Divide-and-conquer problem set

Please complete this problem set by September 29, 2022 at 11:59PM.

Problem 0 – Recurrences (20%)

Give upper ($O(\cdot)$) asymptotic bounds for the following recurrences. You may assume a $O(1)$ base case for small n. Justify your answer by some combination of the following: deriving how much total work is done at an arbitrary level k, how many levels there are, and how much work is required to merge (function body). For each recurrence, state whether or not it is top-heavy, bottom-heavy, or even work. Answers that only cite the Master theorem will not receive full credit.

1. $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$
2. $T(n) = 2T\left(\frac{n}{2}\right) + O(1)$
3. $T(n) = 7T\left(\frac{n}{2}\right) + O(n^3)$
4. $T(n) = 7T\left(\frac{n}{2}\right) + O(n^2)$
5. $T(n) = 4T\left(\frac{n}{2}\right) + O(n^2\sqrt{n})$
6. $T(n) = 4T\left(\frac{n}{2}\right) + O(n\log_2(n))$

Problem 1 – Covering a chess board (20%)

You are given a $2^k \times 2^k$ board of squares (e.g. a chess board) with the top left square removed. Prove, by giving a divide-and-conquer algorithm or argument, that you can exactly cover the entire board with L-shaped pieces (each covering 3 squares).

Problem 2 – Counting inverted pairs (20%)

You are given an unsorted list L that has $k \geq 0$ pairs of indices $i < j$ such that $L[i] > L[j]$. These are called *inverted pairs*. Develop an $O(n \log n)$ algorithm that counts the number of inverted pairs (i.e. compute the value k).

Problem 3 – Best subset problem (40%)

The *best subset problem* is defined as, given a list $(x_1, x_2, \ldots, x_n)$ of integers (which can be positive, negative, or zero), find (i, j) such that $x_i + x_{i+1} + \cdots + x_j$ is maximum for any $1 \leq i \leq j \leq n$. For example, if $n = 10$ and the input is $(4, -8, -5, 8, -4, 3, 6, -3, 2, -11)$ then the output is $x_4 + x_5 + x_6 + x_7 = 8 - 4 + 3 + 6 = 13$.

1. Develop an $O(n)$ algorithm for the related problem, *best subset middle* or BSM. The input to BSM is a list $(x_1, x_2, \ldots, x_n)$ of integers (which can be positive, negative, or zero) and the output is the maximum value of $x_i + x_{i+1} + \cdots + x_j$ such that $[i, j]$ spans $\frac{n}{2}$, in other words, for all possibilities for i and j such that $1 \leq i \leq \frac{n}{2} \leq j \leq n$.
2. Design a recursive algorithm for the best subset problem with runtime $O(n \log n)$ that uses the BSM function.
3. Argue that your algorithm is indeed correct and prove the runtime is $O(n \log n)$.
4. (Extra credit: 5pts) Design an algorithm for the best subset problem that has $O(n)$ runtime. Argue why your algorithm is correct and has $O(n)$ runtime.