Design and Analysis of Algorithms

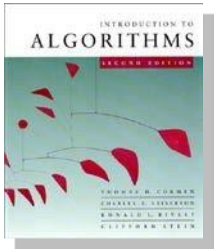
Master Method

Spring 2022

National University of Computer and Emerging Sciences,
Islamabad

Methods to solve recurrence

- **Iteration method**
- **Recursion tree method**
- **Master Theorem**

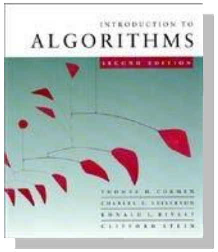


The master method

The master method applies to recurrences of the form

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

where $a \geq 1$, $b > 1$, and f is asymptotically positive.



Three common cases

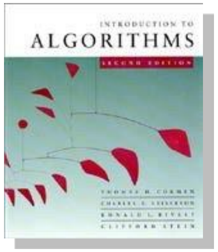
Compare $\boldsymbol{f(n)}$ with $\boldsymbol{n^{\log_b a}}$:

1. $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$.

- $f(n)$ grows **polynomially slower than** $n^{\log_b a}$

  (by an n^{ε} factor).

Solution: $\boldsymbol{T(n) = \Theta(n^{\log_b a})}$.



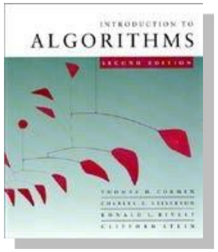
Three common cases (cont.)

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

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$.
   - $f(n)$ grows **polynomially faster than** $n^{\log_b a}$ (by an n^{ε} factor),

   and $f(n)$ satisfies the ***regularity condition*** that $a f(n/b) \le c f(n)$ for some constant $c < 1$.

   Solution: $T(n) = \Theta(f(n))$.

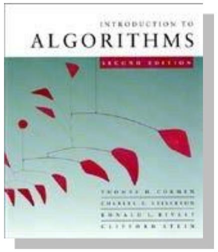


Three common cases

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

2. $f(n) = \Theta(n^{\log_b a} \lg^k n)$ for some constant $k \geq 0$.
   - $f(n)$ and $n^{\log_b a}$ **grow at similar rates**.

   Solution: $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$.



Three common cases

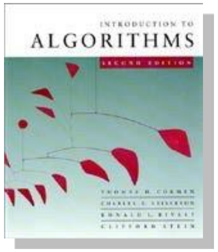
Compare $f(n)$ with $n^{\log_b a}$:

1. $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$.
   - $f(n)$ grows **polynomially slower than** $n^{\log_b a}$ (by an n^{ε} factor).

   Solution: $T(n) = \Theta(n^{\log_b a})$.

2. $f(n) = \Theta(n^{\log_b a} \lg^k n)$ for some constant $k \geq 0$.
   - $f(n)$ and $n^{\log_b a}$ **grow at similar rates**.

   Solution: $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$.



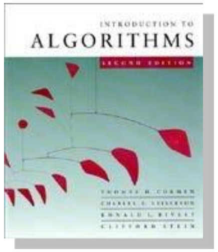
Three common cases (cont.)

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

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$.
   - $f(n)$ grows polynomially faster than $n^{\log_b a}$ (by an n^{ε} factor),

   and $f(n)$ satisfies the ***regularity condition*** that $a f(n/b) \le c f(n)$ for some constant $c < 1$.

   Solution: $T(n) = \Theta(f(n))$.



Examples

CASE 1

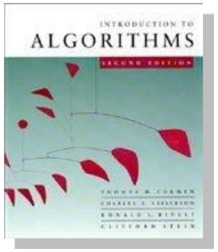
EX1. $T(n) = 4T(n/2) + n$

$a = 4,\ b = 2$

$\Rightarrow n^{\log_b a} = n^2;$

$f(n) = n.$

Compare $n^{\log_b a}$ *and* $f(n)$

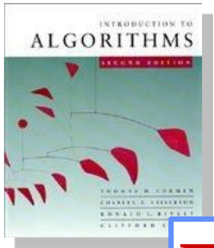


Three common cases

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

1. $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$.
   - $f(n)$ grows **polynomially slower than** $n^{\log_b a}$ (by an n^{ε} factor).

   Solution: $T(n) = \Theta(n^{\log_b a})$.



CASE 1

Examples

EX1. $T(n) = 4T(n/2) + n$

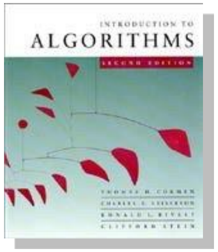
$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;$

$f(n) = n.$

- CASE 1: $f(n)$ grows **polynomially slower than** $n^{\log_b a}$ (by an n^{ε} factor).

$f(n) = O(n^{2-\varepsilon})$ for $\varepsilon = 1$.

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



Examples

CASE 2

Ex2.

$$\boldsymbol{T(n) = 4T(n/2) + n^2}$$

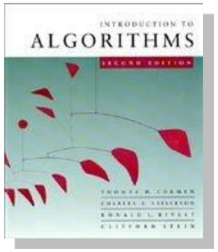
$a = 4,\ b = 2$

$$\Rightarrow n^{\log_b a} = n^2;$$

$$f(n) = n^2.$$

$$f(n) = \Theta(n^{\log_b a} \lg^k n)$$

Compare $n^{\log_b a}$ *and* $f(n)$

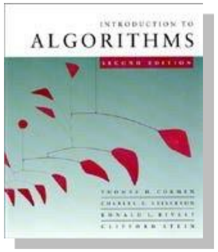


Three common cases

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

2. $f(n) = \Theta(n^{\log_b a} \lg^k n)$ for some constant $k \geq 0$.
   - $f(n)$ and $n^{\log_b a}$ **grow at similar rates**.

   Solution: $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$.



CASE 2

Examples

EX2.

$$\boldsymbol{T(n) = 4T(n/2) + n^2}$$

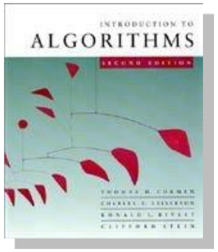
$$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;$$

$$\boldsymbol{f(n) = n^2}.$$

CASE 2:

$f(n) = \Theta(n^2 \lg^0 n)$, that is, $\boldsymbol{k = 0}$.

$$\boldsymbol{\therefore\ T(n) = \Theta(n^2 \lg n)}$$



Examples

CASE 3

EX3.

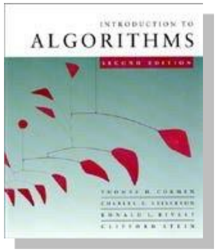
$T(n) = 4T(n/2) + n^3$

$a = 4,\ b = 2$

$\Rightarrow n^{\log_b a} = n^2;$

$f(n) = n^3.$

Compare $n^{\log_b a}$ **and** $f(n)$



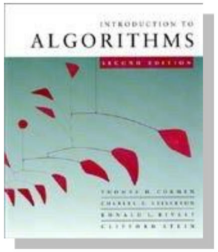
Three common cases (cont.)

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

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$.
   - $f(n)$ grows polynomially faster than $n^{\log_b a}$ (by an n^{ε} factor),

   and $f(n)$ satisfies the ***regularity condition*** that $a f(n/b) \le c f(n)$ for some constant $c < 1$.

   Solution: $T(n) = \Theta(f(n))$.



CASE 3

Examples

EX3.

$$T(n) = 4T(n/2) + n^3$$

$$a = 4,\ b = 2$$

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

CASE 3: $f(n) = \Omega(n^{2+\varepsilon})$ for $\varepsilon = 1$
and $4(n/2)^3 \le cn^3$ (reg. cond.) for $c = 1/2$.

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

Reference

- **Introduction to Algorithms**
- **4.1,4.2, 4.3, 4.4, 4.5**
  - **Chapter # 4**
  - **Thomas H. Cormen**
  - **3rd Edition**