Problem Set 3

Due date: Electronic submission of the pdf file of this homework is due on **2/10/2023 before 11:59pm** on ecampus.

Name: Huy Quang Lai

Resources. (All people, books, articles, web pages, etc. that have been consulted when producing your answers to this homework)

On my honor, as an Aggie, I have neither given nor received any unauthorized aid on any portion of the academic work included in this assignment. Furthermore, I have disclosed all resources (people, books, web sites, etc.) that have been used to prepare this homework.

Signature: ____________________________

Problem 1 (20 points)**.** Use mathematical induction to show that when n is an exact power of 2, the solution of the recurrence

$$T(n) = \begin{cases} 2 & \text{if } n = 2, \\ 2T(n/2) + n & \text{if } n = 2^k \text{ for } k > 1, \end{cases}$$

is $T(n) = n \log_2 n$.

Solution. Base Case:
Let $k = 1$
$T(2) = 2$
$2 \log_2 2 = 2$
The relationship holds when $n = 2$

Inductive Case:
Assume that $T(n) = n \log_2 n$ if $n = 2^k, \forall k > 1$
If $n = 2^{k+1}$, then:

$$\begin{aligned} T(2^{k+1}) &= 2T(2^{k+1}/2) + 2^{k+1} \\ &= 2T(2^k) + 2^{k+1} \\ &= 2\left(2^k \log_2\left(2^k\right)\right) + 2^{k+1} && \text{Inductive Hypothesis} \\ &= 2^{k+1} \log_2(2^k) + 2^{k+1} \\ &= 2^{k+1}\left(\log_2(2^k) + 1\right) \\ &= 2^{k+1} \log_2(2^{k+1}) \end{aligned}$$

By the inductive hypothesis, $T(n) \equiv n \log_2 n$
□

Problem 2 (20 points)**.** We can express insertion sort as a recursive procedure as follows. In order to sort $A[1..n]$, we recursively sort $A[1..n-1]$ and then insert $A[n]$ into the sorted array $A[1..n-1]$. Write a recurrence for the running time of this recursive version of insertion sort.

Solution. There are two steps in the algorithm.

1. sort the sub-array $A[1..n-1]$
2. inserting $A[n]$ into the sorted sub-array.

For $n = 1$, step 1 does not take time since the sub-array is empty. Step 2 takes constant time.
As a result, the algorithm runs in $\Theta(1)$.
For $n > 1$, step 1 calls for the sorting of $A[1..n-1]$. Step 2 takes $\Theta(n)$ since, on average, $A[n]$ will be inserted in the middle of the array.

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1. \\ T(n-1) + \Theta(n) & \text{if } n > 1. \end{cases}$$

Problem 3 (20 points)**.** V. Pan has discovered a way of multiplying 68×68 matrices using $132,464$ multiplications, a way of multiplying 70×70 matrices using $143,640$ multiplications, and a way of multiplying 72×72 matrices using $155,424$ multiplications. Which method yields the best asymptotic running time when used in a divide-and-conquer matrix-multiplication algorithm? How does it compare to Strassen's algorithm?

Solution. For m sub-problems and k multiplications, the recurrence relationship is as follows

$$T(n) = kT\left(\frac{n}{m}\right)$$

Using the master theorem, $T(n) = \Theta\left(n^{\log_m k}\right)$

$\log_{68}(132464) \approx 2.79512848736$
$\log_{70}(143640) \approx 2.79512268975$
$\log_{72}(155424) \approx 2.79514739109$
$\log_2 7 \approx 2.80735492206$
$\because \log_2 7 > \log_{72}(155424)$,
$\therefore$ This algorithm runs asymptotically faster than Stranssen's Algorithm

Problem 4 (20 points)**.** Show how to multiply the complex numbers $a + bi$ and $c + di$ using only three multiplications of real numbers. The algorithm should take a, b, c, and d as input and produce the real component $ac - bd$ and the imaginary component $ad + bc$ separately.

Solution. Using Karatsuba's Algorithm, the three multiplications for calulcate the product would be

1. $S_0 = ac$
2. $S_1 = bd$
3. $S_3 = (a + b)(c + d)$

Using these three products, the final product can be determined as follows.

$$A = S_1 - S_2, B = S_3 - (S_1 + S_2)$$

Problem 5 (20 points)**.** Use the master method to show that the solution to the binary-search recurrence

$$T(n) = T(n/2) + \Theta(1)$$

is $T(n) = \Theta(\lg n)$. Clearly indicate which case of the Master theorem is used.

Solution. $a = 1, b = 2, f(n) = \Theta(1)$
$\because f(n) = \Theta\left(n^{\log_b a}\right)$
$\therefore$ Case II of the Master Theorem is used.

$$T(n) = \Theta\left(n^{\log_b a} \log_2 n\right) = \Theta(\log_2 n)$$