CSOR W4246–Fall, 2022
Eleni Drinea

Homework 1 Theoretical (115 points)

Out: Monday, September 19, 2022
Due: 11:59pm, Monday, October 3, 2022

Homework Instructions.

1. For all algorithms that you are asked to "give" or "design", you must do **all of the following** to get full credit:

   (a) Describe your algorithm clearly in English.

   (b) Give pseudocode.

   (c) Argue correctness, even if you don't give a formal proof and give a convincing argument instead.

   (d) Give with an explanation the best upper bound that you can for the running time.

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

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

3. 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.

4. **You should write up your 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 retroactively if we have reasons to re-evaluate your homework.

Homework Problems

1. (25 points)

   (a) (5 points) Suppose that when doing a Mergesort you recursively break lists into 3 equal-sized sublists (or as close to equal as possible) instead of 2. What is the asymptotic running time of this modified Mergesort?

   (b) (20 points) Give a (deterministic) algorithm that, on input a set S of n distinct integers and another integer x, determines whether or not there exist two elements in S whose sum is exactly x. The worst-case time complexity of your algorithm should be $\Theta(n \log n)$.

2. (20 points) Consider the following high-level description of a recursive algorithm for sorting. On input a list of n distinct numbers, the algorithm runs in three phases. In the first phase, the first $\lceil 2n/3 \rceil$ elements of the list are sorted recursively; the recursion bottoms out when the list has size 1 in which case you do nothing, or if the list has size 2, in which case you return the list if it is ordered, otherwise you swap the elements and return the resulting list. In the second phase, the last $\lceil 2n/3 \rceil$ elements are sorted recursively. Finally, in the third phase, the first $\lceil 2n/3 \rceil$ elements are sorted recursively again.

   Give pseudocode for this algorithm, prove its correctness and derive a recurrence for its running time. Use the recurrence to bound its asymptotic running time. Would you use this algorithm in your next application to sort?

3. (35 points) You want to measure how similar your musical preferences are to those of your classmates.

   To this end, you create a list of n songs and you order them as $\{s_1, s_2, \ldots, s_n\}$, where s_1 is your most preferred song and s_n your least preferred song. So your ranking for this ordered list of songs is $\{1, 2, \ldots, n\}$.

   Now you ask your classmates to also provide a ranking $\{x_1, x_2, \ldots, x_n\}$ for the ordered list $\{s_1, s_2, \ldots, s_n\}$. Intuitively, a classmate who provides a ranking in almost ascending order has similar musical tastes to you. For example, say you ordered 5 songs in the list $\{s_1, s_2, s_3, s_4, s_5\}$. Your ranking for these songs is $\{1, 2, 3, 4, 5\}$. If X ranks the songs as $\{1, 2, 3, 5, 4\}$ while Y ranks them as $\{5, 4, 3, 2, 1\}$, then clearly your preferences are quite similar to X but very different from Y. So one way to measure similarity between your preferences and those of your classmates is by checking how far your classmates' rankings are from being in ascending order.

   In a more abstract setting, you are given a sequence of n distinct numbers $x_1, \ldots, x_n$ and want to understand how far this sequence is from being in ascending order.

   One way to define a measure for this is by counting how many pairs of numbers x_i, x_j appear "out of order" in the sequence, that is, $x_i > x_j$ but x_i appears before x_j.

We will define the *disorder* of a sequence to be the number of pairs (x_i, x_j) such that $x_i > x_j$ but $i < j$. For example, if the input sequence is $\{2, 4, 1, 3, 5\}$, then the pairs $(2, 1)$, $(4, 1)$ and $(4, 3)$ are out of order and the *disorder* of the sequence is 3.

(a) (10 points) Give a brute force algorithm to compute the *disorder* of an input sequence of size n.

(b) (25 points) Can you give a faster algorithm to compute the *disorder*?

4. (35 points) *Clustering*, that is, grouping together "similar" data points, is an important task in machine learning. The first step for several clustering algorithms is to determine the pair of data points that are closest together (and place the pair in the same cluster). This question examines two algorithms for this step.

`Closest pair problem`

Input: a set of n points in the plane $\{p_1 = (x_1, y_1), p_2 = (x_2, y_2), \ldots, p_n = (x_n, y_n)\}$.

Output: the pair (p_i, p_j) with $p_i \neq p_j$ for which the euclidean distance between p_i and p_j is minimized. The euclidean distance $d(p_i, p_j)$ is given by $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

For simplicity, assume all x_i are distinct, all y_i are distinct and n is a power of 2.

(a) (10 points) Give the brute-force algorithm to solve this problem.

(b) (25 points) Now consider the following high-level description of a divide-and-conquer algorithm for this problem.

- Find a value x for which exactly half the points have $x_i < x$ and half have $x_i > x$. On this basis, split the points in two groups, L and R.
- Recursively find the closest pair in L and the closest pair in R. Say these pairs are (p_L, q_L) with $p_L, q_L \in L$, and (p_R, q_R) with $p_R, q_R \in R$, with distances d_L and d_R respectively. Let $d = \min\{d_L, d_R\}$.
- It remains to be seen whether there is a point in L and a point in R that are less than distance d apart from each other. To this end, discard all points with $x_i < x - d$ or $x_i > x + d$ and **sort** the remaining points by y coordinate.
- Now go through the sorted list and for each point compute its distance to the *seven* subsequent points in the list. Let (p_M, q_M) be the closest pair found in this way.
- The answer is one of the three pairs $(p_L, q_L), (p_R, q_R), (p_M, q_M)$, whichever has the smallest euclidean distance.

i. (15 points) Give the recurrence for the running time of this algorithm; **explain carefully your derivation**.

ii. (10 points) Solve the recurrence.

RECOMMENDED EXERCISES (do NOT submit, they will not be graded)

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
$6n \log^2 n$	$n^2 \log n$	Y	y	n	N	y
$\sqrt{\log n}$	$(\log \log n)^3$	n	n	y	y	n
$10n \log n$	$n \log (10n^2)$	y	n	y	n	y
$n^{4/5}$	$\sqrt{n} \log n$	n	n	y	y	n
$5\sqrt{n} + \log^2 n$	$3\sqrt{n}$					
$\frac{3^n}{n^3}$	$n^3 2^n$					
$\sqrt{n} 2^n$	$2^{n/2 + \log n}$					
$n \log (2n)$	$\frac{n^2}{\log n}$					
$n!$	n^n					
$\log n!$	$\log (n^{n/2})$					