CS 561: Homework 11

Prof. Jared Saia, University of New Mexico

Remember: You are encouraged to work on the homework in groups, but please observe the "Star Trek" rule from the syllabus.

1. **Trees and Induction.** Prove by induction that any tree with n nodes has exactly $n-1$ edges. State what quantity you are doing induction over, and explicitly state the base case (BC), inductive hypothesis (IH), and inductive step (IS).

For Problems 2 and 3, use the following setup from the single-source shortest-path lecture. Let $G = (V, E)$ be a directed, weighted graph with source s, and assume no negative-weight cycle is reachable from s. Initialize $dist(s) = 0$, $dist(v) = \infty$ for $v \neq s$, and $pred(v) = \text{NONE}$ for every vertex. Repeatedly choose a *tense* edge (u, v), meaning $dist(u) + w(u, v) < dist(v)$, and perform

$$dist(v) \leftarrow dist(u) + w(u, v), \qquad pred(v) \leftarrow u.$$

The algorithm halts when no tense edge remains. As usual, $\infty + a = \infty$ for every finite a.

2. **Predecessor Paths (Claim 1).** Prove that after any number of successful relaxations, every vertex v with finite $dist(v)$ has a predecessor chain that reaches s. In forward order, this chain is a directed path P from s to v satisfying

   $$dist(v) \geq w(P),$$

   where $w(P)$ is the sum of the edge weights on P. Use induction on the number of successful relaxations, and explicitly justify why the predecessor pointers cannot form a cycle.

3. **Distances at Termination (Claim 2).** Suppose the algorithm halts. Prove that for every directed path P from s to a vertex v,

   $$dist(v) \leq w(P).$$

   Use induction on the number of edges in P, stating the base case, inductive hypothesis, and inductive step. Combine this result with Problem 2 to conclude that the final distances are correct.

4. **Currency Arbitrage.** There are n currencies. For every ordered pair of distinct currencies i, j, an exchange rate $r_{ij} > 0$ specifies how many units of currency j you receive for one unit of currency i. Rates are fixed, and there are no transaction fees.

   An *arbitrage opportunity* is a sequence of exchanges that begins and ends with the same currency and turns one unit into strictly more than one unit. Design an $O(n^3)$-time algorithm that determines whether an arbitrage opportunity exists anywhere in the exchange network and, if so, returns such a sequence. Prove correctness and justify the running time. You may treat arithmetic, comparisons, and logarithms as exact constant-time operations.

5. **Safest Route.** Saia Trucking has a directed graph $G = (V, E)$ whose vertices are cities and whose edges are roads. Each edge e has a safety probability $p(e)$ with $0 < p(e) \leq 1$. The safety of a route is defined to be the product of its edge probabilities.

   Given distinct cities s and t, design an algorithm that finds a safest route from s to t, or reports that no route exists, in $O((n + m) \log n)$ time, where $n = |V|$ and $m = |E|$. Describe the algorithm, prove correctness, and explain how to recover the route.