COMP 362 - Winter 2017 - Assignment 1

Due: 6pm Jan 26th

General rules: In solving these questions you may consult your book; you can discuss high level ideas with each other. But each student must find and write his/her own solution. You should drop your solutions in the assignment drop-off box located in the Trottier Building on the 2nd floor.

1. (5 points) What is the running time of the Ford-Fulkerson algorithm for finding the maximum bipartite matching?

   Solution: Each augmentation requires $O(m)$ where m is the number of edges, and there are at most n augmentations, as the size of the matching cannot be larger than n, the number of vertices. So $O(mn)$.

2. (a) (10 points) Show that for every flow network, there exists an execution of the Ford-Fulkerson algorithm that never decreases the value of the flow on any of the edges (i.e. never "pushes back" the flow on any of the edges).

   Solution: See the solution to Question 4(a).

   (b) (5 points) On the other hand, show that if we modify the Ford-Fulkerson algorithm so that it does not decrease the flow on any of the edges (i.e. we do not add the opposite edges to the residual graph), then the algorithm might terminate without finding the maximum flow.

   Solution: Consider the network with vertices s, a, b, t with edges sa, sb, ab, at, bt each with capacity 1. Now if we first augment the path (s, a, b, t), then we cannot find the max-flow if we do not decrease the flow on the edge ab.

3. (20 Points) Consider a variation of the Ford-Fulkerson algorithm in which to find an augmenting path, we run a BFS to find the shortest path from s to t in the residual graph. Prove that throughout the algorithm, the length of the shortest path never decreases. Moreover show that after at most m augmentations the length of the shortest

path increases. Here m is the number of the edges of G. What is the running time of this algorithm?

Solution: See Section 3 of `http://theory.stanford.edu/~tim/w16/l/l2.pdf`.

4. (a) (15 points) Show that for every flow network, there exists an execution of the Ford-Fulkerson algorithm that finds the maximum flow after at most m augmentations. Here m is the number of the edges of G.

   Solution: To prove this, we apply the following algorithm to decompose a maximum flow f into a set of flow-paths:

   - Repeat the following two steps until there is no such s, t-path:
   - Find an s, t-path P with positive flow, i.e., $f(e) > 0$ for all $e \in P$.
   - Let Δ be the minimum value of the flow f on edges of P. Decrease the flow f on each edge $e \in P$ by Δ.

   In every iteration the value of f on at least one edge becomes 0. Hence the number of iterations is at most m. Using these paths as augmenting paths (with augmenting value Δ) leads to the desired result.

   (b) (5 points) On the other hand, construct examples of flow networks that require $\Theta(m)$ augmentations no matter how we choose the augmenting paths.

   Solution: Consider a flow network with vertices $s, t, v_1, \ldots, v_n$ and edges sv_i and v_it for $i = 1, \ldots, n$ all with capacities 1. The number of edges is $2n$ and one needs at least n augmentations to find the max flow.

5. (20 points) Let G be a graph in which some of the edges are directed, and some of them are undirected. We want to assign an orientation to each undirected edge so that in the resulted directed graph the indegree of every vertex is equal to its outdegree. Here the indegree is the number of edges pointing towards the vertex and the outdegree is the number of the edges pointing away from the vertex. Solve this problem by reducing it to a max-flow problem.

   Solution: The idea is that every undirected edge ab can contribute 1 to the indegree of a or b, and we want to decide these contributions so that every vertex a will finally have indegree $\deg(a)/2$, where $\deg(a)$ is the number of edges (directed and undirected) incident to a.

Construct a network flow as in the following. For every vertex a in the graph put a vertex v_a in the network, and for every edge ab in the graph put a vertex v_{ab} in the network. If ab has no direction, then add the edges $v_{ab}v_a$ and $v_{ab}v_b$, each with capacity 1 to the network. If ab has already a direction, say from a to b, then only add the edge $v_{ab}v_b$, and if the direction is from b to a then add $v_{ab}v_a$ instead.

Add a source s and connect it to each vertex v_{ab} with an edge of capacity 1. Add a sink t, and for every vertex v_a, add the edge v_at with capacity $\deg(a)/2$.

6. (20 Points) Consider the problem of finding the largest matching in a bipartite graph H with n vertices. We saw how we can solve this problem by modeling it as a max-flow problem in a flow network G. Suppose we run the Ford-Fulkerson algorithm on G and at some point during the execution we arrive at a flow f (not necessarily maximum yet). Suppose that the length of the shortest s-t path in G_f is larger than $\sqrt{n}$. Prove that $\mathrm{val}(f) \geq M - \sqrt{n}$ where M is max flow (which is equal to the size of the largest matching).

   Solution: Partition the residual graph G_f according to the distance of the vertices from s. In other words let $k > \sqrt{n}$ be the distance from s to t in the residual graph, and for $i = 0, \ldots, k$ let S_i be the set of the vertices at distance exactly i from s. Obviously there exists $1 < j < k$ such that $|S_j| \leq \sqrt{n}$. Note that in the original network, except for s and t, every other vertex has either indegree 1 or outdegree 1. It is easy to see that this property is preserved in the residual graph G_f. Let $A = S_0 \cup \ldots \cup S_{j-1} \cup T$ where T is the vertices in S_j with outdegree 1, and let B be the set of the vertices not in A. We claim that (A, B) is a cut in G_f with capacity at most $|S_j| \leq \sqrt{n}$.

   To prove this note that the only edges from A to B are the edges that either connect S_{j-1} to $S_j \setminus T$ or the ones that connect vertices in T to vertices in B. Since every vertex in $S_j \setminus T$ has indegree at most 1 and every vertex in T has outdegree at most 1, the total number of such edges is at most $|T| + |S_j \setminus T| = |S_j| \leq \sqrt{n}$. Each edge in G_f has residual capacity at most 1 and thus the capacity of the cut is at most $\sqrt{n}$ as desired.