Data Structure & Algorithms
CS210A
Semester I, 2014-15, CSE, IIT Kanpur

Theoretical Assignment I

Deadline : 21th August

Note: The exercises in this assignment are extensions of the problems which we discussed in full details in the lectures till now. Therefore, you are advised to fully internalize the concepts and fundamentals of the corresponding lectures before attempting these exercises. Make sincere attempts to solve these exercises. In case you are stuck, you may contact the instructor during office hours before 20th August to get some hint or pointer (no penalty for such hints). But make sure you think really hard enough for each of these problems. Your solution for each exercise must be formal and complete. In particular,

- The design of each algorithm or data structure must be provided in full details.
- You must provide analysis of time and space complexity of each algorithm and data structure.
- You must prove the correctness of each algorithm as well.

1 Local Minima in a Grid

In the class, we discussed an $O(n \log n)$ time algorithm for finding a local minima in a $n \times n$ grid of distinct numbers. A sketch for an $O(n)$ algorithm was also given. Now your job is to give full details of the $O(n)$ algorithm. Specifically, you are expected to provide the following details:

(a) A clean pseudo code

(b) Proof of correctness

(c) Analysis of time complexity

Be very careful in designing the algorithm. It has subtleties which were not covered in the class.

2 Unsolved Problem in Computer Science

In this section we will be solving a very interesting problem. It has three parts.

(a) Given a **sorted** array A storing n elements and a number x, give an $O(n \log n)$ algorithm to determine if there exist elements $a, b \in A$ such that $a + b = x$.

(b) Improve the solution of part (a) to give an $O(n)$ algorithm for the same problem

(c) Extend the solution of part(b) to solve a more general problem: Given a sorted array A storing n elements, design an $O(n^2)$ algorithm to find if there exist elements $a, b, c \in A$ such that $a + b = c$.

(Hint for (b): What happens if $A[0] + A[n-1] > x$?)
If you are able to find an $O(n^2)$ solution to part (c), you will realize that the solution is fairly simple, and yet this is the best known solution to this problem. Whether there exist a solution faster than $O(n^2)$ time is an open problem in CS, fascinating, isn't it!

3 Range-Minima Problem

Recall the range-minima problem discussed in full details in Lecture 5. We developed a data structure for this problem which occupies $O(n \log n)$ space and answers each query in $O(1)$ time. Suppose, our application where we want to deploy our Range-minima data structure is very conservative about space. Our aim is to decrease the space at the expense of increased query time. So you are asked to design a data structure which should occupy $O(n)$ space and should be able to answer any range-minima query in $O(\log n)$ time. The design of your data structure has to proceed along the following lines:
For an array of size n, the data structure discussed in the class takes $O(n \log n)$ space. How and to what extent should we shrink the array suitably so that this data structure takes $O(n)$ space. Note that you may spend $O(\log n)$ time per query instead of $O(1)$.