Homework 7

Aleck Zhao

October 27, 2017

1 Consistent Lifetimes (34 points)

Suppose that you are an historian, and you are trying to figure out possible dates for when various historical figures may have lived. In particular, there are n people $P_1, P_2, \ldots, P_n$ who you are studying. Based on your research, you have discovered a collection of m facts about when these people lived relative to each other. Each of these facts has one of the following two forms:

- For some i and j, person P_i died before person P_j was born; or
- For some i and j, the lifetimes of P_i and P_j overlapped.

Unfortunately, since the historical record is never fully trustworthy, it's possible that some of these facts are incorrect. Design an $O(n+m)$ time algorithm to at least check whether they are *internally consistent*, i.e., whether there is a birth and a death date for each person so that all of the facts are true. As always, prove running time and correctness (i.e., prove that if there is a possible set of dates then your algorithm will return yes, and if no such dates are possible then your algorithm will return no).

2 Minimum Spanning Trees (33 points)

(a) Suppose that we are given a connected graph $G = (V, E)$ where all edge weights are distinct. Prove that there is a *unique* minimum spanning tree.

As in the previous part, suppose that we are given a connected graph $G = (V, E)$ where all edge weights are distinct. But instead of trying to construct an MST, we are trying to construct a something else. Given a spanning tree $T = (V, E')$ of G, the *bottleneck edge* of T is the edge in E' of maximum weight. A spanning tree T is a *minimum-bottleneck spanning tree* if there is no spanning tree T' of G with a lower weight bottleneck edge.

(b) Is every minimum-bottleneck spanning tree of G also a minimum spanning tree of G? Prove or give a counterexample (and explanation).

(c) Is every minimum spanning tree of G a minimum-bottleneck spanning tree of G? Prove or give a counterexample (and explanation).

3 Shortest Paths (33 points)

(a) We saw in class that Dijkstra's algorithm for computing shortest paths in a directed graph does not work if there are negative edge lengths. Consider the following idea to fix this.

> First add a large enough positive constant to every edge weight so that the "revised" edge lengths are all positive (for example, you could add the value $1 + \max_{v \in V} |len(v)|$). To find the shortest path between two nodes s and t in the original graph, run Dijkstra's algorithm using the revised edge lengths to find the shortest path between s and t using the revised lengths, and return the path which it finds.

Does this work? That is, will this always return the shortest path under the original edge lengths? If so, give a proof. If not, give a counterexample (and explain it).

(b) Now suppose that instead of adding the same value to every length, we instead *multiply* every length by the same value $\alpha > 0$ (note that this preserves the sign of each edge length). Let P be a shortest path from s to t under the original edge lengths. Is P still a shortest path from s to t under the new edge lengths? If yes, give a proof. If no, give a counterexample (and explain it).