CS 341, Fall 2017 A. Lubiw, B. Ma, E. Zima

ASSIGNMENT 2

DUE: Wednesday September 27, 7 PM. DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

Please read `http://www.student.cs.uwaterloo.ca/~cs341` for general instructions and policies.

NOTE: There is a programming question in a separate file.

1. [10 marks] **Divide-and-conquer.** One popular way to rank researchers is by their "h-index". A researcher's h-index is the maximum integer k such that the researcher has at least k papers that have been cited at least k times each. Suppose Professor X has written n papers and paper i has been cited c_i times. Suppose you have these sorted in an array C with $c_1 > c_2 > \ldots > c_n$. Give a divide-and-conquer algorithm to find Professor X's h-index. Your algorithm should run in time $O(\log n)$.

   As noted on the course web page, "giving" an algorithm means: describe the algorithm briefly in words, give high-level pseudocode, justify correctness, and analyze run-time.

2. [15 marks] **Squaring a matrix.**

   (a) [3 marks] Suppose you are given a 2×2 matrix A and you want to compute A^2. Show that you can do this with 5 multiplications.

   (b) [6 marks] Consider the following divide-and-conquer algorithm to compute A^2 when A is an $n \times n$ matrix: The algorithm is like Strassen's algorithm except that we get 5 problems of size $n/2$ (by part (a)) instead of getting 7 subproblems of size $n/2$ as in Strassen's algorithm. This gives a run-time of $O(n^{\log_2 5}) \approx O(n^{2.32})$—even better than Strassen's run-time of $O(n^{2.81})$.

   Explain what is wrong with the above algorithm and analysis.

   (c) [6 marks] Show that squaring matrices is in fact no easier than multiplying matrices. To do this, prove that if there is an algorithm with run time $O(n^c)$ that will square an $n \times n$ matrix, then there is an algorithm with run-time $O(n^c)$ that will multiply two $n \times n$ matrices. (In other words, you will be reducing the problem of multiplying matrices to the problem of squaring matrices.)