COMP 285 (NC A&T, Spr '22) Homework 9

Due. Wednesday, April 20th, 2022 @ 11:59 PM!

Homework Expectations: Please see Homework.

Exercises The following questions are exercises. We encourage you to work with a group and discuss solutions to make sure you understand the material.
Points This assignment is graded out of 50 points. However, you can get up to 60 points if you complete everything. These are not bonus points, but rather points to help make-up any parts you miss.

Fun with Greedy

Written Problems The following questions are to be submitted in written/typed form to gradescope.

1 Greedy Strategies (10 pt.)

Let $G = (V, E)$ be an undirected unweighted graph. Say that a vertex v can "see" an edge if v is an endpoint of that edge. Say that a set $S \subseteq V$ is "all-seeing" if every edge $e \in E$ is seen by at least one vertex in S. In this problem, we will try to greedily construct the smallest all-seeing set possible.

Consider the following greedy algorithm to find an all-seeing subset:

Algorithm 1: findAllSeeingSubset(G)

```
S = {}
while G contains edges do
    choose an edge e = (u, v) in G
    S.add(u)
    S.add(v)
    remove u and all of its adjacent edges from G
    remove v and all of its adjacent edges from G
return S
```

1.1 (5 pt.)

Prove that *findAllSeeingSubset(G)* always returns an all-seeing subset.

[We are expecting: A short but complete argument as to why the algorithm above always returns an all-seeing subset**]**

1.2 (5 pt.)

Give an example of a graph on which *findAllSeeingSubset(G)* does not return a *smallest* all-seeing subset, for at least one way of choosing edges.

[We are expecting: An example graph and a brief justification why it does not return a smallest all-seeing subset.**]**

2 Plucky's Slushies (10 pt.)

Plucky the penguin has a cooler with a capacity of Q ounces. Plucky heads to the convenience store where there are n flavors of slushies. Each slushy flavor i has a cost per ounce $v_i > 0$ (measured in units of dollars per ounce) and a quantity $q_i > 0$ (measured in ounces). There are q_i ounces of slushy i available to Plucky, and for any real number $x \in [0, q_i]$, the total cost from x ounces of slushy i is $x \cdot v_i$.

Note that Plucky can take a fractional amount of each slushy. For example, perhaps there is 36 ounces of blue raspberry flavored slushy; Plucky can choose to put 12.345 ounces of blue raspberry slushy in the cooler.

Plucky wants to create the most expensive slushy concoction in the cooler. Thus, Plucky wants to choose an amount $x_i \geq 0$ to take for each flavor i in order to maximize $\sum_i x_i v_i$ the total cost while satisfying:

1. Plucky does not overfill the cooler (that is, $\sum_i x_i \leq Q$), and
2. Plucky does not take more of a flavor than is available (that is, $0 \leq x_i \leq q_i$ for all i).

Assume that $\sum_i q_i \geq Q$, so there always is some way to fill the cooler.

2.1 Which flavor? (2 pt.)

Suppose that Plucky has already have partially filled the cooler, and there is some amount of each slushy flavor left. Which flavor should Plucky choose next, and how much?

[We are expecting: A short answer.**]**

2.2 Greedy Algorithm (4 pt.)

Design a **greedy algorithm** which takes as input Q along the tuples (i, v_i, q_i) for $i = 0, \ldots, n-1$, and outputs tuples (i, x_i) so that conditions (1) and (2) hold and $\sum_i x_i v_i$ is as large as possible. Your algorithm should take $O(n \log(n))$.

Note that the tuples may be returned in any order.

[We are expecting: Pseudocode AND an English explanation of what it is doing. A justification of the running time.**]**

2.3 Proof of Correctness (4 pt.)

Complete the inductive step below to prove that your algorithm is correct.

- **Inductive hypothesis:** After making the t'th greedy choice, there is an optimal solution that extends the solution that the algorithm has constructed so far.
- **Base case:** Any optimal solution extends the empty solution, so the inductive hypothesis holds for $t = 0$.

[We are expecting: A proof of the inductive step: assuming the inductive hypothesis holds for $t - 1$, prove that it holds for t. A proof by induction conclusion.**]**

3 Monkey Island (15 pt.)

One day, you get stranded on a beach on an island with n monkeys. You will be here a while, so you want to make friends and introduce yourself to all the monkeys. However, you don't want to leave the beach since the rest of the island looks dangerous. Each monkey i will only be on the beach during one time interval $[a_i, b_i]$, inclusive. You can only introduce yourself to monkey i during this time interval.

Your plan is to stand in the center of the beach and use the megaphone you brought with you to say "Hello" at certain times $t_1, \cdots, t_m$. Any monkey on the beach at one of the times $t_j \in \{t_1, \cdots, t_m\}$ will hear your introduction. It's okay if a monkey hears your introduction more than once, but you want to make sure every monkey hears your introduction at least once.

3.1 Greedy Algorithm (6 pt.)

The battery in your megaphone is low, so you want to minimize m, the number of times you use your megaphone.

Devise a greedy algorithm that takes as input the list of intervals $[a_i, b_i]$ and outputs a list of times $t_1, \cdots, t_m$ such that m is as small as possible but every monkey hears your introduction. Your algorithm should run in time $O(n \log n)$ where n is the number of monkeys on the island. You can assume every monkey visits the beach at some point.

[We are expecting: Pseudocode and an English description of your algorithm, as well as a short justification of the runtime**]**

3.2 Proof of Correctness (9 pt.)

Give a formal proof that your algorithm is correct. **[We are expecting:** A proof by induction. So include a base case, inductive hypothesis, inductive step, and conclusion**]**

Coding Problems The following questions are to be submitted as a ".zip" file on Gradescope.

4 Coding (25 pt.)

After completing the written portion of the assignment, you should submit it to Gradescope.

For the coding portion, get your starter C++ code or Python code.

Note that the starter code also include a few test cases you can run on repl.it. However, the full test suite is the one run on Gradescope.

Please reference the `README.md` included in your starter code for detailed instructions.

Submitting the Assignment

This assignment is a combination of written and programming questions. Both portions of the assignment should be submitted through Gradescope.

The "Homework 9: Fun with Greedy" assignment is the written portion, for which you should submit a **typed** response to the non-coding questions (questions 1-3). Each response should clearly be marked with its corresponding number. You are free to use the provided templates, print the questions and write your answers, or to simply type your responses on a blank document (whatever works for you).

The "Homework 9: Coding" is the programming portion of the assignment. For this portion, download the "answers.cpp", "answers.h" (or just "answers.py") and "citations.txt" from your repl.it these files to Gradescope. You can upload the assignment as many times as you want.