T17 A0244126M

Programming Assignment 2

Task 1:

(a) Do you need to implement Θ(n ^ 2.8073) Strassen's algorithm to solve A1? Explain your answer!
No. A simple compute-all-by-definition algorithm Θ(n ^ 3) can pass.

(**b) Do you need to implement Θ(n ^ 2.8073) Strassen's algorithm to solve A2? Explain your answer!**
No. Strassen's algo is not fast enough to pass, will get TLE.

(c) Do you need to implement Θ(n ^ 2.37188) by Duan, Wu, Zhou to solve A2? Explain your answer!
Yes. Although I haven't used this algo but this algo can pass.

(a) Is Freivald's algorithm of type Monte Carlo or Las Vegas algorithm?
Freivald's algo is Las Vegas algo.

(b) Assuming that you have ensured that "inner matrix dimensions agree", then we have:
1). Test cases where A · B = C (you should output "AC")
2). Test cases where A · B not = C (you should output "TLE")
Which test cases that Freivald's algorithm runs a bit faster vs its worst-case time complexity?
Which test cases that Freivald's algorithm is always correct vs may be wrong?
Worst-case time complexity of O(n ^ 2). The algo will run faster when the matrices have few non-zero entries.
Test case 2 is always correct, test case 1 can be wrong.

(c) The standard Freivald's algorithm has an error rate of 1/2 k .
What is your chosen k so that you can solve A2?
K = 10.

(d) How many submission(s), if you setup Freivald's algorithm as answered earlier, until you can solve A2 for the first time, in theoretical expectation?
To answer this question, please do not factor in potential implementation bugs...
I need 2 submission to pass.

Task 2:

(a) Are you able to use C++ std::sort/Java Collections.sort/Python list.sort() (all comparison-based sorts) that runs in O(n log n) per sort to solve B? Explain your answer!
I think we cannot use O(n logn) sorting algo as O(n logn) algo cannot pass TLE.

(b) Can you explain the meaning of the new way unsorted sequence U is generated this time?
The key difference is that previously the element of U is < C but now can be bigger. Requiring to use long long instead of int type.

(c) Are you able to use your PA1-B2/radix sort (or PA1-B1/counting sort) code runs that is supposed to run in linear time to solve B? Explain your answer!
Radix-sort/counting sort cannot pass the TLE.

(d) Do you use worst-case linear time select as discussed at the very end of Lecture 06 or the expected linear time QuickSelect as discussed on Week 07 tutorial?
I tried both but QuickSelect worked for me.

efg-2 If you use expected linear time QuickSelect, elaborate a bit of your implementation details.

1. **Does it actually pass the time limit of B?**
Yes.

2. Is it Monte Carlo or Las Vegas algorithm?
Las Vegas algo.

3. So what should you do if you get Time Limit Exceeded?
I got TLE and what I do is to reduce k (correctness – performance tradeoff)