**EECS 340/454: Algorithms** Spring 2020

# Assignment 5: Dynamic Programming

Instructors: Mehmet Koyutürk and Orhan Özgüner Due: March 22 before 11:59 PM

## Problem 1

Provide a dynamic programming solution to each problem by following the described steps:

### Steps:

(i) Identify the "last" question you need to answer in developing a solution.

*(ii)* Define and prove optimal substructure.

(iii) Define subproblems, express the solution to the overall problem in terms of the subproblems.

(iv) Formulate a recursive solution to the subproblems. Do not forget to specify the base case(s).

(v) Characterize the runtime of the resulting procedure assuming that you would implement your solution using a bottom-up procedure.

*(vi)* Provide the pseudo code of the bottom-up procedure you use to compute the value of the optimal solution, as well as the procedure for reconstructing the optimal solution.

***** Note that, you only need to apply items (ii) and (vi) on <u>one</u> of the problems (of your own choice, you can choose a different problem for each item). Clearly express which problems you choose.

### Problems:

(a) We are given an arithmetic expression $x_1 o_1 x_2 o_2 ... x_{n-1} o_{n-1} x_n$ such that $x_i$ for $1 \leq i \leq n$ are positive numbers and $o_i \in \{+, \times\}$ for $1 \leq i \leq n-1$ are arithmetic operations (summation or multiplication). We would like to parenthesize the expression in a such a way that the value of the expression is <u>maximized</u>. For example, if the expression is $3 + 4 \times 2 + 6 \times 0.5$, then the optimal parenthesization is $(3 + 4) \times (2 + (6 \times 0.5))$, with a value of 35.

(b) We are given $n$ types of coin denominations with integer values $v_1$, $v_2$, ..., $v_n$. Given an integer $t$, we would like to compute the <u>minimum</u> number of coins to make change for $t$ (i.e., we would like to compute the minimum number of coins that add up to $t$, where repetitions are allowed). We know that one of the coins has value 1, so we can always make change for any amount of money $t$. For example, if we have coin denominations of 1, 2, and 5, then the optimal solution for $t = 9$ is 5, 2, 2.

(c) Given two strings $X =< x_1, x_2, ..., x_m >$ and $Y =< y_1, y_2, ..., y_n >$, the edit distance between $X$ and $Y$ is defined as the <u>minimum</u> number of edit operations (*replacement*, *insertion*, or *deletion* of a character) required to convert $X$ to $Y$. For example, the edit distance between $X$ = *esteban* and $Y$ = *stephen* is 4, comprising of 1 deletion ($e$), 1 insertion ($h$), and 2 replacements ($b \to p$ and $a \to e$). We would like to compute the edit distance between two given strings.

## Problem 2

We are given $n$ currencies and an exchange rate $r_{ij}$ for any pair of currencies $i$ an $j$. Namely, if we exchange 1 unit of currency $i$ with currency $j$, we receive $r_{ij}$ units of currency $j$. If we are given a source currency $s$ and a target currency $t$, then we can go through a path of different currencies to reach $t$ from $s$ so as to maximize our profit. The markets can also charge an exchange fee depending on the number of exchanges we make. For example if the exchange fee is $f(k)$ for making $k$ exchanges and we start with 1 unit of currency $s$, then the path of exchanges $s \to t$ will yield $r_{st} - f(1)$ units of currency $t$, whereas the path of exchanges $s \to i \to j \to t$ will yield $r_{si} \times r_{ij} \times r_{jt} - f(3)$ units of currency $t$. The problem is to find the sequence of exchanges that will maximize the amount of target currency $t$ we can obtain for a given source currency $s$.

(a) Define and prove the optimal substructure for this problem when there is no exchange fee ($f(k) = 0$ for all $k$).

(b) Prove that it is possible to find an exchange fee schedule $f(k)$ so that the optimal substructure you defined above will not hold anymore.