Instructions. Limited collaboration is allowed while solving problems, but you must write solutions yourself. List collaborators on your submission.

You can choose which problems to complete, but must submit at least one problem per assignment. See the course page for information about how challenge problems are graded and contribute to your homework grade. Since you don’t need to complete every problem, you are encouraged to focus your efforts on producing high-quality solutions to the problems you feel confident about. There is no benefit to guessing or writing vague answers.

If you are asked to design an algorithm, please (a) give a precise description of your algorithm using either pseudocode or language, (b) explain the intuition of the algorithm, (c) justify the correctness of the algorithm; give a proof if needed, (d) state the running time of your algorithm, (e) justify the running-time analysis.

Submissions. Please submit a PDF file. You may submit a scanned handwritten document, but a typed submission is preferred. Please assign pages to questions in Gradescope.

This challenge set has extra problems from previous chapters that you may attempt to try to meet your goals for challenge problems:

- Submit at most 6 problems total.
- Course staff won’t provide much support on extra problems. TAs will prioritize questions about intractability problems, and may not be familiar with extra ones.

1 Challenge Problems about Intractability

Problem 1. Summer Camp Recruiting (K&T Ch 8 Ex 3) Suppose you’re helping to organize a summer sports camp, and the following problem comes up. The camp is supposed to have at least one counselor who’s skilled at each of the n sports covered by the camp (baseball, volleyball, and so on). They have received job applications from m potential counselors. For each of the n sports, there is some subset of the m applicants qualified in that sport. The question is: For a given number k < m, is it possible to hire at most k of the counselors and have at least one counselor qualified in each of the n sports? We’ll call this the Efficient Recruiting Problem.

Show that Efficient Recruiting is NP-complete.

Problem 2. Hitting set (K&T Ch 8 Ex 5). Consider a set A = \{a_1, \ldots, a_n\} and a collection B_1, \ldots, B_m of subsets of A (i.e., B_i \subseteq A for all i). We say that H \subseteq A is a hitting set for the collection if H contains at least one element from each B_i, that is \( H \cap B_i \) is non-empty for all \( i \) (so H “hits” all the sets \( B_i \)).

The Hitting-Set problem is the following: Given a set A = \{a_1, \ldots, a_n\}, subsets B_1, \ldots, B_m \subseteq A, and a number k, is there a hitting set \( H \subseteq A \) of size at most k?

Prove that Hitting-Set is NP-Complete.

Problem 3. Combinatorial Auctions (K&T Ch 8 Ex 13). A combinatorial auction is a particular mechanism developed by economists for selling a collection of items to a collection of potential buyers. Here’s a simple type of combinatorial auction. There are n items for sale, labeled I_1, \ldots, I_n. Each item can only be sold to one person. Now, m different people place bids: The ith bid is a pair (S_i, x_i), where S_i is a subset of the items, and x_i is an offering price the bidder is willing to pay for the items in the set S_i, as a single unit.

An auctioneer now looks at the set of all m bids; they choose to accept some of these bids and to reject the others. If the bid (S_i, x_i) is accepted, then the ith person gets to take the items in S_i. Thus, no two accepted
bids can contain an item in common (since we can’t give the same item to two people). The auctioneer collects the sum of the offering prices of all accepted bids, and their goal is to collect as much money as possible.

Thus, the problem of Winner Determination for Combinatorial Auctions asks: Given items $I_1,\ldots,I_n$, bids $(S_1,x_1),\ldots,(S_m,x_m)$, and a bound $B$, is there a collection of bids that the auctioneer can accept so as to collect an amount of money that is at least $B$?

Prove that the problem of Winner Determination for Combinatorial Auctions is NP-complete.

**Example.** Suppose an auctioneer decides to use this method to sell some excess computer equipment. There are four items labeled PC, monitor, printer, and scanner; and three people place bids. Define

$$S_1 = [\text{PC}, \text{monitor}], S_2 = [\text{PC}, \text{printer}], S_3 = [\text{monitor}, \text{printer}, \text{scanner}]$$

and

$$x_1 = x_2 = x_3 = 1$$

The bids are $(S_1,x_1),(S_2,x_2),(S_3,x_3)$, and the bound $B$ is equal to 2.

Then the answer to this instance is no: The auctioneer can accept at most one of the bids (since any two bids have a desired item in common), and this results in a total monetary value of only 1.

**Problem 4. Evasive Path (K&T Ch 8 Ex 12).** Some friends of yours maintain a popular news and discussion site on the Web, and the traffic has reached a level where they want to begin differentiating their visitors into paying and nonpaying customers. A standard way to do this is to make all the content on the site available to customers who pay a monthly subscription fee; meanwhile, visitors who don’t subscribe can still view a subset of the pages (all the while being bombarded with ads asking them to become subscribers).

Here are two simple ways to control access for nonsubscribers: You could (1) designate a fixed subset of pages as viewable by nonsubscribers, or (2) allow any page in principle to be viewable, but specify a maximum number of pages that can be viewed by a nonsubscriber in a single session. (We’ll assume the site is able to track the path followed by a visitor through the site.)

Your friends are experimenting with a way of restricting access that is different from and more subtle than either of these two options. They want nonsubscribers to be able to sample different sections of the Web site, so they designate certain subsets of the pages as constituting particular zones—for example, there can be a zone for pages on politics, a zone for pages on music, and so forth. It’s possible for a page to belong to more than one zone. Now, as a nonsubscribing user passes through the site, the access policy allows them to visit one page from each zone, but an attempt by the user to access a second page from the same zone later in the browsing session will be disallowed. (Instead, the user will be directed to an ad suggesting that he or she become a subscriber.)

More formally, we can model the site as a directed graph $G = (V,E)$, in which the nodes represent Web pages and the edges represent directed hyperlinks. There is a distinguished entry node $s \in V$, and there are zones $Z_1,\ldots,Z_k \subset V$. A path $P$ taken by a nonsubscriber is restricted to include at most one node from each zone $Z_i$.

One issue with this more complicated access policy is that it gets difficult to answer even basic questions about reachability, including: Is it possible for a nonsubscriber to visit a given node $t$? More precisely, we define the Evasive Path Problem as follows: Given $G,Z_1,\ldots,Z_k,s \in V$, and a destination node $t \in V$, is there an $s-t$ path in $G$ that includes at most one node from each zone $Z_i$? Prove that Evasive Path is NP-complete.

**Hint:** try a reduction from 3-Sat. This is not as hard as it seems. How can you use zones to force an evasive path to make a binary choice, akin to selecting true or false for a Boolean variable? After doing that, how can you use an evasive path to model whether or not clauses are satisfied?
2 Challenge Problems from Previous Chapters

Problem 5. K&T Chapter 3, Exercise 10. Counting shortest paths. A number of art museums around the country have been featuring work by an artist named Mark Lombardi (1951–2000), consisting of a set of intricately rendered graphs. Building on a great deal of research, these graphs encode the relationships among people involved in major political scandals over the past several decades: the nodes correspond to participants, and each edge indicates some type of relationship between a pair of participants. And so, if you peer closely enough at the drawings, you can trace out ominous-looking paths from a high-ranking U.S. government official, to a former business partner, to a bank in Switzerland, to a shadowy arms dealer.

Such pictures form striking examples of social networks, which, as we discussed in Section 3.1, have nodes representing people and organizations, and edges representing relationships of various kinds. And the short paths that abound in these networks have attracted considerable attention recently, as people ponder what they mean. In the case of Mark Lombardi’s graphs, they hint at the short set of steps that can carry you from the reputable to the disreputable.

Of course, a single, spurious short path between nodes \( v \) and \( w \) in such a network may be more coincidental than anything else; a large number of short paths between \( v \) and \( w \) can be much more convincing. So in addition to the problem of computing a single shortest \( v - w \) path in a graph \( G \), social networks researchers have looked at the problem of determining the number of shortest \( v - w \) paths.

This turns out to be a problem that can be solved efficiently. Suppose we are given an undirected graph \( G = (V, E) \), and we identify two nodes \( v \) and \( w \) in \( G \). Give an algorithm that computes the number of shortest \( v - w \) paths in \( G \). (The algorithm should not list all the paths; just the number suffices.) The running time of your algorithm should be \( O(m + n) \) for a graph with \( n \) nodes and \( m \) edges.

Problem 6. K&T Chapter 3 Exercise 12. Oral histories You’re helping a group of ethnographers analyze some oral history data they’ve collected by interviewing members of a village to learn about the lives of people who’ve lived there over the past two hundred years. From these interviews, they’ve learned about a set of \( n \) people (all of them now deceased), whom we’ll denote \( P_1, P_2, \ldots, P_n \). They’ve also collected facts about when these people lived relative to one another. Each fact has one of the following two forms:

- For some \( i \) and \( j \), person \( P_i \) died before person \( P_j \) was born; or
- For some \( i \) and \( j \), the life spans of \( P_i \) and \( P_j \) overlapped at least partially.

Naturally, they’re not sure that all these facts are correct; memories are not so good, and a lot of this was passed down by word of mouth. So what they’d like you to determine is whether the data they’ve collected is at least internally consistent, in the sense that there could have existed a set of people for which all the facts they’ve learned simultaneously hold.

Give an efficient algorithm to do this: either it should produce proposed dates of birth and death for each of the \( n \) people so that all the facts hold true, or it should report (correctly) that no such dates can exist — that is, the facts collected by the ethnographers are not internally consistent.

Problem 7. Connectivity in Weighted Graphs (Chapter 4). Let \( G = (V, E, W) \) be a connected weighted graph where each edge \( e \) has an associated non-negative weight \( w(e) \). We call a subset of edges \( F \subseteq E \) unseparating if the graph \( G' = (V, E \setminus F) \) is connected. This means that if you remove all of the edges \( F \) from the original edge set, this new graph is still connected. For a set of edges \( E' \subseteq E \) the weight of the set is just the sum of the weights of the individual edges that is \( w(E') = \sum_{e \in E'} w(e) \). Give a polynomial time algorithm to find the unseparating set with maximum weight.

Problem 8. Longest common substring (Chapter 6). Given two strings \( x = x_1 x_2 \cdots x_n \) and \( y = y_1 y_2 \cdots y_m \) we wish to find the length of their longest common substring, that is, the largest \( k \) for which there are indices \( i \) and \( j \) with \( x_i x_{i+1} \cdots x_{i+k-1} = y_j y_{j+1} \cdots y_{j+k-1} \). Show how to do this in \( O(mn) \) time.

Problem 9. Split a string into words (Chapter 6). You are given a string of \( n \) characters \( s[1...n] \), which you believe to be a corrupted text document in which all punctuation has vanished (so that it looks something like itwasthebestoftimes...). You wish to reconstruct the document using a dictionary, which is
available in the form of a Boolean function $dict$: for any string $w$,

$$
dict(w) = \begin{cases} 
\text{true} & \text{if } w \text{ is a valid word,} \\
\text{false} & \text{otherwise.}
\end{cases}
$$

1. Give a dynamic programming algorithm that determines whether the string $s$ can be reconstituted as a sequence of valid words. The running time should be at most $O(n^2)$, assuming calls to $dict$ take unit time.

2. In the event that the string is valid, make your algorithm output the corresponding sequence of words.

**Problem 10. Holidays and employees (Chapter 7).** There is a set $H$ of holidays and a set of employees. On each holiday, exactly one employee must be at work. Each employee $w$ has a set $H_w \subseteq H$ of holidays when they can work and a maximum number $m_w$ of holidays that they can work. Give an efficient algorithm that assigns employees to holidays when they should work, or reports this is not possible.