

Algorithms  
Homework 7, due Wednesday, Nov 17, at 11:59 p.m.

Reading: CLRS 24.3, 24.5, 23.1, 23.2.

**“HOMEWORK” exercises: submit on Gradescope by Wednesday, Nov 17, at 11:59 p.m.**

1. Suppose that, in the single-source shortest path problem in weighted graphs, we wish to find not just any shortest (min-weight) path between a source vertex  $s$  and a vertex  $v$ , but among those, the shortest (min-weight) path that has the fewest edges. Given a directed, edge-weighted graph  $G = (V, E)$  with integer edge weights  $w(e) > 0$  and a source vertex  $s \in V$ , give an  $O((V + E) \lg V)$  time algorithm to find the shortest (min-weight) path from  $s$  to  $v$  with the fewest number of edges, for all  $v \in V$ . Assume that  $G$  is in adjacency list format.
  - (a) (2 points) Briefly describe the basic idea of your algorithm.

- (b) (5 points) Describe your algorithm in pseudocode. Comment your code.
  - (c) (2 points) Argue why your algorithm returns the correct answer.
  - (d) (1 point) Analyze the running time of your algorithm with reference to your pseudocode.
2. Given a directed graph  $G = (V, E)$  with non-negative edge weights  $w(e) \geq 0$  and two vertices  $s$  and  $t$ , find the shortest path from  $s$  to  $t$  that has an even number of edges. Such a path need not be a simple path, i.e. it may repeat vertices or edges.
- (a) (5 points) Modify Dijkstra's algorithm to solve the problem. Give the complete pseudocode for your modified algorithm. Analyze the time taken by your algorithm. Argue why your algorithm is correct.
  - (b) (5 points) Modify  $G = (V, E)$  into  $G' = (V', E')$ , such that executing the original Dijkstra's algorithm on  $G'$  solves the problem. Explain clearly, using an example, how you would create  $G'$ , and how you would obtain the required distances. Analyze the time taken by your solution. Argue why your algorithm is correct.
3. The following statements may or may not be correct. In each case, either prove it (if it is correct) or give a counterexample (if it isn't correct). Always assume that the graph  $G = (V, E)$  is undirected and connected. Do not assume that edge weights are distinct unless this is specifically stated.
- (a) (1 point) If graph  $G$  has more than  $|V| - 1$  edges, and there is a unique heaviest edge, then this edge cannot be part of a minimum spanning tree.
  - (b) (2 points) If  $G$  has a cycle with a unique heaviest edge  $e$ , then  $e$  cannot be part of any MST.
  - (c) (1 point) Let  $e$  be any edge of minimum weight in  $G$ . Then  $e$  must be part of some MST.
  - (d) (2 points) If the lightest edge in a graph is unique, then it must be part of every MST.
  - (e) (2 points) If  $e$  is part of some MST of  $G$ , then it must be a lightest edge across some cut of  $G$ .
  - (f) (2 points) Prim's algorithm works correctly when there are negative edges.
4. CLRS 23-1, page 638 as:
- Part (a): Prove that the MST (in this case) is unique, and give a counter example to show that the second-best MST need not be unique. (2 points)
  - Part (b) (2 points)
  - Part (c) Give pseudocode for the algorithm. (3 points)
  - Part (d). Give pseudocode for the algorithm. (3 points)