COMPSCI 311: Introduction to Algorithms **Spring 2019**

Homework 6

Released 4/11/2019 Due 4/26/2019 11:59pm in Gradescope

Instructions. You may work in groups, but you must write solutions yourself. List collaborators on your submission.

If you are asked to design an algorithm, please provide: (a) the pseudocode or precise description in words of the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your 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.

Problem 1. (10 points) Diverse Subset (K&T Ch8Ex2) A store trying to analyze the behavior of its customers will often maintain a two-dimensional array A, where the rows correspond to its customers and the columns correspond to the products it sells. The entry $A[i, j]$ specifies the quantity of product j that has been purchased by customer i.

Heres a tiny example of such an array A.

	detergent	beer	diapers	cat litter
Raj	0	6	0	3
Alanis	2	3	0	0
Chelsea	0	0	0	7

One thing that a store might want to do with this data is the following. Let us say that a subset S of the customers is diverse if no two of the of the customers in S have ever bought the same product (i.e., for each product, at most one of the customers in S has ever bought it). A diverse set of customers can be useful, for example, as a target pool for market research. We can now define the Diverse Subset Problem as follows: Given an mn array A as defined above, and a number $k \leq m$, is there a subset of at least k of customers that is diverse? Show that Diverse Subset is NP-complete.

Problem 2. (20 points) Evasive Path (K&T Ch8Ex12). 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 dont 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. (Well 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 zonesfor example, there can be a zone for pages on politics, a zone for pages on music, and so forth. Its possible for a page to belong to more than one zone. Now, as a nonsubscribing user passes through the site, the access policy allows him or her 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.

Problem 3. (20 points) Strategic Advertising (K&T Ch8Ex10). Your friends at LargeTechCo have recently been doing some consulting work for companies that maintain large, publicly accessible websites-contractual issues prevent them from saying which ones-and theyve come across the following Strategic Advertising Problem. A company comes to them with the map of a website, which well model as a directed graph $G = (V, E)$. The company also provides a set of t trails typically followed by users of the site; well model these trails as directed paths $P_1, P_2, \ldots, P_t$ in the graph G (i.e., each P_i is a path in G). The company wants LargeTechCo to answer the following question for them: Given G, the paths $\{P_i\}$, and a number k, is it possible to place advertisements on at most k of the nodes in G, so that each path P_i includes at least one node containing an advertisement? Well call this the Strategic Advertising Problem, with input $G, \{Pi : i = 1, \ldots, t\}$, and k. Your friends figure that a good algorithm for this will make them all rich; unfortunately, things are never quite this simple.

1. Prove that Strategic Advertising is NP-complete.
2. Your friends at LargeTechCo forge ahead and write a pretty fast algorithm S that produces yes/no answers to arbitrary instances of the Strategic Advertising Problem. You may assume that the algorithm S is always correct. Using the algorithm S as a black box, design an algorithm that takes input G, $\{Pi\}$, and k as in part (a), and does one of the following two things:
   (a) Outputs a set of at most k nodes in G so that each path P_i includes at least one of these nodes, or
   (b) Outputs (correctly) that no such set of at most k nodes exists. Your algorithm should use at most a polynomial number of steps, together with at most a polynomial number of calls to the algorithm S.

Problem 4. (20 Points) Combinatorial Auctions (K&T Ch8.Ex13). A *combinatorial auction* is a particular mechanism developed by economists for selling a collection of items to a collection of potential buyers. (The Federal Communications Commission has studied this type of auction for assigning stations on the radio spectrum to broadcasting companies.)

Heres a simple type of combinatorial auction. There are n items for sale, labeled $I_1, ..., I_n$. Each item is indivisible and can only be sold to one person. Now, m different people place bids: The ith bid specifies a subset S_i of the items, and an offering price x_i that the bidder is willing to pay for the items in the set S_i, as a single unit. (Well represent this bid as the pair (S_i, x_i).)

An auctioneer now looks at the set of all m bids; she chooses to accept some of these bids and to reject the others. Each person whose bid i is accepted gets to take all the items in the corresponding set S_i. Thus the rule is that no two accepted bids can specify sets that contain a common item, since this would involve giving the same item to two different people.

The auctioneer collects the sum of the offering prices of all accepted bids. (Note that this is a one-shot auction; there is no opportunity to place further bids.) The auctioneers goal is to collect as much money as possible.

Thus, the problem of Winner Determination for Combinatorial Auctions asks: Given items $I_1, ..., I_n$, bids

$(S_1, x_1), ..., (S_m, x_m)$, 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

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

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 5. (30 points) Punctuation Fix (Programming). 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,

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

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, your algorithm should return the boolean value true. In the event that the string is invalid, your algorithm should return the boolean value false.

This is a programming assignment that you will upload to Gradescope in Python or Java. Please check Moodle to download a framework that will provide you with the necessary helper functions and classes. The README file inside of the package will give further instructions.

Problem 6. (0 points). How long did it take you to complete this assignment?