EECS 336: Algorithms	Homework 2 Solutions

1 Myth Boosters

Algorithm:

int $j = 1$

for i from 1 to n

if $w_i = t_j$

$j++$

if $j = m + 1$

return True

else return False

Proof of correctness: We must argue that our algorithm returns "True" if and only if T is a valid secret message in W.

First, assume that our algorithm returns "True." Let x_k be the index in W for which j was increased from k to $k+1$. Note that $x_1 < x_2 < \ldots < x_m$ (you could prove this easily by induction, but it's obvious enough to leave out). Moreover, $w_{x_k} = t_k$ for all k. It follows that if we were to delete all words in W except w_{x_k} for each k, we would get T, and hence our algorithm only returns "True" on instances where T is a valid secret message in W.

To prove the other direction, we will use a "greedy stays ahead" argument. To do this, we will define a "matching" of W to T to be a set of indices $y_1 < y_2 < \ldots < y_m$ such that $w_{y_j} = t_j$ for all j. In particular, T is a valid secret message in W if and only if there exists a matching of W to T. We can now state our "greedy stays ahead" lemma:

Lemma 1. *Assume that T is a valid secret message in W. Let y_j be any matching of W to T, and let x_j for $j = 1, \ldots, m$ be as defined in the "only if" paragraph above. Then $y_j \geq x_j$ for all j.*

Proof. (Induction on j.)
Base Case $j = 0$. Both matchings match the first 0 characters in the same trivial way. (Alternatively, define a dummy 0th word at the beginnings of W and T.)

Inductive Hypothesis Assume $y_k \geq x_k$ for some k.

Inductive Case By the construction of our algorithm, x_{k+1} is the first occurrence of t_{k+1} in W after w_{x_k}. Let y^* be the first occurrence occurs before the first occurrence of t_{k+1} in W after w_{y_k}. By the inductive hypothesis, $x_{k+1} \leq y^*$. But y^* is the smallest index possible for y_{k+1}. It follows that $y_{k+1} \geq y^* \geq x_{k+1}$. $\square$

Now to prove the other direction of our correctness proof. Assume that T is a valid secret message in W. By our lemma, we know that our algorithm will have incremented j from k to $k+1$ at an index which is earlier than y_k for any matching y_j. But since a matching must exist, there must be some index y_m in the matching, which implies that our algorithm increments j from m to $m+1$, which in turn means that the algorithm will return "True."

Runtime: This is an $O(n)$ algorithm - it loops through the n words in W, performing constant work on each one.

2 Snow, Men

The claim is true. We will prove the following two claims:

1. Among all connected subgraphs of G, there is a tree which is optimal.
2. Any MST will be optimal among all trees.

Proof of Claim 1. Take a connected subgraph S of G. Delete edges from S, without disconnecting the graph, until the result is acyclic. This new graph will be a tree, and its maximum-cost edge will have no higher cost than the maximum in S. $\square$

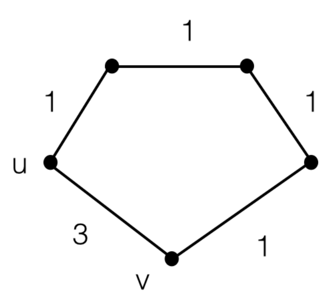
Proof of Claim 2. We argue by contradiction. Assume there is an MST T which is not optimal. By Claim 1, there is another tree T' and an edge $e \in T$ such that $t(e) > t(e')$ for all $e' \in T'$. Consider removing e from T. This creates two disconnected components with vertex sets S and $V \setminus S$. These two sets form a cut. Since T' is connected, it must have at least one edge crossing this cut. By the optimality of T', every edge crossing this cut has lower cost than e. Choose one such edge e'. We know the following things about $T \cup \{e'\} \setminus \{e\}$:

- It is a tree: - it is connected and has $n-1$ edges.
- Its total cost is less than that of T: we have replaced e with a strictly cheaper edge.

It follows that T was not a minimum spanning tree, a contradiction. $\square$

3 MST Trivia

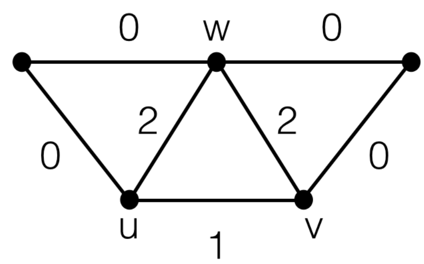
a. This claim is false. Below is a counterexample.



The shortest path from u to v is one edge, of length 3. The MST for this graph is every edge except this one.

b. This is false. A simple counterexample is a two-node graph with one edge. It has a unique heaviest edge, which is also the unique MST.

c. This is false. Below is a counterexample.



The unique lightest edge in the cycle $u - v - w$ is $\{u, v\}$. The minimum spanning tree uses only 0-cost edges, and therefore does not use $\{u, v\}$.

d. This claim is true. Kruskal's algorithm is correct regardless of how ties are broken in sorting. Therefore, if e is one of the cheapest edges in the graph, sorting the edges to put e first will cause Kruskal's algorithm to include e in its output, and therefore in a MST.

e. This claim is true. It is the cycle property. Its proof can be found on pages 147-148 in the text. Though the proof assumes that all edges are distinct, it does not use this assumption.