National University of Computer and Emerging Sciences
Karachi Campus

Design and Analysis of Algorithms (CS2009)

Mid-1 Exam

Total Time (Hrs): 3
Total Marks: 12.5
Total Questions: 5

Date: Sep 23rd 2024

Course Instructor(s)

Dr.Muhammad Atif Tahir, Dr.Nasiruddin, Dr.Kamran,Dr. Fahad Sherwani, Dr. Farrukh Salim Ms.Aanum,Mr.Syed Faisal Ali,Mr.Sandesh,Mr.Minhal Raza ,Mr. Abu Zohran

_____________ ____________ _________________________

Roll No Section Student Signature

Do not write below this line

Attempt all the questions.

CLO #1: To apply acquired knowledge to solve computing problems complexities and proofs

---

Question 1

Marking Scheme:
Strict Marking. Case is correctly identified Either 0 or 0.5 for both

(a): Solve the following recurrence relations by using the Master Theorem [1.5 marks]

a) $T(n) = 100T\left(\frac{n}{10}\right) + n^2 \log n + n^2 + 1$

Solution:

$$a = 100, b = 10, f(n) = \theta(n^2 \log n)$$
$$since\ n^2 = n^{\log_{10} 100}$$
$$Therefore$$
$$T(n) = \theta(n^2 \log^2 n)$$

b) $T(n) = 2T\left(\frac{n}{4}\right) + 3T\left(\frac{n}{2}\right) + n$

Solution:

It has two recurrence branches so we can't apply standard Master Theorem

c) $T(n) = 3T\left(\frac{n}{3}\right) + \sqrt{n} + 1$

Solution:

$$\boldsymbol{a = 3, b = 3\ and\ d} = \mathbf{1/2}$$
$$Since\ a > b^d$$
$$Therefore$$
$$T(n) = \boldsymbol{\theta}\left(n^{\log_3 3}\right) = \boldsymbol{\theta}(n)$$

(b): Solve the following recurrence relations by using the Substitution Method [2 marks]

Marking Scheme:
1 for correct answer with some minor errors
0.25-0.75 for partial attempt

a) $T(n) = 3T\left(\frac{n}{2}\right) + n^2$; apply for (i) $T(n) = O(n2)$ or (ii) $T(n) = O(nlogn)$

$$(i)$$

Let Suppose that $T(n) = O(\boldsymbol{n^2})$

Therefore $\boldsymbol{T(n) = 3\ c\left(\frac{n^2}{4}\right) + n^2}$

According to Assumption $T(n) = O(\boldsymbol{n^2}) = \boldsymbol{cn^2}$

$$\boldsymbol{cn^2 = 3\ c\left(\frac{n^2}{4}\right) + n^2}$$
$$\boldsymbol{\frac{cn^2}{4} = n^2}$$
$$\boldsymbol{\frac{cn^2}{4} = n^2}$$

$$Since\ from\ the\ above\ equation\ c = 4\ which\ positive\ value\ Hence\ our\ is\ correct$$

$$(ii)$$

Let Suppose that $T(n) = O(\boldsymbol{n\ logn})$

Therefore $\boldsymbol{T(n) = 3\ c\left(\frac{n^2}{4}\right) + n^2}$

According to Assumption $T(n) = O(\boldsymbol{n\ logn}) = \boldsymbol{cn\ logn}$

$$\boldsymbol{cn\ logn = 3\ c\left(\frac{n}{2}log\frac{n}{2}\right) + n^2}$$
$$\boldsymbol{\frac{cnlogn}{2} = 3c\left(\frac{n}{2}(logn - 1)\right) + n^2}$$
$$\boldsymbol{cnlogn = 3cnlogn - 3cn + n^2}$$
$$\boldsymbol{3cn - 2cnlogn = n^2}$$

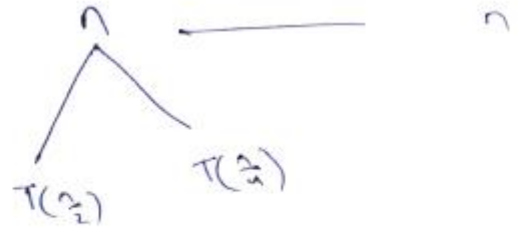
$$Since\ the\ left\ hand\ side\ will\ be\ negative\ for\ large\ values\ of\ n. Hence\ our\ guess\ is\ wrong.$$

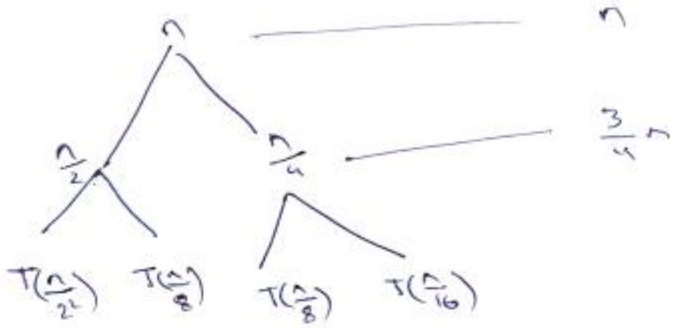
(c): Solve the following recurrence relations by using the Recursion Tree or Iterative Method [2 marks]

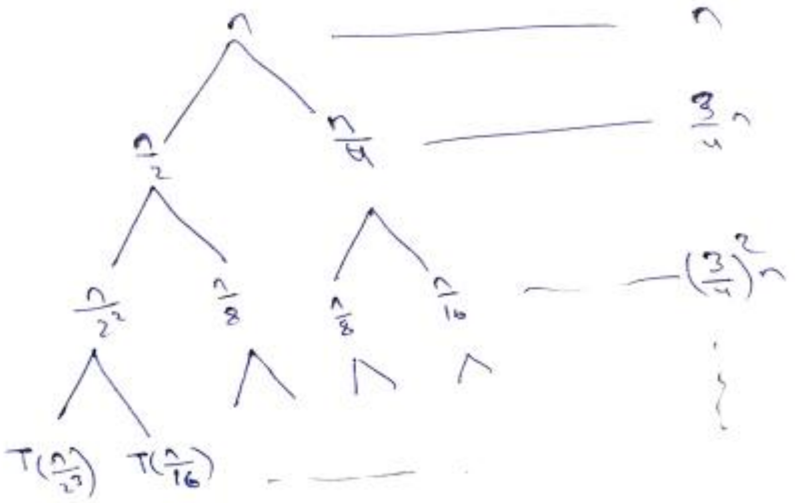
Marking Scheme:
1 for correct answer with some minor errors
0.25-0.75for partial attempt

a) $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + n$

Solution:







$$T(n) = n + \frac{3}{4}n + \left(\frac{3}{4}\right)^2 n + \cdots$$

$$= n\left(\frac{1}{1-\frac{3}{4}}\right) = 4n$$

$$T(n) = O(n)$$

b) $T(n) = 2T\left(\frac{n}{2}\right) + n^2$

Marking Scheme:
1 for correct answer with some minor errors

0.25-0.75for partial attempt

$$T(n) = 2(2T\left(\frac{n}{2^2}\right) + \frac{n^2}{2^2}) + n^2$$
$$T(n) = 2^2T\left(\frac{n}{2^2}\right) + \frac{n^2}{2^1} + n^2$$
$$T(n) = 2^2(2T\left(\frac{n}{2^3}\right) + \frac{n^2}{2^4}) + \frac{n^2}{2^1} + n^2$$
$$T(n) = 2^3T\left(\frac{n}{2^3}\right) + \frac{n^2}{2^2} + \frac{n^2}{2^1} + n^2$$
$$T(n) = n^2 + \frac{n^2}{2^1} + \frac{n^2}{2^2} + \frac{n^2}{2^3} + \frac{n^2}{2^4} \dots\dots.$$
$$T(n) = n^2\left(\frac{1}{1-\frac{1}{2}}\right)$$
$$T(n) = 2\,n^2$$
$$T(n) = \theta(n^2)$$

(d): Prove using following loop invariant that the following function correctly sums the elements in the array passed to it.

Marking Scheme: [1.5 marks]
1.5 for correct answer discussing all three parts with 0.5 allocated for each (initialization, maintenance, termination)
0.25-1for partial attempt

Loop Invariant: At the start of each iteration of the loop, sum is the sum of the first i elements of A, and i is an integer such that 0 <= i <= n.

```
def sum(A, n):
    sum = 0
    i = 0
    while i < n:
        sum = sum + A[i]
        i = i + 1
        return sum
```

Solution:

Invariant: At the start of each iteration of the loop, sum is the sum of the first i elements of A, and i is an integer such that 0 <= i <= n.

Initialization: Before the loop starts, sum is 0 and i is 0. The invariant holds because the sum of the first 0 elements is 0.

Maintenance: If the invariant holds before an iteration of the loop, it holds after the iteration because sum is updated to include A[i] and i is incremented by 1.

Termination: The loop terminates when i equals n. At this point, sum is the sum of the first n elements of A

CLO #2: To analyze complexities of different algorithms using asymptotic notations, complexity classes and standard complexity function

Question 2 : Prove the following statements. If cannot proof, write FALSE while doing proof [1.5 marks]

Marking Scheme:

0.5 for correct answer with no errors
0.-0.25for partial attempt

1. $\boldsymbol{n^3 + 2^n = O(2^n)}$

$\boldsymbol{Solution}$:

Solution

Set up the definition: We need to show:

$$n^3 + 2^n \leq C \cdot 2^n \quad \text{for all } n \geq n_0$$

Choose C:

- 2^n dominates n^3 for large n. Specifically:

$$n^3 \leq (C-1) \cdot 2^n$$

- For large n, choose $C = 2$:

$$n^3 \leq 2^n$$

- For $n \geq 10$, this holds true.

2. $\boldsymbol{\frac{n^2+4}{2n^2+3n+1} = \theta(1)}$

Solution

$$c_1 \le \frac{n^2+4}{2n^2+3n+1} \le c_2 \quad \text{for all } n \ge n_0$$

- For large n, the ratio approximates:

$$\frac{n^2+4}{2n^2+3n+1} \approx \frac{n^2}{2n^2} = \frac{1}{2}$$

Find constants:

- For sufficiently large n, $\frac{n^2+4}{2n^2+3n+1}$ is bounded between two constants. Specifically:

$$\frac{1}{3} \le \frac{n^2+4}{2n^2+3n+1} \le \frac{1}{2}$$

Conclusion:

- $\frac{n^2+4}{2n^2+3n+1} = \Theta(1)$ with $c_1 = \frac{1}{3}$, $c_2 = \frac{1}{2}$, and sufficiently large n_0.

3. $\boldsymbol{n^2 logn = \theta(n^2)}$

   Solution:

   Since the two functions are of different growth and it is not possible to show that $n^2 logn = O(n^2)$ for large values of n. Therefore, this statement is False.

CLO #4: To construct and analyze real world problems solutions using different algorithms design techniques

---

Question 3: Suppose you are given an array A with n entries, with each entry holding a distinct number. You are told that the sequence of values $A[1], A[2], \ldots, A[n]$ is unimodal: For some index p between 1 and n, the values in the array entries increase up to position p in A and then decrease the remainder of the way until position n. You'd like to find the "peak entry" p without having to read the entire array in fact, by reading as few entries of A as possible. Show how to find the entry p by reading at most $O(log\ n)$ entries of A. [4 marks]

Marking Scheme:
4 Points O(logn) for correct solution using Binary Search Solution with Proper Explanation for Constant term.
3.0 Points for correct solution without Constant factor explanation
1.5-2.75 for Partial Correct Solution
0-1.5 for some attempt

Solution:

Binary Search Problem (to find index of the largest number)
Solution: We can view this as a divide-and-conquer approach: for some constant c > 0, we perform at

most c operations and then continue recursively on an input of size at most n/2. As in the chapter, we will assume that the recursion "bottoms out" when n = 2, performing at most c operations to finish the computation. If T (n) denotes the running time on an input of size n, then we have the recurrence

$T(n) = T(n/2) + c$, where $c \geq 2$