STEVENS INSTITUTE OF TECHNOLOGY

Computer Science (Master's)

CS 590 A - ALGORITHMS
ASSIGNMENT 4

Submission to:
Professor In Suk Jang

Submission By:
Chetan Goyal
CWID: 20005334

Date: November 29, 2021

ANALYSIS

Two methodologies of dynamic programming utilizing Smith-Waterman calculation have been used for this assignment’s initial parts. The calculation computes the neighborhood arrangement of two given groupings using insertion and deletion depending upon the match.

The bottom-up approach is iterative and more efficient, but requires us to solve the smaller subproblems first then use the smaller subproblems values for the larger subproblems while the top-down approach is recursive though less efficient, and memoize solution for the return value and use it to reduce recursive calls thereby reducing time though the drawback which is not as significant, is that it increases space complexity as we need to sore the previous values.

The dynamic programming bottom-up approach is the optimal solution in matrix M, where M i,j represents the maximal similarity score for local alignment of the ax...i with by...j, where x, y > 0 are unknown and identified via traceback.

Exercise 15.1.1 - Part 4

In the below table, ‘d’ = Diagonal, ‘l’ = Left and ‘u’ = Up - denoting the direction followed while tracing back to find the common sequence for the path.

X’ = dcdcba-bbbb
Y’ = acdccaab-bb

P[n][m] Table

		a	c	d	c	c	a	b	d	b	b
	0	0	0	0	0	0	0	0	0	0	0
d	0	**d**	**d**	**d**	**l**	**l**	**d**	**d**	**d**	**l**	**l**
c	0	**d**	**d**	**u**	**d**	**d**	**l**	**l**	**u**	**d**	**d**
d	0	**d**	**u**	**d**	**u**	**d**	**d**	**d**	**d**	**l**	**l**

c	0	**d**	**d**	**u**	**d**	**d**	**l**	**l**	**u**	**d**	**d**
b	0	**d**	**u**	**u**	**u**	**d**	**d**	**d**	**l**	**d**	**d**
a	0	**d**	**l**	**u**	**u**	**d**	**d**	**u**	**d**	**d**	**d**
c	0	**u**	**d**	**l**	**d**	**u**	**u**	**d**	**d**	**d**	**d**
b	0	**u**	**u**	**d**	**d**	**u**	**d**	**d**	**l**	**d**	**d**
b	0	**d**	**u**	**d**	**d**	**u**	**d**	**d**	**d**	**d**	**d**
b	0	**d**	**u**	**d**	**d**	**u**	**d**	**d**	**d**	**d**	**d**

H[n][m] Table

		a	c	d	c	c	a	b	d	b	b
	0	0	0	0	0	0	0	0	0	0	0
d	0	**-1**	**-1**	**2**	**1**	**3**	**-1**	**-1**	**2**	**1**	**0**
c	0	**-1**	**1**	**1**	**4**	**3**	**2**	**1**	**1**	**1**	**0**
d	0	**-1**	**0**	**3**	**3**	**5**	**2**	**1**	**3**	**2**	**1**
c	0	**-1**	**1**	**2**	**5**	**4**	**4**	**3**	**2**	**2**	**1**
b	0	**-1**	**0**	**1**	**4**	**3**	**4**	**6**	**5**	**4**	**4**
a	0	**2**	**1**	**0**	**3**	**3**	**6**	**5**	**5**	**4**	**3**
c	0	**1**	**4**	**3**	**2**	**5**	**5**	**5**	**4**	**4**	**3**
b	0	**0**	**3**	**3**	**2**	**4**	**4**	**7**	**6**	**6**	**6**
b	0	**1**	**2**	**2**	**2**	**3**	**3**	**6**	**6**	**8**	**8**
b	0	**-1**	**1**	**1**	**1**	**1**	**2**	**5**	**5**	**8**	**10**

Exercise 15.1-2

I - number of cuts	1	2	3	4	5
Pi- price per inch	1	20	33	36	40
Pi/i- density	1	10	11	9	8

According to greedy strategy, without cutting the rod, the total value would be 40.

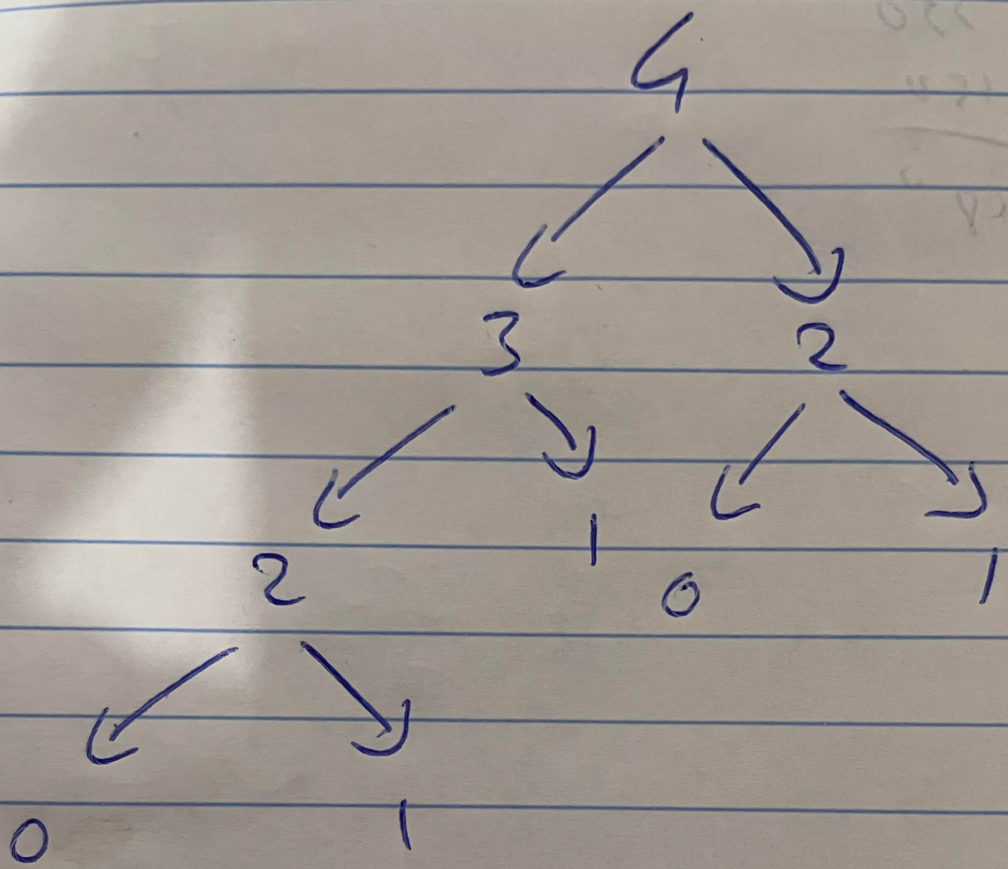
If we cut the rod into lengths of 4 inch and 1 inch, the value would be 38.

If we cut the rod into lengths of 3 inch and 2 inch, the value would be 53.

If we cut the rod into lengths of 3 inch and 2 inch and 1 inch, the value would be 42.
Hence the optimal solution is 53.

Exercise 15.1-5

```
FIBONACCI(n)

    Let fib[0..n] be a new array fib [0] = 1

    fib [1] = 1

    for i = 2 to n

        fib[i] = fib[i - 1] + fib[i - 2]

    return fib[n]
```


4
3
2
2
1
0
1
0
1

there are n+1 vertices in the sub problem graph, i.e. v0, v1, ..., vn - for v0, v1, each has 0 leaving edge.

- for v2, v3, ... , vn, each has 2 leaving edges.

Thus, there are 2n-n edges in the subproblem graph.

Exercise 15.4 - 1

LCS of {1, 0, 0, 1, 0, 1, 0, 1} and {0, 1, 0, 1, 1, 0, 1, 1, 0} is 6

		1	0	0	1	0	1	0	1
	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1
1	0	1	1	1	2	2	2	2	2
0	0	1	2	2	2	3	3	3	3
1	0	1	2	2	3	3	4	4	4
1	0	1	2	3	3	3	4	4	5
0	0	1	2	3	3	3	4	5	5
1	0	1	2	3	4	4	5	5	6
1	0	1	2	3	4	4	5	5	6
0	0	1	2	3	4	5	5	6	6

Answer - 0 1 0 1 0 1