Date:16/04/2019

Title: Algorithm Complexity Analysis

Course: CS201

Section: 1

Assignment: 2

Name/ID: Zeynep Cankara/21703381

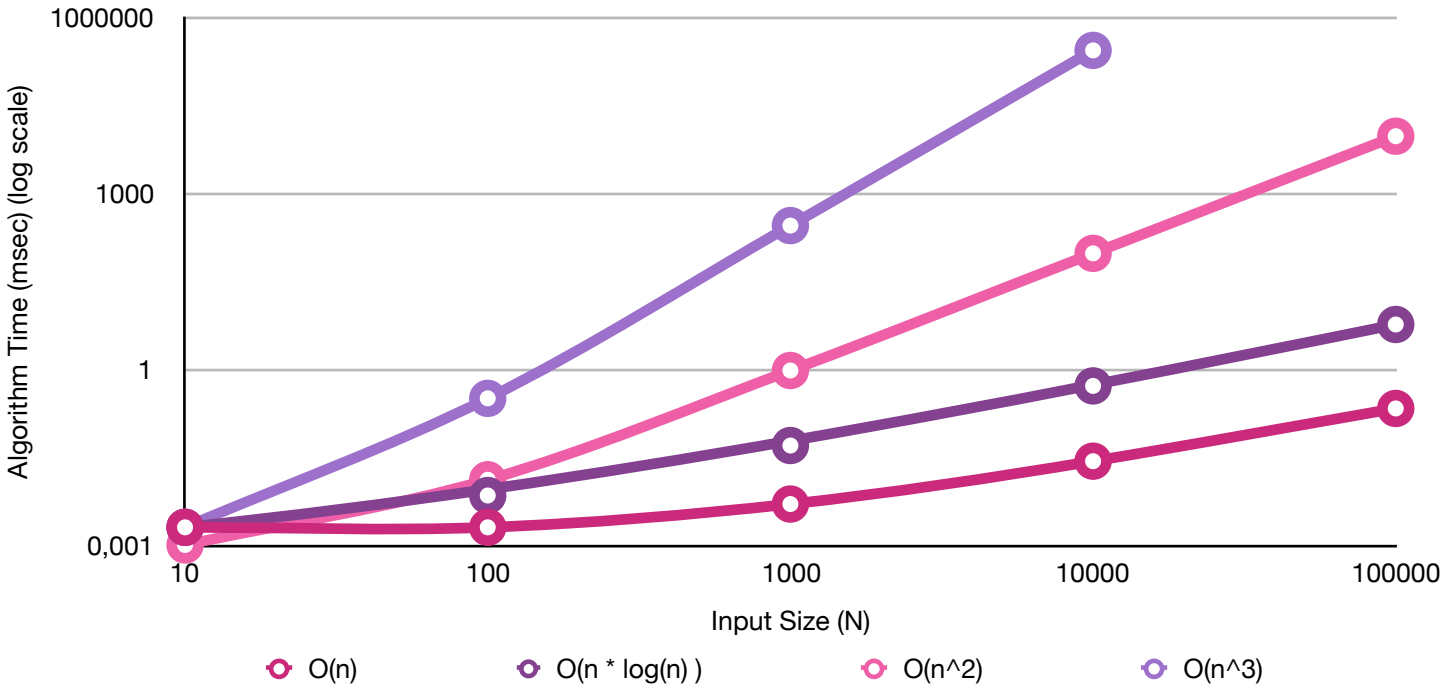
Description: Performing analysis on 4 possible solutions of the Maximum Subsequent Sum Problem

Question 1:

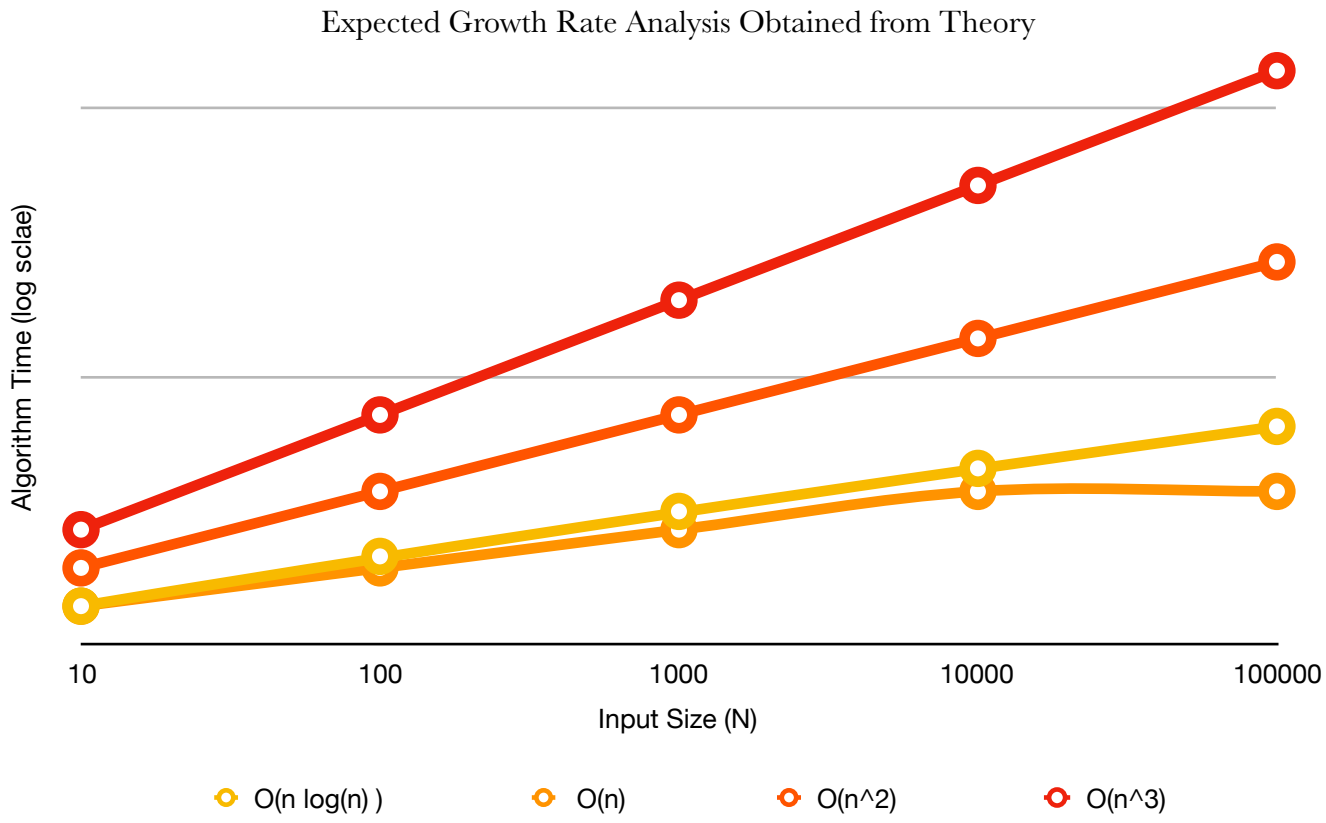
1. Table and Graph for Analyzing the Algorithm Efficiency obtained from Experiment

Algorithm Time (msec)

Input Size n	1 $O(n)$	2 $O(nlog(n))$	3 $O(n^2)$	4 $O(n^3)$
N = 10	0.002	0.002	0.001	0.002
N = 100	0.002	0.007	0.013	0.32
N = 1000	0.005	0.05	0.957	286.803
N = 10000	0.027	0.522	96.11	282833
N = 100000	0.215	5.852	9660.66	(exceed 10 min)



Question 2:



2. Comments on the Algorithm Analysis Plot

Trends of both experimental and theoretical result are close to each other. The arrays for the experiment generated with random integer data and same array generated being used across the 4 algorithms which have varying time complexities. The results are plotted in terms of milliseconds on y-axis (log scale). Even though we can't compare algorithms with small input size (smaller than 100) in practice, because of comping limitations for sensitive calculations, the difference between efficiency starts to become visible for inputs greater than 100. The efficiency of $O(nlog(n))$ algorithm seems to be much more worse off than $O(n)$ algorithm in the experimental setting when compared with theoretical result. This greater difference in the experiment might be the result of allocating extra memory during execution of $O(nlog(n))$ algorithm (It's the algorithm which use divide and conquer approach). Overall best performance obtained in algorithm 4 which uses only one for-loop to find maximum subsequent sum in place.

3.Computer Specifications:

OS: macOS Mojave

MacBook Pro (15-inch, 2018)

Processor: 2,2 GHz Intel Core i7

Memory: 16 GB 2400 MHz DDR4

Graphics: Intel UHD Graphics 630 1536 MB