The maximum flow problem

A. Agnetis[*]

1 Basic properties

Given a network $G = (N, A)$ (having $|N| = n$ nodes and $|A| = m$ arcs), and two nodes s (source) and t (sink), the *maximum flow problem* consists in finding the flow distribution that allows sending the maximum possible amount of flow from s to t. (Though this is not strictly necessary, for simplicity we assume that $\delta^-(s) = \delta^+(t) = \emptyset$, i.e., s and t are assumed not to have any ingoing and outgoing arcs respectively.) More formally, it can be formulated as:

$$\begin{aligned}
\max \quad & v && (1)\\
\sum_{(s,j)\in\delta^+(s)} x_{sj} &= v && (2)\\
-\sum_{(i,t)\in\delta^-(t)} x_{it} &= -v && (3)\\
\sum_{(h,j)\in\delta^+(h)} x_{hj} - \sum_{(i,h)\in\delta^-(h)} x_{ih} &= 0, \quad h \in N \setminus \{s,t\} && (4)\\
x_{ij} &\le k_{ij} \quad (i,j) \in A && (5)\\
x_{ij} &\ge 0 \quad (i,j) \in A && (6)
\end{aligned}$$

where variables x_{ij} indicate the flow in arc (i, j) and v is the *value* of the flow, to be maximized. The value k_{ij} is the *capacity* of arc (i, j).

A vector $x \in I\!R^m$ that satisifes (2)–(6) is referred to as a *feasible flow*. The (4) are called *continuity constraints*, while the (5) are called *capacity constraints*. The problem consists in finding the feasible flow that maximizes v.

We note that the coefficient matrix of (2)–(6) is the node-arc incidence matrix of G, and hence it is totally unimodular. As a consequence, as long as all k_{ij} are integer, the optimal solution is also integer (though this is not required by the problem formulation).

The following fundamental definitions can be given.

A *cut* $(S, \bar{S})$ is a partition of the node set such that $s \in S$ and $t \in \bar{S}$.

[*]Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche - Università di Siena

Given a feasible flow x, the *flow through a cut* $(S, \bar{S})$ is defined as

$$\phi(S) := \sum_{(i,j)\in\delta^+(S)} x_{ij} - \sum_{(i,j)\in\delta^-(S)} x_{ij}. \tag{7}$$

Notice that from (2), $v = \phi(\{s\}) = \sum_{(i,j)\in\delta^+(S)} x_{ij}$. It is easy to show the following property, which simply derives from the fact that in the network there are no leaks, so whatever is generated in node s eventually reaches node t.

THEOREM 1 *Given a feasible flow x of value v, the flow $\phi(S)$ through* any *cut $(S, \bar{S})$ is equal to v.*

□

Another important notion is that of *capacity of a cut* $(S, \bar{S})$, denoted by $k(S)$:

$$k(S) := \sum_{(i,j)\in\delta^+(S)} k_{ij}. \tag{8}$$

THEOREM 2 *Given a feasible flow x of value v, and a cut $(S, \bar{S})$, it holds*

$$v \leq k(S).$$

Proof. From Theorem 1, and from (7) and (8),

$$v = \phi(S) = \sum_{(i,j)\in\delta^+(S)} x_{ij} - \sum_{(i,j)\in\delta^-(S)} x_{ij} \leq \sum_{(i,j)\in\delta^+(S)} k_{ij} - \sum_{(i,j)\in\delta^-(S)} x_{ij}$$

$$\leq \sum_{(i,j)\in\delta^+(S)} k_{ij} = k(S)$$

where the last inequality is due to the fact that $x_{ij} \geq 0$ for all (i, j). □

The two above theorems imply that even the value of the maximum flow, v^*, cannot exceed the capacity $k(S^*)$ of the cut having minimum capacity. In fact, we are going to show that these two values are indeed equal.

To prove such result, it is useful to introduce the concept of *incremental network*, designed to represent possible variations in the arc flows with respect to a certain current feasible flow x. In the following, given a feasible flow x, we say that if $x_{ij} = k_{ij}$, arc (i, j) is *saturated.* On the contrary, if $x_{ij} = 0$, arc (i, j) is *empty.*

Given a network G and a feasible flow x, the incremental network $\bar{G} = (N, \bar{A})$ is obtained from the original network G replacing each arc $(i, j) \in A$ with two arcs:

- a direct arc (i, j), having residual capacity $\bar{k}_{ij} = k_{ij} - x_{ij}$;
- a reverse arc (j, i), having residual capacity $\bar{k}_{ji} = x_{ij}$.

Arcs having zero residual capacity can be eliminated from $\bar{G}$.

The incremental network accounts for the possibility of increasing the flow in non-saturated arcs (corresponding to direct arcs with nonzero residual capacity) and decreasing it in nonempty arcs (corresponding to reverse arcs with nonzero residual capacity). Notice that an incremental flow $\bar{x}_{ji}$ on a reverse arc $(j, i) \in \bar{A}$ must be interpreted as a flow *decrease* in the arc $(i, j) \in A$ of the original network G.

It is apparent that if a path from s to t exists in the incremental network $\bar{G}$, then by modifying the flow in the corresponding arcs in the original network G, the current value v of the flow can be increased. Such a path is called *augmenting path.* Direct arcs of an augmenting path on $\bar{G}$ correspond to arcs in G in which flow can be increased. Reverse arcs of an augmenting path on $\bar{G}$ correspond to arcs in G in which flow can be decreased. We are now in the position of proving the following theorem.

THEOREM 3 *A feasible flow x is optimal for the maximum flow problem if and only if node t is not reachable from node s in the incremental network $\bar{G}$ associated with the flow x^*. In this case there is a cut $(S^*, \bar{S}^*)$ such that $\phi(S^*) = k(S^*)$, and $(S^*, \bar{S}^*)$ is a cut having minimum capacity (*minimum cut*).*

Proof. Given the feasible flow x, let v be the current flow value. If t can be reached from s in $\bar{G}$, there exists an augmenting path P from s to t. Let

$$\delta = \min\{\bar{k}_{uv} : (u, v) \in P\}.$$

Then, for each arc $(u, v) \in P$, it is possible to define a new flow x' as

- $x'_{uv} := x_{uv} + \delta$ if $(u, v) \in P$ and it is a direct arc
- $x'_{uv} := x_{uv} - \delta$ if $(u, v) \in P$ and it is a reverse arc
- $x'_{uv} := x_{uv}$ if $(u, v) \notin P$.

It is easy to check that x' is a feasible flow, and its value is $v + \delta$. Hence, the original flow x was not optimal.

Suppose now that in $\bar{G}$ there is no path from s to t. This implies that *in the incremental network* $\bar{G}$ there is a cut $(S^*, \bar{S}^*)$ with no outgoing arcs, i.e., such that $\delta^+(S^*) = \emptyset$. From the definition of incremental network, this in turn implies that *in the original network G*:

- each arc $(i, j) \in \delta^+(S^*)$ (i.e., each arc outgoing S^*) is saturated
- each arc $(i, j) \in \delta^-(S^*)$ (i.e., each arc ingoing S^*) is empty

As a consequence,

$$v^* = \phi(S^*) = \sum_{(i,j)\in\delta^+(S^*)} x_{ij} - \sum_{(i,j)\in\delta^-(S^*)} x_{ij} = \sum_{(i,j)\in\delta^+(S^*)} k_{ij} = k(S^*).$$

So, in view of Theorem 2, v^* and $k(S^*)$ are respectively the maximum flow value and the smallest capacity value. $\square$

2 Ford-Fulkerson algorithm

Theorem 3 suggests a very simple algorithm to solve maximum flow problems. Starting from any feasible flow x (possibly, $x = 0$ in which all arcs are empty), the algorithm computes a sequence of augmenting paths, and every time it updates the flow in all arcs of the path. If no augmenting paths are found, from Theorem 3 we are ensured that the current flow is optimal and the algorithm stops.

The computation of an augmenting path can be carried out performing a *visit* of the nodes of G (starting from node s). Whenever a new node is visited, it receives a *label*, provided that the following holds:

- a node j can be labeled from node i if $(i,j) \in A$ and (i,j) is not saturated, or
- a node j can be labeled from node i if $(j,i) \in A$ and (j,i) is not empty.

In both cases, labeling a node indicates that the flow in arc (i,j) or (j,i) can be modified in such a way that the value of the flow v can be increased. When node t receives a label, an augmenting path is found. We indicate by $[\varepsilon(j), pred(j)]$ the label assigned to node j during the visit. The value $\varepsilon(j)$ indicates how much additional flow can be sent from s to node j. The value $pred(j)$ denotes the node from which j has been labeled. Hence, if we label node j from node i, then

- $\varepsilon(j) := \min\{\epsilon(i), \bar{k}_{ij}\}$
- $pred(j) := i^+$ if $(i,j) \in A$ and $pred(j) := i^-$ if $(j,i) \in A$.

Clearly, when node t is reached, $\varepsilon(t)$ denotes by how much can the current flow value be increased, while the values $pred(j)$ allow one to backtrack the augmenting path.

Notice that we did not specify any particular order in which the nodes should be visited. In fact, any visiting strategy is fine, as far as the correctness of the algorithm is concerned. However, the order in which the nodes are visited can make a difference in terms of efficiency. In particular, if nodes are visited in depth-first order, the algorithm

Algorithm 1 Structure of the Ford-Fulkerson algorithm.

```
1: Let $x$ be a feasible flow;
2: repeat
3:     Search for an augmenting path;
4:     if an augmenting path exists then
5:         update the flow $x$
6:     end if
7: until no augmenting path exists
8: $x^* := x$
```

may not have polynomial complexity. On the contrary, if nodes are visited in breadth-first order, the algorithm converges in polynomial time. The algorithm is known as Edmonds-Karp algorithm, and the corresponding algorithm for computing an augmenting path is reported.

Algorithm 2 Computation of an augmenting path in Edmonds-Karp algorithm.

```
1: for $j = 1$ to $n$ do
2:     $pred(j) = 0$;
3: end for
4: $\epsilon(s) = +\infty$; $Q := \{s\}$;
5: while $Q \neq \emptyset$ and $pred(t) = 0$ do
6:     let $h \in Q$ be the head of $Q$ and remove it from $Q$;
7:     for each $(h, j) \in \delta^+(h)$ such that $x_{hj} < k_{hj}$ do
8:         $\epsilon(j) := \min\{\epsilon(h), k_{hj} - x_{hj}\}$;
9:         $pred(j) = h^+$;
10:        push $j$ into $Q$;
11:    end for
12:    for each $(i, h) \in \delta^-(h)$ such that $x_{ih} > 0$ do
13:        $\epsilon(i) := \min\{\epsilon(h), x_{ih}\}$;
14:        $pred(i) = h^-$;
15:        push $i$ into $Q$.
16:    end for
17: end while
```