CS 170 Homework 4

Due **Saturday 2/22/2025, at 10:00 pm (grace period until 11:59pm)**

1 Study Group (1 point)

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

2 Sparsity and SCCs (10 points)

Call a directed graph G *weakly-connected* if, upon making all edges bidirectional (i.e. turning the directed graph into an undirected graph), the graph is connected.

(a) (2 points) Argue that every weakly-connected directed graph G with n vertices and $n-1$ edges must have n strongly connected components.

(b) (3 points) Consider a weakly-connected directed graph G with $n \geq 3$ vertices and n directed edges such that for every vertex, v, there are exactly two directed edges incident to v. For every integer k from 1 to n, describe conditions (if any) upon which there are k strongly-connected components.

Hint: this graph structure is very special. If $n = 5$, is it possible for there to be exactly two strongly connected components?

(c) (2 points) For the graph described part (b), describe a *simple* and efficient algorithm using *only one* DFS or BFS traversal that determines the number of SCCs G has.

Note: you may **not** use the SCC-finding algorithm from lecture or another black-box SCC algorithm such as Tarjan's algorithm. Your algorithm should be even simpler.

(d) (3 points) Now, consider any weakly-connected directed graph G' with $n \geq 3$ vertices and n directed edges. Modify your algorithm from part (c) to efficiently count the number of SCCs G' has (still using at most one DFS or BFS traversal).

3 2-SAT (10 points)

In the 2SAT problem, you are given a set of clauses, where each clause is the disjunction (OR) of two literals (a literal is a Boolean variable or the negation of a Boolean variable). You are looking for a way to assign a value true or false to each of the variables so that all clauses are satisfied – that is, there is at least one true literal in each clause. For example, here's an instance of 2SAT:

$$(x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee \overline{x_3}) \wedge (x_1 \vee x_2) \wedge (\overline{x_3} \vee x_4) \wedge (\overline{x_1} \vee x_4)$$

Recall that $\vee$ is the logical-OR operator and $\wedge$ is the logical-AND operator and $\overline{x}$ denotes the negation of the variable x. This instance has a satisfying assignment: set x_1, x_2, x_3, and x_4 to `true, false, false, and true`, respectively.

The purpose of this problem is to lead you to a way of solving 2SAT efficiently by reducing it to the problem of finding the strongly connected components of a directed graph. Given an instance I of 2SAT with n variables and m clauses, construct a directed graph $G_I = (V, E)$ as follows.

- G_I has $2n$ nodes: one for each variable and its negation.
- G_I has $2m$ edges: for each clause $(\alpha \vee \beta)$ of I (where α, β are literals), G_I has an edge from from $\overline{\alpha}$ to β, and one from the $\overline{\beta}$ to α.

Note that the clause $(\alpha \vee \beta)$ is equivalent to each of the implications $\overline{\alpha} \implies \beta$ and $\overline{\beta} \implies \alpha$. In this sense, G_I records all implications in I.

(a) (2 points) Show that if G_I has a strongly connected component containing both x and $\overline{x}$ for some variable x, then I has no satisfying assignment.

(b) (5 points) Now show the converse of (a): namely, that if none of G_I's strongly connected components contain both a literal and its negation, then the instance I must be satisfiable.

Hint: Pick a sink SCC of G_I. Assign variable values so that all literals in the sink are True. Why are we allowed to do this, and why doesn't it break any implications?

(c) (3 points) Conclude that there is a linear-time algorithm for solving 2SAT. Provide the algorithm description and runtime analysis; proof of correctness is not required (in fact, you've already done the bulk of the proof!)

4 Where's the Graph? (12 points)

Each of the following problems can be solved with techniques taught in lecture. Construct a simple directed graph, write an algorithm for each problem by black-boxing algorithms taught in lecture, and analyze its runtime. **You do not need to provide proofs of correctness.**

(a) (4 points) Sarah wants to do an extra credit problem for her math class. She is given three numbers: 1, x, and y. Starting from x, she needs to find the shortest sequence of additions, subtractions, and divisions (only possible when the number is divisible by y) using 1 and y to get to 2025. For example, if $x + 1$ is divisible by y, then from x, she can first add 1 and then divide by y to get $(x + 1)/y$. If there are multiple sequences with the shortest length, return any one of them. She can use 1 and y multiple times. Design an efficient algorithm that Sarah can query to get this sequence of arithmetic operations.

Hint: For finding the runtime, is there an upper bound on the largest possible number that Sally might ever create in an optimal sequence of operations?

(b) (4 points) The CS 170 course staff is helping build a roadway system for PNPenguin's hometown in Antarctica. Since we have skill issues, we are only able to build the system using one-way roads between igloo homes. Before we begin construction, PNPenguin wants to evaluate the accessibility of our design. To do this, they want to determine the number of *accessible* igloos; an igloo is accessible if you are able to leave the igloo along some road and then return to it along a path of other roads. However, PNPenguin isn't very good at algorithms, and they need your help.

Given our proposed roadway layout, design an efficient algorithm that determines the number of accessible igloos.

(c) (4 points) There are s different species of Pokemon, all descended from the original Mew. Also, all species have at most 1 parent. For any species of Pokemon, Professor Juniper knows all of the species *directly* descended from it. She wants to write a program that answers queries about these Pokemon. The program would have two inputs: a and b, which represent two different species of Pokemon. Her program would then output one of three options in constant time (the time complexity cannot rely on s):

(1) a is descended from b.

(2) b is descended from a.

(3) a and b share a common ancestor, but neither are descended from each other.

Unfortunately, Professor Juniper's laptop is very old and its SSD drive only has enough space to store up to $O(s)$ pieces of data for the program. Give an algorithm that Professor Juniper's program could use to solve the problem above given the constraints.

Hint: Professor Juniper can run some algorithm on her data before all of her queries and store the outputs of the algorithm for her program; time taken for this precomputation is not considered in the query run time.

5 Road Trip (11 points)

The CS 170 staff are preparing to drive from Berkeley to Chicago to attend a computer science conference!

Assume that the roads from Berkeley to Chicago can be expressed as a directed weighted graph $G = (V, E, c)$, where each $v \in V$ represents a city, each $(u, v) \in E$ represents a road from city u to city v, and $c(u, v)$ represents the cost of gas required to drive from city u to city v. Each road takes exactly one day to traverse, and after driving across road (u, v), the staff will stop in city v overnight to rest.

Unfortunately, there are too many course staff, so they will have to rent two separate cars for the trip, and each city along the way (not including Berkeley and Chicago) can only accommodate one group of staff per night. Both cars start in Berkeley and depart on the same day.

Additionally, each group has to pay $r > 0$ dollars per day in car rental fees. They are allowed to spend a day in any city without driving, but they still have to pay the rental fee for that day. Once a group arrives in Chicago, the group will return the car and does not have to pay any more rental fees.

Our goal is to design an efficient algorithm to find the routes that minimize the total cost of the trip, including gas and rental fees. (You may assume that no matter the route, the staff will always arrive in Chicago before the conference starts.)

(a) (3 points) Find an example of a graph G along with costs c and r where it is not optimal for either car to take the path that Dijkstra's outputs from Berkeley to Chicago.

(b) (8 points) Devise an efficient algorithm that determines the optimal routes.

Hint: given the graph from part (a), it seems that running Dijkstra's twice on G, even with some modifications on the second run, likely won't work.

Hint 2: refer to Discussion 4 for ideas regarding other ways to handle shortest path variants.

(i) Describe your algorithm.

(ii) Prove the correctness of your algorithm.

(iii) Analyze the runtime of your algorithm.

6 Job Allocation (11 points)

You have n computing jobs to perform, with job i requiring t_i units of CPU time to complete. You also have access to n machines that you can assign these jobs to. Since the machines are in high demand, you can only assign one job to any machine. The jth machine costs c_j dollars for each unit of CPU time it spends running a job, so assigning job i to machine j will cost you $t_i \cdot c_j$ dollars (each job takes the same amount of CPU time to complete, regardless of which machine is used). Your goal is to find an assignment of jobs to machines that minimizes the total cost.

Assume the jobs and machines are sorted and have distinct runtimes/costs, i.e. $t_1 > t_2 > \ldots > t_n$ and $c_1 > c_2 > \ldots > c_n$.

(a) (3 points) Describe the assignment of jobs to machines minimizes the total cost (no proof necessary).

Hint: What machine should we assign the longest job to? What machine should we assign the second longest job to? It might help to solve a small example by hand first.

(b) (4 points) Given an assignment of jobs to machines, consider the following modification: if there is a pair of jobs i, j such that job i is assigned to machine i', job j is assigned to machine j', and $t_i > t_j$ and $c_{i'} > c_{j'}$, instead assign job i to machine j' and job j to machine i'. Show that this modification decreases the total cost of an assignment.

(c) (4 points) Use part (b) to show by induction that the assignment you chose in part a has the minimum total cost.

