University of Washington
Department of Computer Science and Engineering
CSE 417, Winter 2020

March 4, 2020

Homework 8, Problem 1

Problem 1 (10 points) Homework optimization:

Based on problem 20, Page 329 from the text, with a simplification: Suppose it is nearing the end of the quarter and you are taking n courses, each with a final project that still has to be done. Each project will be graded on the following scale: It will be assigned an integer number on a scale of 1 to g, higher numbers being better grades. Your goal is to maximize your average grade on the n projects.

You have a total of H hours in which to work on the n projects cumulatively, and you want to decide how to divide up this time. For simplicity, assume H is a positive integer and you'll spend an integer number of hours on each project. To figure out how best to divide up your time, you have come up with a set of functions $\{f_i : i = 1, 2, \ldots, n\}$ for each of your n courses; if you spend $h \leq H$ hours on the project for course i, you'll get a grade of $f_i(h)$. (You may assume that the functions f_i are non-decreasing.)

The problem is: Given these functions $\{f_i\}$, decide how many hours to spend on each project (in integer values only) so that your average grade, as computed according to the f_i, is as large as possible. In order to be efficient, the running time of your algorithm should be polynomial in n, g, and H; none of these quantities should appear as an exponent in your running time.

Now to simplify the problem - you only signed up for $n = 3$ courses, so you just need to consider functions f_1, f_2, and f_3. Give an algorithm to determine how much time to spend on each assignment that runs in $O(H^2)$ time.

Answer:

algorithm:

```
def homework_optimization(f,H):
    P = [[0 for i in range(4)] for i in range(H)]
    for h in range(H):
        P[0,h] = 0

    for i in range(0, 4):
        for h in range(1, H+1):
            for k in range(1, h):
                P[i,h] = max(f[i](t)+P[i-1, h-k], P[i, h])
```

proof:

Let $0 \leq i \leq 3$ to be the first i course

Let P[i, h] to be maximum total grade for this subproblem to maximize grade on first i course with at most h hours

base case:
When i = 0:
According to the context, $P[0, h] = 0$, since no course means no grade.
induction hypothesis:
$P[i, h] = max : f_i(k) + P[i - 1, h - k]$ where $0 \leq i \leq 3$, $0 \leq k \leq h$
since in the optimal solution to subproblem of first i course with at most h hour,k hours was to used on this course, and was compared among all timeframe among total H hours. There are only H possibilities to allocate project of course i, so for each subproblem, we use funtion to each h to locate the optimal time.
Started from the no course and time h = 0, couse 1 optimal value is base on base case and attained the optimal grade with all time. And then add course 2, and memorized course 2's optimal solution build upon the prior subproblem and then do the same until course 3. Since adding each course is garantee to be optimal because the solution built upon the 1 to i-1 couses. Therefore, the algorithm garantee to get the optimal result.

The runtime is $O(H^2)$, since there are 3 * H subproblem with first i course with at most h hours, and each subproblem trace back H hours.