CSOR W4231–Spring, 2022
Eleni Drinea

Homework 1 (160 points)

Out: Friday, February 4, 2022
Due: 11:59pm, Friday, February 18, 2022

Homework Instructions.

1. For all algorithms that you are asked to "give" or "design", you should

   - Describe your algorithm clearly in English.
   - Give pseudocode.
   - Argue correctness; you may provide a formal proof or a convincing argument.
   - Provide, with an explanation, the best (smallest) upper bound that you can for the running time. All bounds should be **worst-case** bounds, unless explicitly stated otherwise.

   You are also encouraged to analyze the space required by your algorithm. We will not remove marks if you don't, unless the problem explicitly asks you to analyze space complexity.

2. **Full credit will be given to the fastest correct solution.** For example, an algorithm that solves the problem but runs in time $O(n^2)$ will not receive full marks if there is another algorithm that solves the problem and runs in $O(n)$ (possibly at the expense of some additional space).

3. You should submit this assignment as a **pdf** file to Gradescope. Other file formats will not be graded, and will automatically receive a score of 0.

4. I recommend you type your solutions using LaTeX. For every assignment, you will earn 5 extra credit points if you type your solutions using LaTeX or other software that prints equations and algorithms neatly. If you do not type your solutions, make sure that your hand-writing is very clear and that your scan is high quality.

5. You should write up the solutions **entirely on your own**. Collaboration is limited to discussion of ideas only. You should adhere to the department's academic honesty policy (see the course syllabus). Similarity between your solutions and solutions of your classmates or solutions posted online will result in receiving a 0 in this assignment, and possibly further disciplinary actions. There will be no exception to this policy and it may be applied retro-actively if we have reasons to re-evaluate this homework.

Homework Problems

1. (20 points) Give an efficient algorithm for the following problem:

   Input: A sorted array A of n distinct integers.

   Output: An index i such that $A[i] = i$, if one exists; -1, otherwise.

2. (32 points) Design an efficient algorithm for each of the following problems and give an upper bound for its worst-case running time.

   (a) **Input:** An *unsorted* array A of n integers.
   Output: Elements $x, y \in A$ such that $|x - y|$ is maximum.

(b) **Input:** A *sorted* array A of n integers.
Output: Elements $x, y \in A$ such $|x - y|$ is maximum.

(c) **Input:** An *unsorted* array A of n integers.
Output: Elements $x, y \in A$ such $|x - y|$ is minimum.

(d) **Input:** A *sorted* array A of n integers.
Output: Elements $x, y \in A$ such $|x - y|$ is minimum.

3. (25 points) An array A with n entries is said to have a *majority* element if more than half of its entries are the same. Given A, we want to find such a majority element, if one exists. A question of the form *"Is $A[i] = A[j]$?"* can be answered in constant time; however questions of the form *"Is $A[i] > A[j]$?"* are **not** permitted (for example, the elements of the array could be images so there is no order among them).

   Design an efficient algorithm to solve this problem in $\Theta(n \log n)$ time.

4. (28 points) Consider a complete undirected binary tree T on $n = 2^d - 1$ nodes, for some integer $d > 1$. Each node v in T is labeled with a real number x_v; all x_v are distinct. A node v in T is a *local minimum* if its label x_v is less than the label x_u for every node u joined to v by an edge.

   You are given such a complete binary tree T but the labeling is only specified in the following implicit way: for each node v, you can determine x_v by *probing* the node v. Give an efficient algorithm to find a local minimum of T. (You may assume that each probe takes constant time.)

5. (30 points) You're helping some security analysts monitor a collection of networked computers tracking the spread of an online virus. There are n computers in the system, labelled $C_1, C_2, \ldots, C_n$. You are given a trace indicating the times at which pairs of computers communicated. The trace consists of m triples (C_i, C_j, t_k); such a triple indicates that C_i and C_j communicated at time t_k. At this time, a virus could have spread from C_i to C_j (and vice versa).

   We assume that the trace holds the triples sorted by time. For simplicity, we assume that each pair of computers communicate at most once over the time of the trace. Also, it is possible to have pairs (C_i, C_j, t_k) and (C_i, C_ℓ, t_k); this would indicate that C_i communicated with both C_j and C_ℓ at time t_k allowing a virus to spread in any way among the 3 machines.

   We would like to answer questions of the following form: If a virus was introduced at C_i at time x, could it have spread to C_j at time y? That is, is there a sequence of communications that could have led from the virus moving from C_i to C_j?

   Design an algorithm that, given as input a collection of (sorted) trace data and a virus query, gives a yes/no answer to the query. The algorithm should run in time $O(m + n)$.

6. (25 points) Give an efficient algorithm that takes as input a directed graph $G = (V, E)$ and determines whether or not there is a vertex $s \in V$ from which all other vertices are reachable.

RECOMMENDED EXERCISES: Do NOT submit, they will not be graded. Solutions to the first three exercises will be provided.

1. Give tight asymptotic bounds for the following recurrences.

   - $T(n) = 4T(n/2) + n^3 - 1$.
   - $T(n) = 8T(n/2) + n^2$.
   - $T(n) = 6T(n/3) + n$.
   - $T(n) = T(\sqrt{n}) + 1$.

2. Show that, if λ is a positive real number, then $f(n) = 1 + \lambda + \lambda^2 + \ldots + \lambda^n$ is

   (a) $\Theta(1)$ if $\lambda < 1$.

   (b) $\Theta(n)$ if $\lambda = 1$.

   (c) $\Theta(\lambda^n)$ if $\lambda > 1$.

   Therefore, in big-Θ notation, the sum of a geometric series is simply the first term if the series is strictly decreasing, the last term if the series is strictly increasing, or the number of terms if the series is unchanging.

3. In the table below, indicate the relationship between functions f and g for each pair (f, g) by writing "yes" or "no" in each box. For example, if $f = O(g)$ then write "yes" in the first box. Here $\log^b x = (\log_2 x)^b$.

f	g	O	o	Ω	ω	Θ
$\log^2 n$	$6 \log n$					
$\sqrt{\log n}$	$(\log \log n)^3$					
$4n \log n$	$n \log (4n)$					
$n^{3/5}$	$\sqrt{n} \log n$					
$5\sqrt{n} + \log n$	$2\sqrt{n}$					
$n^5 4^n$	5^n					
$\sqrt{n} 2^n$	$2^{n/2+\log n}$					
$n \log (2n)$	$\frac{n^2}{\log n}$					
$n!$	2^n					
$\log n!$	$(\log n)^{\log n}$					

4. Recall Problem 3 above. Design an algorithm to solve this problem in $O(n)$ time.

5. You are given 12 coins and a scale with 2 pans that are balanced against each other. One of the coins is heavier or lighter than the rest. Identify this coin in just three weighings.

6. Read Sections 8.0 and 8.1 in your textbook on lower bounds for comparison-based sorting.