CSCI 3104 Spring 2022
Instructors: Profs. Chen and Layer

Quiz 23 - DP: Bellman-Ford algorithm

Due Date .... April 8
Name .... **Julia Troni**
Student ID **109280095**

Contents

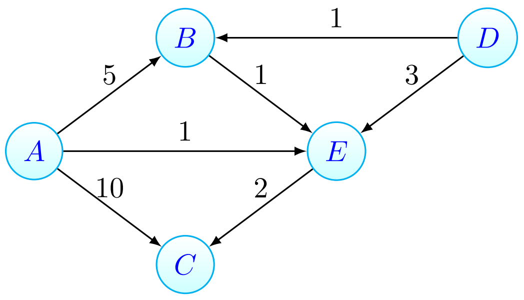
1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You **may not collaborate with other students**. **Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Discord, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.

2 Standard 23 - DP: Bellman-Ford algorithm

Problem 1. Consider the weighted directed graph $G(V, E, w)$ pictured below. Work through the Bellman-Ford algorithm with the destination vertex C.

- Clearly specify the cost $d(v)$ of the current best path from each node $v \in V$ to C as well as the corresponding successor node at each iteration/pass. You may want to make a table to store the costs and successors.
- Give the shortest path tree, i.e., the union of all the shortest paths to C from all other vertices. For your convenience, you may want to modify the "latex code" for the given graph to draw the shortest path tree.



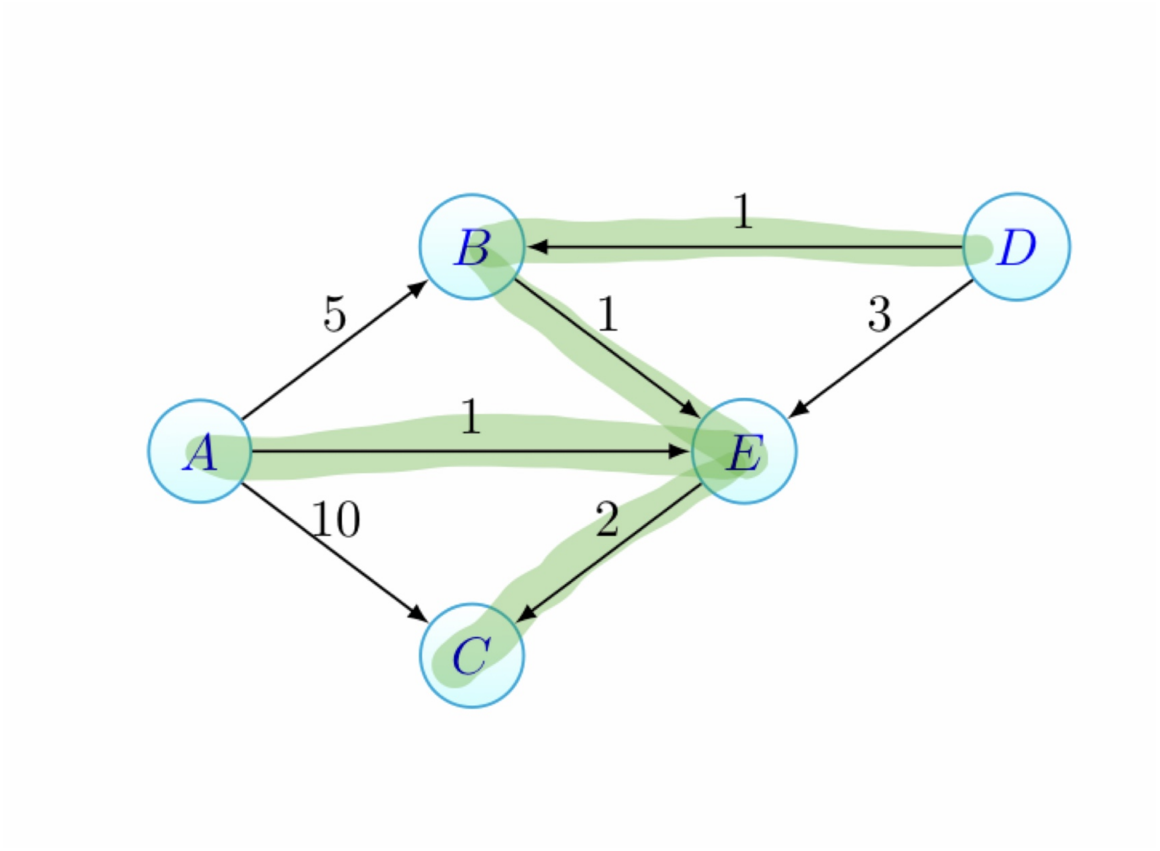
Answer. Apply Bellman-Ford to the graph:
Let $c(u, v)$ denote cost of the edge from vertex u to vertex v. Let $D(v)$ denote the cost of the current path from vertex v to the destination vertex C.

1. 1st iteration: begin by examining source vertex C incoming edges.
   - Poll edge (A,C): $D(A) = \infty > D(C) + c(A, C) = 0 + 10$ so set $D(A) = 10$ and $successor[A] = C$
   - Poll edge (E,C): $D(E) = \infty > D(C) + c(E, C) = 0 + 2$ so set $D(E) = 2$ and $successor[E] = C$
2. 2nd iteration:examine the incoming edges for each of the updated vertices, which were A, C
   - Poll edge (A,E): $D(A) = 10 > D(E) + c(A, E) = 2 + 1 = 3$ so set $D(A) = 2$ and $successor[A] = E$
   - Poll edge (D,E): $D(D) = \infty < D(E) + c(D, E) = 2 + 3$ so set $D(D) = 5$ and $successor[D] = E$
   - Poll edge (B,E): $D(B) = \infty > D(E) + c(D, E) = 2 + 1 = 3$ so set $D(B) = 3$ and $successor[B] = E$
3. 3rd iteration:examine the incoming edges for each of the updated vertices, which were A,D,B
   - Poll edge (A,B): $D(A) = 3 < D(B) + c(A, B) = 3 + 5 = 8$ so no change
   - Poll edge (D,B): $D(D) = 5 > D(B) + c(D, B) = 3 + 1 = 4$ so set $D(D) = 4$ and $successor[D] = B$
4. 4th iteration:no updates made

At the end of the algorithm the paths are the following

- A-E-C with cost of 3
- B-E-C with cost 3
- E-C with cost 2
- D-B-E-C with cost 4

See the shortest path tree below



□