CS624: Analysis of Algorithms

Midterm exam 2 – practice

Instructor – Nurit Haspel

General Instructions

1. You may use up to 20 pages of hand written notes (or 10 double sided). No other printed/written material is allowed. No electronic devices are allowed.
2. The work is to be your own and you are expected to adhere to the UMass Boston honor system.
3. Write your answers in the available spaces, using the back of the page if needed. Write clearly and concisely and try to avoid cursive.
4. Please explain your answers if needed **but do it briefly**.
5. You may use any proof technique we showed in class or any other technique, as long as it constitutes a mathematical proof. Remember that a proof by example is generally good only to show that something is NOT true.
6. If you base your answer on a homework question state exactly which question it was.

Good Luck!

1. **Medians and Order Statistics:** (30%) Given two *sorted* arrays, A and B, each of size n. Describe an $O(\log n)$ worst case algorithm to find the median of all the 2n elements in A and B. Provide a brief but accurate runtime analysis (**Hint 1:** It's not unlike binary search... **Hint2:** Use the medians of A and B to guide you)
2. **Binary search trees:**
   (a) (15%). Let x be a leaf node in a binary search tree T. Let y be x's parent. Show that y.key is either the smallest key in T larger than x.key or the largest key in T smaller than x.key.
   (b) (15%) Is the following claim true or false? Explain: in order to determine whether two binary search trees are identical one has to perform an in-order walk on them and compare the results.
3. **Dynamic Programming:** Given an array A of n numbers, the maximum subarray problem is the task of finding the contiguous subarray A[i..j] of numbers which has the largest sum. For example, if $A = \{-2, 1, -3, \mathbf{4}, \textbf{-1}, \mathbf{2}, \mathbf{1}, -5, 4\}$ then the subarray that gives the maximum sum is $\{4, -1, 2, 1\}$ with sum 6 (emphasized in bold font). Let us define MS(i) as the maximum sum subarray that ends at A[i] (and must include A[i]). For example, in a 1-based index, – MS(1) $=\{-2\}$. MS(2) $= \{1\}$ (since concatenating -2 and 1 gives a smaller sum, so MS(2) includes only A[2]). In other words – for MS(i) we ask ourselves which one is better – for A[i] to extend MS(i-1) or be its own subarray.

(a) Show that the problem has the optimal substructure (Hint: A[i] either extends the maximum sub-array that ends in A[i-1] or alternatively, includes only A[i] itself. Use a cut-and-paste argument for MS(i-1) with respect to MS(i)).

(b) Define a recursive algorithm that calculates MS(i), that can be used as a basis for a dynamic programming calculation. Remember to also return the overall maximum sum. It doesn't have to be MS(n) (why?).

(c) Based on that, calculate MS(i) for every index in the array above.