CS4810 / CS7800: Advanced Algorithms — Fall 2022
Homework 2: Due Friday, September 30, 2022

Mahsa Derakhshan and Jonathan Ullman

Collaboration and Honesty Policy Reminder. Collaboration in the form of discussion is allowed and encouraged. However, all forms of cheating are not allowed and will be penalized harshly. These include, but are not limited to, copying parts of an assignment from a classmate, finding answers to problems on the internet or from anyone not enrolled in the class, and plagiarizing from research papers or old posted solutions. A rule of thumb is that you should be able to walk away from discussing a homework problem with no notes and write your solution on your own.

- *You must write up all solutions by yourself, and may not share any written solutions, even if you collaborate with others to solve the problem.*
- *You must identify all of your collaborators. If you did not work with anyone, you should write "no collaborators" or something to that effect on your solutions.* You may have a maximum of two collaborators per assignment, and collaboration is transitive (if you list a collaborator they must list you).
- Asking and answering questions in class forums (lectures, office hours, Piazza) is allowed and encouraged, and you do not need to list these interactions as collaborators.
- Seeking out alternative sources (e.g. fellow students, textbooks, the internet) for general concepts you need for assignments (e.g. proof-by-induction, greedy algorithms, probability) is allowed and encouraged.

Assigned Problems

Problem 1 Show that there are instances of the stable matching problem with exponentially many distinct stable matchings. Specifically, for every n, construct preferences for n hospitals and n doctors/residents and argue that the number of distinct stable matchings for these preferences is $\Omega(b^n)$ for some constant $b > 1$.[1]

The next three questions ask you to design greedy algorithms. For each question, include:

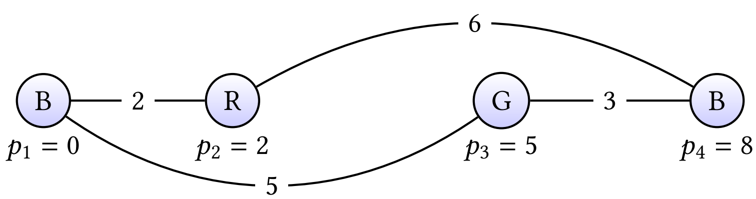
- *A clear pseudocode description of your algorithm.*
- *A proof of its correctness. Your proof doesn't need to be overly formal—just clear and convincing.*
- *An analysis of its asymptotic ("big O") running time.*

Problem 2 Your local school is planning its annual bake sale to raise funds and needs parents to help staff the event. The event runs from time 0 to time T. The parents were surveyed, and each of n parents provided one specific time window where they can volunteer, with parent i volunteering for the

[1] A construction that achieves any $b > 1$ will receive full credit, but if you'd like to challenge yourself, try to make b as large as possible, or show that there cannot be more than $O(c^n)$ stable matchings for some constant $c < \infty$.

window of s_i until f_i. Since each parent will eat something from the table as compensation, you'd like to use the smallest number of different parents to staff the event. Specifically, find the smallest set $S \subseteq \{1, 2, \dots, n\}$ so that $\bigcup_{i \in S} [s_i, f_i] \supseteq [0, T]$. Design a greedy algorithm that solves this problem. For simplicity, you may assume that all the values s_i and f_i are distinct, and that it is possible to staff the entire event from time 0 to time T. For full credit your algorithm should run in time $O(n \log n)$, but slower algorithms will receive significant partial credit.

Problem 3 Rita and Gwen live on a straight road with n tall dorms, with dorm i located at the point p_i on the street. For snowy weather, the school has decided to build bridges that go directly between pairs of dorms. Building a bridge between i and j has a cost of $|p_i - p_j|$ proportional to their distance, so they would like to build the cheapest possible set of bridges that makes it possible to go from any dorm i to any other dorm j. However, due to some incidents at parties, Rita is not allowed to go into certain dorms, and Gwen is not allowed to go into certain other dorms, and we need to ensure that both Rita and Gwen can get around without going into these dorms. Each dorm is labeled **R** (meaning **R**ita can enter), **G** (meaning **G**wen can enter), or **B** (meaning **B**oth can enter), and we need to build enough bridges so that Rita can get between any pair of dorms labeled **R** or **B** without entering any dorm labeled **G**, and the analogous condition for Gwen. We'd of course like to build the cheapest set of bridges possible. For example, given the input below one (not necessarily optimal) solution is to build four bridges with a total cost of $2 + 3 + 5 + 6 = 16$ and Rita can go between any of the **B**/**R** nodes and Gwen can get between any of the **B**/**G** nodes.



Design a greedy algorithm that solves this problem in $O(n \log n)$ time.[2]

Problem 4 As city mayor, you have received complaints about a lack of public spaces in your city. You thus decide on closing-off some of the roads in your city's road network and converting them into public outdoor dining areas. You need to ensure that the road network remains connected, but you don't see why there needs to be more than one route to drive anywhere, and so you decide to eliminate enough roads so that the resulting network is acyclic. However, converting roads into outdoor dining areas is expensive, so you'd like to find the cheapest set of roads to eliminate. More specifically, you are given an undirected, connected graph $G = (V, E)$ that represents the road network with n vertices and m edges. Each road/edge $e \in E$ has a cost w_e for converting it to a dining area. You want to find the cheapest subset of edges $S \subseteq E$ such that the graph $G = (V, E \setminus S)$ is connected and acyclic. Design a greedy algorithm that solves this problem. For full credit your algorithm should run in time $O(m \log n)$ or better, but slower algorithms will receive significant partial credit.[3]

[2] **Hint:** Start by considering what you would do for small examples where the first and last dorm on the street are B and everything in the middle is G or R.

[3] **Hint:** How is this problem related to minimum spanning tree?