MEF University
Department of Computer Engineering

COMP 303 Analysis of Algorithm

Course Project

Maximum Subarray Sum

In this project, you are given a one-dimensional array that may contain both positive and negative integers, find the sum of contiguous subarray of numbers which has the largest sum. For example, if the given array is {-2, -5, **6, -2, -3, 1, 5**, -6}, then the maximum subarray sum is 7 with the start and end indices 2 and 6, respectively.

1. Write pseudocode of the brute-force algorithm to find the subarray with a running time complexity of $\Theta(n^2)$. Analyze the complexity of your algorithm by considering the cost of each line. Then implement the algorithm in Python 3 and show that your program works correctly by considering the following three small arrays:

   A1 = {-2, -5, **6, -2, -3, 1, 5**, -6}

   A3 = {13,- 3,-25, 20, -3, -16, -23, **18, 20, -7, 12**, -5, -22, 15, -4, 7}

   Finally test the program on the arrays of varying sizes such as n = 10, 50, 100, 500, 1000, 5000, 10000, 50000,100000 (if possible). Measure the running time and plot it. Check whether the running time coincides with the theoretical complexity.
2. Repeat the same steps for an algorithm with a running time of $\Theta(n \lg n)$.
3. Repeat the same steps for an algorithm with a running time of $\Theta(n)$.
4. Prepare a report. Compare and Interpret the results of your experiments. Determine the problem size n_0 which gives the crossover point at which algorithm with the running time $\Theta(n\lg n)$ beats the algorithm with the running time $\Theta(n^2)$. In the project report, describe the problem, basic idea behind the algorithms, the running time analysis of the algorithms, give the list of your code, show that it works for the examples, compare the algorithms.