CSE 6140 Assignment 1
due Wednesday, Sept. 28, 2016 at 11:55pm EDT on T-Square

Please upload a PDF with solutions to all the problems. If you handwrite the solutions first, please make sure all answers are legible; no credit will be given to illegible answers.

1 Greedy - The Great Ice Cream Sale

Amrita's Amazing Ice Cream shop is the talk of Georgia Tech. It produces the same n delicious flavors freshly every day, and never resells old ice cream made on previous days. In order to get rid of the extra ice cream at the end of each day, Amrita's Amazing Ice Cream has a special deal at closing time. They sell 1 gallon buckets of ice cream for a fraction of the usual price, packed with your choice of leftover ice cream. You can choose as many flavors as you like that can fit in the bucket; however, because Amrita's wants to get rid of their ice cream as quickly as possible, their only stipulation is that once you choose a flavor, you need to take it all (or as much as you can until your bucket is full).

Being a poor college student, you love quality ice cream as much as the next person, but can only afford to buy Amrita's ice cream with the special deal. Your goal is to come home with the best ice cream bucket every time, i.e. packed to maximize the total likeability score L of all its flavors. The total score L is the sum of each individual ice cream's score s_i, which you decide when you see how much ice cream of each flavor is left for the day. This score s_i assumes you are getting the full quantity d_i of the ice cream flavor i for that day; if you only receive a fraction of the ice cream flavor i for that day, you only receive the proportional fraction of the score. For example, consider that the Ice Cream Shop has 3 flavors with leftover amounts of $d_1 = 1, d_2 = 0.5, d_3 = 0.25$ gallons, and your scores for them are s_1, s_2, s_3. If you pick flavor 1 to fill your bucket, this gives you a score of $L = s_1$. If you decide to pick flavor 2 first, and fill the other half gallon with ice cream 1, this would give you a score of $L = s_2 + \frac{s_1}{2}$ (you only get half the score of 1 because you only get half the amount of 1). If you first decide to get ice cream 3 with its quantity of a fourth gallon ($d_3 = 0.25$), then flavor 2, then flavor 1, then you will be getting all of flavor 3, all of flavor 2, and a fourth of flavor 1's amount; your score is thus $L = s_3 + s_2 + \frac{s_1}{4}$.

The amount of each ice cream d_i left at the end of the day changes daily, and so your optimal ice cream bucket may also change. Assume your scores s_i are already calibrated to the amount d_i in question. Hence, instead of spending time every day to figure out your optimal ice cream bucket, you've decided to design an algorithm to help you out. Give an efficient algorithm to decide on the best combination of ice cream to get to maximize its score L. Prove that your algorithm does indeed yield an optimal bucket. (For the sake of analysis, you may assume that each flavor's amount can be measured as an integer number of ounces.)

2 Dynamic Programming: George Learns Algorithms

George P. Burdell has been taking the algorithms course at Tech every year since the College of Computing opened in 1964. As he looks at his transcripts over the decades, he wonders if he has been getting the hang of the material. The difficulty of the course varied based on which instructor taught it and how much time George spent at Georgia Tech football games, so naturally his final scores fluctuated from year to year. He decides the best way to measure his progress is to look at the sequence of final scores he earned (thankfully they are all positive and $\leq$100) and find the longest subsequence of scores over which his grade strictly improved. He looks at his grades from 1980-1989:

$$\{82, 77, 65, 89, 83, 68, 88, 71, 91, 90\}$$

George does not want to restrict himself to only contiguous years, since then it looks like he is never able to improve his performance for longer than 2 years ($\{65, 89\}$ or $\{68, 88\}$ or $\{71, 91\}$). If he allows himself to leave out whatever years he likes, he could see his grade improving over the following 3 scores: $\{77, 89, 91\}$. Or better yet, the following 4 scores: $\{77, 83, 88, 91\}$.

Feeling better about himself already, George defines a subsequence to be the original sequence with some (or all) of the elements removed. Now, he enlists the help of one his clever classmates from this year's algorithms class (you) to find his longest subsequence of improving grades.

1. George is not convinced that his problem can be solved any way besides a brute force search through all 2^n possible subsequences (where n is the number of times George took algorithms). Prove this problem has optimal substructure to show George there may be a better approach.
2. Write down a recurrence relation for this problem.
3. Design a bottom up DP algorithm to solve the problem. Include the time and space complexity of this algorithm to conclusively show George that your algorithm is better than an exhaustive search.

3 Dynamic Programming: Most Mysterious Mansion

You are playing a game in which you must manage a team of 2 adventurers as they travel through a cursed mansion filled with monsters. The explorers accumulate stress every time they encounter a monster, starting with 0 stress at the beginning of the excursion. Each of them can handle at most S units of stress. Each monster causes an even amount of stress s_i (in other words: $s_i/2$ will be an integer) and defeating it yields reward r_i. In order to defeat a monster, one of the 2 adventurers must step up and fight it, causing their own stress level to increase by s_i, but earning reward r_i for the team. (Note, if the fight would cause the adventurer to exceed their stress threshold, they cannot defeat the monster.) Alternatively, the team may choose to evade the monster, in which case there is no reward and all of them will split the stress level, s_i, evenly. Your campaign is over when either evading or fighting the next monster will raise either adventurer's stress level over S. For example: in the case where the next monster causes a stress level of 12, where the first adventurer's stress level is at 8, the second adventurer's stress level is at 7, and their max stress level allowed is 10, then the campaign will be over because any action (fighting or evading) would cause an adventurer's stress level to go above 10. Suppose you are now given the order in which monsters will appear, their stress costs and their rewards. Use dynamic programming to devise a strategy that will maximize the total reward your team can earn before they must abandon their mission.

1. Prove optimal substructure.
2. Provide the recurrence relation.
3. Design a top-down algorithm with memoization. Include the time and space complexity of your approach.

4 NP-complete

Now consider the case when Amrita's Amazing Ice cream runs out of gallon containers. They offer to pack your ice cream into two smaller containers, and ask you which flavors to put in each. Wanting to produce two equally pleasing smaller buckets, you wonder if there's a way to divide up the flavors such that the two smaller buckets B_1 and B_2 have exactly the same score $L_1 = L_2 = \frac{L}{2}$ (you may assume L is even). You do not care that the quantities of the buckets may be different, only that their likeability scores are the same. Unfortunately for you, you cannot come up with a good algorithm to quickly find the best arrangement. Even worse, your best friend tells you that you may just have to settle for containers with lopsided likeability scores the next time this happens. Prove that your best friend is right by showing that this problem is NP-Complete. Remember to follow the steps from lecture to prove NP-completeness; lack of any of the steps will result in a suboptimal grade. *Hint*: The Subset Sum problem is that, given a set of integers S, is there a subset of S that sums to a value k? Because the subset sum problem is NP-complete, you can use it for your reduction.