Greedy Algorithms: Proof Practice

Proof Practice

- I will give you a problem, as well as the optimal greedy algorithm to solve the problem.
- You will break up into groups of size 2-3, and write the best proof of correctness you can for the algorithm.
- Each group should email a single solution to me (aaroncot@usc.edu) by 2:20pm. Do **not** include your names in your solution, as I will be sharing them with the rest of the class.
- The goal is to get practice writing proofs, as well as reading and recognizing good and bad proof attempts.

The Problem

A friend in CSCI 270 has decided to open a new pizza chain called **Algorithmic Pizza**. She is interested in servicing a very long stretch of road which currently has no pizza parlors. There are houses located at various locations on this road (specified by an x-coordinate), and she wants to make sure each house is within 4 miles of an Algorithmic Pizza. She wants to open the fewest pizza parlors possible to accomplish this goal.

The Algorithm:
Going from left to right, place the next pizza parlor 4 miles after the first uncovered household.

A Solution

<u>Base Case</u>: There is an optimal solution that includes our first 0 pizza parlors.

<u>Inductive Hypothesis</u>: Assume there is an optimal solution OPT that includes our first k pizza parlors.

<u>Inductive Step</u>: Assume that p_{k+1}, our (k+1)st pizza parlor, is not in OPT (otherwise we're done).

- Order OPT's pizza parlors from left to right: the first k are all in our algorithm, so consider O_{k+1}, the (k+1)st left-most pizza parlor in OPT.
- O_{k+1} must be further left than p_{k+1}, otherwise there is a house that isn't covered by OPT. Replace O_{k+1} with p_{k+1} to produce OPT'
- OPT' has the same number of parlors, so it is optimal. We are moving O_{k+1} to the right, but not uncovering any households in that area, so it is still valid.

The Problem

You are the manager at Algorithmic Pizza, and have received n different pizza orders. Making a pizza is divided into two stages: (1) preparation, and (2) baking. Each order i takes p_i time in stage 1, and then b_i time in stage 2.

With only a single chef, only one pizza can be prepped at any given time. You have a large stove, however, so you can bake all the pizzas simultaneously.

You want to choose an order to prepare the pizzas so as to minimize the time when the last pizza finishes baking.

The Algorithm:
Prepare pizzas in decreasing order of b_i.

A Solution

Base Case: There is an optimal solution with $\leq C(n,2)$ inversions.

Inductive Hypothesis: Assume there is an optimal solution OPT with $\leq k$ inversions.

Inductive Step: Assume OPT has a consecutive inversion (i,j), where OPT schedules i immediately preceding j (otherwise we're done).

- Create OPT' by instead scheduling j immediately before i.
- This is a valid schedule, but is it optimal?
- None of the pizzas before i are affected.
- None of the pizzas after j are affected, since they still need to wait for both i and j to be prepped.
- j will now finish earlier
- We need to show that i finishes no later in OPT' than j finished in OPT.

Solution, Part 2

- Let X be the sum of the prep times for all pizzas before i in OPT.
- In OPT, j will finish at $X + p_i + p_j + b_j$.
- In OPT', i will finish at $X + p_j + p_i + b_i$.
- We scheduled j before i, which means that $b_i \leq b_j$.
- Therefore, i in OPT' finishes no later than j did in OPT.
- Therefore, OPT' is optimal
- Our algorithm is optimal by induction.