

CMSC 451: Dynamic Programming Examples Supplementary handout, Spring 2002

1. Please note that the correct formula for $\text{MaxV}(i)$ in the problem *Finding the Maximum Consecutive Subsequence*, should be $\text{MaxV}(i) = \max(\text{MaxV}(i-1), x_i, \text{SMaxV}(i-1)+x_i)$ instead of $\text{MaxV}(i) = \max(\text{MaxV}(i-1), \text{SMaxV}(i-1)+x_i)$ stated in the original handout.

2. In deriving the recurrence formula for the problem *Minimum Edit distance*, we consider two subcases: $x[i] = y[j]$ and $x[i] \neq y[j]$. For the first subcase, the handout simply states that $C[i, j] = C[i-1, j-1]$. Reducing the problem to transforming $x[1..i-1]$ to $y[1..j-1]$ is a valid reduction; this supplementary handout is going to explain why this will lead to an optimal solution.

For a valid transformation from $x[1..i]$ to $y[1..j]$, the last character of the transformed string must equal $y[j]$. If a transformation does not match $x[i]$ to $y[j]$, then there could be two subcases.

(1) $y[j]$ is added to the transformed string by insertion or by replacing some character $x[i']$ for some $i' < i$. Consider two further subcases: (a) $x[i]$ is not retained and is deleted and (b) $x[i]$ is retained and matched to $y[j']$ for some $j' < j$. For (a), the transformation is not optimal as the cost of transformation can be reduced by matching $x[i]$ to $y[j]$. For (b), the transformation will involve insertion of $y[j'+1], \dots, y[j]$ and matching $y[j']$ to $x[i]$. Instead of inserting $y[j]$ and matching $y[j']$ to $x[i]$, we could have a transformation of the same cost by first matching $y[j]$ to $x[i]$ and then inserting $y[j']$ through $y[j-1]$.

(2) $y[j]$ is matched to $x[i']$ for some $i' < i$. The transformation will involve deletion of $x[i'+1], \dots, x[i]$ and matching $y[j]$ to $x[i']$. Similar to subcase 1b, we could have a transformation of the same cost by first matching $y[j]$ to $x[i]$, then deletion of $x[i'], \dots, x[i-1]$.

Combining the two cases, we could see that there exists at least one optimal transformation that involves matching the character $x[i]$ to $y[j]$; hence $C[i, j] = C[i-1, j-1]$.

For the second subcase, ($x[i] \neq y[j]$), we only need to consider making a change at the end of the substring $x[1..i]$. The reason is that we must handle $x[i]$ somehow. It is either deleted, or modified into some character in y . In the latter case, either $x[i]$ is modified to $y[j]$ or $x[i]$ is modified to $y[j']$ for some $j' < j$ (which accounts for deletion of $y[j]$). Making changes somewhere else first do not save us the trouble or reduce the cost from dealing with $x[i]$.