Bulletin of the *Transilvania* University of Braşov • Vol 4(53) No. 1 - 2011
Series III: Mathematics, Informatics, Physics, 113-116

MINIMUM CUT WITH THE SMALLEST NUMBER OF ARCS

Laura CIUPALĂ[1]

Abstract

In this paper we focus on determining from among all minimum cuts in a given network, of a minimum cut containing the smallest number of arcs.

We will solve this problem by reducing it to a standard minimum cut problem in a transformed network.

2000 *Mathematics Subject Classification:* 90B10, 90C90.
Key words: network flow, minimum cut, maximum flow.

1 The minimum cut problem

The minimum cut problem is closely related to the maximum flow problem, which is one of the fundamental problems in network flow theory, that was studied extensively in the last six decades. The importance of both problems (maximum flow problem and minimum cut problem) is also due to the fact that they arise in almost all industries, including agriculture, communications, defense, education, energy, health care, medicine, manufacturing, retailing and transportation. Indeed, both of these problems are pervasive in practice.

Let $G = (N, A)$ be a directed graph, defined by a set N of n nodes and a set A of m arcs. Each arc $(x, y) \in A$ has a capacity $c(x, y)$. In the directed network $G = (N, A, c, s, t)$, two special nodes are specified: s is the source node and t is the sink node.

Let X and Y be two subsets of the node set N. We define the set of arcs $(X, Y) = \{(x, y)|(x, y) \in A,\ x \in X, y \in Y\}$.

For any function $g : N \times N \to \mathbb{R}^+$ and for any function $h : N \to \mathbb{R}^+$ we define

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

.

If $X = \{x\}$ or $Y = \{y\}$ then we will use $g(x, Y)$ or $g(X, y)$ instead of $g(X, Y)$.

A *flow* from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$ is a function $f : A \to \mathbb{R}^+$ which meets the follwing conditions:

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

[1]Faculty of Mathematics and Informatics, *Transilvania* University of Braşov, Romania, e-mail: laura_ciupala@yahoo.com

$$0 \leq f(x,y) \leq c(x,y), \qquad \forall (x,y) \in A. \tag{2}$$

We refer to v as the *value* of the flow f. A flow whose value is maximum is called a *maximum flow*.

Let f be a flow from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$. The *residual capacity* of the arc (x, y) corresponding to the flow f is defined as $r(x, y) = c(x, y) - f(x, y) + f(y, x)$. By convention, if an arc $(x, y) \notin A$, then we can add (x, y) to A and we will consider that $c(x, y) = 0$.

The *residual network* $G(f) = (N, A(f))$ corresponding to the flow f contains all those arcs with strictly positive residual capacity.

Let $X \subset N$ be a nonempty subset of the node set N and let $\overline{X} = N \backslash X$.

A *cut* is the following set of arcs $[X, \overline{X}] = (X, \overline{X}) \cup (\overline{X}, X)$, where $(X, \overline{X})$ is the set of forward arcs and $(\overline{X}, X)$ is the set of backward arcs of this cut.

A cut $[X, \overline{X}]$ where $s \in X$ and $t \in \overline{X}$ is named an $s - t$ *cut* .

The *capacity* $c[X, \overline{X}]$ of the $s-t$ cut $[X, \overline{X}]$ is definited as $c[X, \overline{X}] = c(X, \overline{X})$. An $s-t$ cut with a minimum capacity is referred to as a *minimum* $s - t$ *cut*.

The *residual capacity* $r[X, \overline{X}]$ of the $s-t$ cut $[X, \overline{X}]$ is definited as $r[X, \overline{X}] = r(X, \overline{X})$, where $r(X, \overline{X}) = c(X, \overline{X}) - f(X, \overline{X}) + f(\overline{X}, X)$.

Let f be a flow from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$ and let $[X, \overline{X}]$ be an $s-t$ cut, then $f[X, \overline{X}] = f(X, \overline{X}) - f(\overline{X}, X)$ is the net flow over the cut $[X, \overline{X}]$.

Theorem 1. *[1] If f is a flow of value v from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$ and if $[X, \overline{X}]$ is an $s - t$ cut, then $v = f[X, \overline{X}] \leq c[X, \overline{X}]$.*

Theorem 2. *(Max-flow min-cut theorem)[1] If f is a flow of value v from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$ and if $[X, \overline{X}]$ is an $s - t$ cut such that $v = c[X, \overline{X}]$ then f is a maximum flow and $[X, \overline{X}]$ is a minimum $s - t$ cut.*

The max-flow min-cut theorem shows the relationship between maximum flows and minimum cuts. For determining a maximum flow several algorithms were developed, starting from the well known labeling algorithm due to Ford and Fulkerson.

The algorithms for maximum flows can be divided into two classes:

1. augmenting path algorithms
2. preflow algorithms.

The augmenting path algorithms work in the following way: they start with a feasible flow and they proceed by determining augmenting paths and by increasing the flow along these paths. Any augmenting path algorithm terminates when the network contains no augmenting path, which means that the flow obtained is a maximum flow. The generic augmenting path algorithm for maximum flow does not specify any rule for determining the augmenting paths. By specifying different rules, many different algorithms were developed,

which have better running times than the running time of the generic augmenting path algorithm. The most known augmenting path algorithms can be found in [1].

The preflow algorithms for maximum flow begin with a feasible flow and send as much flow as possible from the source node to its neighbors, creating excesses in these nodes. The basic operation of any preflow algorithm for maximum flow is to select an active node (which is an intermediate node with excess) and to send the flow entering in it forward, closer to the sink node. For measuring closeness, distance labels are used. Any preflow algorithm for maximum flow terminates when the network contains no more active nodes, which means that the preflow is already a flow. Moreover, it is a maximum flow. The generic preflow algorithm for maximum flow does not specify any rule for selecting active nodes. By specifying different rules we can develop many different algorithms, which can have better running times than the generic preflow algorithm. The most known preflow algorithms can be found in [1].

If we have a maximum flow in a network, it is easy to determine a minimum cut, based on the following theorem.

Theorem 3. *[1] Let f be a flow from the source node s to the sink node t in the directed network $G = (N, A, c, s, t)$. Then f is a maximum flow if and only if the residual network $G(f) = (N, A(f))$ corresponding to the flow f contains no directed path from s to t.*

Consequently, if a maximum flow f is determined, we can easily establish a minimum $s - t$ cut $[X, \overline{X}]$ in the following manner: X is the set of nodes reachable from s in the residual network $G(f) = (N, A(f))$ and $\overline{X} = N \backslash X$.

2 An algorithm for determining a minimum cut with the smallest number of arcs

In the previous section we have shown how a minimum cut can be determined. In this section we will focus on a more complex problem, which is the problem of establishing a minimum cut with the smallest number of arcs. We will transform this problem into a standard minimum cut problem in a modified network.

Let $G = (N, A, c, s, t)$ be a directed network in which we need to find a minimum $s - t$ cut with the smallest number of arcs. We construct the directed network $G' = (N, A, c', s, t)$, where $c'(x, y) = m \cdot c(x, y) + 1$, $\forall (x, y) \in A$.

Let $[X, \overline{X}]$ be an $s - t$ cut in G'. Then the capacity of this cut in G' is

$$c'[X, \overline{X}] = m \cdot c[X, \overline{X}] + \left|[X, \overline{X}]\right|,$$

where $c[X, \overline{X}]$ is the capacity of this cut in G.

The algorithm for determining a minimum cut with the smallest number of arcs is the following:

MinCut with smallest arc Algorithm;
Begin

determine the transformed network $G' = (N, A, c', s, t)$;
determine a minimum $s - t$ cut $[X, \overline{X}]$ in G';
$[X, \overline{X}]$ is a minimum $s - t$ cut with the smallest number of arcs in G;
end.

Theorem 4. *(Correctness theorem) The algorithm determines correctly a minimum $s - t$ cut with the smallest number of arcs in the network $G = (N, A, c, s, t)$.*

Proof. The algorithm determines a minimum $s - t$ cut $[X, \overline{X}]$ in the transformed network $G' = (N, A, c', s, t)$. We will prove that $[X, \overline{X}]$ is a minimum $s - t$ cut with the smallest number of arcs in the directed network $G = (N, A, c, s, t)$ based on the fact that any minimum $s - t$ cut in G' is a minimum $s - t$ cut with the smallest number of arcs in G.

Let $[X, \overline{X}]$ be a minimum $s - t$ cut in G'. It follows that its capacity in G', which is $c'[X, \overline{X}] = m \cdot c[X, \overline{X}] + \left|[X, \overline{X}]\right|$ has to be minimum.

Ad absurdum, $[X, \overline{X}]$ is not a minimum $s - t$ cut with the smallest number of arcs in G. This means that there is another $s - t$ cut $[Y, \overline{Y}]$ in G which is a minimum $s - t$ cut with the smallest number of arcs in G. Two cases are possible:

1. $c[Y, \overline{Y}] < c[X, \overline{X}]$. In this case the capacity of the $s - t$ cut $[Y, \overline{Y}]$ in G' is
$$c'[Y, \overline{Y}] = m \cdot c[Y, \overline{Y}] + \left|[Y, \overline{Y}]\right| < m \cdot c[X, \overline{X}] + \left|[X, \overline{X}]\right| = c'[X, \overline{X}],$$
which is a contradiction to $[X, \overline{X}]$ being a minimum $s - t$ cut in G'.

2. $c[Y, \overline{Y}] = c[X, \overline{X}]$ and $\left|[Y, \overline{Y}]\right| < \left|[X, \overline{X}]\right|$. In this case the capacity of the $s - t$ cut $[Y, \overline{Y}]$ in G' is
$$c'[Y, \overline{Y}] = m \cdot c[Y, \overline{Y}] + \left|[Y, \overline{Y}]\right| < m \cdot c[X, \overline{X}] + \left|[X, \overline{X}]\right| = c'[X, \overline{X}],$$
which, again, is a contradiction to $[X, \overline{X}]$ being a minimum $s - t$ cut in G'.

Consequenlty, if $[X, \overline{X}]$ is a minimum $s - t$ cut in G' then it is a minimum $s - t$ cut with the smallest number of arcs in G. Thus, the algorithm terminates with a minimum $s - t$ cut with the smallest number of arcs.

□

Theorem 5. *(Complexity theorem) The algorithm determines a minimum $s - t$ cut with the smallest number of arcs in the network G in $M(n, m)$ time, where $M(n, m)$ is the time needed to determine a minimum $s - t$ cut in a network with n nodes and m arcs.*

Proof. The complexity of the algorithm for determining a minimum $s - t$ cut with the smallest number of arcs in the network G is the complexity of determining a minimum $s - t$ cut in the transformed network G', which is $M(n, m)$. □

References

[1] Ahuja, R.; Magnanti, T.; Orlin, J.: *Network Flow. Theory, Algorithms and Applications*, Prentice Hall, New Jersey, 1999.