CS/ECE 374 Fall 2018
Homework 6 Problem 2

Anqi Yao (anqiyao2@illinois.edu)
Zhe Zhang (zzhan157@illinois.edu)
Ray Ying (xinruiy2@illinois.edu)

Let $X = x_1, x_2, \ldots, x_r$, $Y = y_1, y_2, \ldots, y_s$ and $Z = z_1, z_2, \ldots, z_t$ be three sequences. A common *supersequence* of X, Y and Z is another sequence W such that X, Y and Z are subsequences of W. Suppose $X = a, b, d, c$ and $Y = b, a, b, e, d$ and $Z = b, e, d, c$. A simple common supersequence of X, Y and Z is the concatenation of X, Y and Z which is $a, b, d, c, b, a, b, e, d, b, e, d, c$ and has length 13. A shorter one is b, a, b, e, d, c which has length 6. Describe an efficient algorithm to compute the *length* of the shortest common supersequence of three given sequences X, Y and Z. You may want to first solve the two sequence problem to get you strated.

Solution:

Let $X[0...m-1]$, $Y[0...n-1]$, $Z[0...k-1]$ be the three sequences with lengths m, n, and k. Let len be the length of shortest supersequence. The base case is when all or some of the lengths are 0s. The recursion is to reduce the shortest supersequence of X, Y, and Z into the shortest supersequence of X, Y, Z's subsequences based on the values of their last elements. For example, if they have the same last element ($X[m-1] = Y[n-1] = Z[k-1]$), add 1 to len, and continue call the shortest supersequence function with input $X[0...m-2]$, $Y[0...n-2]$, $Z[0...k-2]$. Other cases would follow the same logic and they would be shown in the pseudo code in details.
The pseudo code is as following:

Let $X[0...m-1]$, $Y[0...n-1]$, $Z[0...k-1]$
SCS(X, Y, Z, m, n, k):

Base cases:
if($m == 0$ && $n == 0$ && $k == 0$) return 0
else if($m == 0$ && $n == 0$) return k
else if($m == 0$ && $k == 0$) return n
else if($n == 0$ && $k == 0$) return m
else if($m == 0$) let X be the shorter one of Y and Z
else if($n == 0$) let Y be the shorter one of X and Z
else if($k == 0$) let Z be the shorter one of X and Y

Recursion:
if($x[m-1] == Y[n-1] == Z[k-1]$) return $1+SCS(X,Y,Z,m-1,n-1,k-1)$
else if($x[m-1] == Y[n-1]$) return $1 + min(SCS(X,Y,Z,m-1,n-1,k), SCS(X,Y,Z,m,n,k-1))$
else if($x[m-1] == Z[k-1]$) return $1 + min(SCS(X,Y,Z,m-1,n,k-1), SCS(X,Y,Z,m,n-1,k))$
else if($Y[m-1] == Z[k-1]$) return $1 + min(SCS(X,Y,Z,m,n-1,k-1), SCS(X,Y,Z,m-1,n,k))$
else return $1 + min(SCS(X,Y,Z,m-1,n,k), SCS(X,Y,Z,m,n-1,k), SCS(X,Y,Z,m,n,k-1))$

The running time of this algorithm is approximately $O(3^n)$ because in the worst case we have three case for each sub-problem and the height of the recursion tree could be $O(n)$. Thus the total running time is $O(3^n)$. ■