CMPT 706 — Algorithms for Big Data

Homework Assignment 2

Instructor: Igor Shinkar *Due date: February 10, 2020*

Submit your solutions, printed or written in readable handwriting, to the assignment boxes in CSIL.

Question 1 (20 points) *In this question you need to check if $N = 1729$ is prime.*

(a) Consider the Fermat's test for testing whether $N = 1729$ is prime. What will the test output for $a = 9$?

(b) Consider the Miller-Rabin test for testing whether $N = 1729$ is prime. What will it output for $a = 9$?

(c) Consider the Miller-Rabin test for testing whether $N = 1729$ is prime. What will it output for $a = 25$?

Is 1729 a prime number?

Question 2 (20 points) *Consider the Merge Sort algorithm we saw in class where the merging is done in linear time using an additional array.*

(a) What is the minimal *number of comparisons that the merge part performs on any two subarrays of length $n/2$ each?*

(b) What is the minimal *number of comparisons that the Merge Sort algorithm performs on any arrays of length 16? Show an example of an array achieving the* minimal *number of comparisons.*

Question 3 (20 points) *Consider the Merge Sort algorithm we saw in class where the merging is done in linear time using an additional array.*

(a) What is the maximal *number of comparisons that the merge part performs on any two subarrays of length $n/2$ each?*

(b) What is the maximal *number of comparisons that the Merge Sort algorithm performs on any arrays of length 16? Show an example of an array achieving the* maximal *number of comparisons.*

Question 4 (20 points) *Suppose that your input is an array A of length n such that each entry in the array is a number between 1 and 100. Design an algorithm that given such an array sorts it in time $O(n)$.*

Question 5 (20 points) *Suppose you are choosing between the following three algorithms:*

1. *Algorithm A solves problems of size n by dividing them into four subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.*
2. *Algorithm B solves problems of size n by recursively solving two subproblems of size $n - 1$, and then combining the solutions in constant time.*
3. *Algorithm C solves problems of size n by dividing them into eight subproblems of size $n/3$, recursively solving each subproblem, and then combining the solutions in $O(n^2)$ time.*

Use big-O notation to express the running times of each of theses algorithms. Which of them would you choose?