ESO207A: Data Structures and Algorithms

Homework *1* **Due: August 23, 2017, end of class.**

Instructions.

1. Start each problem on a new sheet. For each problem, Write your name, Roll No., the problem number, the date and the names of any students with whom you collaborated.
2. For questions in which algorithms are asked for, your answer should take the form of a short write-up. The first paragraph should summarize the problem you are solving and what your results are (time/space complexity as appropriate). The body of the essay should provide the following:
   (a) A clear description of the algorithm in English and/or pseudo-code, where, helpful.
   (b) At least one worked example or diagram to show more precisely how your algorithm works.
   (c) A proof/argument of the correctness of the algorithm.
   (d) An analysis of the running time of the algorithm.

   Remember, your goal is to communicate. *Full marks will be given only to correct solutions which are described clearly.* Convoluted and unclear descriptions will receive *low marks.*

Problem 1. Recurrence Relations. Assume $T(1) = 1$ as the base case for all the recurrences below.

1.1 Solve the following recurrence relations by unfolding the recursion tree. Assume n to be an appropriate power of 2.

a. $T(n) = 4T(n/2) + n \log n$.

b. $T(n) = 4T(n/2) + n^2$.

c. $T(n) = 2T(n/4) + \sqrt{n} \log n$.

1.2 Solve using a combination of the recursion tree method and the substitution method. Assume that for the base case, $T(n) = O(1)$, for $n \leq$ some constant. (*Hint:* First use the recursion tree method to obtain a good guess and then use the substitution method to prove that the guess is correct, up to $\Theta(\cdot)$.)

a. $T(n) = 7T(\lceil n/2 \rceil + 2) + n^2$

Problem 2. Hadamard Matrices The Hadamard matrices $H_0, H_1, H_2, \ldots$ are defined as follows.

1. H_0 is the 1×1 matrix $[1]$.

2. For $k \geq 1$, H_k is the $2^k \times 2^k$ matrix

$$H_k = \frac{1}{\sqrt{2}} \begin{bmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{bmatrix}$$

Show that if v is a column vector of length $n = 2^k$, then the matrix-vector product $H_k v$ can be calculated in $O(n \log n)$ operations. Assume that the numbers in v are small enough so that basic arithmetic operations like addition and multiplication take unit time.

(*P.S.* Another interesting property of the Hadamard matrices is that it is (real) orthonormal, that is, $H_k^T H_k = I$. As shown in the class, the DFT matrix is also unitary (complex orthonormal).)

Problem 3. Counting inversions–the Kendall Tau distance. A permutation is an array of n integers where each of the integers between 1 and n appears exactly once. The Kendall tau distance between the two permutations is the number of pairs that are in different order in the two permutations (inversions). ~~For example, the Kendall tau distance between 0 3 1 6 5 2 4 and 1 0 3 6 4 2 5 is 4 because the pairs (0, 1), (3, 1), (2, 4), (5, 4) are in different relative order in the two rankings but all other pairs are in the same relative order.~~

1. Design an efficient $O(n \log n)$ time algorithm for computing the Kendall tau distance between a given permutation and the identity permutation (that is, 0 1 $\ldots n-1$). (***Hint:*** Augment Merge-sort with inversion counting calculations.)

2. Extend the above algorithm to give an $O(n \log n)$ time algorithm for computing the Kendall tau distance between two permutations.

Problem 4. FFT Given two n-dimensional vectors $a = (a_0, a_1, \ldots, a_{n-1})$ and $b = (b_0, b_1, \ldots, b_{n-1})$, a wrap-around convolution is defined as an n-dimensional vector $c = (c_0, c_1, \ldots, c_{n-1})$ whose coordinates are defined as follows.

$$c_k = a_0 b_k + a_1 b_{k-1} + \ldots + a_k b_0 + a_{k+1} b_{n-1} + a_{k+2} b_{n-2} + \ldots + a_{n-1} b_{k+1}$$

Show how to evaluate this transform in $O(n \log n)$ time by viewing it as a convolution. (*Hint:* Express $\sum_{j=0}^{k} a_j b_{k-j}$ and $\sum_{j=k+1}^{n-1} a_j b_{n+k-j}$ as convolutions and add them.)