This homework has a total of 4 problems on 2 pages. Solutions should be submitted to T-square before 12:05pm on Friday Sep 23.

The problem set is marked out of 15, you can earn up to $16 = 1 + 4 + 7 + 4$ points.

If you choose not to submit a typed write-up, please write neat and legibly. If the electronic version appears unclear, please also submit the originals, stapled, in addition to the T-square submission.

Collaboration is allowed/encouraged on problems, however each student must independently complete their own write-up, and list all collaborators. No credit will be given to solutions obtained verbatim from the Internet or other sources.

0. (1 point, only if all parts are completed)

(a) The answer to each question except question 0 should start on a new page.

(b) Please put your name and GT-ID on each of the answer sheets.

(c) Submit either a scan or electronic version of your homework in a single pdf file on T-square.

GRADING: 0 if any of these requirements is not met.

1. (4 points in 2 parts) Wile E. Coyote and The Road Runner are at it again, except in this problem, there is no road or desert or ‘tunnels’. Both of the Coyote and Road Runner can only travel between vertices of a network by running through one of $n$ special one-way pipes labeled 1...m.

These pipes are special because they affect The Road Runner’s size. Each pipe has two different values $P_i$ and $Q_i$, which are the multipliers of the roadrunner’s and coyote’s heights as they run through it. For example, if the Road Runner and Coyote both enter a pipe $k$ with $P_k = 0.6$ and $Q_k = 1.2$, they will emerge with heights 0.6 and 1.2 respectively: the Coyote is now twice as big as the Road-Runner.

Given this network, decide whether it’s possible for the Roadrunner to lead Wile E. Coyote through a sequence of pipes (possibly repeating) such that it gets infinitely larger than the Coyote. You may assume all arithmetic are exact.

Relevant YouTube video: https://www.youtube.com/watch?v=KJJW7EF5aVk.

(a) (2 points) Show that if we assign

$$l_i = \log(P_i) - \log(Q_i)$$

1
for the edges (here log is with base 2), and take a path

\[ e_1, e_2 \ldots e_k, \]

then we have

\[ \frac{\text{FinalRoadRunnerSize}}{\text{FinalCoyoteSize}} = 2^{\sum_{i=1}^{k} l_{e_i}}. \]

(b) (2 points) Reduce this problem to checking whether a graph has a negative cycle. Make sure to specify precisely the edges and their directions / lengths.

2. (7 points in 4 parts) Consider a minimum spanning tree \( T \) for an undirected, weighted graph \( G \). We want to see what happens to the minimum spanning tree when a new edge \( e \) is added to \( G \). You may assume that both endpoints of \( e \) are already in \( G \). As in class, we will assume that the edges have distinct weights for simplicity.

In class we showed the correctness of Kruskal’s algorithm:

\[
\begin{align*}
&a) \text{ Initialize } H = \emptyset. \\
&b) \text{ Sort edges in increasing order of weights.} \\
&c) \text{ Loop through edges in increasing order of weights.} \\
&\quad i. \text{ If endpoints of } e \text{ are not connected in } H. \\
&\quad \quad A. \text{ Add } e \text{ to } H. \\
&d) \text{ Return } H \text{ as the minimum spanning tree.}
\end{align*}
\]

(a) (2 points) Show that if \( e = uv \) is not chosen by Kruskal’s algorithm, there exists a cycle on which \( e \) is the maximum weight edge.

(b) (2 points) Show that if \( e = uv \) is chosen by Kruskal’s algorithm, there does not exists a cycle on which \( e \) is the maximum weight edge.

\textbf{NOTE:} Parts (a) and (b) give the \textbf{cycle property}, which states that an edge \( e \) is in the MST iff there does not exist a cycle containing \( e \) where \( e \) is the maximum cost edge.

(c) (2 points) Use the cycle property to show that if \( T \) is a MST of \( G \), a MST of \( T \cup \{e\} \) is also a MST of \( G \cup \{e\} \).

(d) (1 point) Describe an \( O(n) \) time algorithm to find a minimum spanning tree of \( G \cup \{e\} \) when given a minimum spanning tree \( T \) of \( G \).

\textbf{NOTE:} this shows that a MST can be maintained under a sequence of \( m \) edge insertions in \( O(nm) \) total time, better than the \( O(m^2 \log n) \) from rerunning MST at each step.
3. (4 points in 4 parts) Shortest odd-walk. We want to get from $s$ to $t$ in some directed graph with positive edge lengths, but use an odd number of edges.

(a) (2 points) Show how to construct another graph $G'$ such that the length of the shortest $s$ to $t$ odd-walk in $G$ equals to the length of the shortest $s'$ to $t'$ walk in $G'$. Your $G'$ should have at most $2n$ vertices and $2m$ edges, which is the size in our solution.

Hint: split each vertex into two, add labels indicating the parity of the edge count up to that point, and add edges appropriately.

(b) (1 point) Give a proof of correctness for your answer to part (a).

**HINT:** use induction on number of edges on the path.

(c) (1 point) Generalize your graph construction to show that that the shortest walk whose edge count is $1 \mod 3$ ($x \equiv 1 \mod 3$ if $x = 3y + 1$ for some integer $y$) can be computed by finding a shortest path on a graph with at most $3n$ vertices and $3m$ edges. **You only need to give the graph construction, and do not need to prove correctness.**