Honors Algorithms — G22.3520-010 Fall 2006 — Problem Set 5
Due: Nov. 22

1. **Sightseeing.** You are a tourist in a city, and you want to take in as many sights as possible on a single tour. Let's model this as a directed graph $G = (V, E)$, where the nodes in the graph represent the sights, and the edges represent roads between them. For a path p in the graph, define the *sightseeing value* of p to be the number of *distinct* nodes in p. For a node $v \in V$, define the *sightseeing value* of v to be the maximum sightseeing value of any path starting at v. Design and analyze an algorithm to find a node of maximum sightseeing value. Your algorithm should run in time $O(|V| + |E|)$.

2. **Colorful paths (I).** You are given a directed graph $G = (V, E)$, and nodes s, t. Nodes are colored *red*, *yellow*, *blue*, or *white*. Let us say a path (not necessarily simple) is *colorful* if it runs from s to t, and contains at least one red, one yellow, and one blue node. Design and analyze an algorithm that determines if there is a colorful path, and if so, finds one containing a minimum number of white nodes. Your algorithm should run in time $O(|V| + |E|)$.

3. **Colorful paths (II).** You are given a directed graph $G = (V, E)$, and nodes s, t. Nodes are colored *red*, *yellow*, *blue*, or *white*. A path (not necessarily simple) is called k-colorful if it runs from s to t and contains at least k red nodes, k yellow nodes, and k blue nodes (counting any repetitions of nodes). G is called *dazzling* if there is a k-colorful path for every $k > 0$. Design and analyze an algorithm to determine if G is dazzling. Your algorithm should run in time $O(|V| + |E|)$.

4. **Running on empty (I).** In this problem, we want to determine how to drive from point s to point t, using the least amount of gas. Our car has a gas tank that is initially filled up to its capacity c. We may have to stop to refuel along the way: we can never allow the amount of gas in the tank to become negative, and we can never fill our tank beyond its capacity. Let us model this problem using a directed graph $G = (V, E)$ with non-negative edge weights $w : E \to \mathbb{R}_{\geq 0}$, along with nodes $s, t \in V$. For an edge $(u, v) \in E$, $w(u, v)$ represents the amount of gas required to drive from u to v. Also, a subset $F \subset V$ represents those points where we may stop to refuel (assume each node v is marked with a bit $f(v)$ indicating whether there is a gas station located at v). Show how to solve this problem in time $O(|F|(|V| \log |V| + |E|))$.

5. **Running on empty (II).** This problem considers a variant of the previous one, in which gas stations sell gas at different prices, and the goal is to minimize the amount of money spent buying gas along the way from s to t, assuming we start out with a full tank of gas at s (free of charge). To simplify the problem, assume that our gas tank holds c units of gas, where c is a positive integer; also, assume that edge weights are non-negative integers, and that gas stations only sell gas in integral units. For each node u, $p(u)$ denotes the price per unit of gas sold at u ($p(u) \in \mathbb{R}_{>0} \cup \{\infty\}$, where $p(u) = \infty$ indicates no gas is available at u). Thus, if we arrive at u with m units of gas in our tank, and $p(u) < \infty$, we can buy k units of gas at u, where $k \in \{0, \ldots, c - m\}$, at a cost of $k \cdot p(u)$. Show how to solve this problem in a reasonable amount of time, assuming c is not too large. In particular, your algorithm should run in time $O((|V| + c)^\alpha)$ for some constant α.

6. **Invasion.** Two armies simultaneously invade a country. Let's call them the "red army," and the "blue army." The red army starts out occupying city a, and the blue army starts out occupying city b. Both armies fan out simultaneously in all directions, and whichever army arrives at a city first, occupies that city, and blocks the other army from either occupying or transiting through that city. The occupying army leaves a small occupation force at that city, but the remainder of the army continues to fan out to all neighboring cities. *In case of a tie, the city is occupied by neither army, and neither army may transit the city.* The army that occupies the most cities wins the war. The question is: which army wins? Let's model this problem as a directed graph with *positive* edge weights. The nodes in the graph represent the cities, and the edges represent roads between cities. The weight of an edge (u, v) represents the amount of time required for either army to travel from city u to city v. Design and analyze an efficient algorithm to solve this problem. The input is a directed, weighted graph, along with distinct nodes a and b. The output is "red wins," "blue wins," or "tie."

[See next page]

7. **Min-max graphs.** Consider a directed graph $G = (V, E)$. Each node is labeled as either a *min* node, a *max* node, or a *constant* node. A constant node u has no outgoing edges, and is also labeled with $c(u) \in \mathbb{R}$. A *value assignment* for G is a function $\alpha : V \to \mathbb{R} \cup \{-\infty, \infty\}$. For any given value assignment α, we define a new value assignment α^G as follows:

   (i) for all *min* nodes u, $\alpha^G(u) = \min\big(\{\alpha(v) : (u, v) \in E\} \cup \{\infty\}\big)$;

   (ii) for all *max* nodes u, $\alpha^G(u) = \max\big(\{\alpha(v) : (u, v) \in E\} \cup \{-\infty\}\big)$;

   (iii) for all *constant* nodes u, $\alpha^G(u) = c(u)$.

   α is called a *fixed point* of G if $\alpha^G = \alpha$. Observe that for any fixed point, a *min* node with no outgoing edges is assigned ∞, while a *max node* with no outgoing edges is assigned $-\infty$.

   There may be more than one fixed point for G. A fixed point α is called a *least fixed point* if for any other fixed point β, we have $\alpha(u) \leq \beta(u)$ for all $u \in V$. It is clear that a least fixed point, if it exists, must be unique.

   (a) Your first task is to prove that a least fixed point always exists. To this end, let α_0 be the value assignment that assigns $-\infty$ to all nodes. For $i = 0, 1, 2, \ldots$, let $\alpha_{i+1} := \alpha_i^G$. Show that there exists some $k \geq 0$ such that
   $$\alpha_k = \alpha_{k+1} = \alpha_{k+2} = \cdots,$$
   and moreover, for this k, α_k is the least fixed point of G.

   (b) Suppose that G contains N distinct values labeling its constant nodes. Show that the value k in part (a) is at most $|V|(N + 1)$. Based on this, describe a simple algorithm to compute the least fixed point of G in time $O(|E||V|(N + 1))$.

   (c) Suppose that G is acyclic. Show how to compute the least fixed point of G in time $O(|V| + |E|)$.

   (d) Suppose that G contains no constant nodes. In this situation, we can naturally interpret $-\infty$ as *false*, ∞ as *true*, *min* as *and*, and *max* as *or*, and the least fixed point represents the minimal, consistent truth-value assignment. Show how to find the least fixed point of G in this case in time $O(|V| + |E|)$.

   Note: the solution to part (d) may be viewed as a special case of the solution to part (e), but it may be easier to tackle this simpler problem first; alternatively, you can first show how to solve part (e), and then adapt that solution to part (d).

   (e) In the general setting, show how to find the least fixed point of G in time $O(|V| \log |V| + |E|)$.