Assignment 4: Report

Finding the optimal values and shortest path.
DVA427 - Lärande System.

Joaquín García Benítez.
Clara Torre García-Barredo.
Mälardalen Högskola.
April 2019.

Explanation of the problem.

This problem consists of creating a program that finds the shortest path from every city in a map to a goal city named “F”. As well as finding the shortest path, we also need to find the total distance between the two cities.

The cities are given in a document that has all connections between two cities and the distance between them. The challenge is to use Bellman equations to find the solutions.

Solution of the problem.

In this report we are asked to solve a miniature version of the problem by hand, to show the algorithm behind our program with a smaller map of cities.

In this case, the goal city is “E”, and the map is the following:

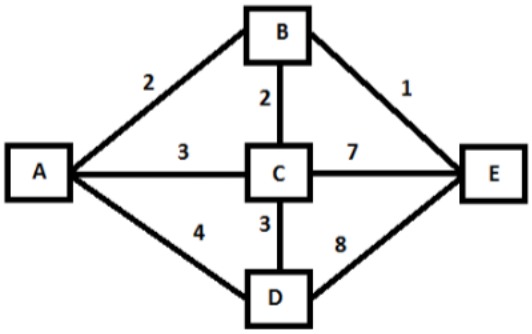

Figure 1: The graph for a small problem

First, we know the distance to the Goal city from the Goal city is obviously 0. We keep that in mind to do the first check.

We check which cities have a direct connection with the Goal city. In this case they are B, C and D. We add them to our Bellman table. A is not directly connected, so for now the distance is ∞, and there is no path.

Path: [], Dist: ∞	Path: [E,B], Dist: 1	Path: [E,C], Dist: 7	Path: [E,D], Dist: 8	Path: [E], Dist: 0

Now we select the one with the smallest distance (apart from E, which we selected in the first place).

In this case, the one with the smallest distance is B. We repeat the process. The cities in direct connection with B are E, C and A. E and C have already a path to reach E, but A doesn’t. So, with E and C, we check if there is an improvement of the distance when passing trough B. For A, we add the new path.

Path: [E,B,A], Dist: 3	Path: [E,B], Dist: 1	Path: [E,B,C], Dist: 3	Path: [E,D], Dist: 8	Path: [E], Dist: 0

Now we can choose between city A or city C as the ones with the smallest distance that have not been visited already. We select the first one, A. And we repeat the process.

The cities in direct connection with A are B, C and D. All of them are already in our table, so we check the improvement. D improves, so we change its path and distance for the new ones.

Path: [E,B,A], Dist: 3	Path: [E,B], Dist: 1	Path: [E,B,C], Dist: 3	Path: [E,B,A,D], Dist: 7	Path: [E], Dist: 0

So now we return and visit C. C has direct connection to all of the cities in the map, so we check if any of them improves. In this case, D improves, so we change its path and distance.

Path: [E,B,A], Dist: 3	Path: [E,B], Dist: 1	Path: [E,B,C], Dist: 3	Path: [E,B,C,D], Dist: 6	Path: [E], Dist: 0

And, since D is the last city to visit, we visit it and finish the process.

The solutions would be:

- Path from A to E: A→B→E. Distance: 3.
- Path from B to E: B→E. Distance: 1.
- Path from C to E: C→B→E. Distance: 3.
- Path from D to E: D→C→B→E. Distance: 6.
- Path from E to E: E. Distance: 0.