Fall 2024, CS 598: Topics in Graph Algorithms
Homework 1
Due: Tue, 17th Sept, 2024

Instructions and Policy: You can work in groups of up to two. Each group needs to submit only one solution. You need to indicate the names of the people you discussed a problem with. Solutions to most of these problems can be found from one source or the other. Try to solve on your own first, and cite your sources if you do use them.

Please write clearly and concisely. Refer to known facts. You should try to convince me that you know the solution, as quickly as possible.

Problem 1 Yao's $O(m \log\log n)$-time algorithm for MST. See Prob 3 from CMU's homework `https://www.cs.cmu.edu/~15850/hws/h1.pdf`.

Problem 2 Let $G = (V, E)$ be an undirected graph with non-negative edge weights $w : E \to \mathbb{R}_+$.

- For a spanning tree $T = (V, E_T)$ define its bottleneck weight, denoted by $b(T)$, as the weight of the maximum weight edge in T, that is $b(T) = \max_{e \in E_T} w(e)$. Describe a deterministic linear time algorithm, that given a graph, outputs a spanning tree with *minimum* bottleneck weight. *Hint:* You may want to recall the linear-time Selection algorithm.
- For a path P we define its bottleneck weight, denote by $b(P)$, to be the weight of the maximum weight edge on P, that is $b(P) = \max_{e \in P} w(e)$. The bottleneck shortest path distance between two distinct vertices s, t, denoted by $d_b(s,t)$, is defined as $d_b(s,t) = \min_{P \in \mathcal{P}_{s,t}} b(P)$ where $\mathcal{P}_{s,t}$ is the set of all s-t paths in G. Prove that, in an undirected graph, the MST is a compact structure that contains a bottleneck weight path for *every* pair of vertices s, t.
- Now let $G = (V, E)$ be a directed graph with non-negative edge-weights. Give a source vertex s we wish to compute the single source shortest bottleneck weight distances to every vertex in G, that is we wish to compute $d_b(s, v)$ for all v. Describe an algorithm based on Fibonacci heaps that can do this in $O(m + n \log n)$ time. Note that this is the same time as that for Dijkstra's shortest path algorithm. **Extra credit:** Obtain faster running time without looking at the known literature.

Problem 3 Consider the dual of the tree packing LP given by

$$\min \sum_{e \in E} c_e x_e \text{ s.t } \sum_{e \in E_T} x_e \geq 1 \quad \forall T \in \mathcal{T}, \quad x_e \geq 0 \quad \forall e \in E.$$

In the above $\mathcal{T}$ is the set of spanning trees of G. Suppose we are given a feasible solution x' to the LP. Describe a near-linear time algorithm that outputs a partition P of the vertex set such that

$\frac{c(E_P)}{|P|-1} \leq \sum_e c_e x_e$. In lecture we saw proof that relies on x' being an optimum solution, and used complementary slackness. Here we want a fast algorithm that converts *any* fractional solution to a good partition with respect to the value of the solution. *Hint:* Try running Kruskal's algorithm with x' and consider partitions along the way.

Problem 4 Let $G = (V, E)$ be a directed graph and let $c : E \to \mathbb{Z}_+$ be non-negative edge capacities. Let r be a root vertex and we can assume that it has no in-coming arcs. A spanning arborescence rooted at r is a directed out-tree with root r that contains an r-v path for each vertex $v \in V$. Let $\lambda(r, v)$ be the minimum r-v cut value in G. A famous theorem of Edmonds shows that G has k arc-disjoint spanning arborescences rooted at r iff $\lambda(r, v) \geq k$ for all $v \in V - \{r\}$. The integer packing version implies the fractional packing version. That is, the value of a maximum fractional packing of spanning-arborescences in G rooted at r (subject to edge capacities given by c) is equal to $\min_v \lambda(r, v)$. Prove the fractional packing theorem via the primal-dual method. Write down an LP for the maximum fractional packing problem and use complementary slackness like we did in the proof of the fractional packing version of the Tutte-Nash-Williams tree packing theorem.

Problem 5 Let $G = (V, E)$ be a directed graph with edge lengths $\ell : E \to \mathbb{Z}$ (which could be negative). Suppose we are promised that each vertex v has a shortest path from s with at most k negative length edges. Show how to compute all $d(s, v)$ distances in $O(mk \log n)$ time.

Problem 6 Let $G = (V, E)$ be a directed graph with edge capacities $c : E \to \mathbb{Q}_+$, and let h be an integer between 1 and $|V|$. Typically we write the s-t maximum flow via edge-variables but there is an advantage to using an exponential number of path variables. This is particularly useful when one wants to restrict flow to paths of length at most h. Let $\mathcal{P}_h$ be the set of all s-t paths which contain at most h edges.

- Write an exponential sized LP that maximizes flow along paths only restricted to paths from $\mathcal{P}_h$. Write this as a packing LP with variables x_p for each path $p \in \mathcal{P}_h$.
- Write its dual.
- Prove that there is a polynomial-time separation oracle for the dual.