CS590 homework 4

Sparsh Oza

Q4. Find the maximum alignment for X = dcdcbacbbb and Y = acdccabdbb by  
using the Smith-Waterman algorithm (see slides). Execute the pseudocode algorithm and fill the necessary tables H and P in a bottom-up fashion. Reconstruct the strings X‘ and Y‘ using the tables H and P.

**H Matrix:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | D | C | D | C | B | A | C | B | B | B |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | -1 | -1 | 2 | 1 | 0 | -1 | -1 | 2 | 1 | 0 |
| C | 0 | -1 | 1 | 1 | 4 | 3 | 2 | 1 | 1 | 1 | 0 |
| D | 0 | -1 | 0 | 3 | 3 | 3 | 2 | 1 | 3 | 2 | 1 |
| C | 0 | -1 | 1 | 2 | 5 | 5 | 4 | 3 | 2 | 2 | 1 |
| C | 0 | -1 | 0 | 1 | 4 | 4 | 4 | 6 | 5 | 4 | 4 |
| A | 0 | 2 | 1 | 0 | 3 | 3 | 6 | 5 | 5 | 4 | 3 |
| B | 0 | 1 | 4 | 3 | 2 | 5 | 5 | 5 | 4 | 4 | 3 |
| D | 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 | 2 | 2 | 5 | 5 | 8 | 10 |

**P Matrix:**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | D | C | D | C | B | A | C | B | B | B |
|  | - | - | - | - | - | - | - | - | - | - | - |
| A | - | d | d | d | l | L | D | d | d | l | l |
| C | - | d | d | u | d | D | L | l | u | d | d |
| D | - | d | u | d | u | D | D | d | d | l | l |
| C | - | d | d | u | d | D | L | l | u | d | d |
| C | - | d | u | u | u | D | D | D | l | d | d |
| A | - | d | l | u | u | D | D | u | d | d | d |
| B | - | u | d | l | d | D | U | d | d | d | d |
| D | - | u | u | d | d | U | D | d | l | d | d |
| B | - | d | u | d | D | U | D | d | d | d | d |
| B | - | d | u | d | D | U | D | d | d | d | d |

Exercise 15.1-2:

Show, by means of a counter example, that the following "greedy" strategy does not always determine an optimal way to cut rods. Define the density of a rod of length i to be pi/i, that is, its value per inch. The greedy strategy for a rod of length n cuts off a first piece of length i, where 1 <= i <= n, having maximum density. It then continues by applying the greedy strategy to the remaining piece of length n – i.

Table

Description automatically generated

In order to find the optimal solution locally, we try to work with a greedy strategy. We require a rod of length 5 as an illustration. As a result, we will chose the item with the highest price/weight ratio. So, we ultimately decide to go with Item 1, which can be as long as 1, Item 2, as long as 2, and the rod, which may be as long as 3. Furthermore, the cost is 6 + 10 = 16. This may not be the most ideal option, but it may be locally good. In order to achieve the greatest cuts, we use dynamic programming. In contrast to the greedy, where the length of 2 was squandered, dynamic programming prevents us from having any length left over.

Here, using dynamic programming, we'll pick a length of 2 and 3, adding them together to make 4, and a price gain of 10 + 12 = 22. As a result, whenever we need to cut a rod of length n, we do so using the height Pi/Wi ratio, which isn't the ideal option because we aren't able to employ fractional length values. Here, we pick the most pressing issue and then search for the best local solution for the subsequent subproblem. However, dynamic programming is used, where we first answer the subproblem before selecting the present.

**Exercise 15.1-5: The Fibonacci numbers are defined by recurrence (3.22). Give an O(n) time dynamic-programming algorithm to compute the n-th Fibonacci number. Draw the subproblem graph. How many vertices and edges are in the graph?**

Algorithm (FIBONACCI (x))

let F [0…..x+1] be new array

F[0] = 0 and F[1] =1

for (2 ≤ i ≤ x) do

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

return F[x]

Diagram

Description automatically generated

The above graph is for the value of n = 6.

Vertex: The subproblem graph has n + 1 vertices, which are 0, 1, 2, 3, 4, 5, and 6.

Edges: The subproblem graph contains 2n - 2 edges.

**Exercise 15.4-1**

**Determine an LCS of ⟨1,0,0,1,0,1,0,1⟩ and ⟨0,1,0,1,1,0,1,1,0⟩**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 0 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |
| 0 | 1 | 2 | 2 | 3 | 3 | 3 | 4 | 4 | 4 |
| 0 | 1 | 2 | 3 | 3 | 3 | 4 | 4 | 4 | 5 |
| 0 | 1 | 2 | 3 | 4 | 4 | 4 | 5 | 5 | 5 |
| 0 | 1 | 2 | 3 | 4 | 4 | 5 | 5 | 5 | 6 |
| 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 6 | 6 |

The LCS is (1,0,0,1,1,0) or (1,0,1,0,1,0).