CS 430 Lecture 18 Activities

Proving a Greedy Choice Property

To prove a Greedy Choice Property for a problem (that local optimal choice leads to global optimal solution) use the following "cut and paste" proof.

1. Assume you have an optimal answer that does not contain the greedy choice you are trying to prove.
2. Show that you can "cut" something out of that optimal answer and "paste" in the greedy choice you are trying to prove. Therefore either the assumed optimal answer already contained the greedy choice, or you can make the assumed optimal contain the greedy choice.
3. Therefore there is always an optimal answer that contains the greedy choice (can be continued like an inductive proof).

For the Activity Selector Problem, . . . , Assume you have an optimal answer that is a subset of meetings with k # of activities. . . . I know I can remove the earliest end meeting from the optimal answer and paste in the greedy choice.

1. Try various "common sense" greedy approaches that divide the problem into a sub-problem(s) and try to come up with counter-examples or prove the greedy choice is correct. As stated before, the earliest end time/lastest start time methods are the only greedy choices that'll work.

Does every Optimization Problem exhibit Optimal Substructure?

Consider the following two problems in which we are given a directed graph $G = (V, E)$ and vertices $u, v \in V$.

1. Un-weighted shortest path: Find a path from u to v consisting of the fewest edges.
2. Un-weighted longest simple[1] path: Find a simple path from u to v consisting of the most edges.

2. Try to prove Optimal Substructure for the above two problems.

   a) Assume there is an optimal answer $u \to v' \to v'' \to \cdots \to v^{k-1} \to v$ of k edges. If w is on the sortest path from u to v, then along the path from u to w must be the shortest path from u to w. Also, along that path from w to v must be the shortest path from w to v.

   b) Along the path from u to v, the path from u to w that you would use to get to v could go through v on its way to w, but that would make it a cycle, so it wouldn't be simple.

3. Why does optimal substructure fail for some optimization problems? The path you're trying to optimize isn't summative in the overall problem.

[1] Doesn't go in a cycle, because the longest path could infinitely cycle through before approaching the destination.