CSE 421 Spring 2025: Set 3

Instructor: Chinmay Nirkhe
Due date: April 23rd, 2025 11:59pm

Instructions: Solutions should be legibly handwritten or typeset (ideally in LaTeX). Mathematically rigorous solutions are expected for all problems unless explicitly stated.

You are encouraged to collaborate on problems in small teams but everyone must individually submit solutions. Solutions for the problems may be found online or in textbooks – but do not use them.

For grading purposes, list, with each problem, the names of your collaborators. Please start each problem on a new page.

Notation: Graphs are assumed to have n vertices and m edges in this problem set.
Notation: "Iff" is a portmanteau of "if and only if" commonly used in mathematics.

Problem 1 (Shortest paths with multiple goals)**.** A robotics company needs to procure k unique parts to build a new robot. The parts are available at various stores with each item being priced the same at all the stores that sell it. The company models the problem as a *directed* graph $G = (V, E)$ with non-negative weights $w : E \to \mathbb{R}_{\geq 0}$ and lists $L(v) \subseteq [k]$ per vertex describing the subset of parts available at each vertex. The weight of each edge $u \to v$ describes the cost of driving from u to v.

The company starts their procurement truck at vertex s and their procurement truck should return to s after retrieving all k parts. When the truck visits a vertex v, it can procure multiple parts (or none).

1. **[13 points]** Give an algorithm along with its proof of runtime and correctness that finds the *minimal* cost route that retrieves all k items. For full credit on this problem, you should produce an algorithm with an asymptotic runtime[1] of $2^k m(k + \log n)$ or better! Assume the input is the graph G expressed as an adjacency list.

   Hint: How can we express the "state" of the truck at any given time? In most graph problems, the state of the truck would be a vertex $v \in V$. What is it for this problem?

2. **[2 points]** Now, suppose that the prices of items vary by store. For each vertex $v \in V$ and item $i \in [k]$, there is a price c_{vi} denoting the price at vertex v of item i (with the price being ∞ if the item is not sold there). How would you adjust your previous algorithm to calculate the new minimum cost route? [You do not need to prove that your adjusted algorithm is correct.]

[1]The road network dataset for the USA has approximately $n = 2 \cdot 10^7$ vertices and $m = 3 \cdot 10^7$ edges. So for $k \lesssim \log n \approx 24.3$, this would still be efficient.

Problem 2 (Profit-based job scheduling)**. [20 points]** In this variant of the job scheduling problem, there are n jobs each with a integer deadline $d_i \in \{1, \ldots, n\}$ and a profit $p_i > \mathbb{R}_{>0}$. Each job takes unit time to complete and the profit earned per job is p_i if the job's termination time t_i is $\leq d_i$ and the profit earned is 0 if the job's termination time t_i is $> d_i$. The earliest start time for any job is 0.

Construct an algorithm that outputs the termination times for each of the jobs in a profit maximizing schedule. It is okay to answer ∞ for the termination time for a job indicating that you do not intend on undertaking said job. Assume that arithmetic of deadlines and profits is unit time and space complexity. Provide proofs of runtime and correctness.

For full credit, your algorithm should run in time $O(n^2)$. For full credit plus **2 extra credit points**, your algorithm should run in time $O(n \log n)$.

Hints and suggestions:

1. One step in your argument will be showing that any optimal solution assigns only integer termination times. Convince yourself this is true. To make your life and our grading easier, you don't need to give a proof of this statement in your answer. You may assume it for free.
2. Like many examples in class, a good place to start a proof of correctness is to consider the **first** time your algorithm and a strategy deviate.
3. In our solution, there was significant casework required to prove our algorithm's correctness.

Problem 3 (Independent set (once more))**.** Last week, we proved that when $d \geq 1$, there exists an independent set of size $\geq n/(2d)$ in a graph. This week, we construct a greedy algorithm for finding an independent set of size $\geq n/(d+1)$.

The high level idea of the algorithm is fairly simple and should remind you of examples we have observed in class: Given the input undirected graph, pick the vertex of smallest degree to add to the independent set, prune[2] it and it's neighbors from the graph, and repeat.

1. **[1 point]** Give an example of a graph where the greedy algorithm fails to find the optimal independent set.
2. **[1 point]** Give an example of a graph with a maximum independent set of size exactly $n/(2d)$. Give an example of a graph with an independent set of size exactly $n/(d+1)$.
3. **[4 points]** Spell out the algorithm in more detail using better data structures and prove its runtime.
4. Analyzing a greedy approximation algorithm is something you have only seen once in class. We'll walk you through another analysis here. Let's observe that the algorithm terminates once all vertices are pruned. Let $I \subseteq V$ be the independent set produced by this greedy algorithm. For $v \in I$, let $f(v)$ be the number of vertices (including v) that are pruned when v is added to I.
   (a) **[1 point]** What is the equation comparing n and $\{f(v)\}_{v \in I}$?
   (b) **[2 points]** In terms of $f(v)$, give a *lower bound* for the sum of the degrees of all the vertices pruned when v is added to I.
   (c) **[2 points]** Express this as an inequality relating d and $\{f(v)\}_{v \in I}$.
   (d) **[2 points]** Using the property that $\sum_{v,w \in I}(f(v) - f(w))^2 \geq 0$, prove that

$$\left(\sum_v f(v)\right)^2 \leq |I| \sum_v f(v)^2. \tag{1}$$

   (e) **[2 points]** Now conclude that $|I| \geq n/(d+1)$.

[2] Pruning a vertex from a graph means removing it and all its associated edges.