University of Colombo School of Computing

SCS 1308 - Foundations of Algorithms

Tutorial - 02

01. Use the recursion tree method to find an asymptotic upper bound for the given recurrence forms.

A.
$$T(n) = \begin{cases} T(n/2) + cn^2 & n \geq 2 \\ c & n = 1 \end{cases}$$

B.
$$T(n) = \begin{cases} 3T(n/4) + cn^2 & n \geq 4 \\ 1 & n = 1 \end{cases}$$

C.
$$T(n) = \begin{cases} 2T(n/2) + 1 & n \geq 2 \\ 1 & n = 1 \end{cases}$$

D.
$$T(n) = \begin{cases} 3T(n/2) + cn & n \geq 2 \\ 1 & n = 1 \end{cases}$$

E.
$$T(n) = \begin{cases} 7T(n/2) + cn^2 & n \geq 2 \\ 1 & n = 1 \end{cases}$$

I. For the above each recurrence

A. Draw the tree
B. Compute depth
C. Compute cost per level
D. Sum and produce the final upper bound

II. After drawing the recursion tree for the given recurrence:

A. Explain how the **cost at each level** changes as the tree goes deeper.
B. Identify which part of the tree (top, middle, or bottom) **dominates the total cost**, and justify your reasoning.

02. Draw a recursion tree for the following recurrences and use it to obtain Big O.

A. $T(n) = T(n/5) + T(4n/5) + n$
B. $T(n) = T(n/3) + T(2n/3) + n$
C. $T(n) = T(n-1) + \log n$
D. $T(n) = 2T(n-1) + 1$