****

CEN 504 Building Bridges Assignment

Authors: Bahadir Can YILDIZ

Lecturer: Prof. Dr. Aydin Ozturk

You are given a two-dimensional map with a river passing through its center horizontally. Given the *x*-coordinates of *n* cities on the southern bank of the river with *x*-coordinates *a*1, *a*2 ... *an* and *n* cities on the northern bank with *x*-coordinates *b*1, *b*2 ... *bn*. Clearly, for each city on the southern bank there is a corresponding city on the northern bank.Without loss of generality assume that the *x*-coordinates in the southern bank is given in increasing order.You are asked to build bridges between pair of cities in northern and southern banks of the river as many as possible such that no two bridges cross. In other words you must build as many non-crossing bridges as possible.When building bridges between cities, you can only connect city *i* on the northern bank to city *i* on the southern bank.

1. Design a brute force algorithm and find its running time.
2. Design a divide and conquer algorithm and find its running time.
3. Design a dynamic programming algorithm and find its running time.
4. Apply your algorithms on the following data

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *bi* | 0 | 8 | 4 | 12 | 2 | 10 | 6 | 14 | 1 | 9 | 5 | 13 | 3 | 11 | 7 | 15 |
| *ai* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

This problem is actually about having a subsequence that flows with both banks of the river. Since nodes of one of the banks will be put as incrementing, other bank should have an incrementing sequence for best case. This problem is actually about having longest incrementing subsequence of given array. So, our problems will turn into this;

1. Design a brute force algorithm for finding longest incrementing subsequence and find its running time.
2. Design a divide and conquer algorithm for finding longest incrementing subsequence and find its running time.
3. Design a dynamic programming algorithm for finding longest incrementing subsequence and find its running time.

## Solutions

1. Here are the segments of finding LIS with brute force;

* Finding all subsets of given array (2^n)
* Traversing subsets to eliminate non-incrementing subsets (n^2) (since subsets are hold in 2D array, nested loop is used.)
* Finding largest and optimum result depending on subset lengths and minimal entities (n^2)

Complexity = O(2^n) , Example given in “d” has been tested and approx. time of processing is 450 ms.

1. Here are the segments of finding LIS with divide & conquer.

* Dividing given array into two pieces and calling divide & conquer function with these arrays recursively (logn)
* Finding LIS of every array bits and combining them (used brute force in here so complexity will be 2^n)

Since departed arrays are used in brute force algorithm,

Complexity = O(2^logn) Example given in “d” has been tested and approx. time of processing is 4 ms.

1. Here are the segments of finding LIS with dynamic programming.

* Starting from the first element of the array, sequences are created depending on if next element is bigger or smaller than previous element. In detail, if next element is bigger, push it to the current segment. If next element is smaller, finish recent segment and start a new one with next element. Adding bigger elements to the previous segments will be provided with binary search (nlogn)
* Find biggest segment within created array (n)

Complexity = O(nlogn) Example given in “d” has been tested and approx. time of processing is 2 ms.

Implementation of these methods can be found at scripts/controller.js file. For working the code, extract files into the same folder and open index.html in a web browser.

HTML, CSS, JavaScript and Angular technologies have been used in this project.