Homework 10

Dushan Terzikj

due 07 May, 2018, 23:55 hours

Problem 1

Consider the array $A = (a1, a2, ...an)$. We call a subarray a succession of numbers in A where the position of any value in the subarray in the array is always bigger than the value of the previous one. Use dynamic programming to find a subarray of maximal ordered length of the array A. You can assume there are no duplicate values in the array and that there exists just one optimal solution.

Solution: Please check file *p1.cpp*.

Problem 2

Consider a triangle formed from n lines ($1 < n \leq 100$), each line containing natural numbers between $[1, 10000]$.

a) Use dynamic programming to determine the biggest sum of numbers existent between the road from the number in the first line and a number from the last line and print the respective road to the output. Each number in this road is seated to the left or to the right of the other value above it.

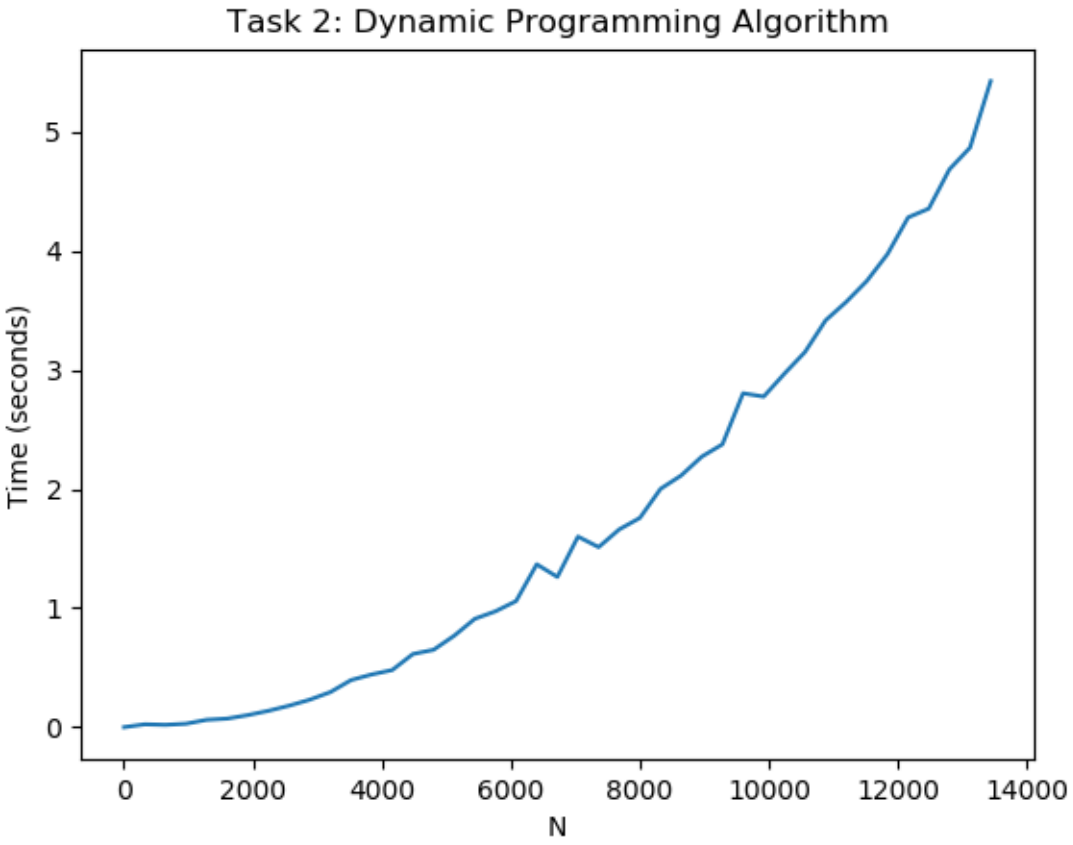
Solution: Please check file *p2a.cpp* and *algorithm.h*. Run the *p2a.cpp* file. The algorithm is in *algorithm.h*.

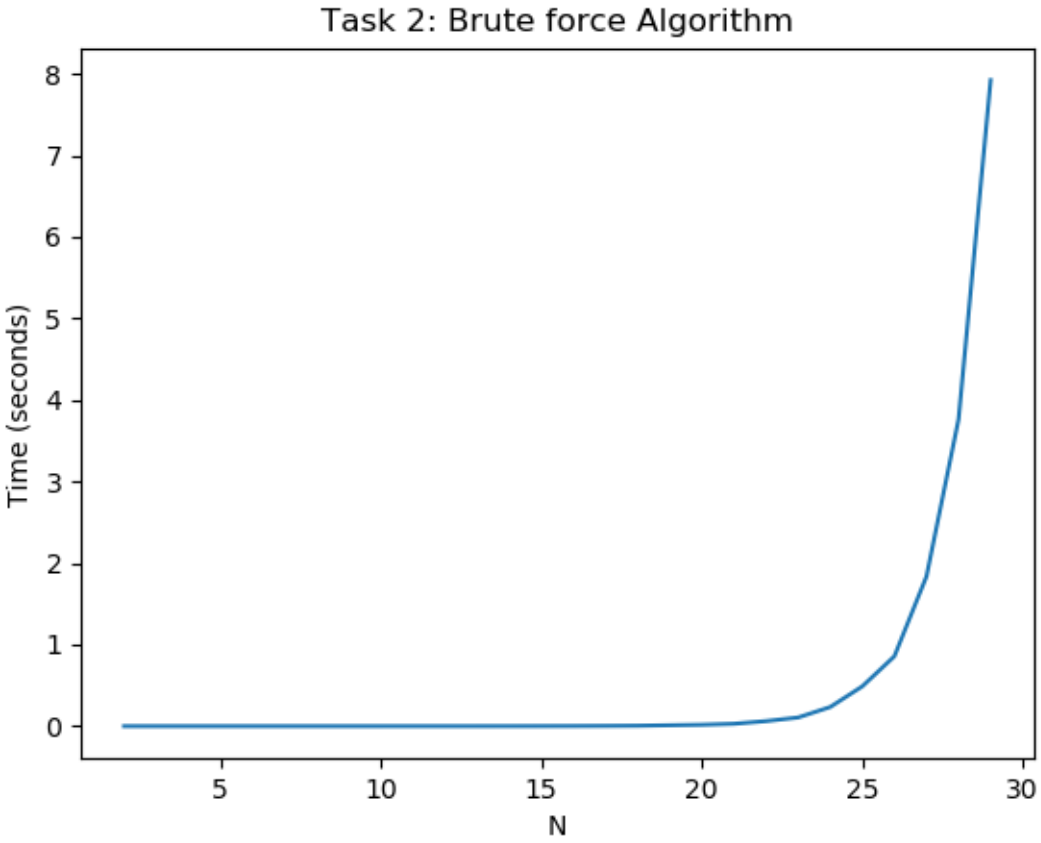
b) Analyze the runtime of your solution and compare it to a brute force approach.

Solution: Dynamic programming approach has a running time of $\Theta(g(n))$, where $g(n) = \frac{n(n+1)}{2}$. Since we do not consider constants and uneffective variables, we get total time complexity of $\Theta(n^2)$.

Bruteforce approach has a running time of 2^k, where $k = \frac{n(n-1)}{2}$. The reason for this is that every point in the triangles has two possible paths that can take, except the points on the lowest level. Since each point has 2 options and we have $k = \frac{n(n-1)}{2}$ points in total, that is 2^k operations.

Check the graphs for comparison.





c) Explain why a greedy algorithm does not work for this problem.

Solution: Greedy approach for this would be the following: starting from the top, always choose the next maximum point. The only problem here is that the solution is not always globally optimal.
Consider the example in the assignment. We go in the following manner: $7 \to 8 \to 1 \to 7 \to 5$. The result is 28, where the optimal solution is 30.

Problem 3

A scuba diver uses a special equipment for diving. He has a cylinder with two containers: one with oxygen and the other with nitrogen. Depending on the time he wants to stay under water and the depth of diving the scuba diver needs various amount of oxygen and nitrogen. The scuba diver has at his disposal a certain number of cylinders. Each cylinder can be described by its weight and the volume of gas it contains. In order to complete his task the scuba diver needs specific amounts of oxygen and nitrogen. Theoretically, the diver can take as many cylinders as he wants/needs. Use dynamic programming to find the minimal total weight of cylinders he has to take to complete the task and which those cylinders are. In case of several acceptable solutions, printing just one of them is enough.

Solution: Check file *p3.cpp*.