minimum flow from sink node t to the source node s. In view of the Decreasing Path Theorem, there is no directed path from t to s in the residual network $\widehat{G}$. This implies that there is a $t-s$ cut $[T,S]$ with

$$\widehat{r}(y,x) = 0, \forall (y,x) \in (T,S) \tag{11}$$

Equivalently,

$$c(x,y) - \widehat{f}(x,y) + \widehat{f}(y,x) - l(y,x) = 0, \forall (y,x) \in (T,S) \tag{12}$$

In view of the condition (1b), this implies that

$$\widehat{f}(x,y) = c(x,y), \forall (x,y) \in (S,T) \tag{13a}$$

and

$$\widehat{f}(y,x) = l(y,x), \forall (y,x) \in (T,S) \tag{13b}$$

Equivalently,

$$\begin{aligned}\widetilde{r}(x,y) = c(x,y) - \widehat{f}(x,y) + \widehat{f}(y,x) - l(y,x) = 0,\\ \forall (x,y) \in (S,T)\end{aligned} \tag{14}$$

Since [T,S] is a $t-s$ cut, $[S,T]$ is a $s-t$ cut. We obtain the residual network $\widetilde{G}$ which contains no directed path from the source node s to the sink node t. In view of the Augmenting Path Theorem, $\widetilde{f} = \widehat{f}$ is a maximum flow from s to t in the network G.

Theorem 4. *The complexity of the maxmin algorithm is the complexity of the flow algorithm used for determining a feasible flow and for establishing a minimum flow from t to s.*

The proof of this theorem is obvious.

4. Example

We will study the network in figure 1 which has a feasible flow.

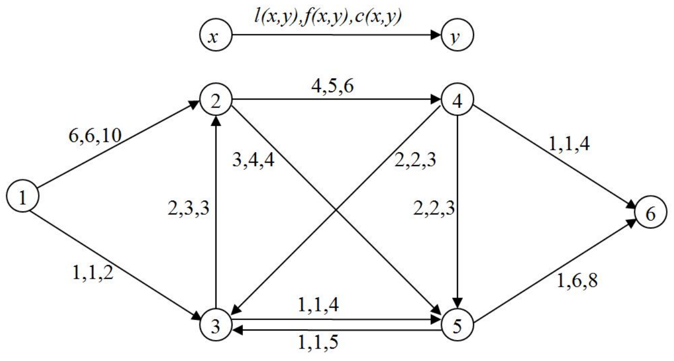

Figure 1. Initial network

The corresponding residual network $\widehat{G} = (N, A, \widehat{r})$ is presented in figure 2.

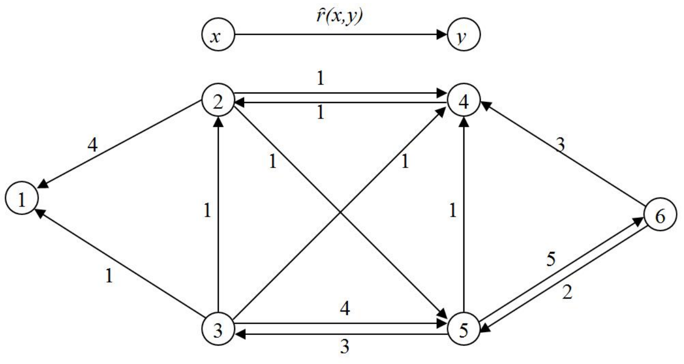

Figure 2. Initial residual network

By applying a decreasing path algorithm we can obtain three decreasing paths from $t = 6$ to $s = 1$.

The first directed path determined is $\widehat{D}_{6,1} = (6, 4, 2, 1)$, $\widehat{r}(\widehat{D}_{6,1}) = 1$. The resulting residual network $\widehat{G}$ is presented in figure 3.

The second directed path determined is $\widehat{D}_{6,1} = (6, 5, 3, 1)$, $\widehat{r}(\widehat{D}_{6,1}) = 1$. The new residual network is presented is presented in figure 4.

The third directed path is $\widehat{D}_{6,1} = (6, 5, 3, 2, 1)$, $\widehat{r}(\widehat{D}_{6,1}) = 1$. The new residual network $\widehat{G}$ is presented in figure 5.

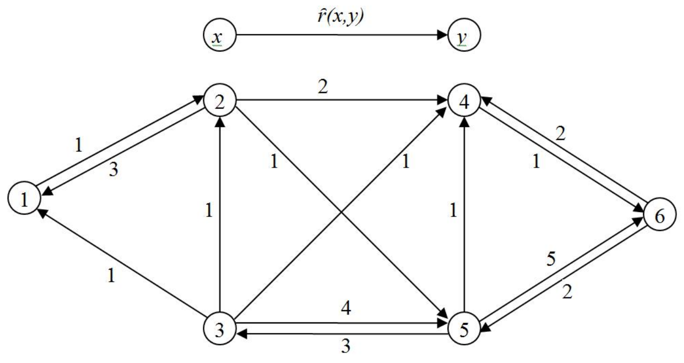

FIGURE 3. Residual network after the first decreasing path

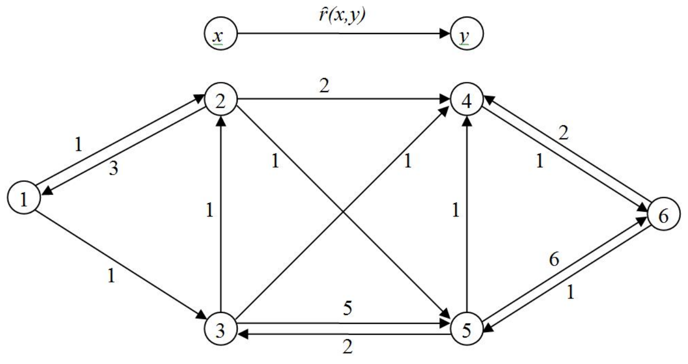

FIGURE 4. Residual network after the second path

We obtain the residual network presented in figure 5 which doesn't contain any decreasing path from $t = 6$ to $s = 1$.

By determining the minimum flow from $t = 6$ to $s = 1$ we obtain a maximum flow $\widetilde{f}$ from $s = 1$ to $t = 6$ in the network $G = (N, A, l, \widetilde{f}, c)$ which we present in figure 6.

Acknowledgements The research of the first author was supported by the Transilvania University of Brasov and by the Grant IDEI 134/2007.

References

[1] R., AHUJA, T., MAGNANTI, J., ORLIN, **Network Flows. Theory, algorithms and applications**, Prentice Hall, Inc., Englewood Cliffs, NY, 1993

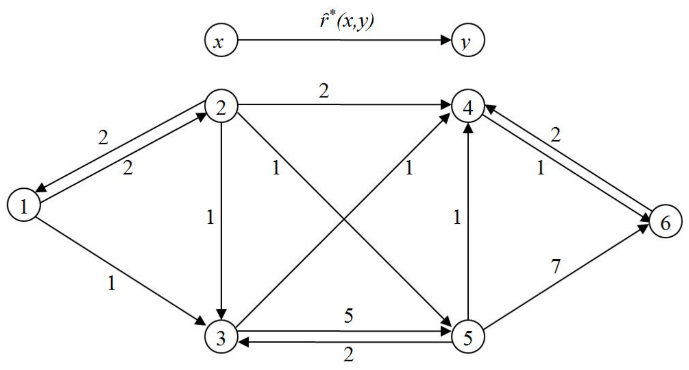

Figure 5. Residual network with no decreasing paths

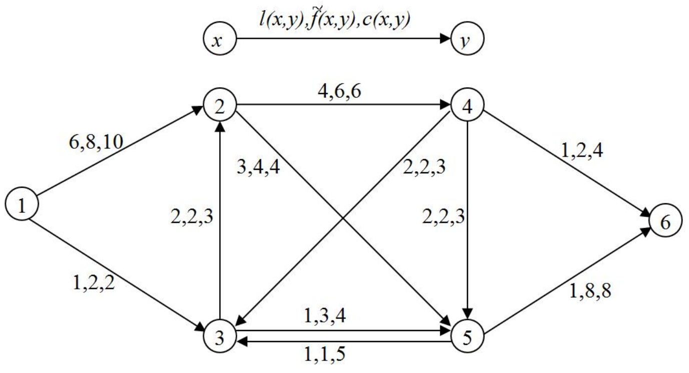

Figure 6. The resulting maximum flow

[2] L., CIUPALĂ, E., CIUREA, **About preflow algorithms for the minimum flow problem**, WSEAS Transaction on Computer Research, Issue 1, Vol 3, 2008, 35-42
[3] L., CIUPALĂ, E., CIUREA, **Sequential and parallel deficit scaling algorithms for minimum flow in bipartite networks**, WSEAS TRANSACTIONS on COMPUTERS, Crete, Greece, Issue 10, Vol 7, 2008, 1545-1554
[4] L., CIUPALĂ, E., CIUREA, **A Highest-Label Preflow Algorithm for the Minimum Flow Problem**, Proceedings of hte 11th WSEAS International Conference on COMPUTERS, Crete, Greece, 2007, 167-170
[5] L., CIUPALĂ, E., CIUREA, **A Parallel FIFO Preflow Algorithm for the Minimum Flow Problem**, International Journal of Computers, Communications and Control, Vol I, 2006, 135-139
[6] E., CIUREA, L., CIUPALĂ, **A C-Scaling Algorithm for the Minimum Flow Problem**, Proceedings of the Seventh Balkan Conference on Operation Research, Romania, 2005, 1-4

[7] E., CIUREA, L., CIUPALĂ, **Sequential and parallel algorithms for minimum flows**, Journal of Applied Mathematics and Computing, Vol 15, No 1-2, 2004, 53-75

[8] E., CIUREA, L., CIUPALĂ, **Algorithms for Minimum Flows**, Computer Science Journal of Moldova, Vol 9(3), Moldova, 2001, 275-290

[9] E., DINIC, **Algorithm for solution of a problem of maximum flow in network with power estimation**, Soviet Mathematics Doklady, Vol 11, 1970, 1277-1280

[10] S., EVEN, R., TARJAN, **Network flow and testing graph connectivity**, SIAM Journal on Computing 4, 1975, 507-518

[11] L.R., FORD, D.R., FULKERSON, **Flows in networks**, Princeton University Press, Princeton, NJ, 1962

[12] B.J., JENSEN, G., GUTIN, **Digraphs. Theory, Algorithms and Applications**, Springer Verlag, London, 2001

[13] D., JUGNICKEL, **Graphs, Networks and Algorithms**, Springer, Berlin, 1999

[14] G., RUHE, **Algorithmic Aspects of Flows in Networks**, Kluwer Academic Publishers, Dordrecht 1991.

[15] A., KARZANOV, **Determining the maximal flow in a network by the method of preflows**, Soviet Mathematics Doklady, Vol. 15, 1974, pp. 434-437

Eleonor CIUREA ”Transilvania” University of Brasov
Facultatea de Matematica si Informatica
Bd. Iuliu Maniu 50, cod 500091
Brasov, Romania
Email: e.ciurea@unitbv.ro and Mihai - Stefan IOLU ”Transilvania” University of Brasov
Facultatea de Matematica si Informatica
Bd. Iuliu Maniu 50, cod 500091
Brasov, Romania
Email: mihaiiolu@yahoo.com