Programming Assignment 4

Joni Vrapi

12/11/2022

Statement of Integrity: I, Joni Vrapi, attempted to answer each question honestly and to the best of my abilities. I cited any, and all, help that I received in completing this assignment.

Problem 1c.

```
PROCESS-SIGNAL($x, y, s$)
xSet = ySet = noiseSet = []
xMovingIndex = yMovingIndex = xCompleted = yCompleted = 0
for index = 0 to length(s)
    if s[index] is the i'th character of both x and y
        if x has been completed more than y
            ySet.append(index + 1)
            move y index by 1 through y
        elseif xCompleted == yCompleted
            if xMovingIndex is ahead of yMovingIndex
                xSet.append(index + 1)
                move x index by 1 through x
            else
                ySet.append(index + 1)
                move y index by 1 through y
        else
            xSet.append(index + 1)
            move x index by 1 through x
        continue

    if s[index] is the i'th character of x
        xSet.append(index + 1)
        move x index by 1 through x
        continue

    if s[index] is the i'th character of y
        ySet.append(index + 1)
        move y index by 1 through y
        continue

    if length(noiseSet != 0)
        assign to noiseSet an array of integers from 1 to length(s)
        that do not include any numbers that are in xSet or ySet

    if length(xSet) + length(ySet) + length(noiseSet) == length(s)
        this is an interweaving
```

Problem 1d. This algorithm processes the signal ”as it comes in” via a single for loop which iterates through the input string only once for an $O(n)$ time. There are many comparisons that are made, all of which are $O(1)$ operations. Finally, to get the noise in the signal, it generates another array of size $m < n$ in $O(m)$ time. In total, this is $O(n) + O(m)$ which is linear with respect to inputs, so this is an $O(n)$ algorithm.

Problem 2c. In three out of the four test cases, there was no noise, so per my analysis in Problem 1d, I would expect the length of the input to equal the number of iterations my program made. The instrumentation in tests $s1, s2, s4$ confirm this. The third test case $s3$ has 12 elements of noise, on an input of length 33. Per my analysis in Problem 1d, I would expect the total number of iterations to be $33 + 12 = 45$, and the instrumentation also confirms this. Therefore, we can conclude that this algorithm does in fact run in $O(n)$ time.