Fall 2015, MATH-566

Network flows - Fast(er) Algorithm

Edmonds-Karp Algorithm
Input: Network (G, u, s, t).
Output: and s-t-flow f of maximum value

1. $f(e) = 4$ for all $e \in E(G)$
2. while f-augmenting path exists:
3. find shortest f-augmenting path P
4. compute $\gamma := \min_{e \in E(P)} u_f(e)$
5. augment f along P by γ (as much as possible)

Note that shortest path can be implemented by
BFS (Breath First Search) algorithm:
Input: Graph G, $s \in V(G)$.
Output: spanning tree T of shortest paths to s

1. $R = \{s\}$, $Q = (s)$, $T = (V, \emptyset)$.
2. while Q is not empty:
3. remove the first entry in Q, denote it by u.
4. $\forall uv \in E(G)$, if $v \notin R$
5. add v at the end of Q; add v to R; add uv to T

1: What is running time of BFS?

Lemma 8.13 Let $f_1; f_2; \dots$ be a sequence of flows such that f_{i+1} results from f_i by augmenting along P_i, where P_i is a shortest f_i-augmenting path. Then
(a) $|E(P_k)| \leq |E(P_{k+1})|$ for all k.
(b) $|E(P_k)| + 2 \leq |E(P_l)|$ for all $k < l$ such that $P_k \cup P_l$ contains a pair of reverse edges.

2: Prove (a). Consider edges X of P_k and P_{k+1} (with multiplicity) together (and erase reverse edges). Show that $|P_k|$ is at most half of the number of edges in X.

3: Prove (b). Fix k and consider the smallest $l > k$ such that P_l uses a reverse edge of P_k. Use that there was a reverse edge.

4: How many augmentations are needed in Edmonds-Karp Algorithm? What is the resulting running time?

Network flows as linear programs

5: Formulate the maximum flow problem for network (G, u, s, t) as a linear program (P). (Hint: Similar to shortest path.) Assume $G = (V, E)$.

6: Write the dual (D) to (P). Use dual variables y_v, where $v \in V \setminus \{s, t\}$ for $\sum_{uv} f_{uv} - \sum_{vw} f_{vw} = 0$, and z_e such that $e \in E$ for $f_e \leq u(e)$.

7: Add two artificial variables $y_s = 0$ and $y_t = -1$. Then the constraints all unify to the form $-y_v + y_w + z_{vw} \geq 0$ for all $vw \in E$. Write the new program (D').

8: Recall that every s-t-flow can be decomposed into weighted s-t-paths. Try to interpret (D') using s-t paths.