

# CS220/SE220 Design and Analysis of Algorithms

School of Computer Science and Engineering  
Macau University of Science and Technology

## Assignment 1

Due Day: 7 March 2024 (Thursday)

1. Use the **substitution method** to show that each of the following recurrences defined on the reals has the **asymptotic solution** specified. **You have to show both the upper ( $O$ ) and lower ( $\Omega$ ) bounds of  $T(n)$ .**
  - a.  $T(n) = 2T(n/3) + \Theta(n)$  has solution  $T(n) = \Theta(n)$ .
  - b.  $T(n) = 4T(n/2) + \Theta(n)$  has solution  $T(n) = \Theta(n^2)$ .
2. For each of the following recurrences, sketch its **recursion tree**, and guess a good **asymptotic upper bound** on its solution. **You DO NOT need to use the substitution method to verify your answer.**
  - a.  $T(n) = T(n/3) + \Theta(n^3)$ .
  - b.  $T(n) = 4T(n/3) + \Theta(n)$ .
3. Use the **master method** to give **tight asymptotic bounds** for the following recurrences:
  - a.  $T(n) = 2T(n/4) + \sqrt{n} \lg^2 n$ .
  - b.  $T(n) = 2T(n/2) + n^3$ .

problem 1

a. proof

Upper bound

Guess:  $T(n) \leq dn$  ( $d > 0$ ,  $d$  is constant)

By definition of  $\Theta$ -notation

$\exists c_1, c_2$  are positive constants, such that

$c_1 g(n) \leq f(n) \leq c_2 g(n)$  for  $n \geq n_0$

To prove upper bound, write

$T(n) \leq 2T(\frac{n}{3}) + c_2 n$

Assume  $T(\frac{n}{3}) \leq d \cdot \frac{n}{3}$  holds

Substitution:

$\Rightarrow T(n) \leq 2T(\frac{n}{3}) + c_2 n$

$\leq 2d \cdot \frac{n}{3} + c_2 n$

$= (\frac{2d}{3} + c_2) n$

$\leq dn \quad \text{if } \frac{2d}{3} + c_2 \leq d$

$\Rightarrow d \geq 3c_2$

Thus,  $T(n) = O(n)$

Lower bound

Guess:  $T(n) \geq dn$  ( $d > 0$ ,  $d$  is constant)

To prove lower bound, write

$T(n) \geq 2T(\frac{n}{3}) + c_1 n$

Assume  $T(\frac{n}{3}) \geq d \cdot \frac{n}{3}$  holds

Substitution:

$\Rightarrow T(n) \geq 2T(\frac{n}{3}) + c_1 n$

$\geq 2d \cdot \frac{n}{3} + c_1 n$

$= (\frac{2d}{3} + c_1) n$

$\geq dn \quad \text{if } \frac{2d}{3} + c_1 \geq d$

$\Rightarrow d \leq 3c_1$

Thus,  $T(n) = \Omega(n)$

Therefore,  $T(n) = \Theta(n)$

b. proof

upper bound

Guess:  $T(n) \leq dn^2$  ( $d > 0$ ,  $d$  is constant)

To prove upper bound, write

$$T(n) \leq 4T\left(\frac{n}{2}\right) + c_2 n \quad (c_2 > 0)$$

Assume  $T\left(\frac{n}{2}\right) \leq d\left(\frac{n}{2}\right)^2$  holds

Substitution:

$$\begin{aligned} T(n) &\leq 4T\left(\frac{n}{2}\right) + c_2 n \\ &\leq 4d\left(\frac{n}{2}\right)^2 + c_2 n \\ &= dn^2 + c_2 n \end{aligned}$$

Obviously, we cannot prove it from what we obtained above.

Thus, change the guess into

$$T(n) \leq dn^2 - n$$

Assume  $T\left(\frac{n}{2}\right) \leq d\left(\frac{n}{2}\right)^2 - n$

Substitution:

$$\begin{aligned} T(n) &\leq 4T\left(\frac{n}{2}\right) + c_2 n \\ &\leq 4\left[d\left(\frac{n}{2}\right)^2 - \frac{n}{2}\right] + c_2 n \\ &= dn^2 + (c_2 - 4)n \\ &\leq dn^2 \quad \text{if } c_2 - 4 \leq 0 \\ &\Rightarrow c_2 \leq 4 \end{aligned}$$

Thus,  $T(n) = O(n^2)$

Lower bound

Guess:  $T(n) \geq dn^2$  ( $d > 0$ ,  $d$  is constant)

To prove lower bound, write

$$T(n) \geq 4T\left(\frac{n}{2}\right) + c_1 n \quad (c_1 > 0)$$

Assume  $T\left(\frac{n}{2}\right) \geq d\left(\frac{n}{2}\right)^2$  holds

Substitution:

$$\begin{aligned} T(n) &\geq 4T\left(\frac{n}{2}\right) + c_1 n \\ &\geq 4d\left(\frac{n}{2}\right)^2 + c_1 n \\ &= dn^2 + c_1 n \\ &\geq dn^2 \end{aligned}$$

## Problem 2

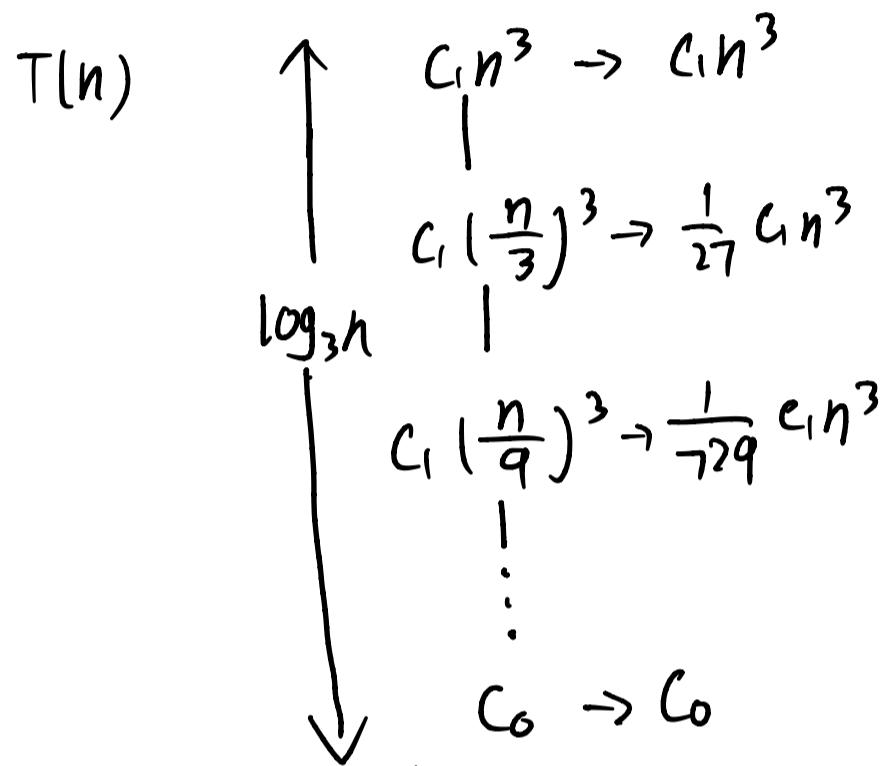
a. Solution:

$$\text{For } T(n) = T\left(\frac{n}{3}\right) + \Theta(n^3)$$

Write  $T(n)$  as

$$T(n) = \begin{cases} C_0 & n=1 \\ T\left(\frac{n}{3}\right) + C_1 n^3 & n>1 \end{cases}$$

Sketch the recursion tree



$$\begin{aligned} \text{Thus, } T(n) &= \sum_{i=0}^{\log_3 n-1} \left(\frac{1}{3}\right)^i C_1 n^3 + C_0 \\ &= C_1 n^3 \sum_{i=0}^{\log_3 n-1} \left(\frac{1}{3}\right)^i + C_0 \\ &= \frac{3}{2} C_1 n^3 \left(1 - \frac{1}{n}\right) + C_0 \\ &= \frac{3}{2} C_1 n^3 - \frac{3}{2} C_1 n^2 + C_0 \\ &= O(n^3) \end{aligned}$$

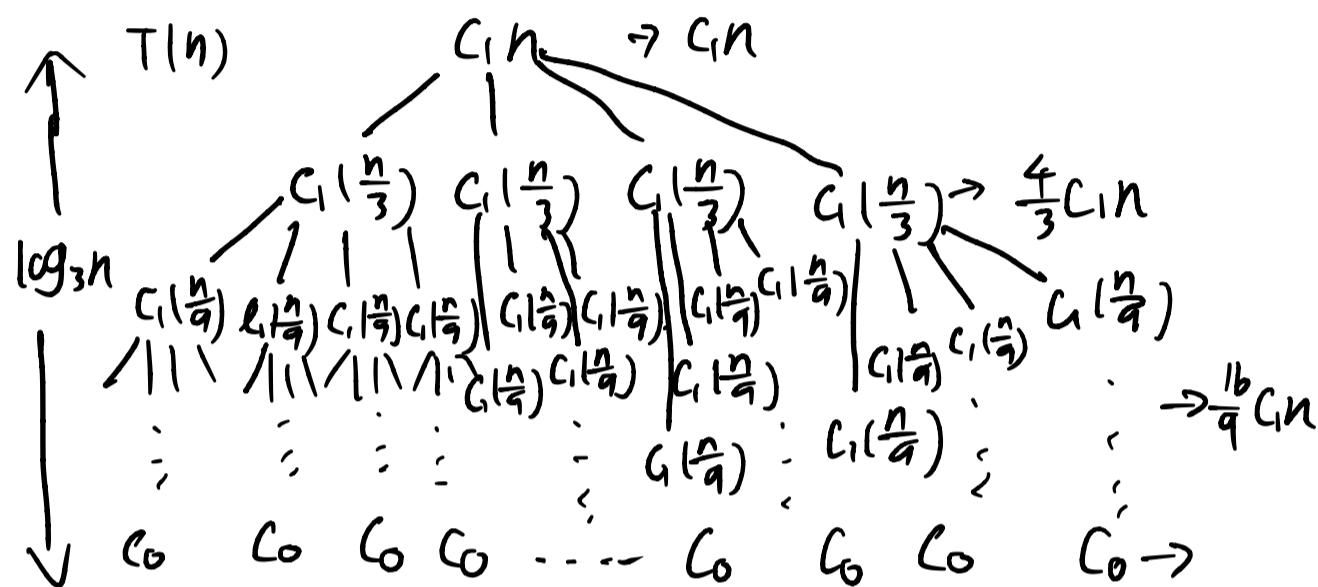
b. Solution:

$$\text{For } T(n) = 4T\left(\frac{n}{3}\right) + \Theta(n^3)$$

Write  $T(n)$  as

$$T(n) = \begin{cases} C_0 & n=1 \\ \frac{4}{9} C_1 n \cdot 4 & n>1 \end{cases}$$

Sketch the recursion tree



$$\begin{aligned} \text{Thus, } T(n) &= \sum_{i=0}^{\log_3 n-1} 4^i \left(\frac{1}{3}\right)^i C_1 n + C_0 \cdot 4^{\log_3 n} \\ &= C_1 n \sum_{i=0}^{\log_3 n-1} \left(\frac{4}{3}\right)^i + C_0 \cdot 4^{\log_3 n} \\ &= 3C_1 n \left(\frac{2}{n} - 1\right) + C_0 \cdot 4^{\log_3 n} \\ &= 3C_1 n (n^{2\log_3 2} - 1) + C_0 \cdot n^{2\log_3 2} \\ &= 3C_1 n^{2\log_3 2} + C_0 n^{2\log_3 2} - 3C_1 n \\ &= (3C_1 + C_0) n^{\log_3 4} - 3C_1 n \\ &= O(n^{\log_3 4}) \end{aligned}$$

### problem 3

a. Solution:

$$\text{For } T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n}(\log_2 n)^2,$$

we have  $a=2$  and  $b=4$

$$\Rightarrow n^{\log_b a} = n^{\log_2 4} = \sqrt{n} = \Theta(\sqrt{n})$$

$$\text{since } f(n) = \sqrt{n}(\log_2 n)^2 = \Theta(n^{\log_b a} \log^2 n),$$

which corresponds with case 2 in textbook

$$\Rightarrow T(n) = \Theta(\sqrt{n} \log^3 n)$$

b. Solution:

$$\text{For } T(n) = 2T\left(\frac{n}{2}\right) + n^3,$$

we have  $a=2$  and  $b=2$

$$\Rightarrow n^{\log_b a} = n^{\log_2 2} = n = \Theta(n)$$

$$\text{since } f(n) = n^3 = \Omega(n^{1+\varepsilon}) = \Omega(n^3),$$

we can find  $\varepsilon=2$  such that  $f(n) = \Omega(n^{1+\varepsilon})$

Assume  $2f\left(\frac{n}{2}\right) \leq c f(n)$  ( $c < 1$  and  $c$  is constant)

holds

$$\Rightarrow 2\left(\frac{n}{2}\right)^3 \leq c n^3$$

$$\Rightarrow c \geq \frac{2\left(\frac{n}{2}\right)^3}{n^3} = \frac{1}{4}$$

Therefore,  $\exists c \in [\frac{1}{4}, 1)$  such that  $2f\left(\frac{n}{2}\right) \leq c f(n)$

$$\Rightarrow T(n) = \Theta(n^3)$$