CS525DA
HW#5

DUE: Thursday, November 13

1. (10 points) EDIT DISTANCE Given two strings of characters $u = x_1 \ldots x_m$ and $v = y_1 \ldots y_n$ drawn from the alphabet $\{a, b, c\}$, we want to know the minimal cost of converting u to v, where $c_{ch} > 0$ is the cost of changing a symbol, $c_{ins} > 0$ is the cost of inserting a symbol, and $c_{del} > 0$ is the cost of deleting a symbol, and the cost of applying a sequence of operations is the sum of the costs of the operations that comprise the sequence. For example, if $c_{ch} = c_{ins} = c_{del} = 1$ and u=*abbaac* and v=*abcbc*, then the cost of converting u to v is 3 because of the sequence

$$abbaac \underset{c_{del}}{\rightarrow} abbac \underset{c_{ch}}{\rightarrow} abcac \underset{c_{ch}}{\rightarrow} abcbc \quad .$$

Write a dynamic programming algorithm to accept as input u, v, c_{ch}, c_{ins} and c_{del} and return the minimal cost of a sequence of operations to convert u to v.

2. (6 points) An instance of the PARTITION PROBLEM is a set U of integers, and the question is whether or not U can be partitioned into two sets T and $U \setminus T$ such that the sums of the integers in the two sets are equal. That is, does $\sum_{s \in T} s = \sum_{s \in U \setminus T} s = \frac{\sum_{s \in U} s}{2}$. For example, the instance $U = \{2,3,9,15,19\}$ admits the solution U={9,15}, and the instance $U = \{2,3,9,15,18\}$ does not admit a solution. Give a dynamic programming solution to the PARTITION PROBLEM, and analyze your solution. (Hint: Consider reducing this problem to a problem we have solved.)

CS525DA
HW#5 SOLUTIONS

1. For $0 \le i \le m$, $0 \le j \le n$, we let $\delta(i, j)$ denote the minimal cost of a sequence of operations to convert $x_1 \ldots x_i$ to $y_1 \ldots y_j$. The answer to the problem is $\delta(m,n)$. To start the process, we note that $\delta(0,0) = 0$, $\delta(1,0) = c_{del}$ and $\delta(0,1) = c_{ins}$. For $1 \le i \le m \wedge 1 \le j \le n$, if x_i=y_j, then $\delta(i, j) = \delta(i-1, j-1)$. Otherwise,
$\delta(i, j) = \min(\delta(i-1, j-1) + c_{ch}, \delta(i-1, j) + c_{del}, \delta(i, j-1) + c_{ins})$. The corresponding program to fill in array $\delta[0..m, 0..n]$ is

> $\delta[0,0] \leftarrow 0$
> $\delta[1,0] = c_{del}$
> $\delta[0,1] = c_{ins}$
> **for** $i \leftarrow 1$ **to** m
>   **for** $j \leftarrow 1$ **to** n
>   **if** x_i=y_j **then** $\delta[i, j] \leftarrow \delta[i-1, j-1]$
>   **else**
> $\delta[i, j] \leftarrow \min(\delta[i-1, j-1] + c_{ch}, \delta[i-1, j] + c_{del}, \delta[i, j-1] + c_{ins})$
> **return** $\delta[m,n]$

2. The instance $U = \{s_1, \ldots, s_n\}$ of the PARTITION PROBLEM ADMITS a solution if and only if the instance v=w=$(s_1, \ldots, s_n)$, $W = \frac{\sum_{1 \le i \le n} s_i}{2}$ of the KNAPSACK PROBLEM admits a packing of value W. The algorithm takes time in $\Theta\left(n \sum_{1 \le i \le n} s_i\right)$.