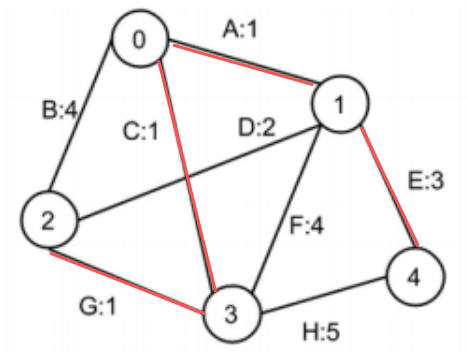
CS 61B Spring 2018

Shortest Paths and MST

Exam Prep 12: April 9, 2018

1 Warmup with MST and SP

(a) For the graph below, use Kruskals algorithm to find the MST. The number on each edge is the weight, and the letter is a unique label you should use in your answer to specify that edge. **Provide the edges in the order theyd be inserted into the MST by Kruskals algorithm**. Break any ties using the alphabetical label. Use the blanks below. You may not need all blanks. **Give your answer in terms of the alphabetical labels**, e.g. if Kruskals starts with the edge between vertices 3 and 4, you would write H in the first blank.



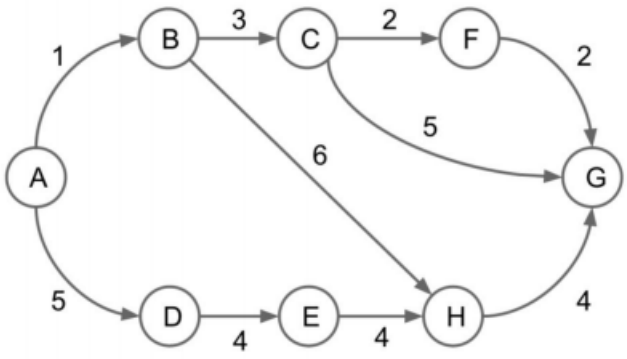
A 1 C 1 G 1 D 2 E 3 B4 F4 H5.
✓ ✓ ✓ ✗

A C G E ___ ___

(b) Repeat part a, but using Prims algorithm, starting from vertex #3. **As before, give your answer in the order added to the MST and in terms of the alphabetical labels**. You may not need all of the blanks.

C A G E ___ ___

(c) For the graph below, give the order in which Dijkstras Algorithm would visit each vertex, starting from vertex A.



A B C D F H G E

① (AB,1) (AD,5) A
② (BC,4), (AD,5), (BH,7) B.
③ (AD,5), (CF,6), (BH,7), (CG,9) C.
④ (CF,6), (BH,7), (CG,9), (DE,9) D
⑤ (BH,7), (FG,8) (CG,9), (DE,9) F
⑥ (HG,8), (CG,9) (DE,9), (HG,11) H.
⑦ (DE,9), (HG,11) G.
⑧ (HG,11) E.
⑨

2 Conceptual MST and SP

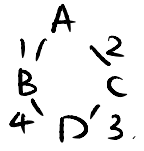
Answer the following questions regarding MSTs and shortest path algorithms for a **weighted, undirected graph**. If the question is T/F and the statement is true, provide an explanation. If the statement is false, provide a counterexample.

(a) (T/F) If all edge weights are equal and positive, breadth-first search starting from node A will return the shortest path from a node A to a target node B.

T, in the case, the graph is the same as unweighted

(b) (T/F) If all edges have distinct weights, the shortest path between any two vertices is unique.

F

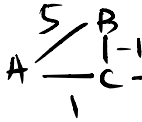


(c) (T/F) Adding a constant positive integer k to all edge weights will not affect any shortest path between vertices.

F

(d) Draw a weighted graph (could be directed or undirected) where Dijkstra's would incorrectly give the shortest paths from some vertex.

F



Dijkstra's algorithm never looks back.
⇒ the bellman example indicates that Dijkstra's algorithm fails to update the distance of D as it has been already visited

(e) (T/F) Adding a constant positive integer k to all edge weights will not affect any MST of the graph.

T Since all MST has exactly the same number of edges for a given graph, adding a positive constant integer k will only increase the total weight by (V-1)·k.

(f) (T/F) Prim's algorithm works even when there are negative edge weights in the graph.

T Since you can always make the graph positive weighted by adding a suitable positive constant integer k.

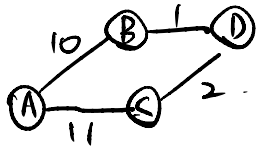
(g) Why are disjoint sets used in Kruskal's algorithm?

To make sure there is no circles

(h) (T/F) The last edge added to the MST by Prim's algorithm is always the highest weight edge of the MST.

if we start from A. we add 2 last.

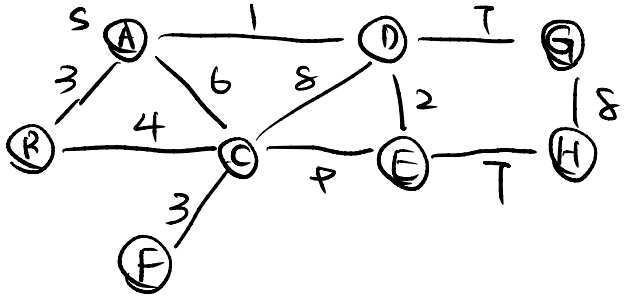
F



3 Shortest Path Algorithm Design

Design an efficient algorithm for the following problem: Given a weighted, undirected, and connected graph G where the weights of every edge in G are all integers between 1 and 10, and a starting vertex s in G, find the distance from s to every other vertex in the graph (where the distance between two vertices is defined as the weight of the shortest path connecting them).

Your algorithm must run asymptotically faster than Dijkstra's.
Hint: What other shortest path algorithms have we learned? Could we possibly modify the graph to apply other SP algorithms?



For exp.: A & B, we make it A—O—O—B

virtual vertices

now we can re-use the BFS method without any bazziers.

The second bucket method can be found in Hug's solution.