TD: Flows

Algorithmique et Programmation

Pierre Senellart
pierre.senellart@ens.fr

November 24, 2016

The goal of this exercise session is to study some applications of network flows and of the maximum-flow/minimum-cut theorem, and an efficient algorithm to compute maximum flows.

1 Flows and Maximum Bipartite Matchings

Let $G = (V, E)$ be a bipartite undirected graph: V is partitioned into two subsets $V = X \cup Y$.

1a. Propose (for now without proof) a flow network whose maximum flow is equal to the size of a maximum matching in G.

1b. Prove that, in such a flow network, the flow computed by the Ford–Fulkerson algorithm can be used to compute a maximum matching.

1c. What is the asymptotic complexity of computing a maximum matching in a bipartite graph by such a procedure? Contrast this with the $O(|V|^2 \times |E|)$ complexity of the blossom algorithm on arbitrary undirected graphs.

1d. Use flow networks to prove Hall's marriage theorem:
Theorem (Hall, 1935)**.** *Let G be an undirected bipartite graph (V, E) with bipartition $V = X \cup Y$ such that $|X| = |Y|$. For $S \subseteq X$, let $N(S) = \bigcup_{u \in S} \{ v \in Y \mid (u, v) \in E \}$. There is a perfect matching in G (i.e., a matching of size $|X| = |Y|$) if and only if:*

$$\forall S \subseteq X, \quad |S| \leqslant |N(S)|.$$

2 Goldberg–Tarjan Preflow-Push Algorithm

The preflow-push algorithm, a solution to the maximum flow/minimum cut problem, is based on the notion of *preflow*, which relaxes the flow conservation constraint. Let $\mathcal{T} = (V, E, c, s, t)$ be a flow network. We assume that we do not have $(u, v) \in E$ and $(v, u) \in E$ at the same time for simplicity. A *preflow* in $\mathcal{T}$ is a function $f : V^2 \to \mathbb{R}$ which satisfies:

- (*Symmetry*) $\forall (u,v) \in V^2, f(u,v) = -f(v,u)$
- (*Capacity constraint*) $\forall (u,v) \in V^2, f(u,v) \leqslant c(u,v)$
- (*Relaxed flow conservation*) $\forall u \in V \setminus \{s,t\}, \sum_{v \in V} f(u,v) \leqslant 0$

This definition means that, in a preflow, a node u can have some *overflow*

$$o(u) \coloneqq \sum_{v \in V} f(v,u),$$

which means it can receive more from the nodes it is pointed by than it sends to the nodes it points to. The preflow-push algorithm, as well as other algorithms working with preflows, maintains at each step a preflow in $\mathcal{T}$, converging finally toward a flow in $\mathcal{T}$ which is maximal.

All nodes are assigned a height (0 in the beginning for all nodes except the source). At each iteration, the preflow is *pushed* from a node with overflow to a lower node. If there are no lower nodes to unload a node with overflow, this node is *raised*. The algorithm ends when there are no nodes with overflow any longer.

Formally, the algorithm is as follows:

Algorithm: Goldberg–Tarjan Preflow-Push

Input: Flow network $\mathcal{T} = (V, E, c, s, t)$
Output: Maximum flow f in $\mathcal{T}$

begin
  $h(s) \leftarrow |V|$;
  for $u \in V \setminus \{s\}$ **do**
    $h(u) \leftarrow 0$;
  for $(u,v) \in V^2$ **do**
    $f(u,v) \leftarrow 0$;
  for $u \in V, c(s,u) > 0$ **do**
    $f(s,u) \leftarrow c(s,u)$;
    $f(u,s) \leftarrow -c(s,u)$;
  while $\exists u \in V \setminus \{s,t\}, o(u) > 0$ **do**
    if $\exists v \in V, f(u,v) < c(u,v) \wedge h(v) = h(u) - 1$ **then**
      $f(u,v) \leftarrow f(u,v) + \min(c(u,v) - f(u,v), o(u))$;
      $f(v,u) \leftarrow f(v,u) - \min(c(u,v) - f(u,v), o(u))$;
    else
      $h(u) \leftarrow 1 + \min\{h(v) \mid v \in V, f(u,v) < c(u,v)\}$;
  return f;

2a. We first prove the correctness of the algorithm

  2aα) Show that at all steps in the algorithm, $\forall (u,v) \in V^2, f(u,v) \leqslant c(u,v)$ (we write $c(u,v) = 0$ if $(u,v) \notin E$).

  2aβ) Show that at all steps in the algorithm, if there exists v such that $f(u,v) < c(u,v)$, then $h(u) \leqslant h(v) + 1$.

2aγ) Show that at all steps in the algorithm, there is no path from s to t through only under-capacity edges (i.e., $f(u,v) < c(u,v)$ for all (u,v) on the path).

2aδ) Show that the algorithm indeed computes a maximum flow in $\mathcal{T}$.

2b. We now establish the asymptotic complexity of the algorithm.

2bα) Show that if a node u is such that $o(u) > 0$ then there is a path from u to s through only under-capacity edges (possibly edges not in E).

2bβ) Deduce from this an upper bound on the height of any given node.

2bγ) Deduce an upper bound on the number of times a node is raised.

2bδ) Show that the number of times flow is pushed on an edge saturating the capacity of its edge is in $O(|V| \times |E|)$.

2bε) By considering $\sum_{\substack{v \in V \\ o(v)>0}} h(v)$, show that the number of times flow is pushed on an edge that remains under-capacity is in $O(|V|^2 \times |E|)$.

2bζ) Conclude on the complexity of Goldberg–Tarjen preflow-push algorithm and compare it to the complexity of Edmonds–Karp algorithm.

3 Can a Team Still Win?

We consider a sports league championship, where $n+1$ teams $X_0, X_1, \ldots, X_n$ compete. Every team faces every other team several times during the season. Every victory yields one point, defeats yield 0. There are no ties, every game is a victory for one team and a defeat for the other.

We are mid-season and we want to know if team X_0 can still win. It has won w_0 games so far, and has still k games to play. For $i, j \in \{1, \ldots, n\}$, we let $a_{i,j}$ be the number of games remaining between teams X_i and X_j, and w_i the number of victories for team i up to now.

3a. Model this problem with a flow network whose maximum flow value is $\sum_{i,j} a_{i,j}$ if and only if team X_0 can still win the championship (we assume every team having the maximum number points wins the competition). Prove this fact.

3b. Deduce that X_0 can win the championship if and only if:

$$\forall F \subseteq \{1, \ldots, n\}, \qquad |F| \times (w_0 + k) \geqslant \sum_{i \in F} w_i + \sum_{(i,j) \in F^2} a_{i,j}.$$