CS590 Assignment 4

1. **Abstract**

In this experiment, two versions of the Smith-Waterman algorithm are implemented. The first version of the algorithm is a dynamic approach, where the subproblems are solved using a nested for loop. Upon completion, the entire H and P tables are fully populated. The second approach to the algorithm uses the top-down approach, starting from the total problem, and recursively solving the subproblems at every step. Both algorithms yield the same results with theoretically similar performance.

1. **Results and Discussion**

Both algorithms correctly recreate all of the example problems given. The output of each algorithm running example 4 is shown below in figure 1. Only one example is shown for brevity.

|  |  |
| --- | --- |
| X = caacbdacca  X'= caacb-dacc-a  Y'= c--cbcd-ccba  Y = bccbcdccba  -----------------  M(n,m) = 9 | X = caacbdacca  X'= caacb-dacc-a  Y'= c--cbcd-ccba  Y = bccbcdccba  -----------------  M(n,m) = 9 |

**Figure 1.** “Example 4” output for bottom-up (left) and top-down memorized (right) implementations of the Smith-Waterman algorithm.

We can run our two algorithms on the provided input, finding the maximum alignment of X = “dcdcbacbbb” and Y = “acdccabdbb” as per question 4. The results can be seen below.

|  |  |
| --- | --- |
| X = dcdcbacbbb  X'= dcdcbacb-bb  Y'= acdcca-bdbb  Y = acdccabdbb  -----------------  M(n,m) = 11 | X = dcdcbacbbb  X'= dcdcbacb-bb  Y'= acdcca-bdbb  Y = acdccabdbb  -----------------  M(n,m) = 11 |

**Figure 2.** Question 4 output for bottom-up (left) and top-down memorized (right) implementations of the Smith-Waterman algorithm.

The resultant H and P tables are shown in figure 3, below. The path traced back in P is highlighted in green, with the start (11) and stop (0) shown in orange.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **H** | **-** | **a** | **c** | **d** | **c** | **c** | **a** | **b** | **d** | **b** | **b** | | **-** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | **d** | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | | **c** | 0 | 0 | 2 | 1 | 4 | 3 | 2 | 1 | 1 | 1 | 0 | | **d** | 0 | 0 | 1 | 4 | 3 | 3 | 2 | 1 | 3 | 2 | 1 | | **c** | 0 | 0 | 2 | 3 | 6 | 5 | 4 | 3 | 2 | 2 | 1 | | **b** | 0 | 0 | 1 | 2 | 5 | 5 | 4 | 6 | 5 | 4 | 4 | | **a** | 0 | 2 | 1 | 1 | 4 | 4 | 7 | 6 | 5 | 4 | 3 | | **c** | 0 | 1 | 4 | 3 | 3 | 6 | 6 | 6 | 5 | 4 | 3 | | **b** | 0 | 0 | 3 | 3 | 2 | 5 | 5 | 8 | 7 | 7 | 6 | | **b** | 0 | 0 | 2 | 2 | 2 | 4 | 4 | 7 | 7 | 9 | 9 | | **b** | 0 | 0 | 1 | 1 | 1 | 3 | 3 | 6 | 6 | 9 | 11 | | |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **P** | **-** | **a** | **c** | **d** | **c** | **c** | **a** | **b** | **d** | **b** | **b** | | **-** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | **d** | 0 | ↖ | ↖ | ↖ | ← | ← | ↖ | ↖ | ↖ | ← | ← | | **c** | 0 | ↖ | ↖ | ↑ | ↖ | ↖ | ← | ← | ↑ | ↖ | ↖ | | **d** | 0 | ↖ | ↑ | ↖ | ↑ | ↖ | ↖ | ↖ | ↖ | ← | ← | | **c** | 0 | ↖ | ↖ | ↑ | ↖ | ↖ | ← | ← | ↑ | ↖ | ↖ | | **b** | 0 | ↖ | ↑ | ↑ | ↑ | ↖ | ↖ | ↖ | ← | ↖ | ↖ | | **a** | 0 | ↖ | ← | ↑ | ↑ | ↖ | ↖ | ← | ↖ | ↖ | ↖ | | **c** | 0 | ↑ | ↖ | ← | ↖ | ↖ | ↑ | ↖ | ↖ | ↖ | ↖ | | **b** | 0 | ↑ | ↑ | ↖ | ↖ | ↑ | ↖ | ↖ | ← | ↖ | ↖ | | **b** | 0 | ↖ | ↑ | ↖ | ↖ | ↑ | ↖ | ↖ | ↖ | ↖ | ↖ | | **b** | 0 | ↖ | ↑ | ↖ | ↖ | ↑ | ↖ | ↖ | ↖ | ↖ | ↖ | |

**Table 1.** Question 4 resultant tables “H” (left) and “P” (right), with the traceback path colored.

The two algorithms function very similarly. The bottom-up dynamic approach solves every single subproblem, using a nested for loop. In light of the fact results in the lower right of the table depend on the results in the upper right, this approach is simple and straightforward. At every iteration of our nested for loop, all of the required values have already been previously calculated.

The top down approach, while still relatively simple, has a few considerations. The implementation of the algorithm requires memorization in order to be performant. This memorization can be implemented using our H table, adding a sentinel value in cells that have yet to be processed. For this, the constant “INT\_MIN” was selected, as there is no chance our calculated values will ever approach this. With this in place, the top-down SW algorithm is called on the bottom right of the table. When the algorithm examines the surrounding cells, if a sentinel value is encountered, the top-down SW algorithm is recursively called on that cell.  
 Overall, the two algorithms are expected to perform similarly, with a slight edge going to the bottom-up approach. Both algorithms have to run the sequence alignment for every cell, but the top-down implementation has a slight increase in overhead, in that it must check for the sentinel values every cell calculation. There is also a memory overhead, in that the recursive calls will build a call stack not required for the bottom-up approach.

1. **Exercises**

Please see the pdf.