CSCI 270 Lecture 12: Common Proof Errors

We want to prove that Prim's Algorithm is correct. We will call the solution produced by Prim's Algorithm 'US'.

Mistake 1: Inducting on the wrong variable

Base Case: Our algorithm is optimal on a graph with 0 edges. (oops!)

Mistake 2: Assuming something about OPT

Base Case: There is an optimal solution which uses the first 0 edges added by US (vacuously true).

Inductive Hypothesis: Assume there is an optimal solution OPT which uses the first k edges added by US. (this is okay)

Suppose US adds edge e_{k+1} next, but OPT does not include this edge (if it doesn't, your inductive step is done, so this is okay)

It must have added some edge $e_{k+2}, e_{k+3}, ..., e_{n-1}$ from a later step in our algorithm instead. (oops!)

Mistake 3: Assuming OPT is an algorithm, not a solution

Base Case: There is an optimal solution which uses the first 0 edges added by US (vacuously true).

Inductive Hypothesis: Assume there is an optimal solution OPT which uses the first k edges added by US.

Suppose US adds edge e_{k+1} next, but OPT does not include this edge.

OPT made some other choice at its $(k+1)$st step OPT_{k+1}. Replace OPT_{k+1} with e_{k+1}. (oops!)

Mistake 4: Merely proving your algorithm is greedy

Base Case: There is an optimal solution which uses the first 0 edges added by US (vacuously true).

Inductive Hypothesis: Assume there is an optimal solution OPT which uses the first k edges added by US.

Suppose US adds edge e_{k+1} next, but OPT does not include this edge.

Since our algorithm chooses the smallest cost edge, e_{k+1} must cost no more than any other edge in OPT.

Therefore, our algorithm is at least as good as OPT. (oops!)

The Correct Proof

Base Case: There is an optimal solution which uses the first 0 edges added by US (vacuously true).

Inductive Hypothesis: Assume there is an optimal solution OPT which uses the first k edges added by US.

Suppose US adds edge e_{k+1} next, but OPT does not include this edge.

- Let S be the set of nodes US has connected with the first k edges.
- Add e_{k+1} to OPT to produce a cycle C.
- Since e_{k+1} has one endpoint in S and the other endpoint in $V - S$, part of C is in S and the rest is in $V - S$.
- Therefore there is another edge in C, e_f, with one endpoint in S and the other in $V - S$.
- $e_f \notin \{e_1, e_2, ..., e_k\}$, otherwise S would be a different set of nodes.
- Exchange e_f with e_{k+1} in OPT to create OPT'.
- OPT' is valid (that is, OPT' is a spanning tree) because:
  - We still have the same number (n-1) of edges, as we just added and then deleted an edge from the valid spanning tree OPT.
  - We still have a connected graph, as any path that used the removed edge e_f can just use the rest of the cycle C instead.
  - Therefore the graph is acyclic and a tree.
- OPT' is optimal because edge e_{k+1} costs no more than e_f. If e_f cost less, then our algorithm would have chosen it instead. Therefore, OPT' must cost no more than OPT and is therefore optimal.