Homework 11

Dushan Terzikj

due 21 May, 2018, 23:55 hours

Problem 1

Your friend (who hasn't taken this course) asks you for help on implementing an algorithm for finding the shortest path between two nodes u and v in a directed graph (possibly containing negative edge weights). She proposes the following algorithm:

1. Add a large constant to each edge weight such that all weights become positive.
2. Run Dijkstra's algorithm for the shortest path from u to v.

Prove or disprove the correctness of this algorithm to find the shortest path (note that in order to disprove, you only need to give a counterexample).

Solution: This might work in several cases, but it is **not** correct. Consider the following counterexample:

$$a \xrightarrow{-3} b \xrightarrow{5} c \tag{1}$$

$$a \xrightarrow{-3} b \xrightarrow{1} d \xrightarrow{3} c \tag{2}$$

The shortest path would be the (2) and the weight is 1. In order to use Dijkstra with negative edges, let us add 3 (since -3 is the smallest negative edge) to all the edges:

$$a \xrightarrow{0} b \xrightarrow{8} c \tag{3}$$

$$a \xrightarrow{0} b \xrightarrow{4} d \xrightarrow{6} c \tag{4}$$

Now the shortest path would be (1) whose weight is 8. This happens because we add a constant on each edge, therefore the weight of the shortest path depends on how many edges we use. The real result can be done by subtracting the number of edges times the constant, but Dijkstra algorithm does not do that. Optimization for this to work can be done, however the current problem statement does not state any optimization to be done.