Note: Your TA probably will not cover all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. They are deliberately made long so they can serve as a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

1 Dijkstra’s Algorithm Fails on Negative Edges

Draw a graph with five vertices or fewer, and indicate the source where Dijkstra’s algorithm will be started from.

(a) Draw a graph with no negative cycles for which Dijkstra’s algorithm produces the wrong answer.

(b) Draw a graph with at least two negative weight edge for which Dijkstra’s algorithm produces the correct answer.

2 Waypoint

You are given a strongly connected directed graph $G = (V, E)$ with positive edge weights, and there is a special node $v_0 \in V$. Give an efficient algorithm that computes the length of the shortest path from $s$ to $t$ that passes through $v_0$ for all pairs $s, t$. Your algorithm should take $O(|V|^2 + |E| \log |V|)$ time.

3 Dijkstra Tiebreaking

We are given a directed graph $G$ with positive weights on its edges. We wish to find a shortest path from $s$ to $t$, and, among all shortest paths, we want the one in which the longest edge is as short as possible. How would you modify Dijkstra’s algorithm to this end? Just a description of your modification is needed.

(If there are multiple shortest paths where the longest edge is as short as possible, outputting any of them is fine).
4 Updating a MST

You are given a graph $G = (V, E)$ with positive edge weights, and a minimum spanning tree $T = (V, E')$ with respect to these weights; you may assume $G$ and $T$ are given as adjacency lists. Now suppose the weight of a particular edge $e \in E$ is modified from $w(e)$ to a new value $\hat{w}(e)$. You wish to quickly update the minimum spanning tree $T$ to reflect this change, without recomputing the entire tree from scratch.

There are four cases. In each, give a description of an algorithm for updating $T$, a proof of correctness, and a runtime analysis for the algorithm. Note that for some of the cases these may be quite brief. For simplicity, you may assume that no two edges have the same weight using both $w$ and $\hat{w}$.

(a) $e \notin E'$ and $\hat{w}(e) > w(e)$
(b) $e \notin E'$ and $\hat{w}(e) < w(e)$
(c) $e \in E'$ and $\hat{w}(e) < w(e)$
(d) $e \in E'$ and $\hat{w}(e) > w(e)$

5 MST practice

Let $G = (V, E)$ be an undirected, connected graph.

(a) Prove that there is a unique MST if all edge weights are distinct.

(b) True or False? If $G$ has more than $|V| - 1$ edges, and there is a unique heaviest edge, then this edge cannot be part of a MST.

(c) True or False? If the lightest edge in $G$ is unique, then it must be a part of every MST.