CS 161 W19: Recitation 6 Problems

February 2019

Exercise 0

We have a network of n nodes and m directed links between them. Each of the n nodes send/forward packets to each other along the m links. The links are not reliable and may fail due to a variety of reasons. Each link between node i and node j is available with probability $p_{i,j}$. We assume that each link fails independently, so the probability that a path, P, is available is simply the product of the probabilities that all of the links in the path are available, $\pi(P) = (p_{1,2})(p_{2,3}) \cdots (p_{k-1,k})$.

We can represent this network with a directed graph G where the edge weights are the probabilities. Give an efficient algorithm for finding the path with the greatest probability of successfully transmitting a packet from node i to node j.

Exercise 1

Dijkstra's algorithm will in general fail on graphs with negative-weight edges. In this problem, we will explore an interesting special case with negative edges for which we can still solve the shortest paths problem as quickly as Dijkstra's algorithm.

Recall that any directed graph $G = (V, E)$ can be broken up into its *strongly connected components* (SCCs)—in other words, a partition of V into disjoint vertex sets C_i such that within each C_i, any two vertices have a path to one another.

Suppose that in addition to the graph $G = (V, E)$, we are given an *ordered* set of its SCCs, $V = C_1 \cup C_2 \cup \cdots \cup C_k$, with the promise that:

- Within each component C_i, all edges have nonnegative weight.
- Edges between different components may have negative weight. However, they all obey the following rule: for any edge (u, v) such that $u \in C_i$ and $v \in C_j$ for $i \neq j$, we are promised that $i < j$. In other words, an edge out of one component can only point to a component "to the right".

It is helpful to draw some examples of graphs that obey this structure.

Devise an algorithm that, given G, an ordered set $\{C_1, C_2, \ldots, C_k\}$ of its SCCs obeying the above constraints, and a starting node s, finds the shortest path lengths from s to all other nodes $v \in V$. It should have running time $O(|V| \log |V| + |E|)$.