✖

Problem Set #2

Back to Week 2

5/5 points earned (100%)

Quiz passed!

---

1 / 1 points

1.
Suppose we are given a *directed graph* $G = (V, E)$ in which every edge has a distinct positive edge weight. A directed graph is *acyclic* if it has no directed cycle. Suppose that we want to compute the maximum-weight acyclic subgraph of G (where the weight of a subgraph is the sum of its edges' weights). Assume that G is weakly connected, meaning that there is no cut with no edges crossing it in either direction.

Here is an analog of Prim's algorithm for directed graphs. Start from an arbitrary vertex s, initialize $S = \{s\}$ and $F = \emptyset$. While $S \neq V$, find the maximum-weight edge (u, v) with one endpoint in S and one endpoint in $V - S$. Add this edge to F, and add the appropriate endpoint to S.

Here is an analog of Kruskal's algorithm. Sort the edges from highest to lowest weight. Initialize $F = \emptyset$. Scan through the edges; at each iteration, add the current edge i to F if and only if it does not create a directed cycle.

Which of the following is true?

- ○ Only the modification of Kruskal's algorithm always computes a maximum-weight acyclic subgraph.
- ● Both algorithms might fail to compute a maximum-weight acyclic subgraph.

Correct Response
Indeed. Any ideas for a correct algorithm?

- ○ Both algorithms always compute a maximum-weight acyclic subgraph.

Only the modification of Prim's algorithm always computes a maximum-weight acyclic subgraph.

---

1 / 1 points

2.
Consider a connected undirected graph G with edge costs that are *not necessarily distinct*. Suppose we replace each edge cost c_e by $-c_e$; call this new graph G'. Consider running either Kruskal's or Prim's minimum spanning tree algorithm on G', with ties between edge costs broken arbitrarily, and possibly differently, in each algorithm. Which of the following is true?

- (selected) Both algorithms compute a maximum-cost spanning tree of G, but they might compute different ones.

Correct Response
Different tie-breaking rules generally yield different spanning trees.

- Kruskal's algorithm computes a maximum-cost spanning tree of G but Prim's algorithm might not.
- Both algorithms compute the same maximum-cost spanning tree of G.
- Prim's algorithm computes a maximum-cost spanning tree of G but Kruskal's algorithm might not.

---

1 / 1 points

3.
Consider the following algorithm that attempts to compute a minimum spanning tree of a connected undirected graph G with distinct edge costs. First, sort the edges in decreasing cost order (i.e., the opposite of Kruskal's algorithm). Initialize T to be all edges of G. Scan through the edges (in the sorted order), and remove the current edge from T if and only if it lies on a cycle of T.

Which of the following statements is true?

- The output of the algorithm will always be connected, but it might have cycles.
-

The output of the algorithm will never have a cycle, but it might not be connected.

The algorithm always outputs a minimum spanning tree.

Correct Response

During the iteration in which an edge is removed, it was on a cycle C of T. By the sorted ordering, it must be the maximum-cost edge of C. By an exchange argument, it cannot be a member of any minimum spanning tree. Since every edge deleted by the algorithm belongs to no MST, and its output is a spanning tree (no cycles by construction, connected by the Lonely Cut Corollary), its output must be the (unique) MST.

The algorithm always outputs a spanning tree, but it might not be a minimum cost spanning tree.

---

1 / 1
points

4.

Consider an alphabet with five letters, $\{a, b, c, d, e\}$, and suppose we know the frequencies $f_a = 0.32$, $f_b = 0.25$, $f_c = 0.2$, $f_d = 0.18$, and $f_e = 0.05$. What is the expected number of bits used by Huffman's coding scheme to encode a 1000-letter document?

2400

3450

2230

Correct Response

For example, $a = 00$, $b = 01$, $c = 10$, $d = 110$, $e = 111$.

3000

---

1 / 1
points

5.

Which of the following statements holds for Huffman's coding scheme?

If the most frequent letter has frequency less than 0.5, then all

- If the most frequent letter has frequency less than 0.5, then all letters will be coded with more than one bit.
- If a letter's frequency is at least 0.4, then the letter will certainly be coded with only one bit.
- A letter with frequency at least 0.5 might get encoded with two or more bits.
- If the most frequent letter has frequency less than 0.33, then all letters will be coded with at least two bits.

Correct Response

Such a letter will endure a merge in at least two iterations: the last one (which involves all letters), and at least one previous iteration. In the penultimate iteration, if the letter has not yet endured a merge, at least one of the two other remaining subtrees has cumulative frequency at least $(1 - .33)/2 > .33$, so the letter will get merged in this iteration.