Proof of Menger's theorem

Algo Telecoms - Jean–Claude Bermond

We give here a proof of Menger's theorem (see 7.45 in the chapter 7 of the book of Kleinberg Tardos) which is a direct adaptation of the flow algorithm of Ford and Fulkerson (see section 7.2) but with integer capacities = 1.

Theorem 1 *Let G be a digraph and s and t two vertices. The maximum number of pairwise arc disjoint paths from s to t is equal to the minimum number of arcs whose removal separates s from t (= destroy all the paths from s to t)*

Easy part : If there exist k pairwise arc disjoint paths from s to t, then one needs to delete at least one arc in each of these paths to separate s from t. So to the minimum number of arcs whose removal separates s from t is greater than or equal to the maximum number of pairwise arc disjoint paths from s to t (equivalent to property (7.8) pp 347-348).

To prove the converse we will use a recursive algorithmic approach ("augmenting flow") as follows. Suppose we have found a set of k pairwise arc disjoint paths from s to t, then we will either construct a set of $k+1$ pairwise arc disjoint paths from s to t or find a set of k arcs whose removal separates s from t. The induction can be started with $k = 0$ or $k = 1$ by finding a path from s to t.

Let $\mathscr{P}$ be a set of k pairwise arc disjoint paths from s to t, $\mathscr{P} = \{P_1, P_2, \ldots, P_k\}$. Let us construct a set S of vertices as follows

Algorithm to construct S

- (i) Put s in S.
- (ii) If $x \in S$ and if there exists an arc (x, y) such that $(x, y) \notin \mathscr{P}$ (i.e. (x, y) is in none of the paths P_i); then put y in S.
- (iii) If $x \in S$ and if there exists an arc (y, x) such that $(x, y) \in \mathscr{P}$, then put y in S.

[(ii) corresponds to push one unit of flow forwards and (iii) backwards (see page 341)].

Two cases can appear at the end of the algorithm.

Case A : $t \in S$. In that case we will construct a set of $k+1$ pairwise arc disjoint paths from s to t.

Then let $P_{k+1} : s = x_0, e_1, x_1, \ldots, x_j, e_{j+1}, x_{j+1}, \ldots, e_p, x_p = t$ be a chain (not directed) from s to t where e_{j+1} is either an arc from x_j to x_{j+1} and so $e_{j+1} \notin \mathscr{P}$, called of type 1 or an arc from x_{j+1} to x_j and so $e_{j+1} \in \mathscr{P}$, called of type 2. [That corresponds to an augmenting dipath in the residual graph; arcs of type 1 are forward edges and arcs of type 2 are backward edges see definition in chapter 7 pp 341-42].

If P_{k+1} does not contain an arc of type 2, then the set of paths $P_1, P_2, \ldots, P_k, P_{k+1}$ is a a set of $k+1$ pairwise arc disjoint paths from s to t.

Otherwise starting from the set of $k+1$ paths (not pairwise arc disjoint) $P_1, P_2, \ldots, P_k, P_{k+1}$ we will construct a set of $k+1$ paths having one arc of type 2 less. Repeating this construction we get after p steps (if p is the number of arcs of type 2) a set of $k+1$ pairwise arc disjoint paths from s to t.

Let e_{j+1} be the first arc of type 2 from x_{j+1} to x_j in P_{k+1} and suppose $e_{j+1} \in P_{i_0}$. Then the set of paths:

$P'_i = P_i$ for $1 \leq i \neq i_0 \leq k$

$P'_{i_0} = P_{k+1}[s, x_j] \cup P_{i_0}[x_j, t]$

$P'_{k+1} = P_{i_0}[s, x_{j+1}] \cup P_{k+1}[x_{j+1}, t]$

has one arc of type 2 less than $P_1, P_2, \ldots, P_k, P_{k+1}$ as wanted.

Case B : $t \notin S$. In that case we will find a set of k arcs whose removal separates s from t. (equivalent of(7.9) in chapter 7 pp 348-349)

Let $T = V - S$. The set of arcs from S to T (arcs (x, y) with $x \in S$ and $y \notin S$ separates S from T and so s from t. Each arc from S to T belongs to $\mathscr{P}$; otherwise we could have applied step (ii) of the algorithm and so the algorithm was not finished. Furthermore, a path P_i cannot contain two arcs from S from T, otherwise there will exist an arc (y, x) from T to S with $y \in T$ and $x \in S$ and this arc should have been added to the set S by the step (iii) of the algorithm. So we have at most k arcs from S to T and the removal of these k arcs separates s from t.

$[(S, T)$ is a minimum cut equal to the flow].