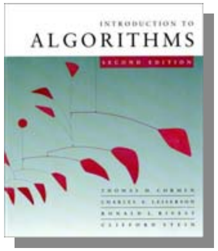
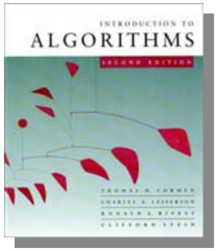


Solving recurrences

- The analysis of merge sort from ***Lecture 1*** required us to solve a recurrence.
- Recurrences are like solving integrals, differential equations, etc. Ex.: Merge Sort:
  - Learn a few tricks.

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

- ***Lecture 3***: Applications of recurrences to divide-and-conquer algorithms.



Substitution method

The most general method:

1. ***Guess*** the form of the solution.
2. ***Verify*** by induction.
3. ***Solve*** for constants.

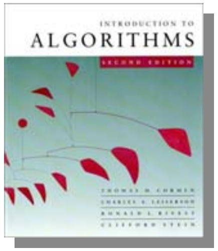
Substitution method

The most general method:

1. ***Guess*** the form of the solution.
2. ***Verify*** by induction.
3. ***Solve*** for constants.

EXAMPLE: $T(n) = 4T(n/2) + n$

- [Assume that $T(1) = \Theta(1)$.]
- Guess $O(n^3)$. (Prove O and Ω separately.)
- Assume that $T(k) \le ck^3$ for $k < n$. I.H. : Inductive Hypothesi
- Prove $T(n) \le cn^3$ by induction.



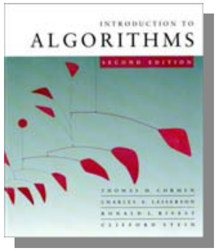
Example of substitution

I.H: $T(k) \le ck^3$ for $k < n$

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^3 + n \\
&= (c/2)n^3 + n \\
&= cn^3 - \underbrace{((c/2)n^3 - n)} \longleftarrow \boxed{\textit{desired} - \textit{residual}} \\
&\le cn^3 \longleftarrow \textit{desired}
\end{aligned}
$$

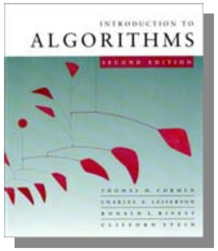
whenever $(c/2)n^3 - n \ge 0$, for example, if $c \ge 2$ and $n \ge 1$.

residual



Example (continued)

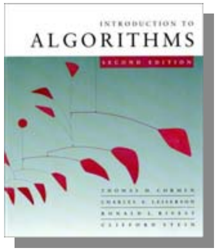
- We must also handle the initial conditions, that is, ground the induction with base cases.
- ***Base:*** $T(n) = \Theta(1)$ for all $n < n_0$, where n_0 is a suitable constant.
- For $1 \le n < n_0$, we have "$\Theta(1)$" $\le cn^3$, if we pick c big enough.



Example (continued)

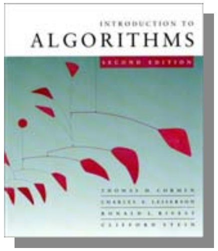
- We must also handle the initial conditions, that is, ground the induction with base cases.
- ***Base:*** $T(n) = \Theta(1)$ for all $n < n_0$, where n_0 is a suitable constant.
- For $1 \le n < n_0$, we have "$\Theta(1)$" $\le cn^3$, if we pick c big enough.

This bound is not tight!



A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

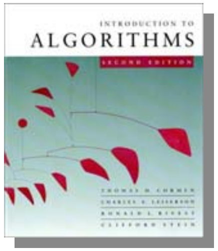


A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \le ck^2$ for $k < n$: — I.H

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^2 + n \\
&= cn^2 + n \qquad \le cn^2 \\
&= O(n^2)
\end{aligned}
$$



A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \le ck^2$ for $k < n$:

$$\begin{aligned} T(n) &= 4T(n/2) + n \\ &\le 4c(n/2)^2 + n \\ &= cn^2 + n \\ &= \cancel{O(n^2)} \end{aligned}$$

Wrong! We must prove the I.H.

A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \le ck^2$ for $k < n$:

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^2 + n \\
&= cn^2 + n \\
&= \cancel{O(n^2)}
\end{aligned}
$$

Wrong! We must prove the I.H.

$$= cn^2 - (-n)$$

[desired – residual]

$$\le cn^2$$

for ***no*** choice of $c > 0$. Lose!

A tighter upper bound!

IDEA: Strengthen the inductive hypothesis.

- ***Subtract*** a low-order term.

Inductive hypothesis: $T(k) \leq c_1k^2 - c_2k$ for $k < n$.

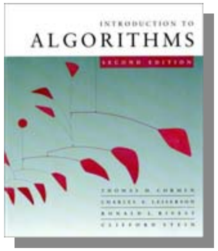
A tighter upper bound!

IDEA: Strengthen the inductive hypothesis.

- ***Subtract*** a low-order term.

Inductive hypothesis: $T(k) \le c_1 k^2 - c_2 k$ for $k < n$.

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&= 4(c_1(n/2)^2 - c_2(n/2)) + n \\
&= c_1 n^2 - 2c_2 n + n \\
&= \underbrace{\boxed{c_1 n^2 - c_2 n}}_{\text{Desired}} - \underbrace{\boxed{(c_2 n - n)}}_{\text{Residual}} \\
&\le c_1 n^2 - c_2 n \quad \text{if } c_2 \ge 1.
\end{aligned}
$$



A tighter upper bound!

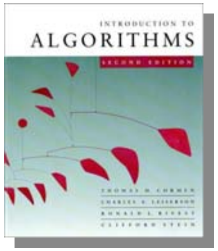
IDEA: Strengthen the inductive hypothesis.

- ***Subtract*** a low-order term.

Inductive hypothesis: $T(k) \le c_1 k^2 - c_2 k$ for $k < n$.

$$\begin{aligned}
T(n) &= 4T(n/2) + n \\
&= 4(c_1(n/2)^2 - c_2(n/2)) + n \\
&= c_1 n^2 - 2c_2 n + n \\
&= c_1 n^2 - c_2 n - (c_2 n - n) \\
&\le c_1 n^2 - c_2 n \ \text{ if } c_2 \ge 1.
\end{aligned}$$

Pick c_1 big enough to handle the initial conditions.

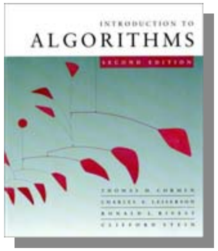


Recursion-tree method

- A recursion tree models the costs (time) of a recursive execution of an algorithm.
- The recursion-tree method can be unreliable, just like any method that uses ellipses (…).
- The recursion-tree method promotes intuition, however.
- The recursion tree method is good for generating guesses for the substitution method.

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



Example of recursion tree

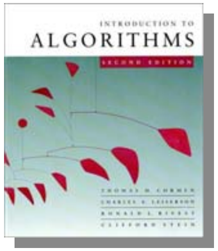
Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$$T(n)$$

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$$
\begin{array}{ccc}
 & n^2 & \\
\diagup & & \diagdown \\
T(n/4) & & T(n/2)
\end{array}
$$



Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

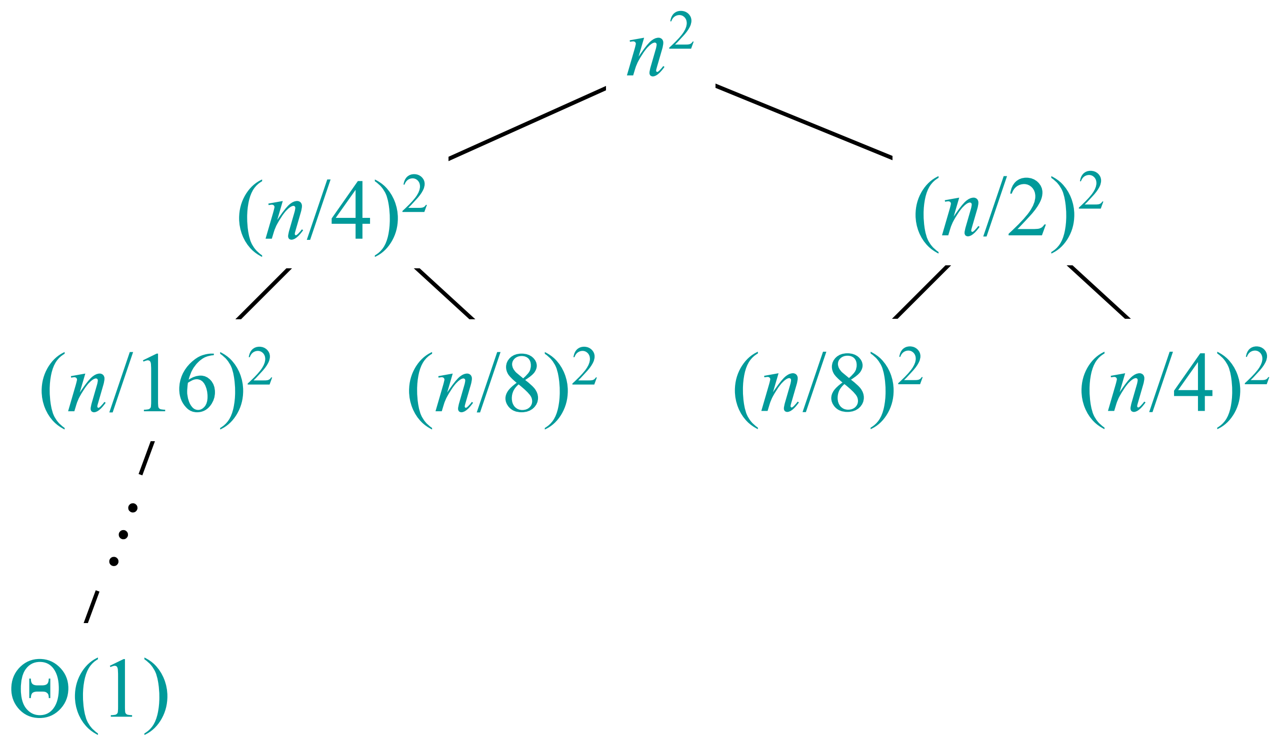
$$n^2$$

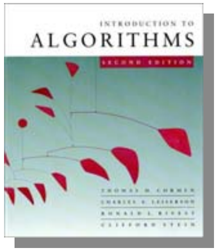
$$(n/4)^2 \qquad\qquad (n/2)^2$$

$$T(n/16) \quad T(n/8) \quad T(n/8) \quad T(n/4)$$

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:





Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

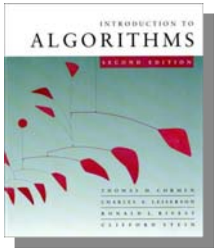
$$n^2 \qquad\qquad n^2$$

$$(n/4)^2 \qquad\qquad (n/2)^2$$

$$(n/16)^2 \qquad (n/8)^2 \qquad (n/8)^2 \qquad (n/4)^2$$

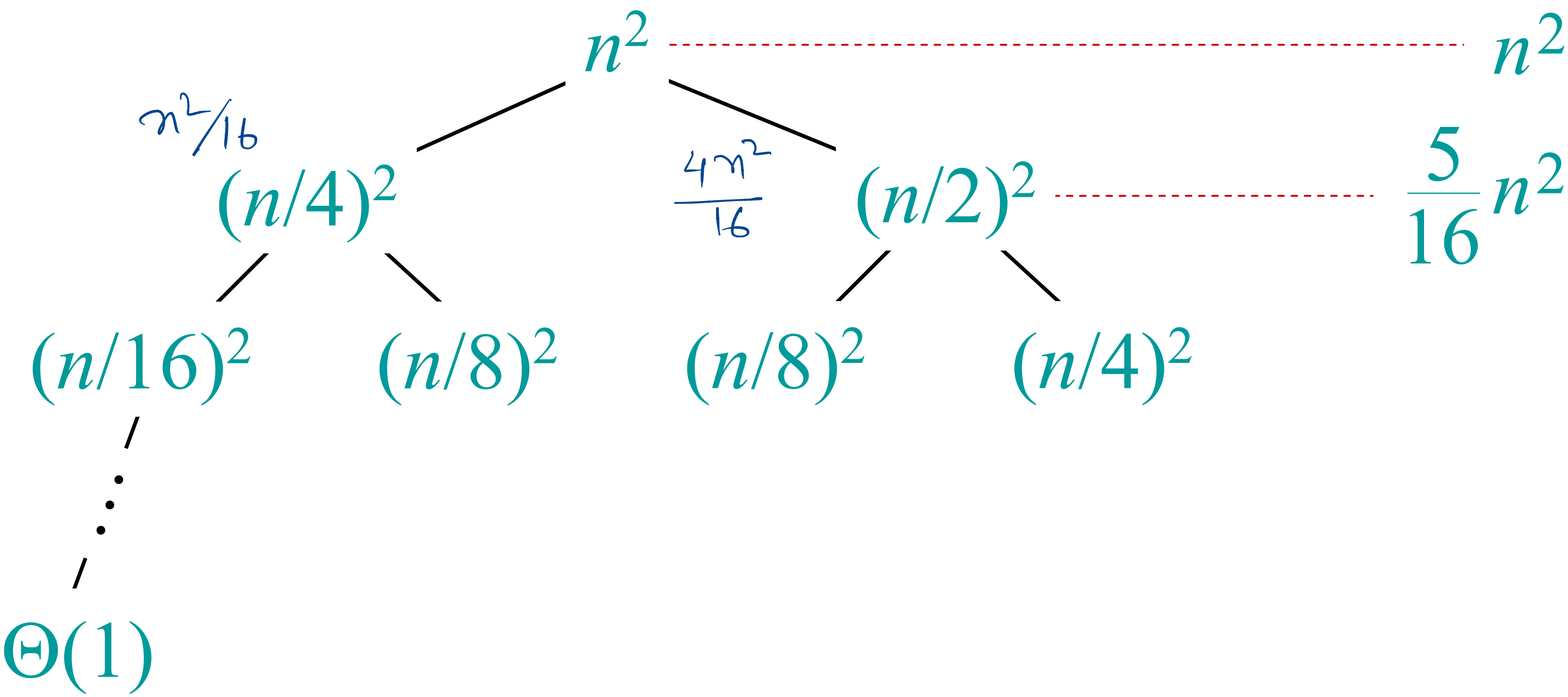
$$\vdots$$

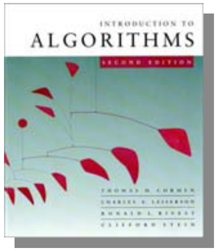
$$\Theta(1)$$



Example of recursion tree

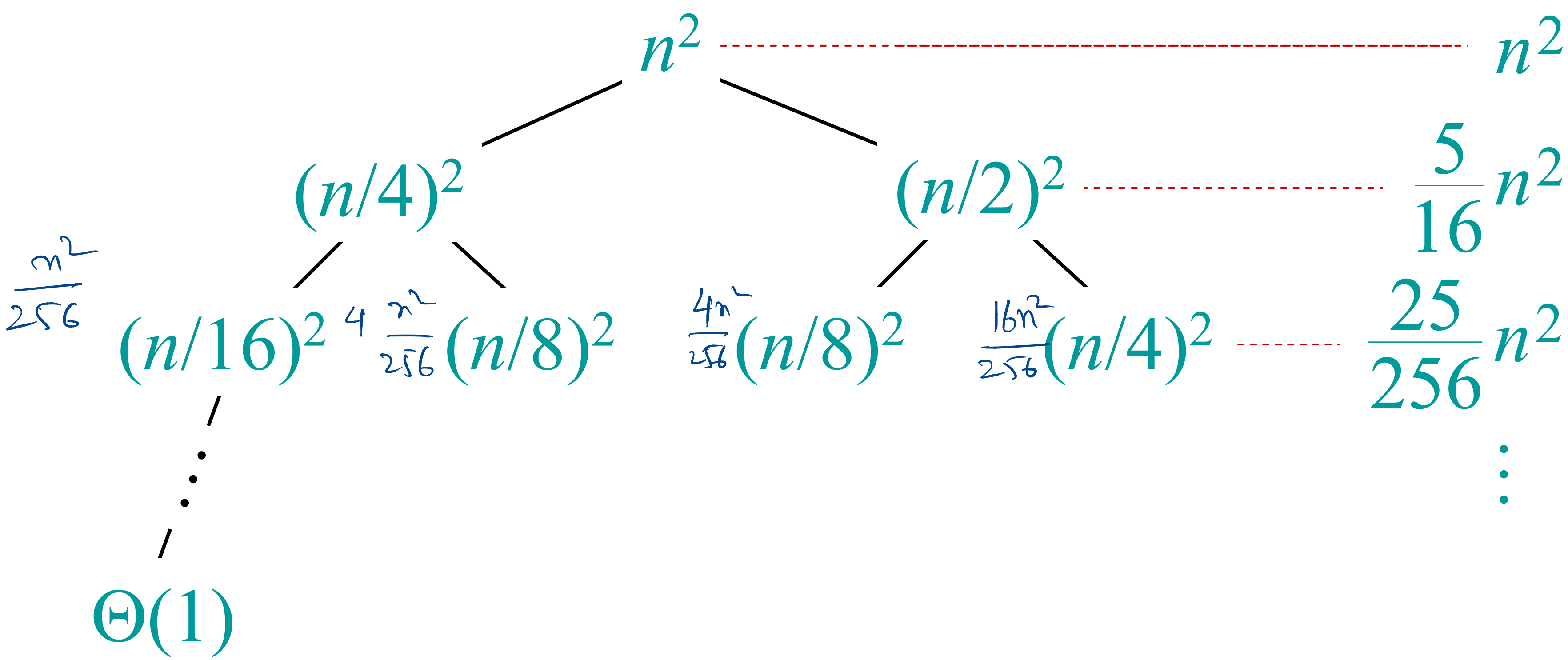
Solve $T(n) = T(n/4) + T(n/2) + n^2$:

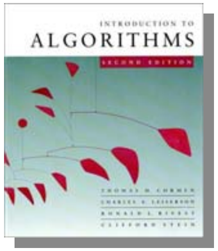




Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:





Example of recursion tree

GP: $(1 + k + k^2 + \cdots\ k^n) = S$; $k<1$

$\Rightarrow S = \left(\frac{1-k^{n+1}}{1-k}\right)$

$\lim_{n\to\infty} S = 1/(1-k)$

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

n^2 -- n^2

$(n/4)^2$ $\quad$ $(n/2)^2$ ------------------ $\frac{5}{16}n^2$

$(n/16)^2$ $\quad$ $(n/8)^2$ $\quad$ $(n/8)^2$ $\quad$ $(n/4)^2$ -------- $\frac{25}{256}n^2$

$\vdots$ $\quad$ $\vdots$

h

$\Theta(1)$

Total $= n^2 \overbrace{\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)}^{16/11}$

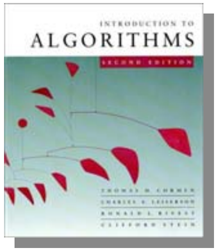
$= \Theta(n^2)$ *geometric series*

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.



$T(n) = aT(n/b) + f(n)$

Three common cases

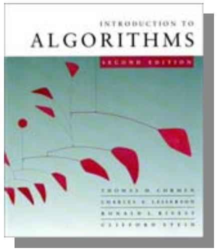
Polynomial

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

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).

   Polynomial factor

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



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

   base 2

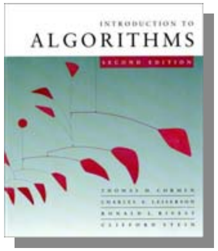
Three common cases (cont.)

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



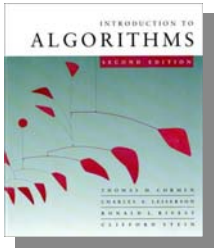
Examples

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

$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;\ f(n) = n.$

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

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



Examples

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

$a = 4$, $b = 2 \Rightarrow n^{\log_b a} = n^2$; $f(n) = n$.

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

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

Ex. $T(n) = 4T(n/2) + n^2$

$a = 4$, $b = 2 \Rightarrow n^{\log_b a} = n^2$; $f(n) = n^2$.

CASE 2: $f(n) = \Theta(n^2 \lg^0 n)$, that is, $k = 0$.

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

Examples

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

$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;\ f(n) = n^3.$

Case 3: $f(n) = \Omega(n^{2+\varepsilon})$ for $\varepsilon = 1$

and $\underline{4(n/2)^3 \le cn^3}$ (reg. cond.) for $c = 1/2$.

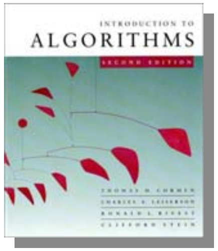
$\therefore\ T(n) = \Theta(n^3)$.

$af(n/b)$

$\Downarrow$

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

$\frac{n^3}{2} \le cn^3$



Examples

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

$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;\ f(n) = n^3.$

Case 3: $f(n) = \Omega(n^{2+\varepsilon})$ for $\varepsilon = 1$

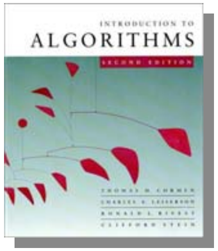
and $4(n/2)^3 \le cn^3$ (reg. cond.) for $c = 1/2$.

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

Ex. $T(n) = 4T(n/2) + n^2/\lg n$

$a = 4,\ b = 2 \Rightarrow n^{\log_b a} = n^2;\ f(n) = n^2/\lg n.$

Master method does not apply. In particular, for every constant $\varepsilon > 0$, we have $n^\varepsilon = \omega(\lg n)$.



Idea of master theorem

$T(n) = aT(n/b) + f(n)$

Recursion tree:

$$f(n)$$

a

$$f(n/b) \quad f(n/b) \quad \cdots \quad f(n/b)$$

a

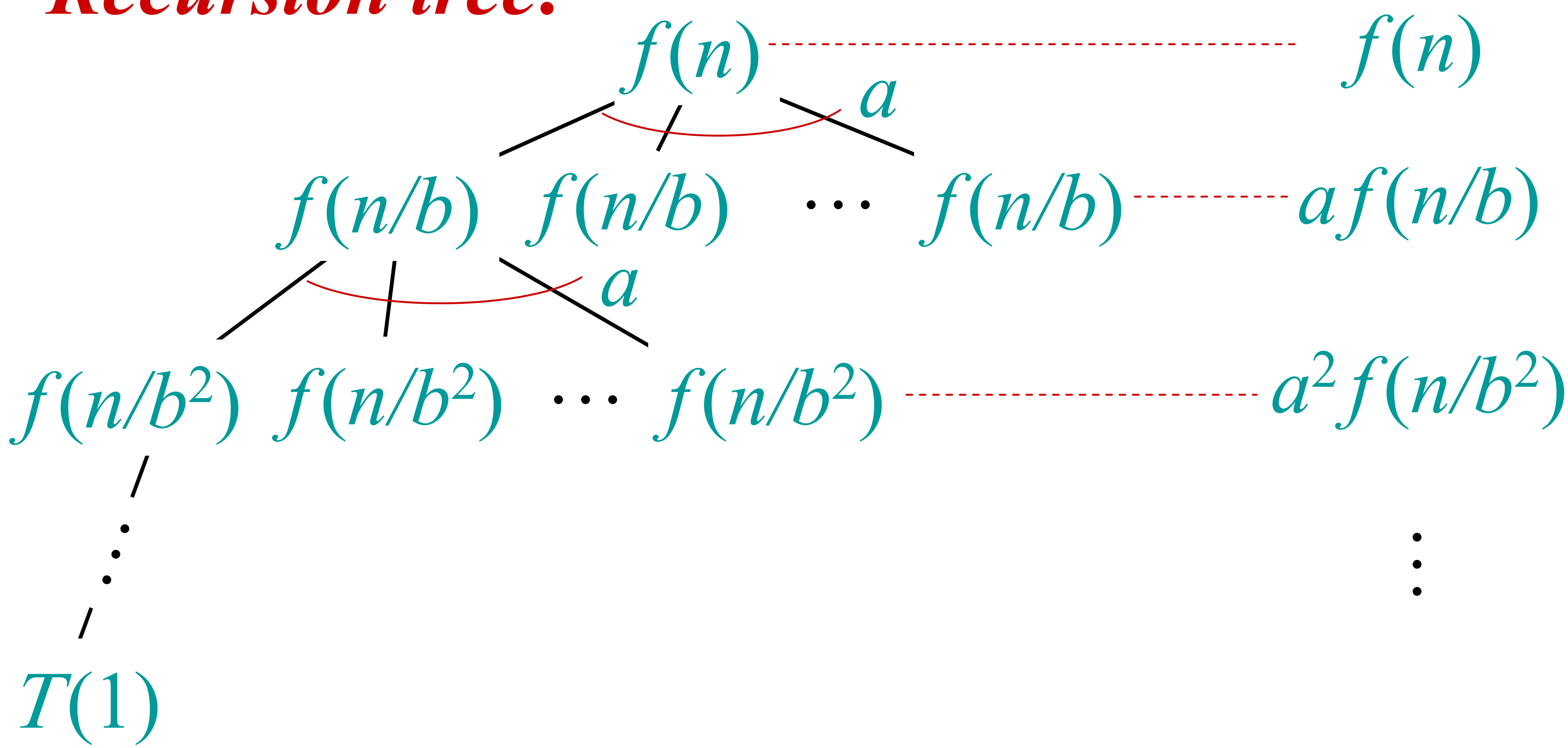
$$f(n/b^2) \quad f(n/b^2) \quad \cdots \quad f(n/b^2)$$

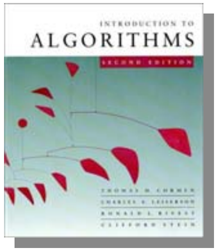
$$\vdots$$

$$T(1)$$

Idea of master theorem

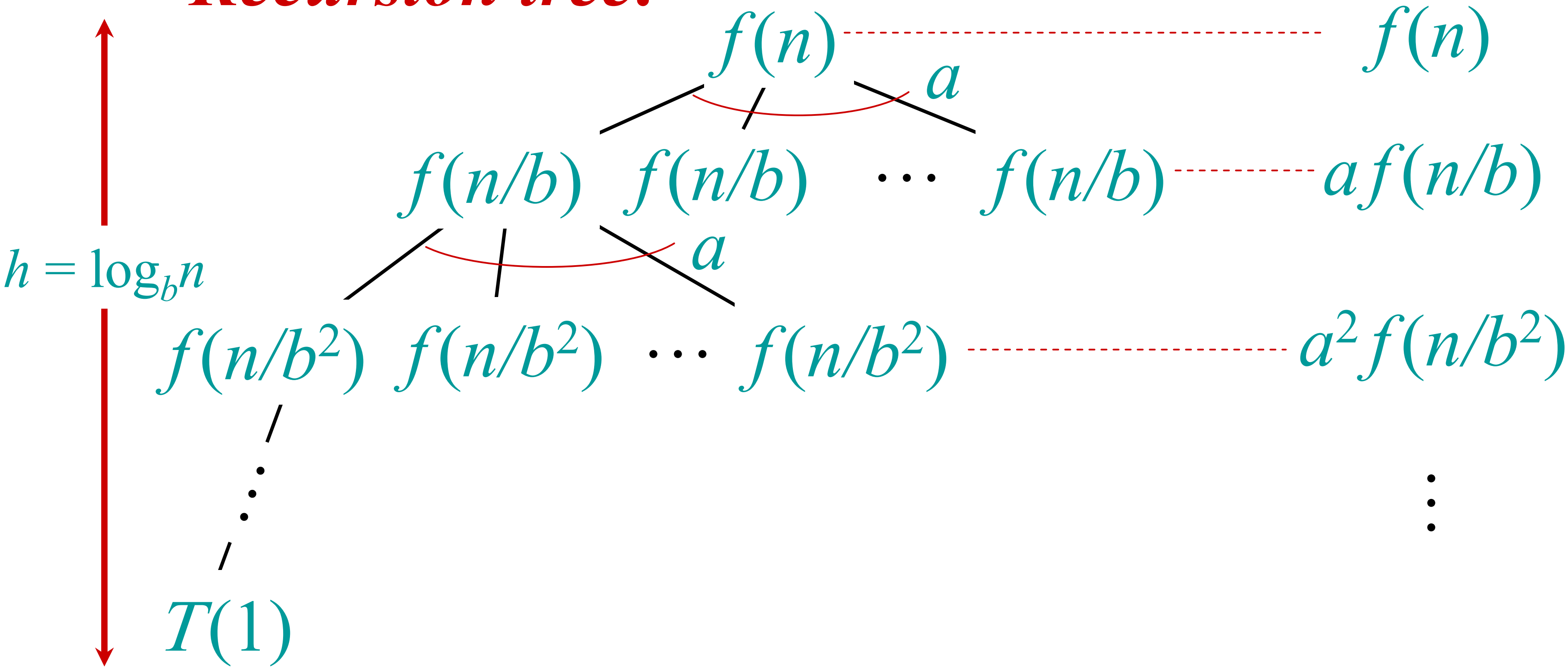
Recursion tree:

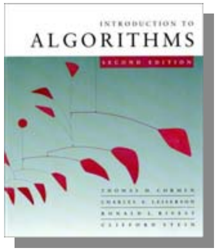




Idea of master theorem

Recursion tree:





Idea of master theorem

Recursion tree:

$$h = \log_b n$$

$$f(n) \quad \cdots\cdots \quad f(n)$$

$$a$$

$$f(n/b) \quad f(n/b) \quad \cdots \quad f(n/b) \quad \cdots\cdots \quad a\,f(n/b)$$

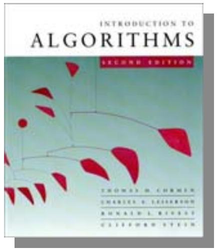
$$a$$

$$f(n/b^2) \quad f(n/b^2) \quad \cdots \quad f(n/b^2) \quad \cdots\cdots \quad a^2 f(n/b^2)$$

$$\vdots$$

$$T(1) \quad \cdots\cdots \quad n^{\log_b a}\,T(1)$$

$$\begin{aligned}\#\text{leaves} &= a^h \\ &= a^{\log_b n} \\ &= n^{\log_b a}\end{aligned}$$



Idea of master theorem

Recursion tree:

$$f(n) \quad \cdots\cdots \quad f(n)$$

$$a$$

$$f(n/b) \quad f(n/b) \quad \cdots \quad f(n/b) \quad \cdots\cdots \quad a\,f(n/b)$$

$$a$$

$$h = \log_b n$$

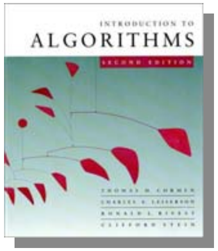
$$f(n/b^2) \quad f(n/b^2) \quad \cdots \quad f(n/b^2) \quad \cdots\cdots \quad a^2 f(n/b^2)$$

$$\vdots$$

$$T(1) \quad \cdots\cdots \quad n^{\log_b a}\,T(1)$$

$$\Theta(n^{\log_b a})$$

CASE 1: The weight increases geometrically from the root to the leaves. The leaves hold a constant fraction of the total weight.



Idea of master theorem

Recursion tree:

$$f(n) \qquad\qquad f(n)$$

$$a$$

$$f(n/b) \quad f(n/b) \quad \cdots \quad f(n/b) \qquad\qquad a\,f(n/b)$$

$$a$$

$$h = \log_b n$$

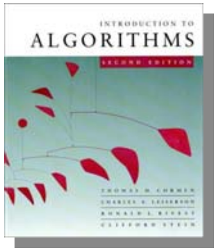
$$f(n/b^2) \quad f(n/b^2) \quad \cdots \quad f(n/b^2) \qquad\qquad a^2 f(n/b^2)$$

$$\vdots \qquad\qquad \vdots$$

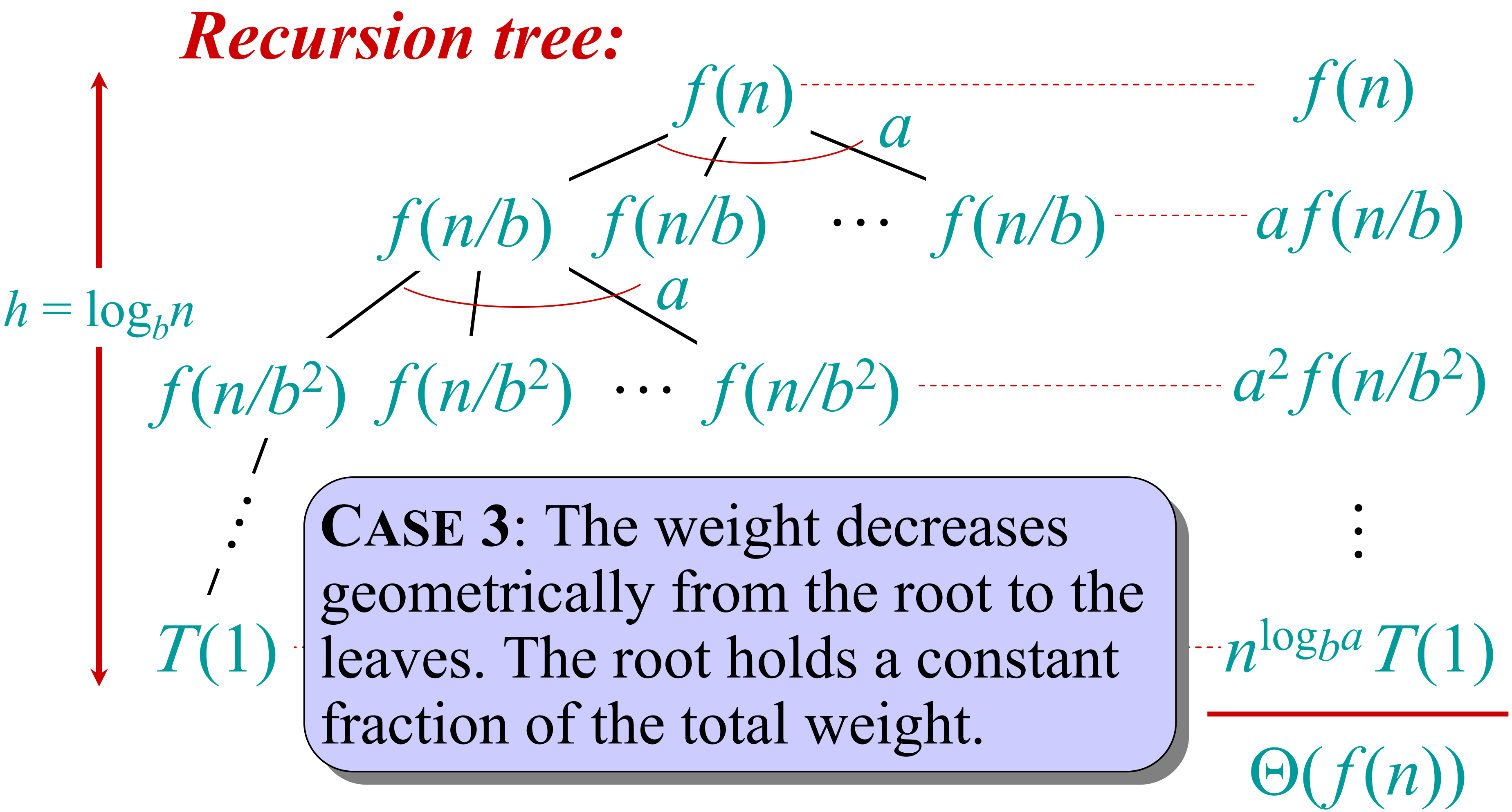
$$T(1) \qquad\qquad n^{\log_b a}\, T(1)$$

CASE 2: ($k = 0$) The weight is approximately the same on each of the $\log_b n$ levels.

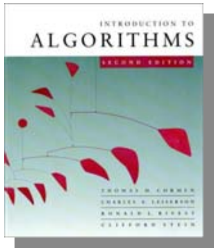
$$\Theta(n^{\log_b a} \lg n)$$



Idea of master theorem



CASE 3: The weight decreases geometrically from the root to the leaves. The root holds a constant fraction of the total weight.



Appendix: geometric series

$$1 + x + x^2 + \cdots + x^n = \frac{1 - x^{n+1}}{1 - x} \quad \text{for } x \neq 1$$

$$1 + x + x^2 + \cdots = \frac{1}{1 - x} \quad \text{for } |x| < 1$$

Return to last
slide viewed.