Chapter 4
Divide and Conquer

Algorithm Analysis

School of CSEE

Recurrence Equation

- In many cases we can use a ***recurrence equation*** to describe the running time of an algorithm.
- In this chapter we will study how to solve recurrence equation.

Recurrences

- **The expression:**

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + cn & n > 1 \end{cases}$$

base case.

is a *recurrence*.

  - **Recurrence: an equation that describes a function in terms of its value on smaller functions.** 점화식.

- **Technicalities**
  - **Floors and ceilings**
  - **Exact vs. asymptotic functions**
  - **Boundary conditions** = base case.

Recurrence Examples

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + c & n > 1 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

$(b > 1)$

Solving Recurrence Equations

4 ways.

- **Substitution method** → 아는걸 증명하는 느낌이 강하다.
- **Recursion-tree method**
- **Iteration method**
- **Master method**

주로 할것들.

Substitution method

- Guess the form of the solution.
- Prove the guess is correct by mathematical induction.

Substitution method

$$T(n) = \begin{cases} \Theta(1) & \text{if} \quad n = 1 \\ 2T(\frac{n}{2}) + n & \text{if} \quad n > 1 \end{cases}$$

Guess $\quad T(n) = O(n \lg n)$

Let us prove that $T(n) \le cn \lg n$ for some c>0.

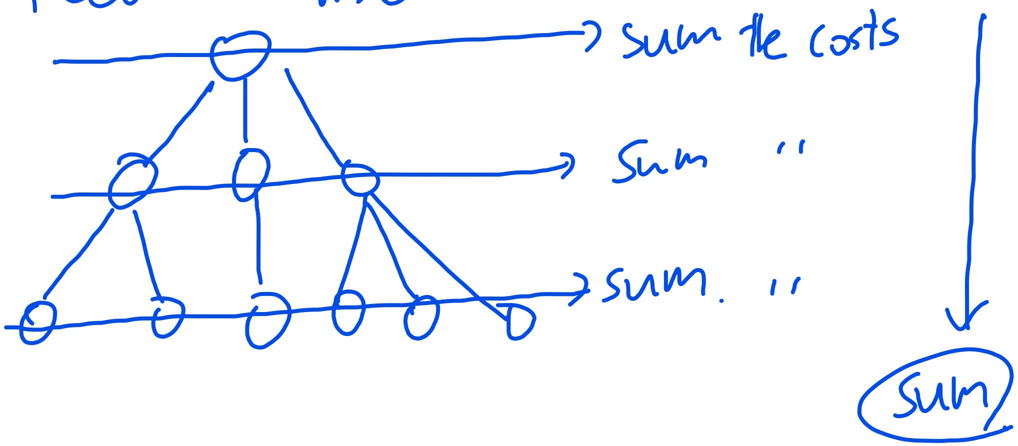
$$\begin{aligned} T(n) &= 2T(\frac{n}{2}) + n \\ &\le 2c\frac{n}{2}\lg\frac{n}{2} + n \\ &= cn\lg n - cn + n \\ &= cn\lg n - (c-1)n \\ &\le cn\lg n \quad \text{for} \quad c \ge 1 \end{aligned}$$

Recursion-tree method

no need to guess.

- Build a recursion tree.
  - Each node is the cost of a single subproblem.
- Sum the costs within each level.
- Sum the costs of all levels.

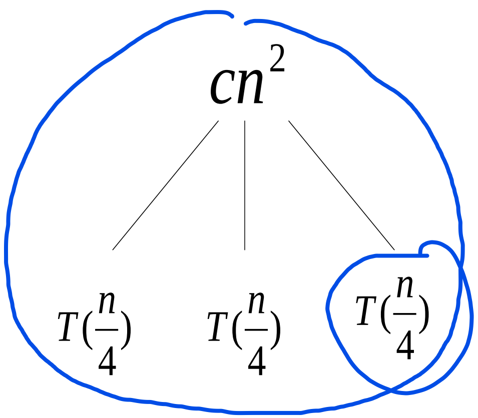
Recursion Tree



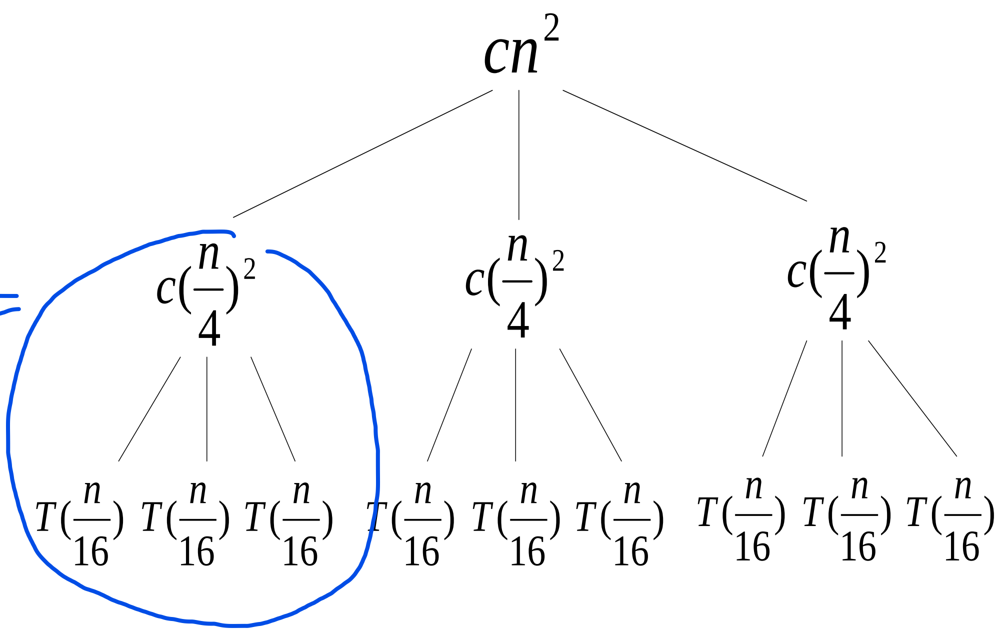
Construction of a recursion tree

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

T(n) =



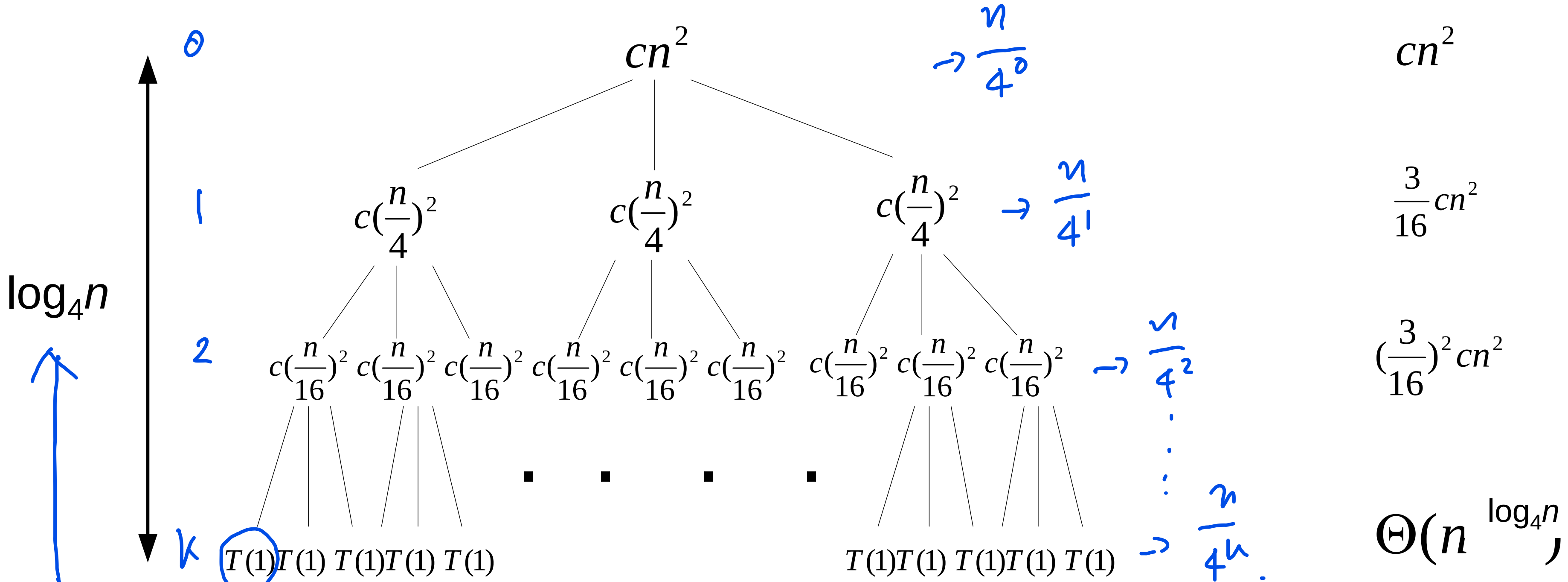
=



Construction of a recursion tree

base case = T(1)=1.

$$T(n) = 3T(\frac{n}{4}) + cn^2$$

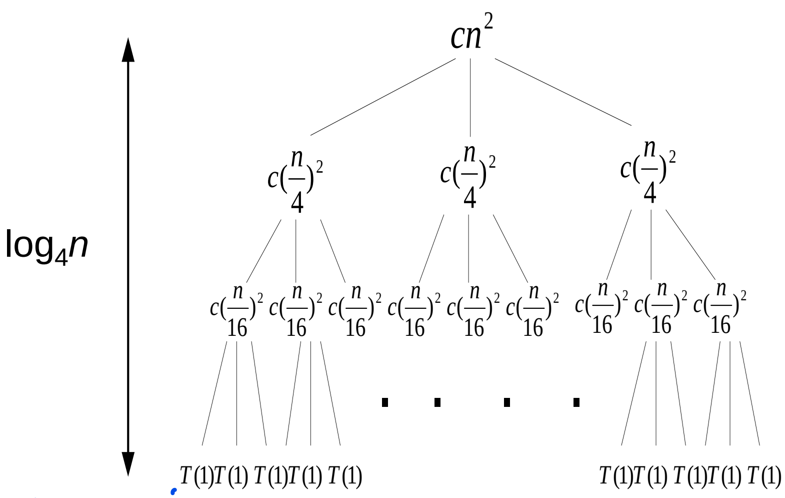


step 1 : get the hight of the tree. : see how the data size is changing.

when $4^k = n$. T(1).

Construction of a recursion tree

$$T(n) = 3T(\frac{n}{4}) + cn^2$$



cn^2

$\frac{3}{16}cn^2$

$(\frac{3}{16})^2 cn^2$

$\Theta(n^{\ln_4 3})$

$$= cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \ldots.\left(\frac{3}{16}\right)^{\ln_4 n-1} cn^2 + \Theta(n^{\ln_4 3})$$

$$\Rightarrow \sum_{i=0}^{\log_4 n-1} \left(\frac{3}{16}\right)^i cn^2 + \theta(n^{\log_4 3})$$

$$= \frac{1-\left(\frac{3}{16}\right)^{\log_4 n}}{1-\frac{3}{16}} cn^2 + \theta(n^{\log_4 3})$$

$$= \frac{16}{13}cn^2 - \frac{16}{13}\left(\frac{3}{16}\right)^{\log_4 n} \cdot cn^2 + \theta(n^{\log_4 3}) < \frac{16}{13}cn^2 + \theta(n^{\log_4 3}) = O(n^2)$$

Exercise

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

$$\left(\frac{1}{2}\right)^0$$

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

$$T\left(\frac{n}{2}\right) = 4T\left(\frac{n}{4}\right) + \frac{1}{2} \cdot n$$

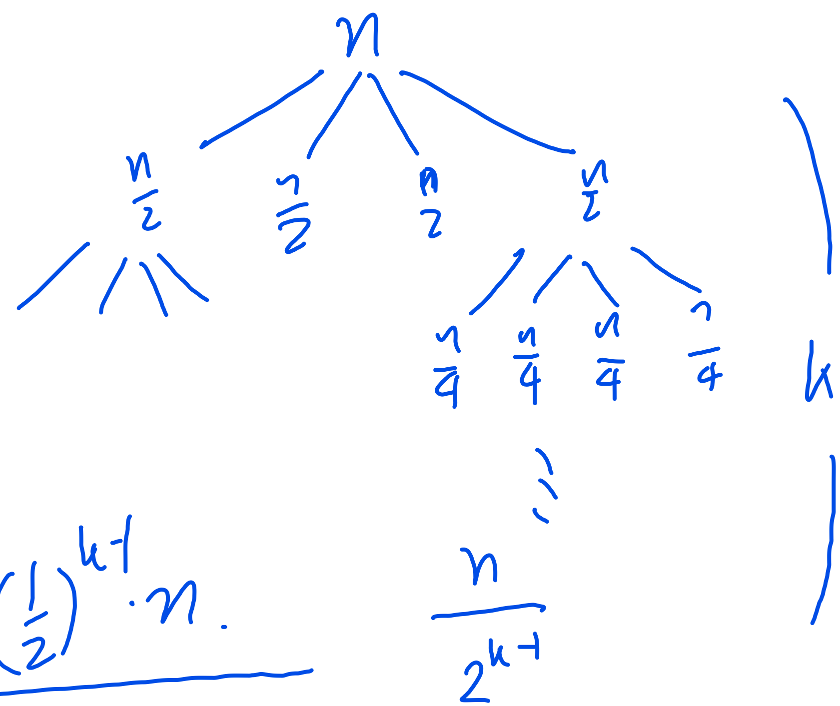
$$T\left(\frac{n}{4}\right) = 4T\left(\frac{n}{8}\right) + \left(\frac{1}{2}\right)^2 \cdot n$$

$$T\left(\frac{n}{8}\right) = 4T\left(\frac{n}{16}\right) + \left(\frac{1}{2}\right)^3 \cdot n$$

k

$\vdots$

$$T\left(\left(\frac{1}{2}\right)^{k-1} \cdot n\right) = 4T\left(\left(\frac{1}{2}\right)^{k} \cdot n\right) + \left(\frac{1}{2}\right)^{k-1} \cdot n$$



Iteration Method

- Another option is the "iteration method"
  - Expand the recurrence
  - Work some algebra to express as a summation
  - Evaluate the summation

- We will show a few examples.

Example

$$s(n)=\begin{cases}0 & n=0\\ c+s(n-1) & n>0\end{cases}$$

- $s(n)$

  $= c + s(n\text{-}1)$

  $= c + c + s(n\text{-}2)$

  $= 2c + s(n\text{-}2)$

  $= 2c + c + s(n\text{-}3)$

  $= 3c + s(n\text{-}3)$

  …

  $= kc + s(n\text{-}k) = ck + s(n\text{-}k)$

Example

- So far for $n >= k$ we have
  - $s(n) = ck + s(n-k)$
- What if $k = n$?
  - $s(n) = cn + s(0) = cn$
- Thus in general
  - $s(n) = cn$ $= \Theta(n)$

Exercise

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

$s(n) = n + s(\cancel{n-1})$

$\cancel{s(n-1)} = n-1 + \cancel{s(n-2)}$

$\vdots$

$+ \; \cancel{s(1)} = n-(n-1) + s(0)$

$s(n) = \frac{n(n+1)}{2} = \Theta(n^2)$

$\sum_{i=1}^{n} i$

$1 \cdots 10$

$\frac{n(n+1)}{2}$

Master method

Master theorem

외울 필요 X. 각각 무엇을 의미하는지 √

Let $a \geq 1$, $b > 1$ be constants, function $f(n) > 0$ and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = a\,T(n/b) + f(n)$$

Then $T(n)$ can be bounded asymptotically as follows.

1. **If $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.**
2. **If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \lg n)$.**
3. **If $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and** if $f(n)$ satisfies the regularity condition $a f(n/b) \leq c f(n)$ for some constant $c < 1$ and all sufficiently large n, then **$T(n) = \Theta(f(n))$.**

Intuition

- Compare $f(n)$ with $n^{\log_b a}$

- Not all the possibilities for $f(n)$ are covered in three cases.

Exercise 1

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

$T(n) = a\,T(n/b) + f(n)$. , $\log_b a = 2$.

① $n = O(n^{2-\varepsilon}) \rightarrow$ True , Then $T(n) = \Theta(n^2)$

② $n = \Theta(n^2) \rightarrow$ False

③ $n = \Omega(n^{2+\varepsilon}) \rightarrow$ False .

Exercise 2

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

$a = 4 \quad b = 2$.

$n^{\log_b a} = n^2 \qquad f(n) = n^2$.

Case 2

$T(n) = \Theta(n^2 \lg n)$

Exercise 3

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

$n^{\log_b a} = n^2 \quad f(n) = n^3.$

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

Proof By Induction

귀납법

- Base case: n=0 or n=1
  - Show formula is true when n = 0 or 1
- Inductive hypothesis:
  - For n greater than 0, assume that the formula holds true for all $k \geq 0$ such that $k < n$
  - By inductive hypothesis, the formula is true when k=n-1
- Proof of goal statement:
  - Show that formula is then true for n

Example: Gaussian Closed Form

- Prove 1 + 2 + 3 + … + *n* = *n*(*n*+1) / 2
  - Base case: $n=0.\ f(n)=0.$ true.
    $n=1\ \ f(1)=1$ true.
  - Inductive hypothesis:
    For n greater than 0, assume that the formular holds true for all $k \geq 0$ such that $k < n$.
    assume. $1+2+\cdots+k = k(k+1)/2$. is true.
  - Proof of goal statement:
    $\underbrace{1+2+\cdots+k}_{=f(k)} + k+1 = (k+1)(k+1+1)/2.$
    $\Rightarrow\ k(k+1)/2 + k+1 = (k+1)(k+2)/2.$ True.

Exercise

- Prove $1^2 + 2^2 + 3^2 + \ldots + n^2 = n(n+1)(2n+1) / 6$
  - Base case:
    - When $n = 1$, then $1 = 1(1+1)(2+1) / 6$
  - Inductive hypothesis:
    - For n greater than 0, assume that $1^2 + 2^2 + 3^2 + \ldots + k^2 = k(k+1)(2k+1) / 6$ holds true all $k \geq 0$ such that $k < n$.
    - By hypothesis the formula is true when $k = n-1$,

      $1^2 + 2^2 + 3^2 + \ldots + (n-1)^2 = (n-1)n(2n-1) / 6$
  - Proof of goal statement:

    $1^2 + 2^2 + \ldots + (n-1)^2 + n^2 = (1^2 + 2^2 + \ldots + (n-1)^2) + n^2$

    $= (n-1)n(2n-1) / 6 + n^2 = n(n+1)(2n+1) / 6$