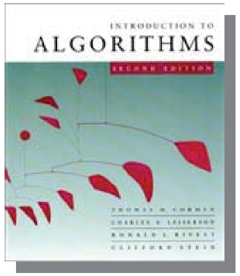
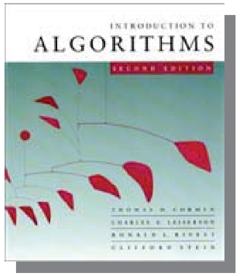


Asymptotic notation

O-notation (upper bounds):

We write $f(n) = O(g(n))$ if there exist constants $c > 0$, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$.



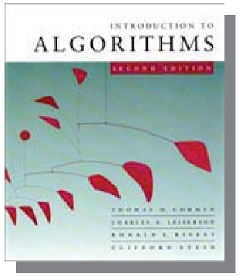
Asymptotic notation

O-notation (upper bounds):

We write $f(n) = O(g(n))$ if there exist constants $c > 0$, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$.

Example: $2n^2 = O(n^3)$ $(c = 1, n_0 = 2)$

by Charles E. Leiserson



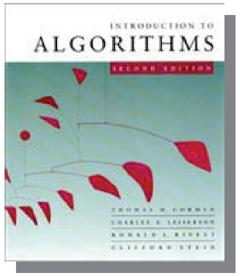
Asymptotic notation

O-notation (upper bounds):

We write $f(n) = O(g(n))$ if there exist constants $c > 0$, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$.

EXAMPLE: $2n^2 = O(n^3)$ $(c = 1, n_0 = 2)$

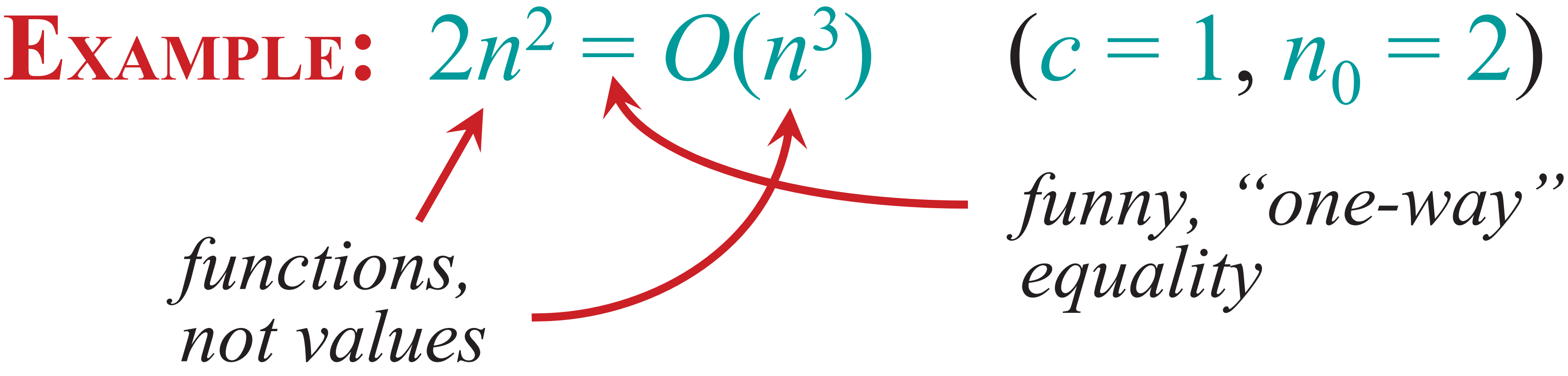
functions, not values



Asymptotic notation

O-notation (upper bounds):

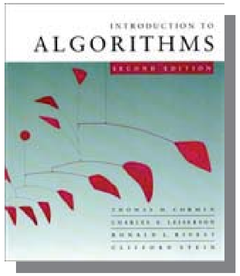
We write $f(n) = O(g(n))$ if there exist constants $c > 0$, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$.



EXAMPLE: $2n^2 = O(n^3)$ $(c = 1, n_0 = 2)$

functions, not values

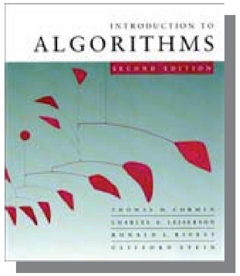
funny, "one-way" equality



Set definition of O-notation

$O(g(n)) = \{ f(n)$: there exist constants $c > 0$, $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0 \}$

by Charles E. Leiserson

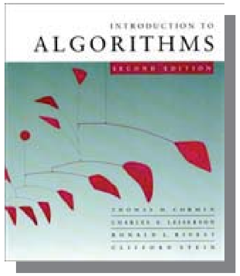


Set definition of O-notation

$O(g(n)) = \{ f(n) :$ there exist constants $c > 0, n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0 \}$

EXAMPLE: $2n^2 \in O(n^3)$

by Charles E. Leiserson



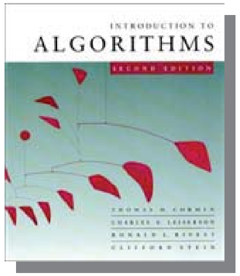
Set definition of O-notation

$O(g(n)) = \{ f(n)$: there exist constants $c > 0$, $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$ $\}$

EXAMPLE: $2n^2 \in O(n^3)$

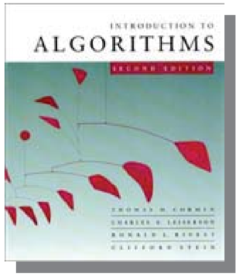
(*Logicians:* $\lambda n.2n^2 \in O(\lambda n.n^3)$, but it's convenient to be sloppy, as long as we understand what's *really* going on.)

by Charles E. Leiserson



Macro substitution

Convention: A set in a formula represents an anonymous function in the set.



Macro substitution

Convention: A set in a formula represents an anonymous function in the set.

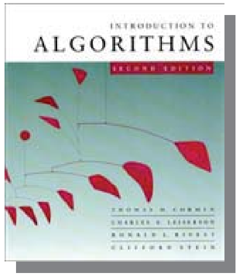
EXAMPLE: $f(n) = n^3 + O(n^2)$

means

$f(n) = n^3 + h(n)$

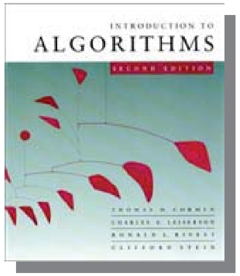
for some $h(n) \in O(n^2)$.

by Charles E. Leiserson



Ω–notation (lower bounds)

O-notation is an *upper-bound* notation. It makes no sense to say $f(n)$ is at least $O(n^2)$.

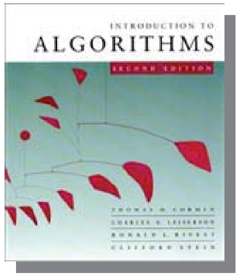


Ω–notation (lower bounds)

O-notation is an *upper-bound* notation. It makes no sense to say $f(n)$ is at least $O(n^2)$.

$\Omega(g(n)) = \{ f(n)$: there exist constants $c > 0, n_0 > 0$ such that $0 \leq cg(n) \leq f(n)$ for all $n \geq n_0$ $\}$

by Charles E. Leiserson

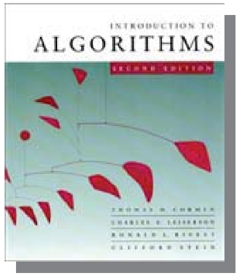


Ω–notation (lower bounds)

$\Omega(g(n)) = \{ f(n)$: there exist constants
$c > 0$, $n_0 > 0$ such
that $0 \leq cg(n) \leq f(n)$
for all $n \geq n_0$ $\}$

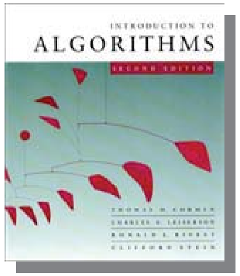
EXAMPLE: $\sqrt{n} = \Omega(\lg n)$

by Charles E. Leiserson



Θ-notation (tight bounds)

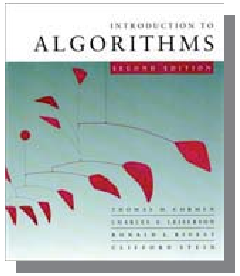
$$\Theta(g(n)) = O\,(g(n)) \;\cap\; \Omega(g(n))$$



Θ-notation (tight bounds)

$$\Theta(g(n)) = O\,(g(n)) \;\cap\; \Omega(g(n))$$

EXAMPLE: $\frac{1}{2}n^2 - 2n = \Theta(n^2)$

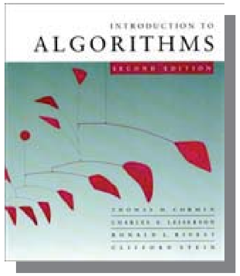


Θ-notation (tight bounds)

$$\Theta(g(n)) = O\,(g(n)) \;\cap\; \Omega(g(n))$$

EXAMPLE: $\frac{1}{2}n^2 - 2n = \Theta(n^2)$

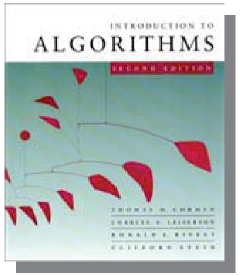
Theorem. The leading constant and low-order terms don't matter. □



Solving recurrences

- The analysis of merge sort from ***Lecture 1*** required us to solve a recurrence.
- Recurrences are like solving integrals, differential equations, etc.
  - Learn a few tricks.
- ***Lecture 3***: Applications of recurrences to divide-and-conquer algorithms.

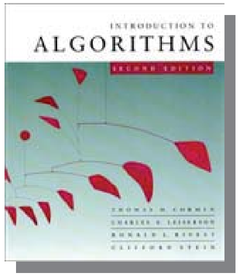
by Charles E. Leiserson



Substitution method

The most general method:

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



Substitution method

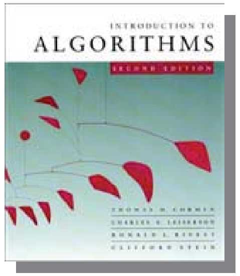
The most general method:

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 $k < n$.
- Prove $T(n) \le cn^3$ by induction.

by Charles E. Leiserson

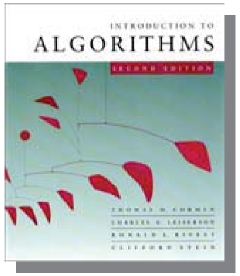


Example of substitution

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

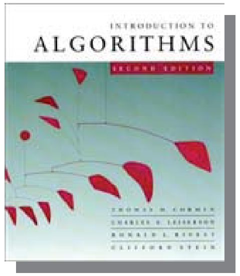
whenever $(c/2)n^3 - n \ge 0$, for example, if $c \ge 2$ and $n \ge 1$.

residual



Example (continued)

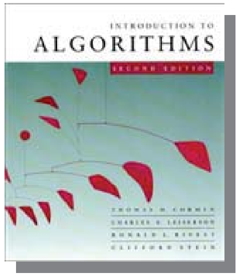
- We must also handle the initial conditions, that is, ground the induction with base cases.
- ***Base:*** $T(n) = \Theta(1)$ for all $n < n_0$, where n_0 is a suitable constant.
- For $1 \leq n < n_0$, we have "$\Theta(1)$" $\leq cn^3$, if we pick c big enough.



Example (continued)

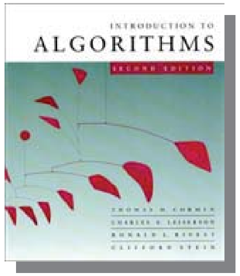
- We must also handle the initial conditions, that is, ground the induction with base cases.
- ***Base:*** $T(n) = \Theta(1)$ for all $n < n_0$, where n_0 is a suitable constant.
- For $1 \le n < n_0$, we have "$\Theta(1)$" $\le cn^3$, if we pick c big enough.

This bound is not tight!



A tighter upper bound?

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

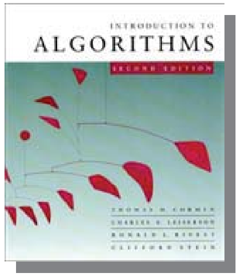


A tighter upper bound?

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

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

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



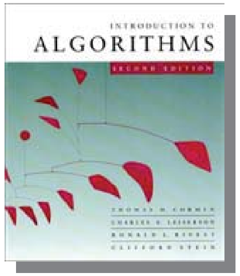
A tighter upper bound?

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 $k < n$:

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

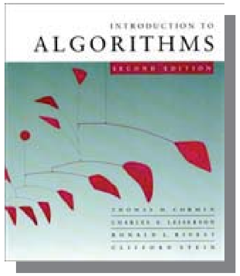
Wrong! We must prove the I.H.

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

[desired – residual]

$$\le cn^2$$

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

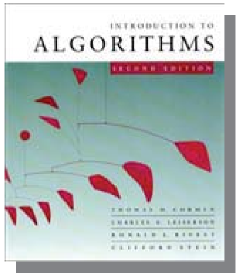


A tighter upper bound!

IDEA: Strengthen the inductive hypothesis.

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

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



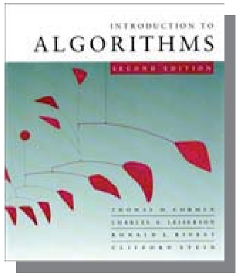
A tighter upper bound!

IDEA: Strengthen the inductive hypothesis.

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

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

$$\begin{aligned} T(n) &= 4T(n/2) + n \\ &= 4(c_1(n/2)^2 - c_2(n/2) + n \\ &= c_1n^2 - 2c_2n + n \\ &= c_1n^2 - c_2n - (c_2n - n) \\ &\le c_1n^2 - c_2n \quad \text{if } c_2 \ge 1. \end{aligned}$$



A tighter upper bound!

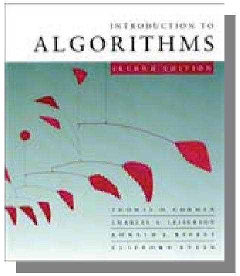
IDEA: Strengthen the inductive hypothesis.

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

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

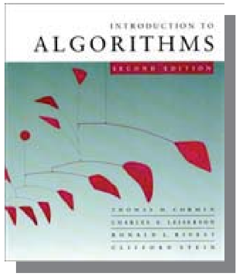
$$\begin{aligned} T(n) &= 4T(n/2) + n \\ &= 4(c_1(n/2)^2 - c_2(n/2) + n \\ &= c_1 n^2 - 2c_2 n + n \\ &= c_1 n^2 - c_2 n - (c_2 n - n) \\ &\le c_1 n^2 - c_2 n \quad \text{if } c_2 \ge 1. \end{aligned}$$

Pick c_1 big enough to handle the initial conditions.



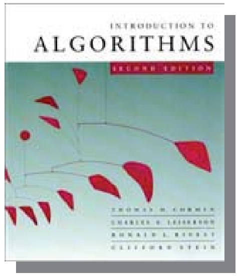
Recursion-tree method

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

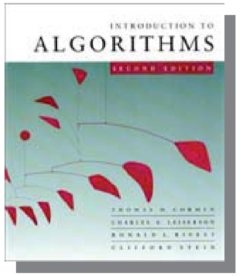


Example of recursion tree

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

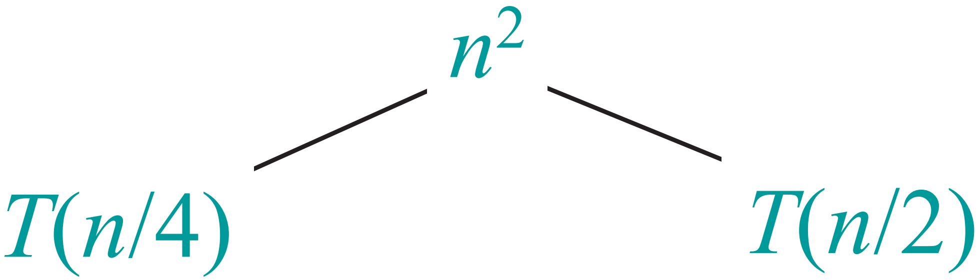
$$T(n)$$

by Charles E. Leiserson



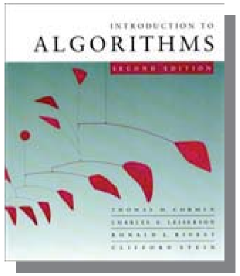
Example of recursion tree

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



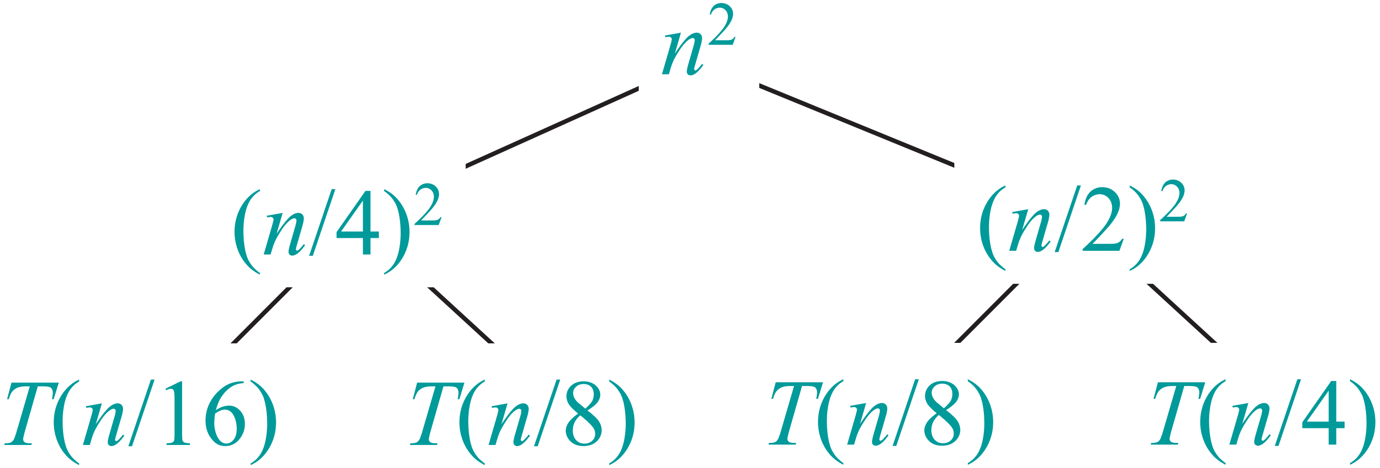
n^2

$T(n/4)$ $T(n/2)$



Example of recursion tree

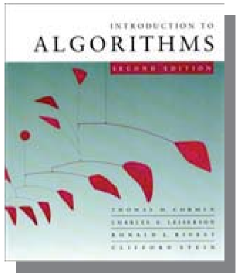
Solve $T(n) = T(n/4) + T(n/2) + n^2$:



n^2

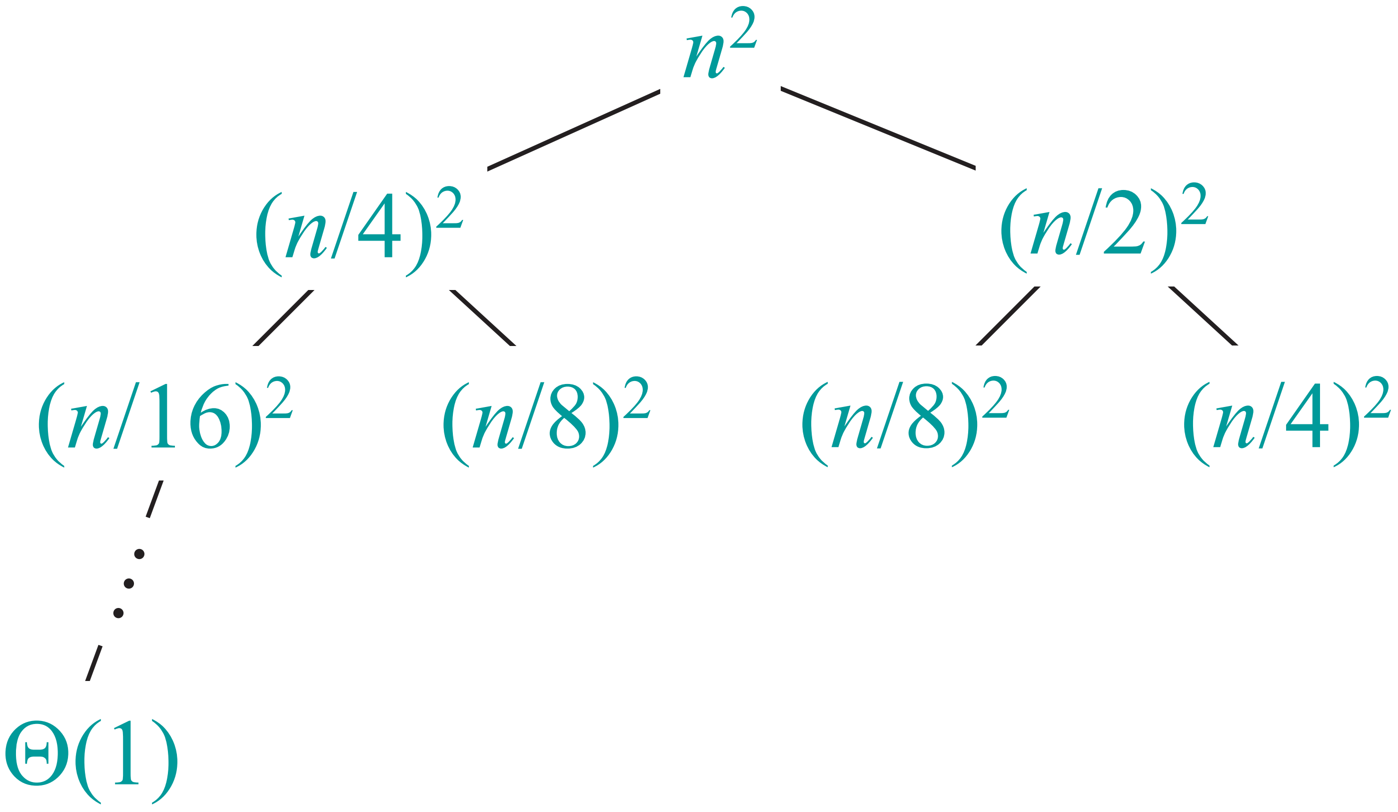
$(n/4)^2$ $(n/2)^2$

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



Example of recursion tree

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

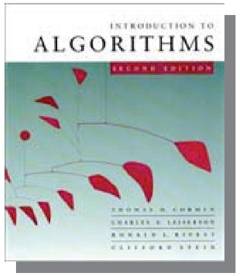


n^2

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

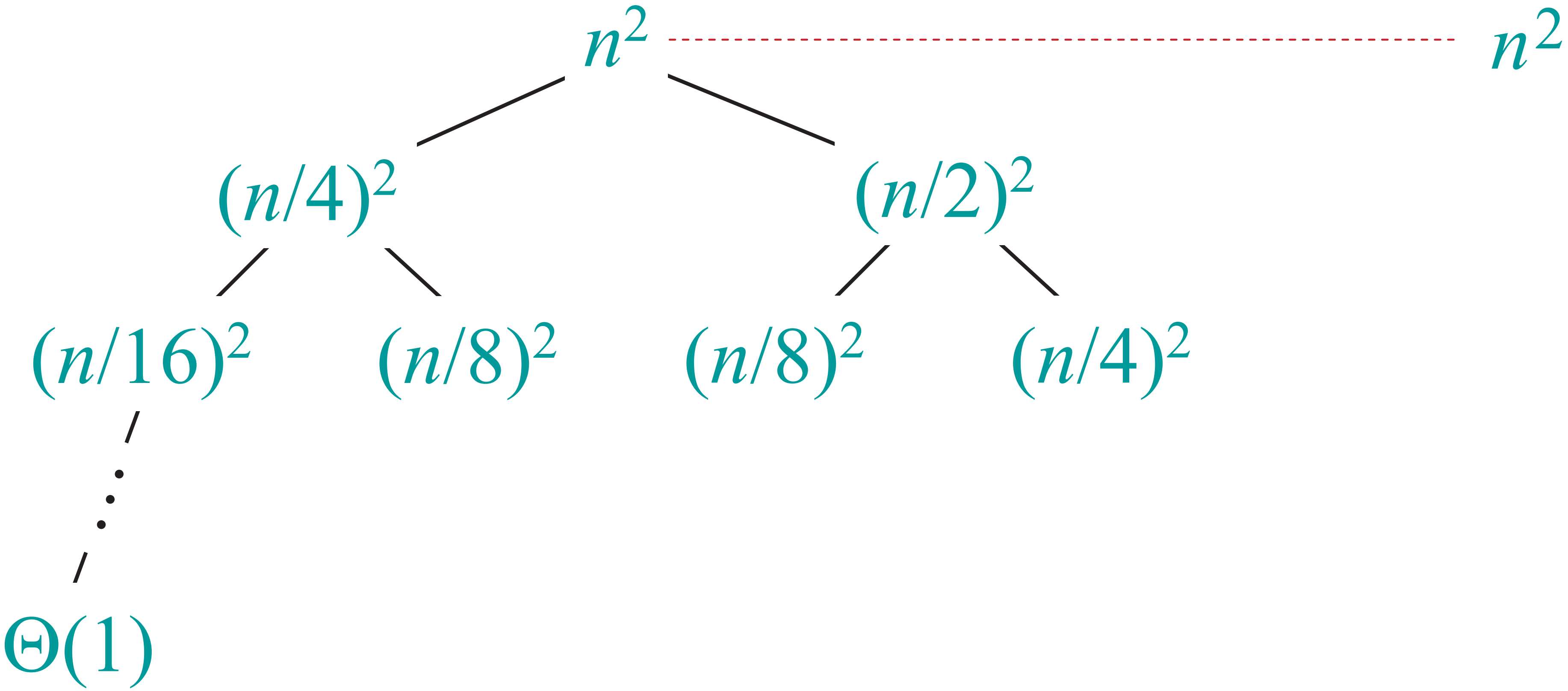
$(n/16)^2$ $(n/8)^2$ $(n/8)^2$ $(n/4)^2$

$\Theta(1)$



Example of recursion tree

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



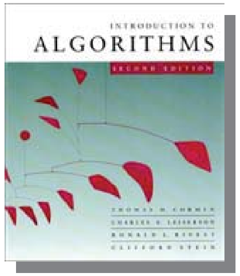
n^2 n^2

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

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

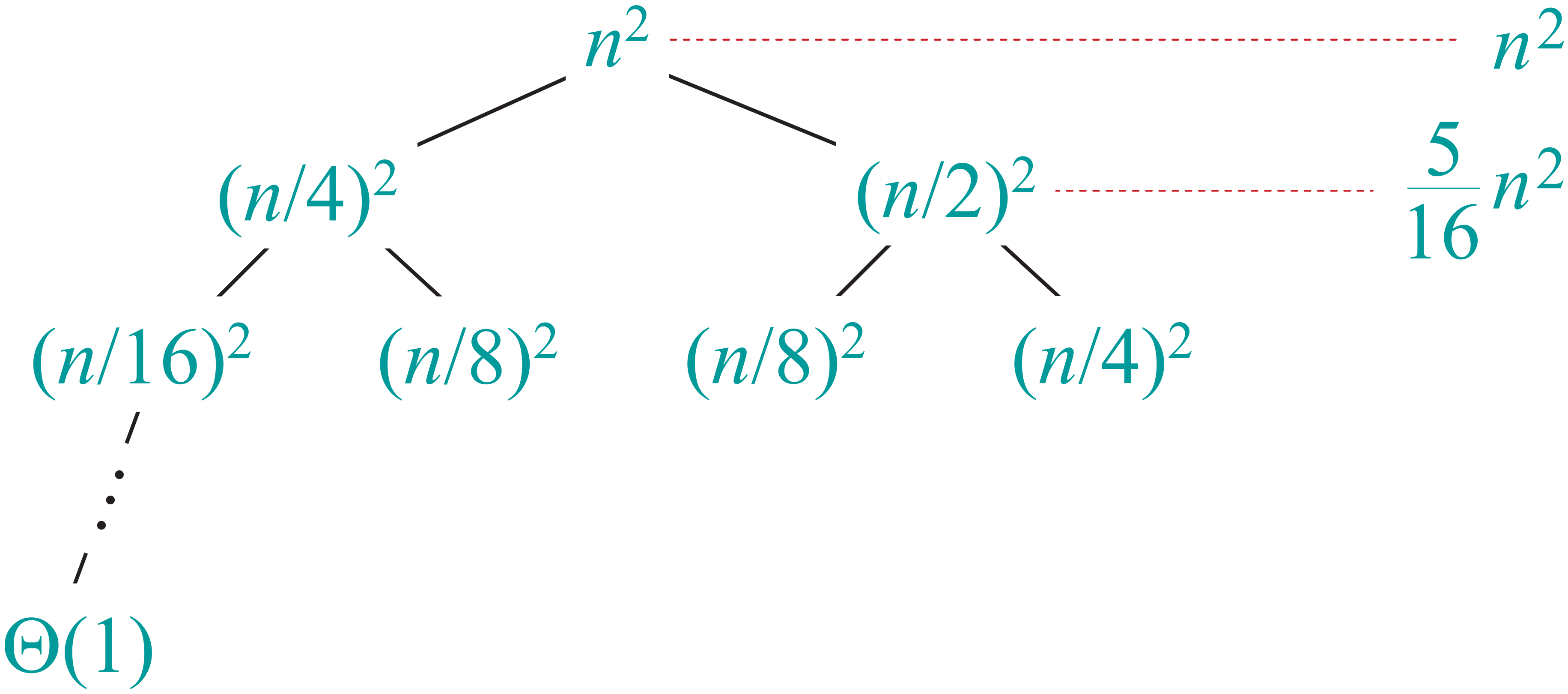
$\vdots$

$\Theta(1)$



Example of recursion tree

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

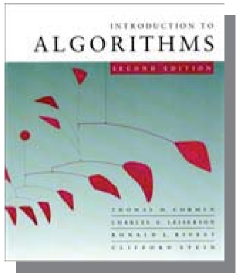


n^2 n^2

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

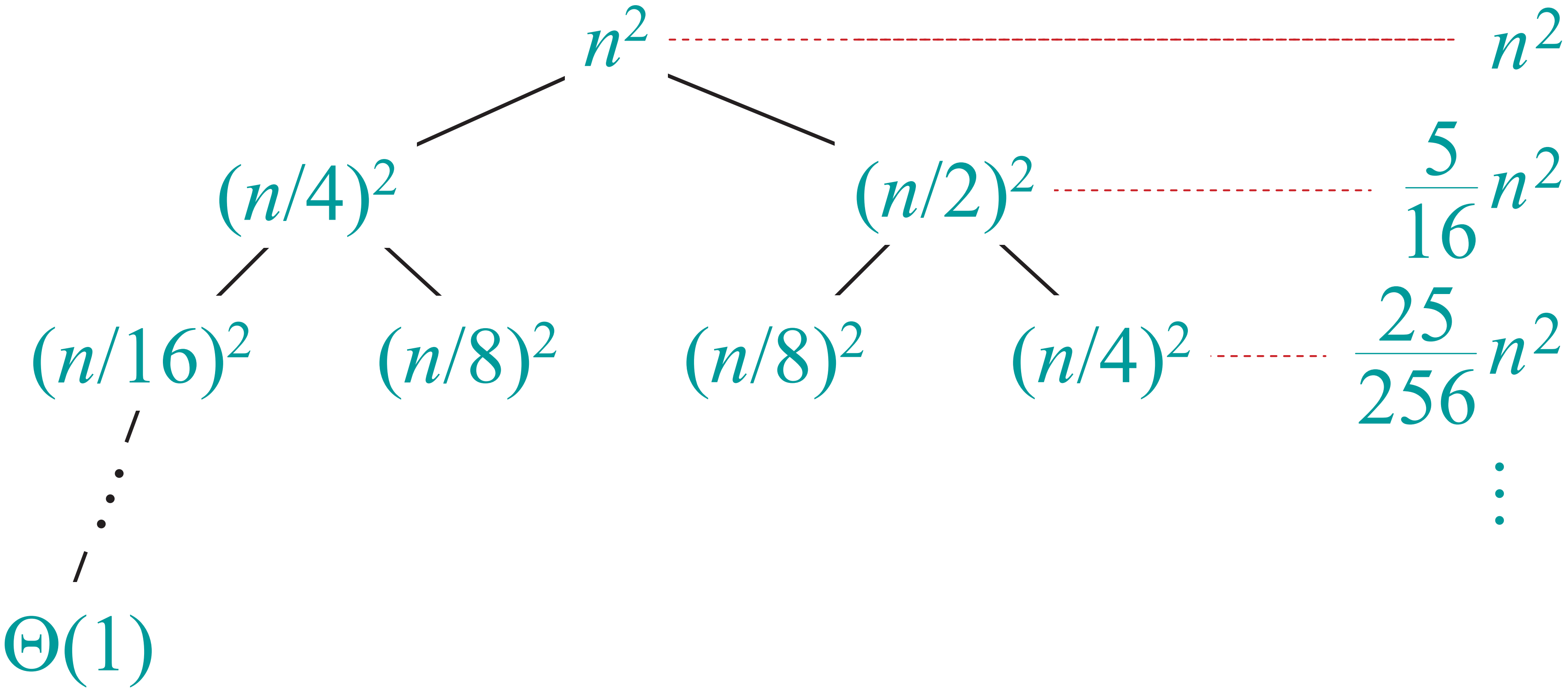
$(n/16)^2$ $(n/8)^2$ $(n/8)^2$ $(n/4)^2$

$\Theta(1)$



Example of recursion tree

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



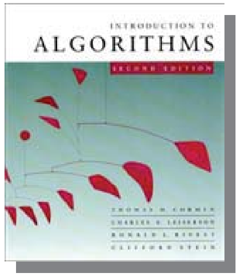
n^2 $\quad$ n^2

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

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

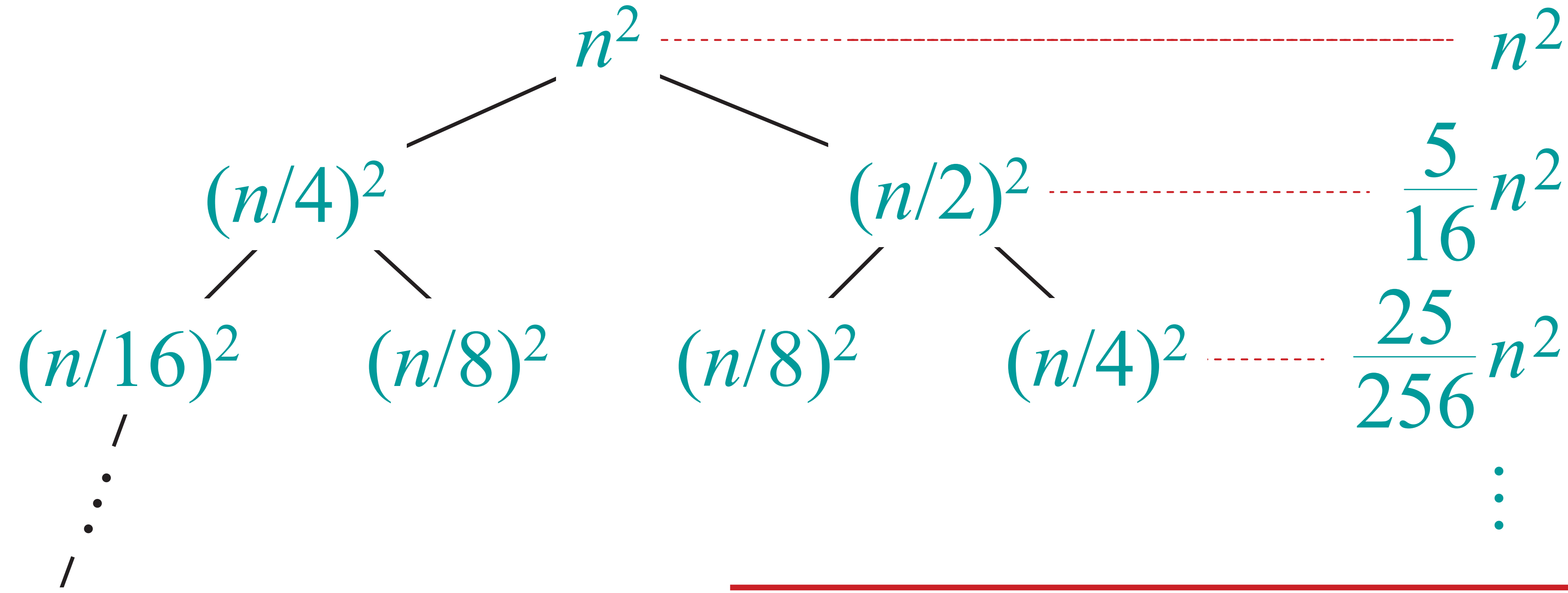
$\vdots$

$\Theta(1)$



Example of recursion tree

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

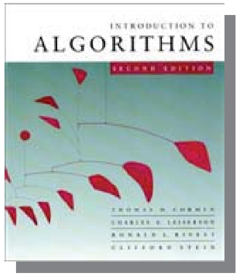


$\Theta(1)$

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

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

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.

Three common cases

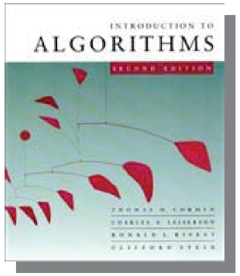
Compare $f(n)$ with $n^{\log_b a}$:

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. $f(n) = \Theta(n^{\log_b a})$.
   - $f(n)$ and $n^{\log_b a}$ grow at similar rates.

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



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 $af(n/b) \le cf(n)$ 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:

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

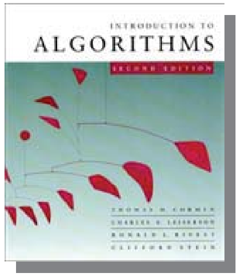
$\therefore \; T(n) = \Theta(n^2).$

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: $f(n) = n^2 = \Theta(n^2).$

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



Examples

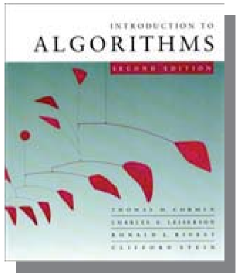
Ex. $T(n) = 4T(n/2) + n^3$

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

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

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

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



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 $\varepsilon = 1$

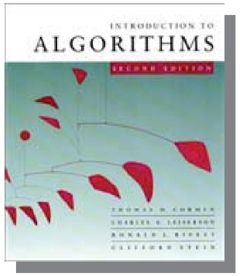
and $4(n/2)^3 \le cn^3$ (reg. cond.) for $c = 1/2$.

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

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

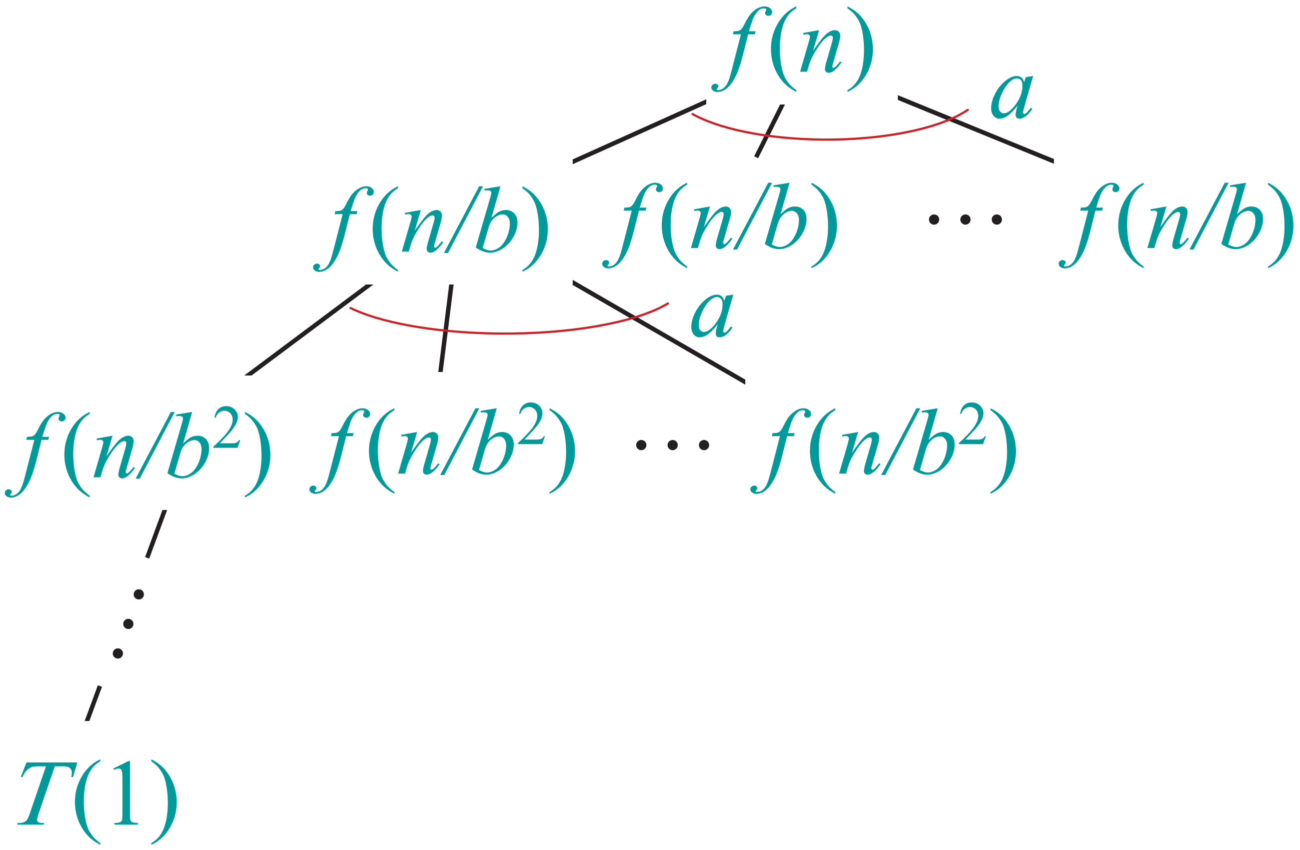
$a = 4, b = 2 \Rightarrow n^{\log_b a} = n^2; f(n) = n^2/\lg n.$

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



Idea of master theorem

Recursion tree:



$f(n)$

a

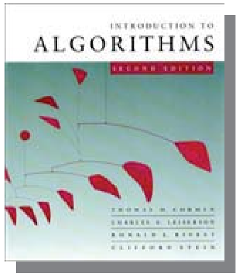
$f(n/b)$ $f(n/b)$ $\cdots$ $f(n/b)$

a

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

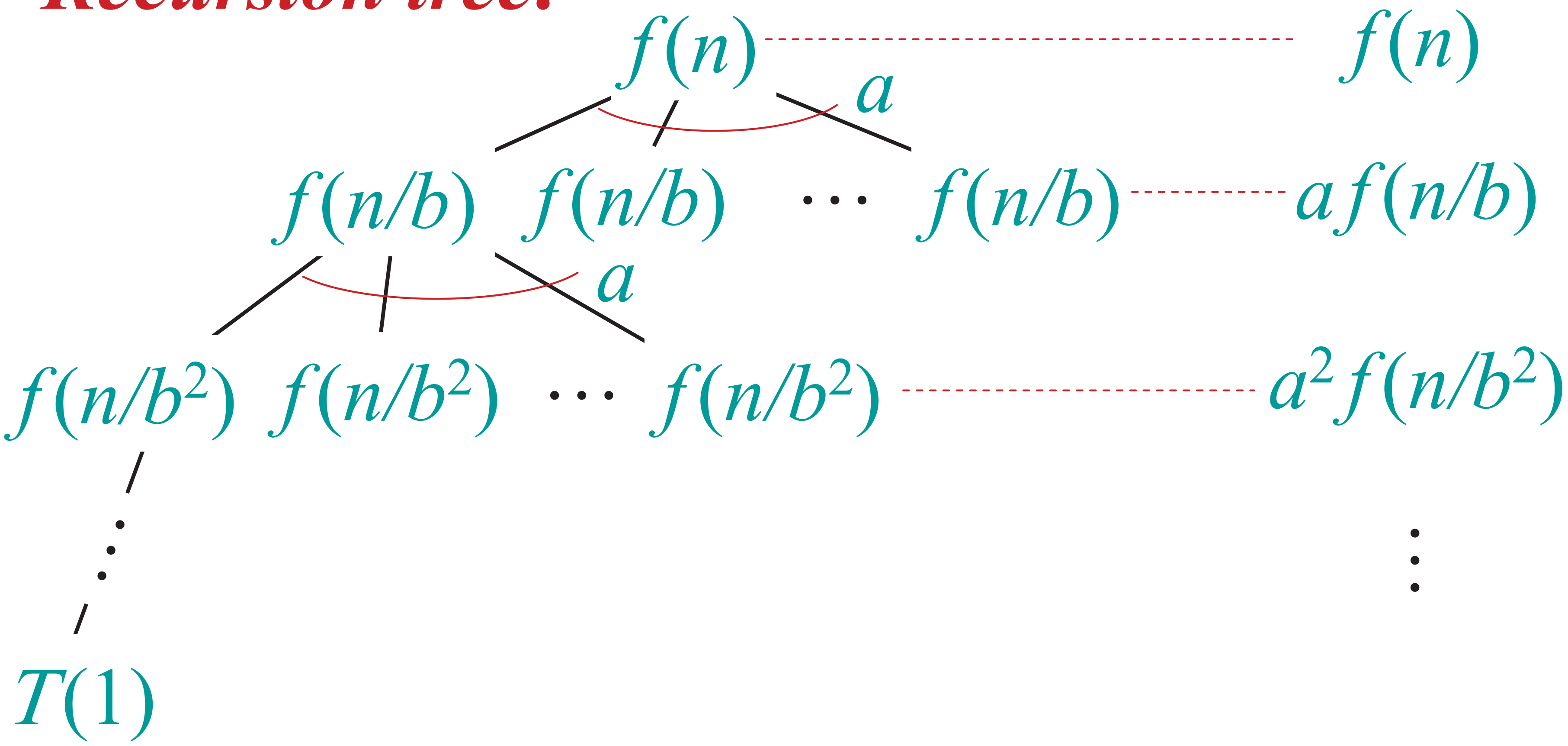
$\vdots$

$T(1)$



Idea of master theorem

Recursion tree:



$f(n)$

a

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

a

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

$\vdots$

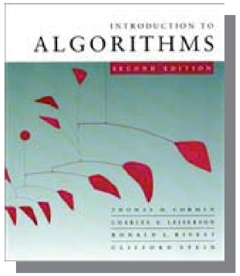
$T(1)$

$f(n)$

$a f(n/b)$

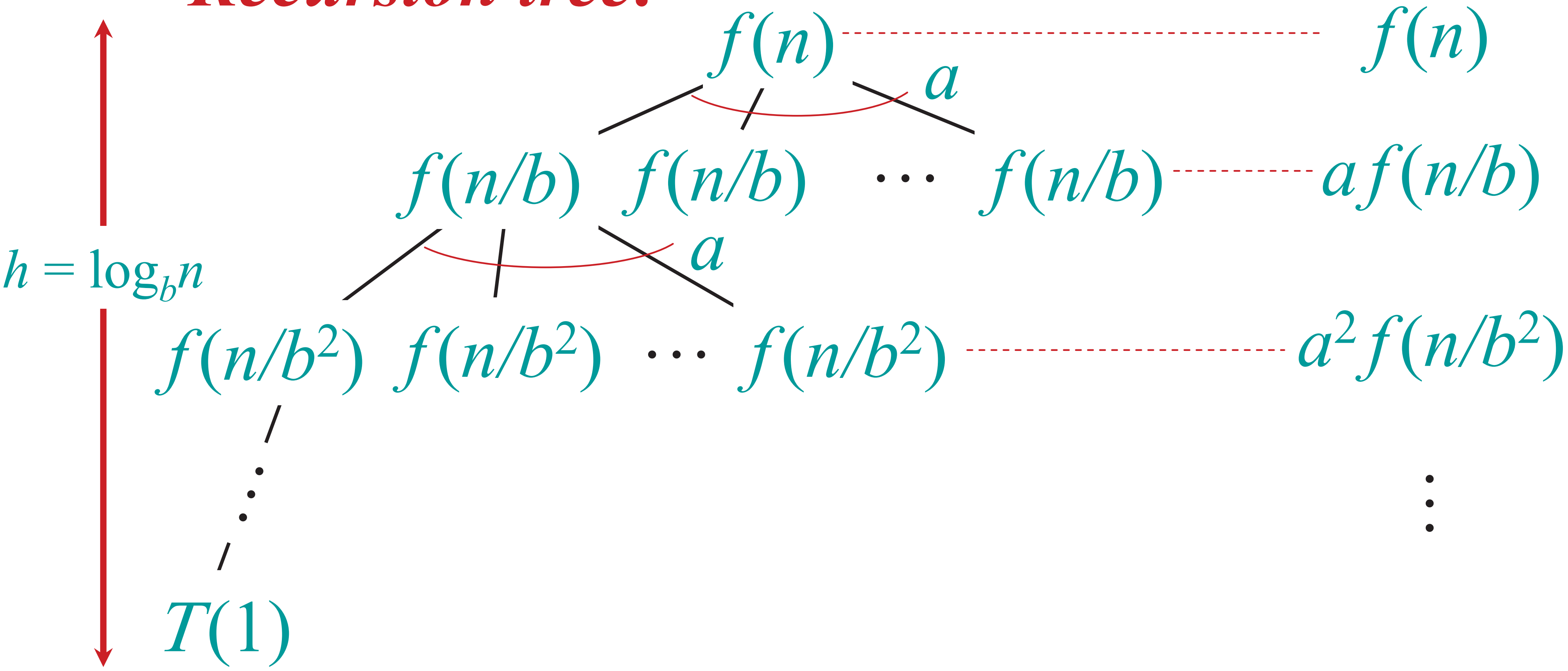
$a^2 f(n/b^2)$

$\vdots$



Idea of master theorem

Recursion tree:



$f(n)$ a $f(n)$

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

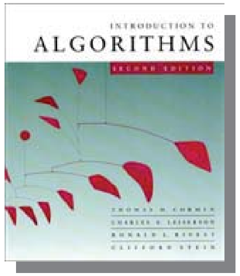
$h = \log_b n$ a

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

$\vdots$ $\vdots$

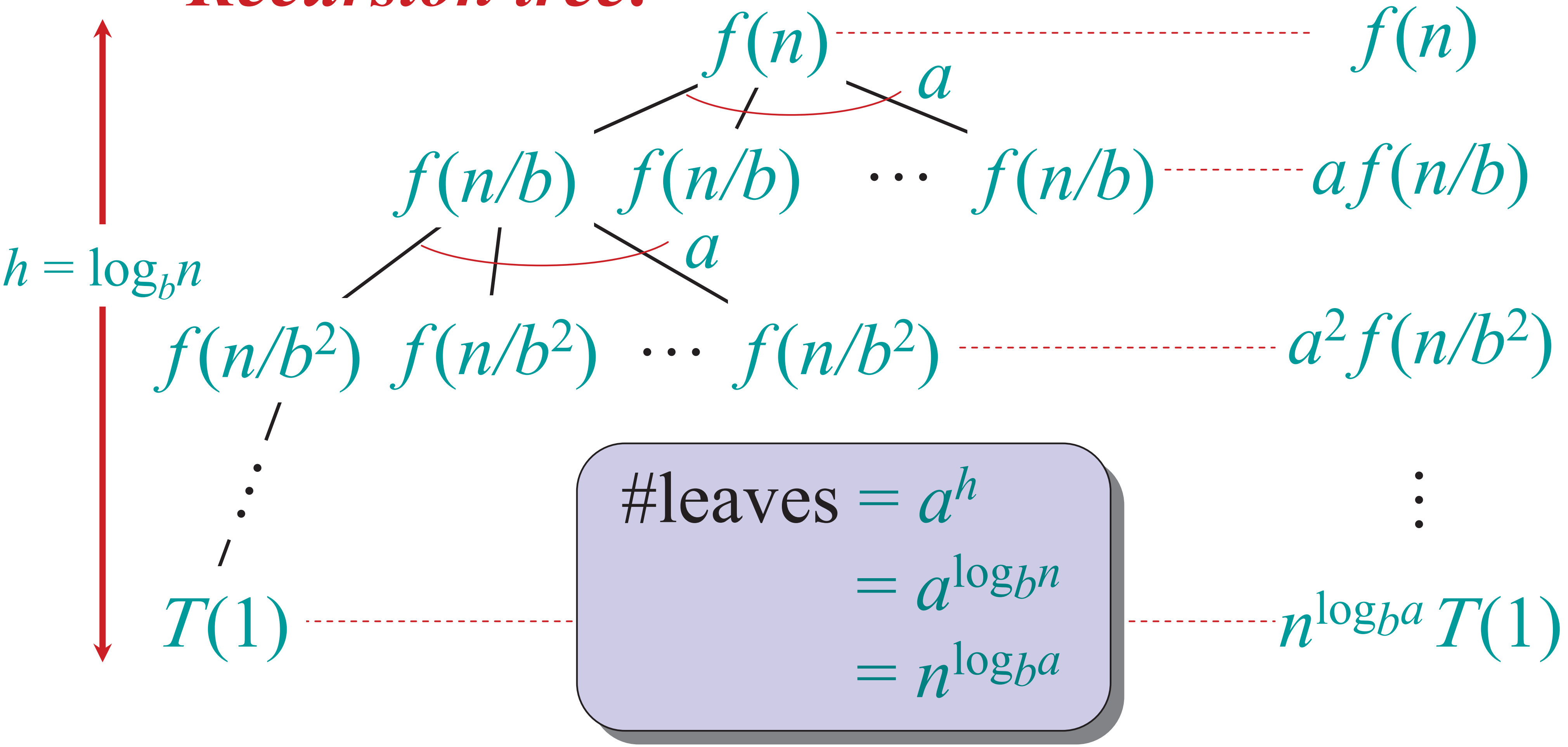
$T(1)$

by Charles E. Leiserson



Idea of master theorem

Recursion tree:



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

a

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

a

$h = \log_b n$

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

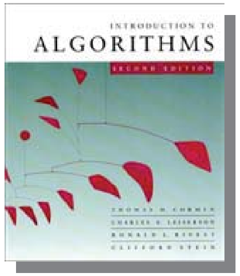
$\vdots$

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

#leaves $= a^h$

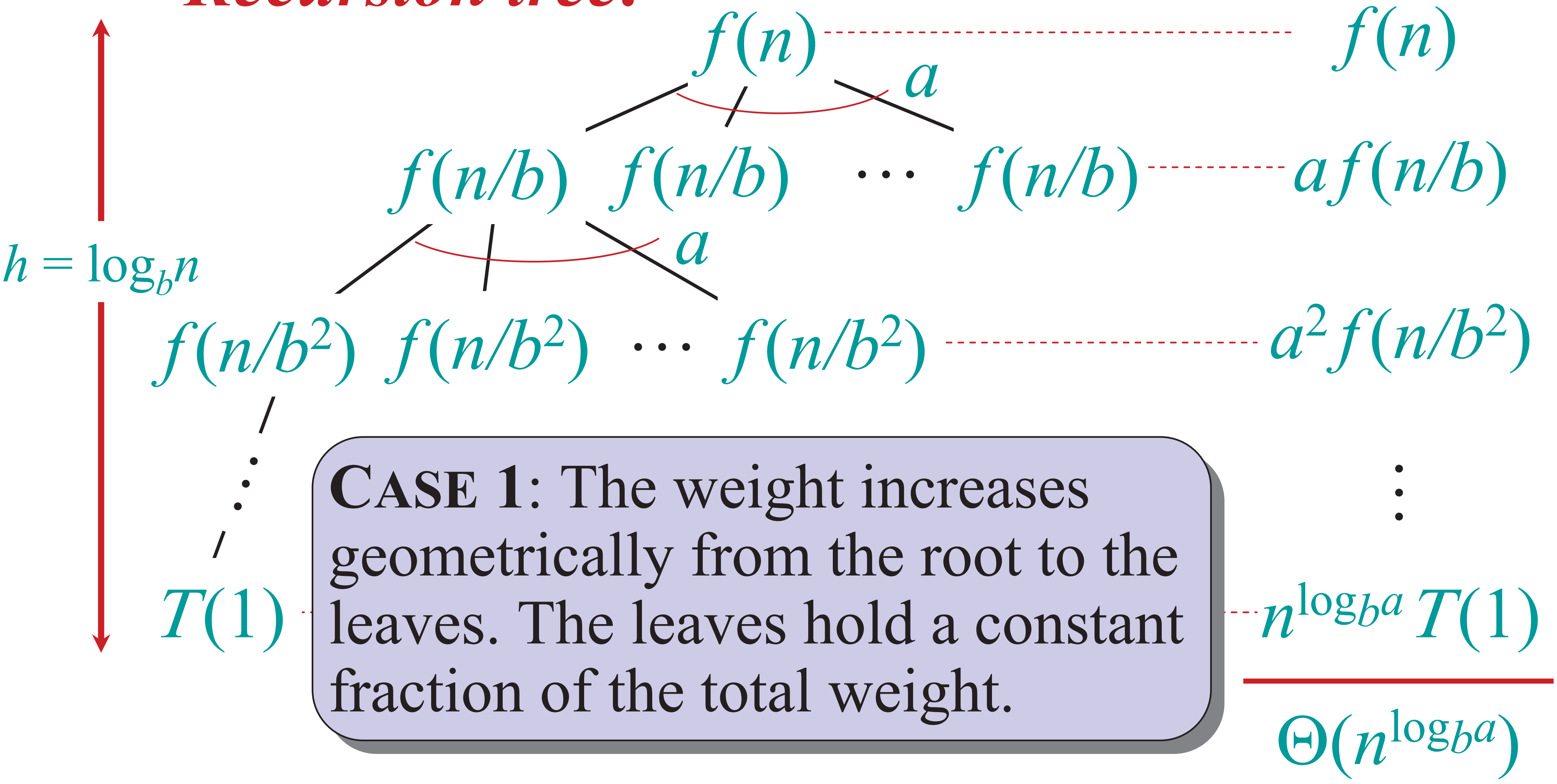
$= a^{\log_b n}$

$= n^{\log_b a}$



Idea of master theorem

Recursion tree:



$f(n)$ .... $f(n)$

a

$f(n/b)$ $f(n/b)$ $\cdots$ $f(n/b)$ .... $af(n/b)$

a

$h = \log_b n$

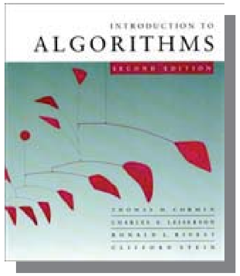
$f(n/b^2)$ $f(n/b^2)$ $\cdots$ $f(n/b^2)$ .... $a^2 f(n/b^2)$

$\vdots$

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

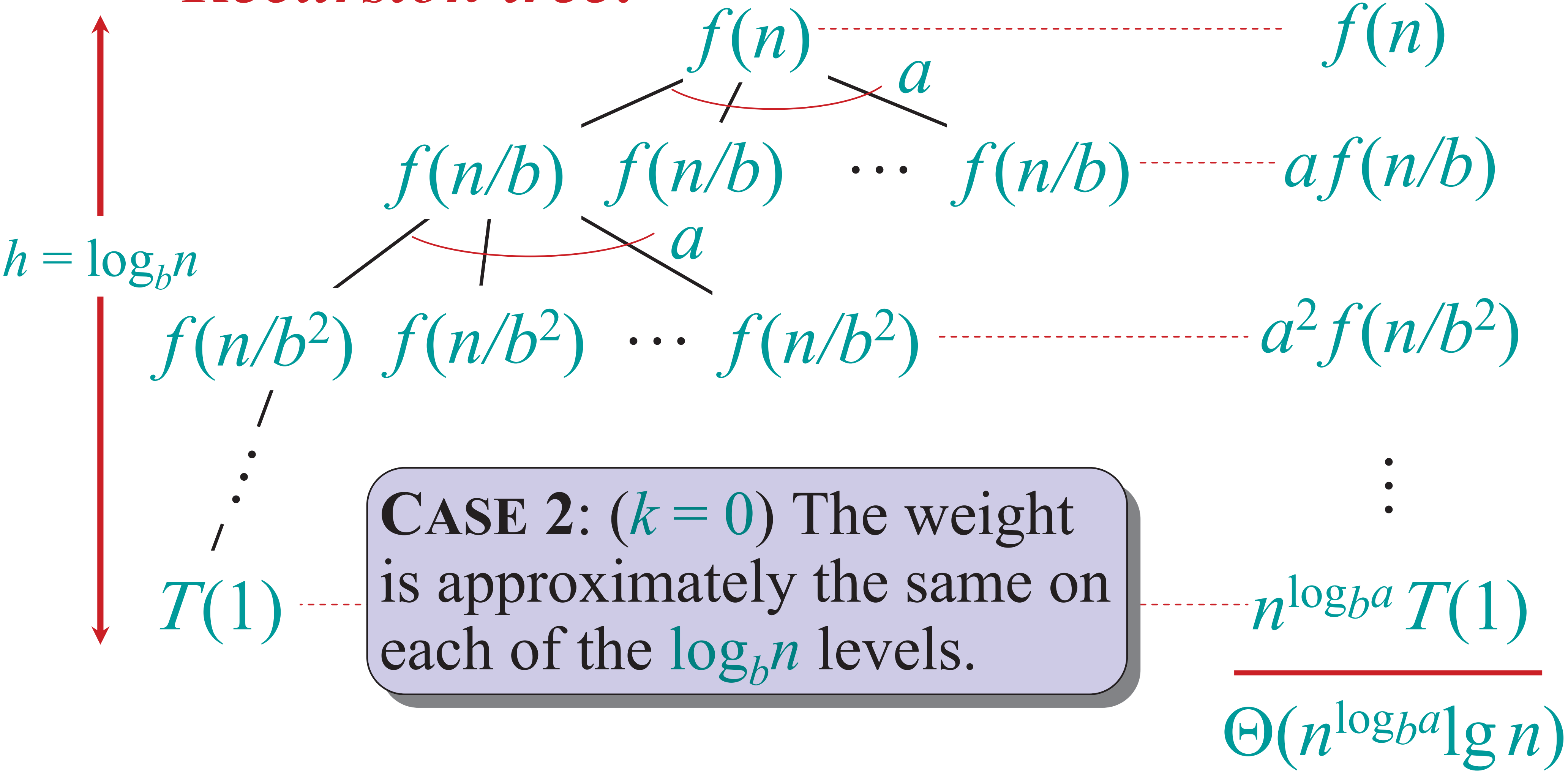
$\Theta(n^{\log_b a})$

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:



$f(n)$ ---------- $f(n)$

a

$f(n/b)$ $f(n/b)$ $\cdots$ $f(n/b)$ ---------- $af(n/b)$

a

$h = \log_b n$

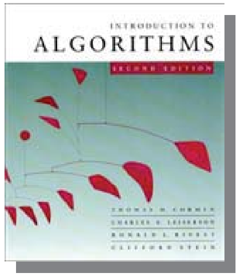
$f(n/b^2)$ $f(n/b^2)$ $\cdots$ $f(n/b^2)$ ---------- $a^2 f(n/b^2)$

$\vdots$

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

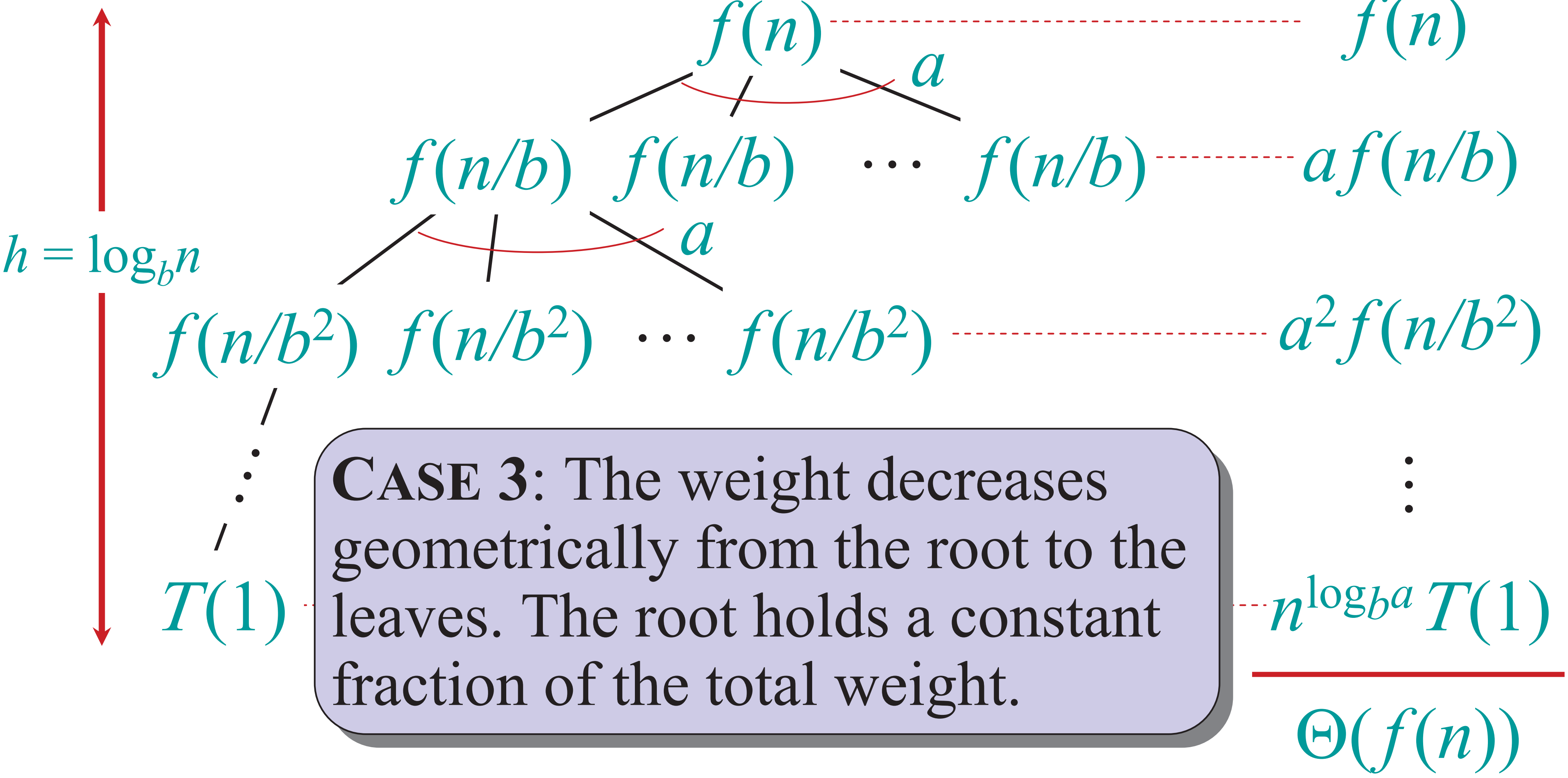
CASE 2: ($k = 0$) The weight is approximately the same on each of the $\log_b n$ levels.

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



Idea of master theorem

Recursion tree:



$f(n)$ ………………………… $f(n)$

a

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

a

$h = \log_b n$

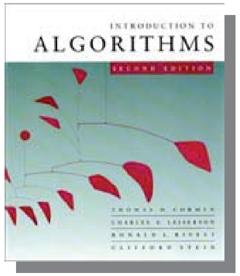
$f(n/b^2)$ $f(n/b^2)$ $\cdots$ $f(n/b^2)$ ………… $a^2 f(n/b^2)$

$\vdots$

$T(1)$ ………… $n^{\log_b a}\,T(1)$

$\Theta(f(n))$

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



INTRODUCTION TO
ALGORITHMS

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.