

## Tutorial Exercise 2: MST and Dynamic Programming

1. **Jot’s Non-deterministic MST Algorithm.** Professor Jot has proposed the following algorithm to solve for the Minimum Spanning Tree of an undirected, connected, weighted graph  $G = (V, E, w)$ :

```
[T] = MinSpanningTree(V, E, w)
% Return the set of edges T in a minimum spanning tree of the
% input graph G = (V, E, w) with edge weights w(e).
% Precondition: G is a connected undirected graph.

Initialize T to be the edges of some spanning tree of G (say by using
depth first search from a starting vertex v ∈ V, details omitted).
That is, suppose (V, T) has been initialized to be a spanning tree of G.

Loop:
  Pick any edge e = (u, v) ∈ E \ T such that the simple path
  from u to v in T contains an edge f = (x, y) such that w(f) > w(e).
  If there is no such edge e in E \ T:
    Break out of this loop.
  Else:
    Set T = (T \ {f}) ∪ {e}
  End if
End loop
return T
```

Note that the Prof. Jot’s algorithm doesn’t specify which set of edges should be used for the initial spanning tree  $(V, T)$ . He is claiming any set will do, so long as the graph  $(V, T)$  is a tree. Moreover, the algorithm doesn’t specify exactly how to select the edge  $e$  to be used at each iteration, nor which of several possible edges  $f$  to use. Multiple choices are possible, and Jot is simply assuming that some such choice is made. This is an example of a non-deterministic algorithm. If you don’t want to think about non-determinism, then whenever the algorithm is faced with a choice, imagine randomly shuffling over the set of possible choices and then selecting whatever choice comes first after shuffling. (Although, note that you are potentially shuffling over a huge number of options.)

- Assume that the initialization is correct, so the original subgraph  $(V, T)$  of  $G$  is indeed a spanning tree. Show that, at the end of each pass through the loop, the updated graph  $(V, T')$  is also a spanning tree, where  $T'$  is the updated set of edges  $T' = (T \setminus \{f\}) \cup \{e\}$ . (**Spoiler-level Hint:** See the proof of Prim’s algorithm at <http://www.cs.toronto.edu/~jepson/csc373/lectures/Prim.txt>.)
- Suppose the “else” clause is executed, and let  $T' = (T \setminus \{f\}) \cup \{e\}$ . That is, let  $T$  be the set of edges in the previous tree, and  $T'$  be the set for the updated tree. Define  $w(T) = \sum_{e \in T} w(e)$  to be the total weight of the edges in tree  $T$ . Then show  $w(T') < w(T)$ .
- Given (a) and (b) above, show the algorithm terminates after a finite number of iterations through the loop. (Hint: **Terminates**, but perhaps not with the correct solution.)
- For the case for which the input graph  $G = (V, E, w)$  has edges with distinct weights, prove the algorithm is correct. That is, assuming  $w(e) \neq w(f)$  for all edges  $e, f \in E$  with  $e \neq f$  then, when this algorithm terminates it outputs the edges  $T$  of the MST  $(V, T)$ .
- [**OPTIONAL**] Is the algorithm correct in the general case of weights, that is, including the cases for which the weights  $w(e)$  need not be distinct? If so, how would you prove it?

2. Consider question 4 from a 2006 final examination:

[http://www.cs.toronto.edu/~jepson/csc373/tutNotes/Q4\\_dynProg\\_csc373h-d06.pdf](http://www.cs.toronto.edu/~jepson/csc373/tutNotes/Q4_dynProg_csc373h-d06.pdf).

Note there are some typos in that question. You should replace all instances of the word “week” with “month”. That is, you are to choose exactly one project per month. Also assume a different internet project is available for you to choose each month.

- (a) Use dynamic programming to solve for the maximum possible profit at the end of each month. Specifically, construct a  $2 \times (n + 1)$  array  $P$  where  $P(k, j)$  stores maximum possible profit at the end month  $j \geq 0$ , given that you are working in city  $k \in \{0, 1\}$  during month  $j = 1, \dots, n$ . Assume  $k = 0$  denotes Toronto and  $= 1$  Vancouver. Assume travel is done at the beginning of a new month. Since you have to start in Toronto, take  $P(0, 0) = 0$  and  $P(1, 0) = -\infty$ . Explain how the remaining entries in the table  $P$  can be iteratively computed.
- (b) Explain how to compute an optimal travel schedule and job selection given the table  $P$  of optimal profits.
- (c) How would you change (a) above if you were allowed to only do at most one internet project over all  $n$  months of work?