CSE 548: Analysis of Algorithms

Lecture 5
(Divide-and-Conquer Algorithms: The Master Theorem)

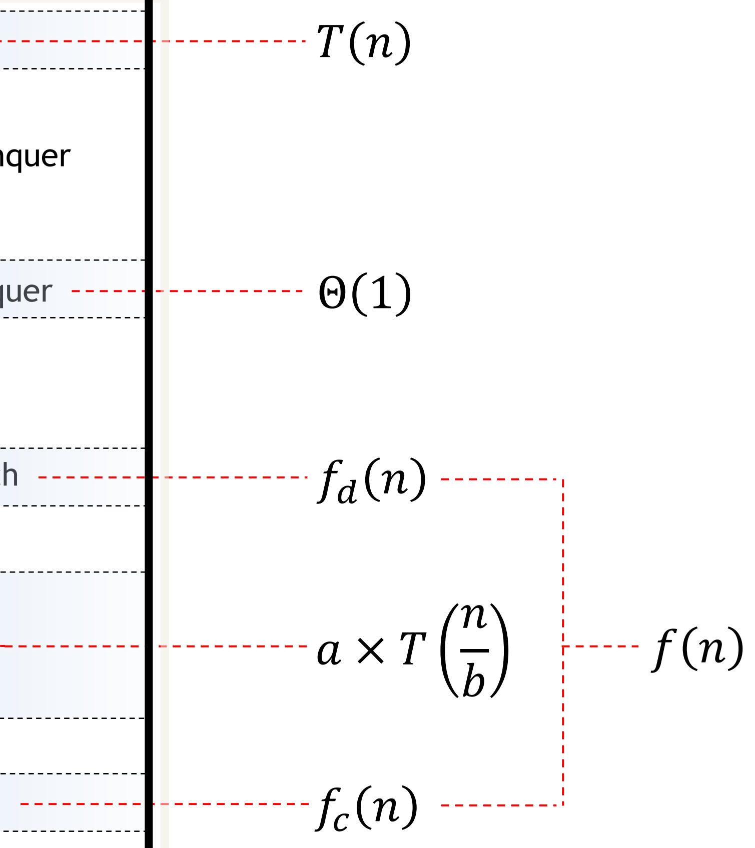
Rezaul A. Chowdhury
Department of Computer Science
SUNY Stony Brook
Fall 2023

A Recurrence for Recursive Divide & Conquer

```
RECURSIVE-DIVIDE-AND-CONQUER ( $P$, $n$ )

INPUT: A problem $P$ of size $n \geq 1$
OUTPUT: Solution $S$ of $P$ computed using recursive divide and conquer

1.  if $n = 1$ then
2.      $S \leftarrow$ solution of $P$ computed directly w/o divide & conquer
3.  else
4.      DIVIDE:
5.          divide $P$ into $a$ subproblems $P_1, P_2, \ldots, P_a$ of size $\frac{n}{b}$ each
6.      CONQUER:
7.          for $i = 1$ to $a$ do
8.              $S_i \leftarrow$ RECURSIVE-DIVIDE-AND-CONQUER ( $P_i$, $\frac{n}{b}$ )
9.      COMBINE:
10.         $S \leftarrow$ solution of $P$ obtained by combining $S_1, S_2, \ldots, S_a$
11. endif
12. return $S$
```



$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$

A Recurrence for Recursive Divide & Conquer

The recurrence:

$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise; \end{cases}$$

where, $a \geq 1$ and $b > 1$.

Arises frequently in the analyses of *divide-and-conquer* algorithms.

Consider the following recurrences.

Karatsuba's Integer Multiplication Algorithm: $T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n)$

Strassen's Matrix Multiplication Algorithm: $T(n) = 7T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)$

Fast Fourier Transform: $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$

How the Recurrence Unfolds

$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$

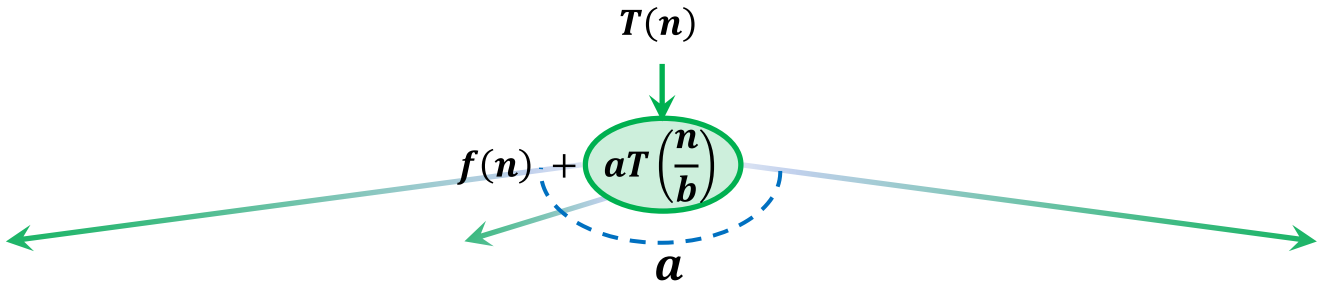
How the Recurrence Unfolds

$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$

$$\begin{tikzcd} \boldsymbol{T(n)} \arrow[d] \\ \boldsymbol{f(n)\ +\ aT\left(\frac{n}{b}\right)} \end{tikzcd}$$

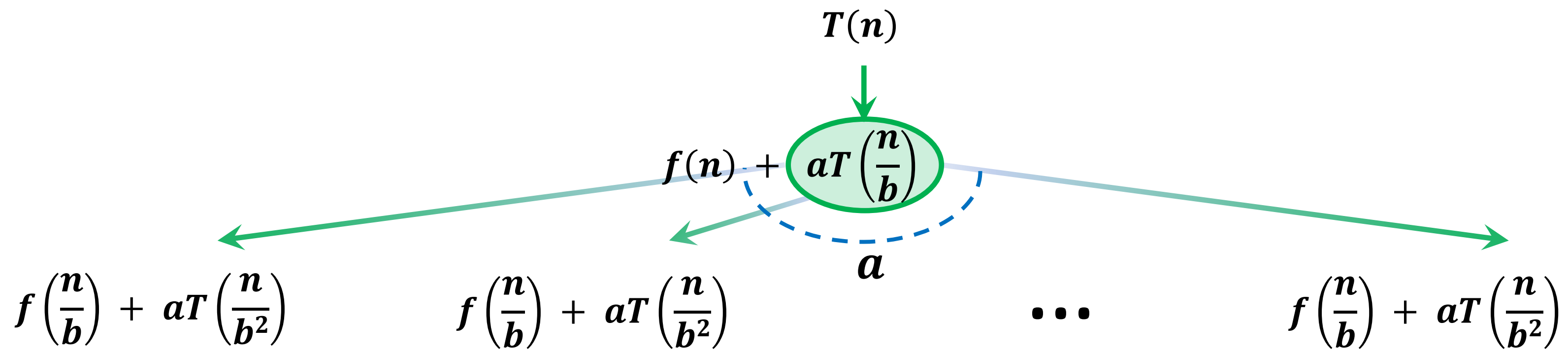
How the Recurrence Unfolds

$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



How the Recurrence Unfolds

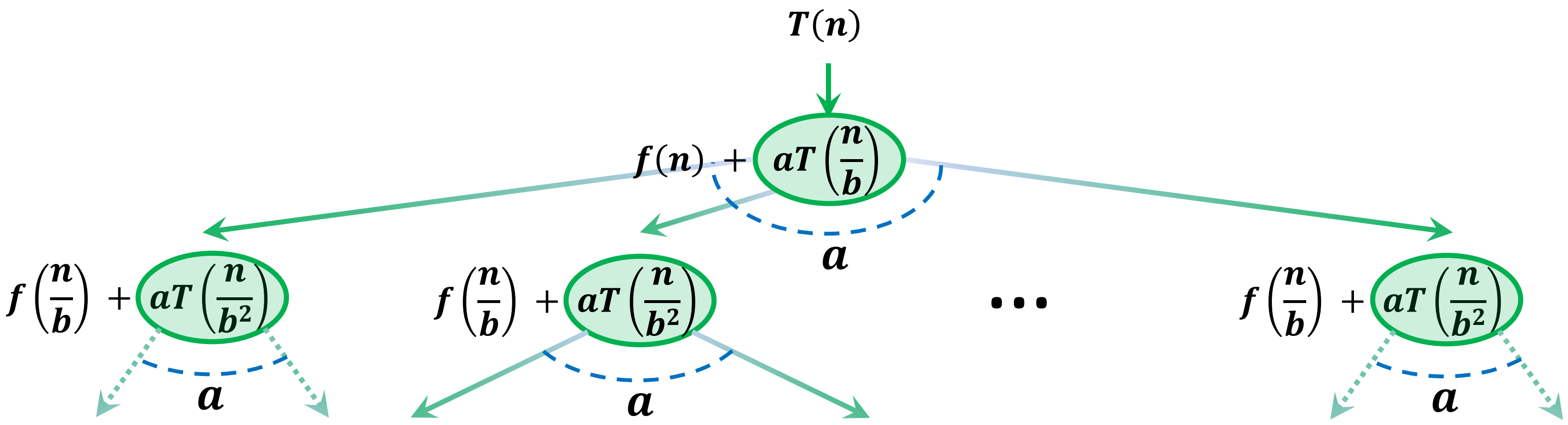
$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
…
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$

How the Recurrence Unfolds

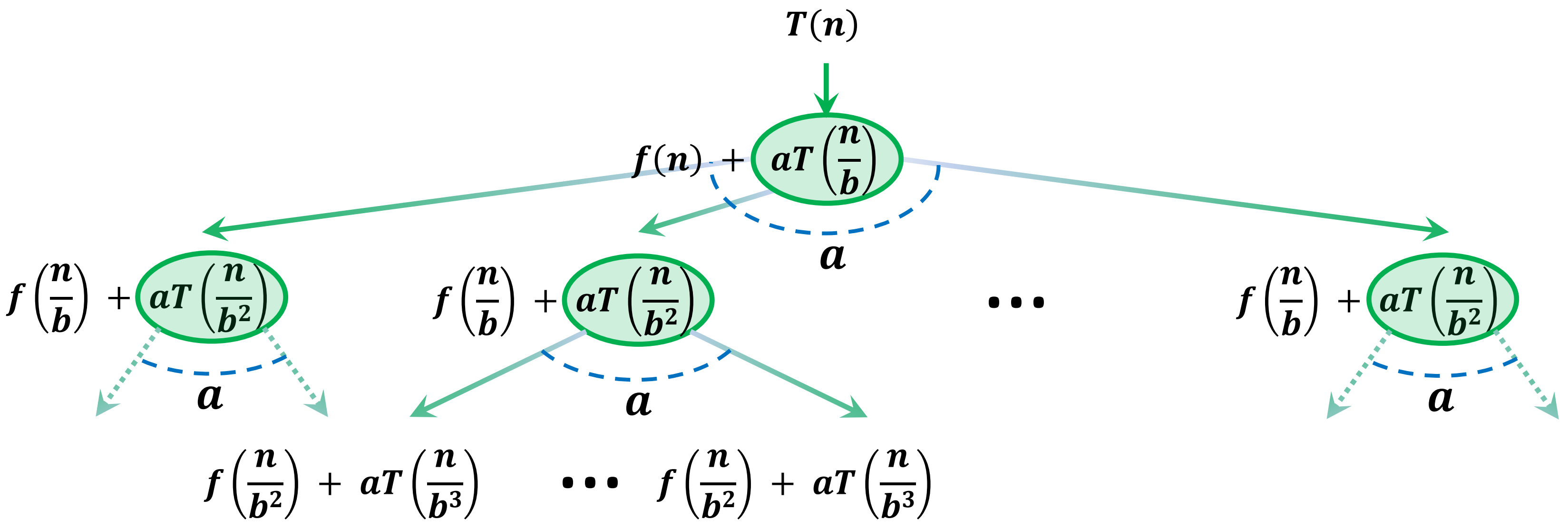
$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$\cdots$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a

How the Recurrence Unfolds

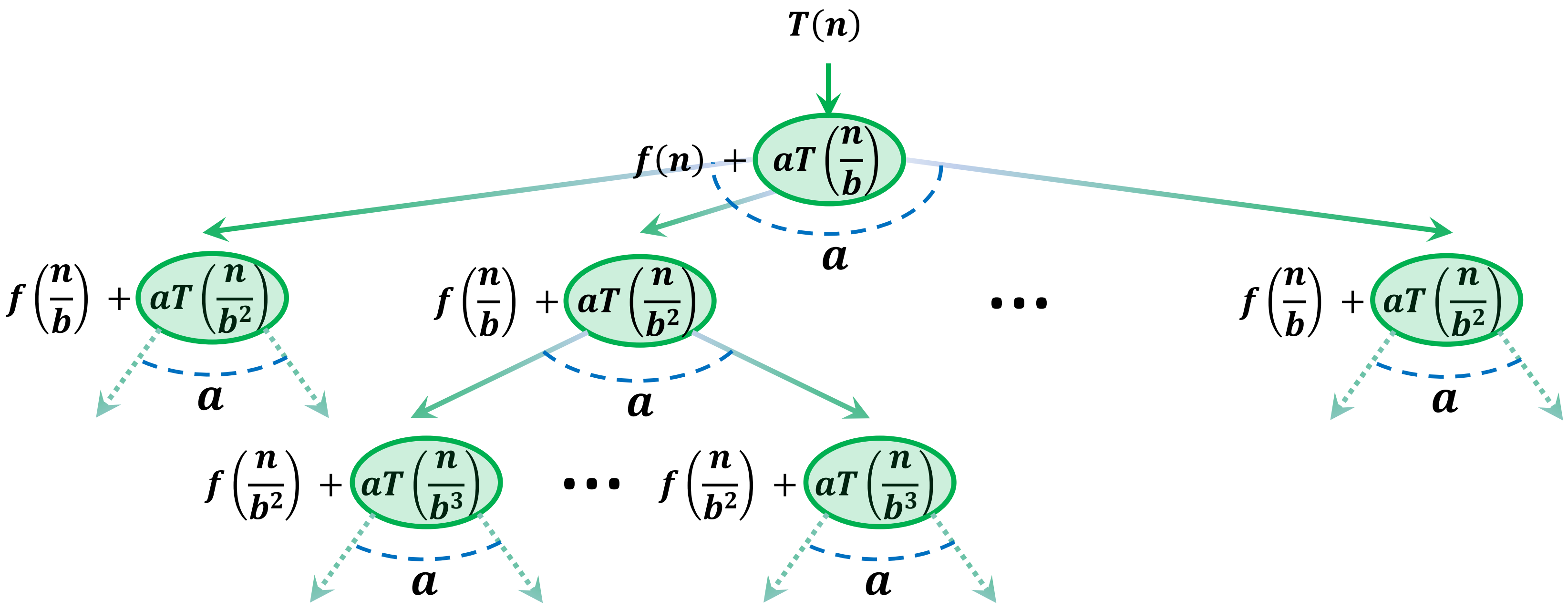
$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$\cdots$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
$\cdots$
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$

How the Recurrence Unfolds

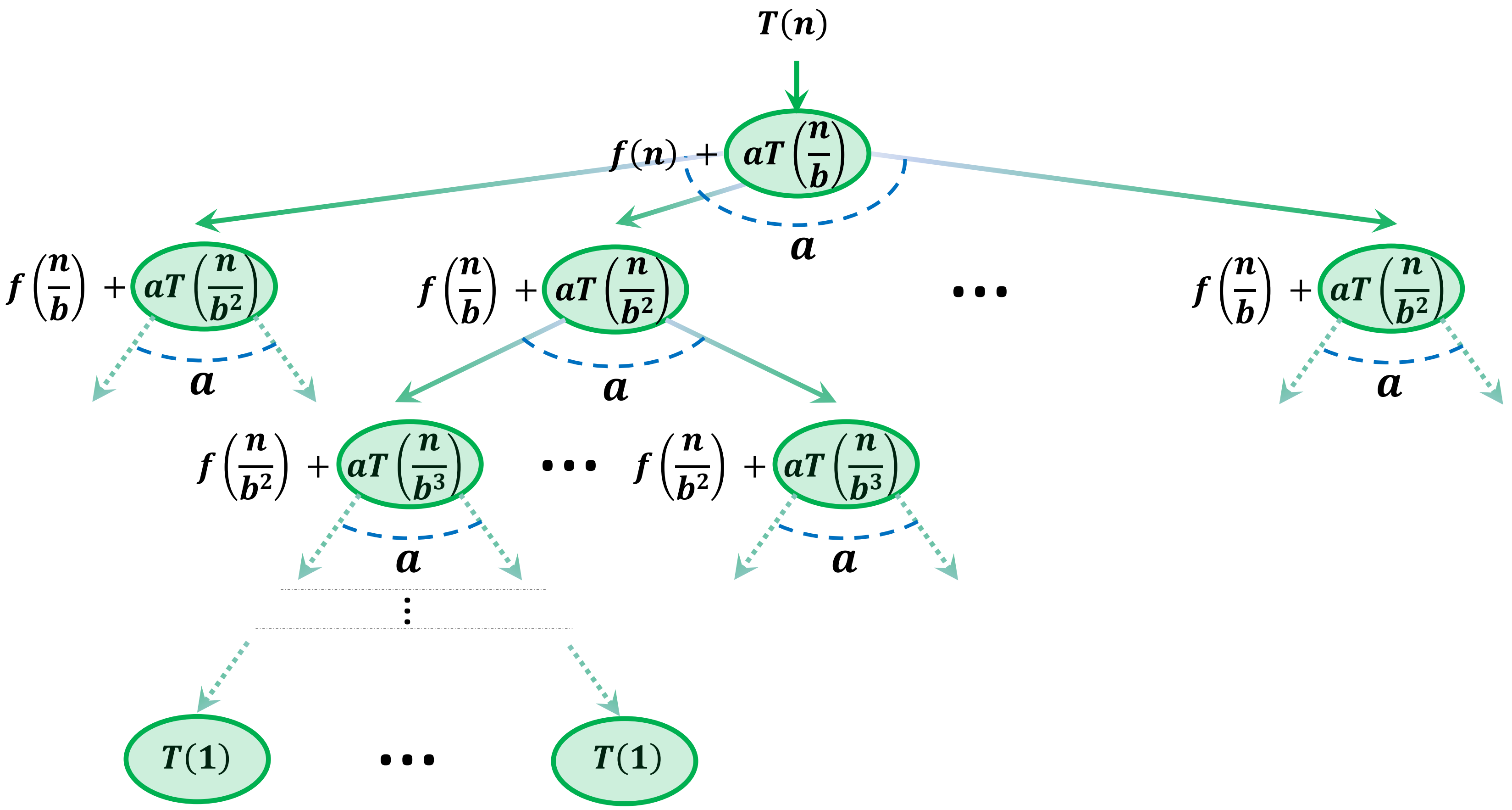
$$T(n) = \begin{cases} \Theta(1), & \text{if } n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & \text{otherwise.} \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$\cdots$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
a
$\cdots$
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
a

How the Recurrence Unfolds

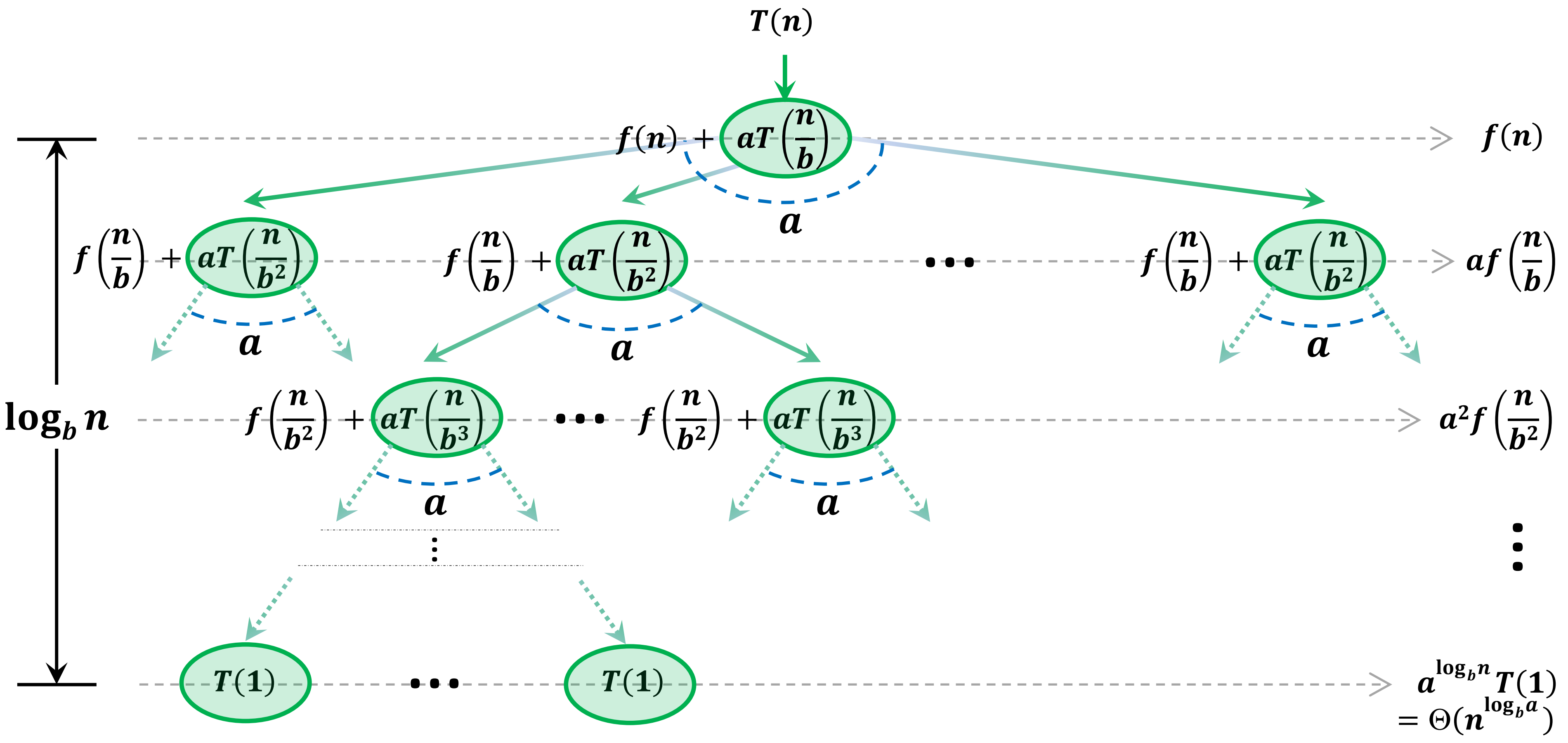
$$T(n) = \begin{cases} \Theta(1), & \text{if } n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & \text{otherwise.} \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$\cdots$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
a
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
a
$\cdots$
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
a
$\vdots$
$T(1)$
$\cdots$
$T(1)$

How the Recurrence Unfolds

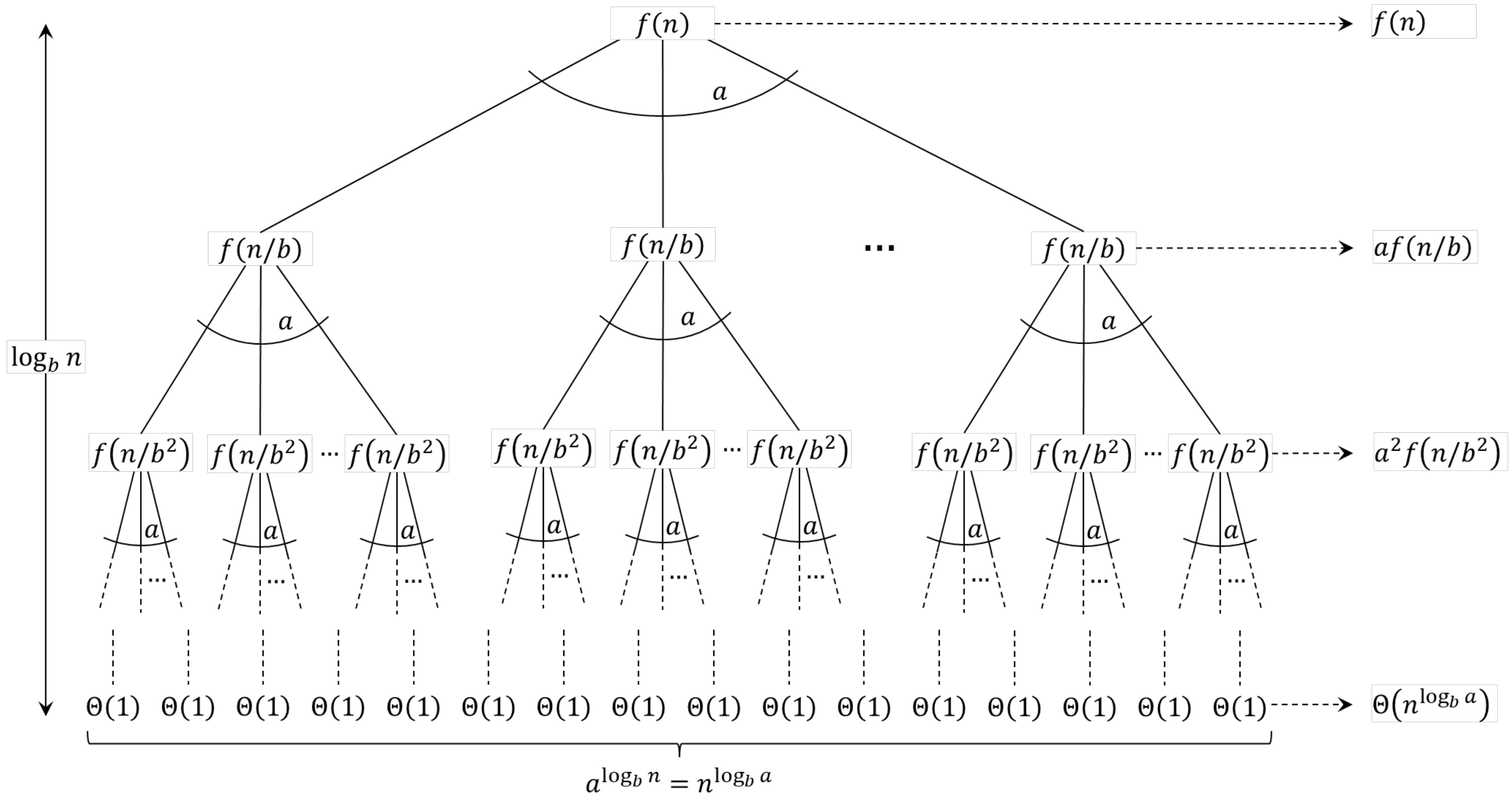
$$T(n) = \begin{cases} \Theta(1), & \text{if } n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & \text{otherwise.} \end{cases}$$



$T(n)$
$f(n) + aT\left(\frac{n}{b}\right)$
a
$f(n)$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
$\cdots$
$f\left(\frac{n}{b}\right) + aT\left(\frac{n}{b^2}\right)$
$af\left(\frac{n}{b}\right)$
a
a
a
$\log_b n$
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
$\cdots$
$f\left(\frac{n}{b^2}\right) + aT\left(\frac{n}{b^3}\right)$
$a^2 f\left(\frac{n}{b^2}\right)$
a
a
$\vdots$
$T(1)$
$\cdots$
$T(1)$
$a^{\log_b n} T(1) = \Theta(n^{\log_b a})$

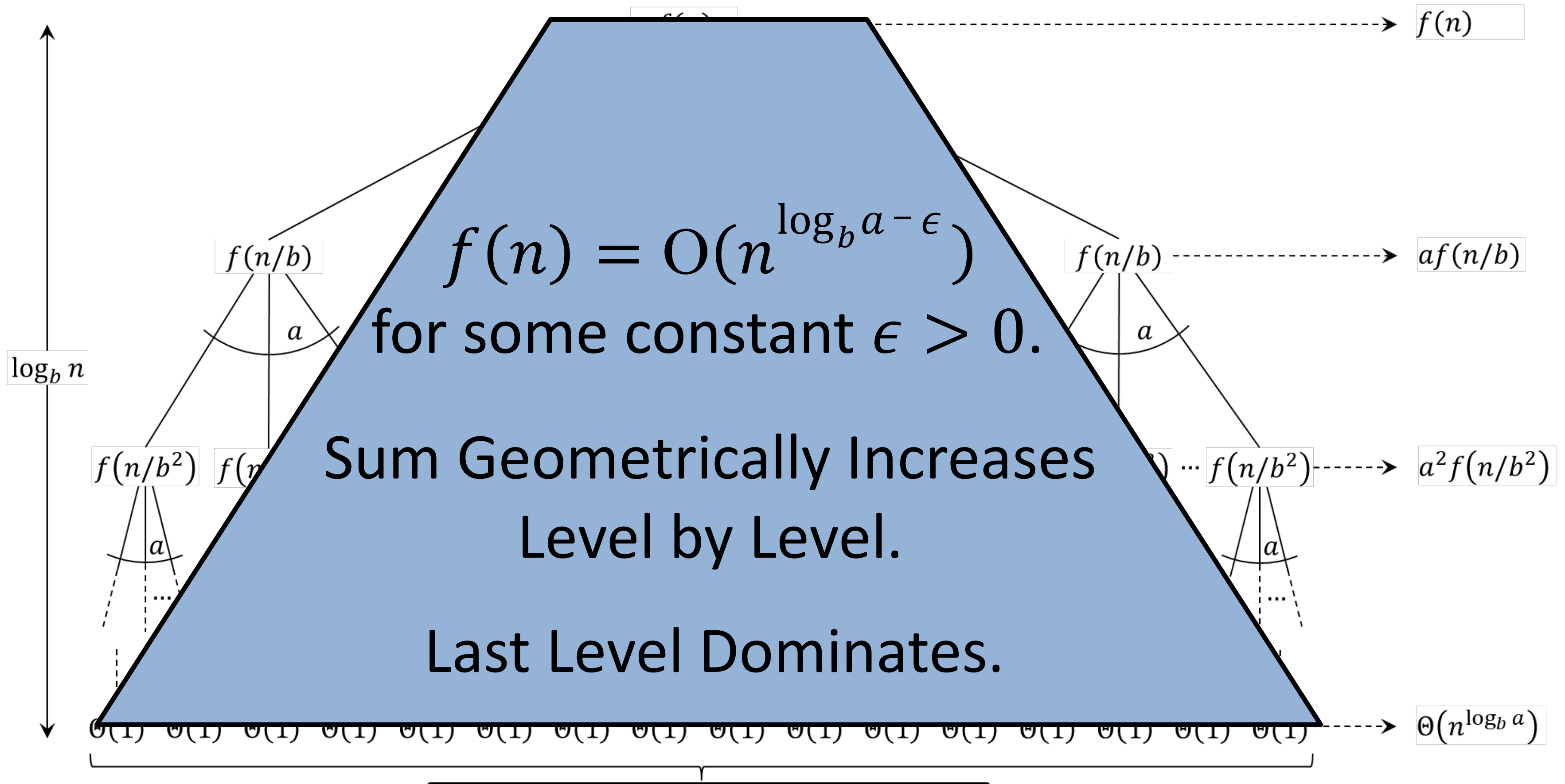
How the Recurrence Unfolds

$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



How the Recurrence Unfolds: Case 1

$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



$f(n) = O(n^{\log_b a - \epsilon})$

for some constant $\epsilon > 0$.

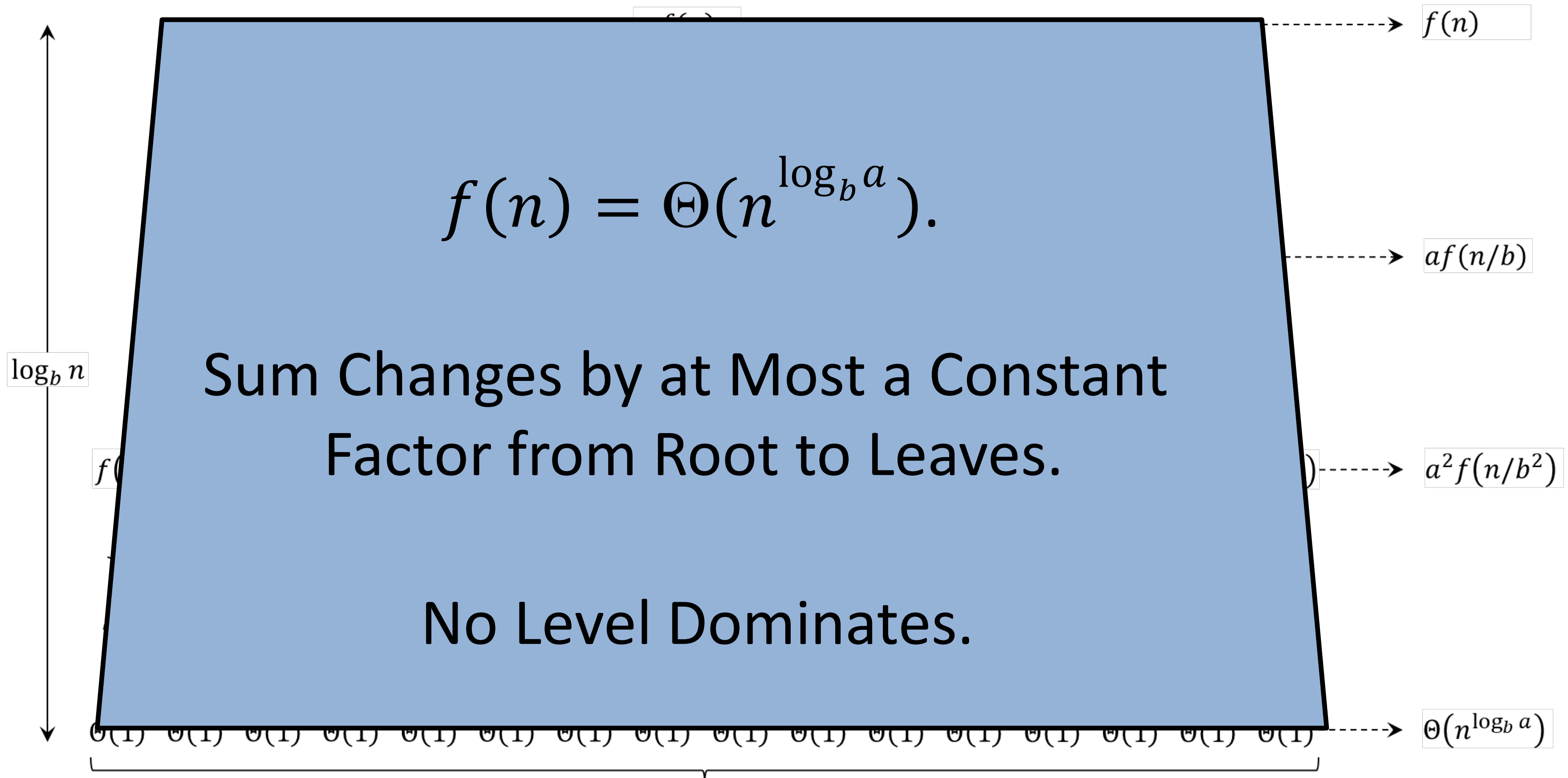
Sum Geometrically Increases Level by Level.

Last Level Dominates.

$$\boxed{\boldsymbol{T(n) = \Theta(n^{\log_b a})}}$$

How the Recurrence Unfolds: Case 2

$$T(n) = \begin{cases} \Theta(1), & \text{if } n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & \text{otherwise.} \end{cases}$$



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

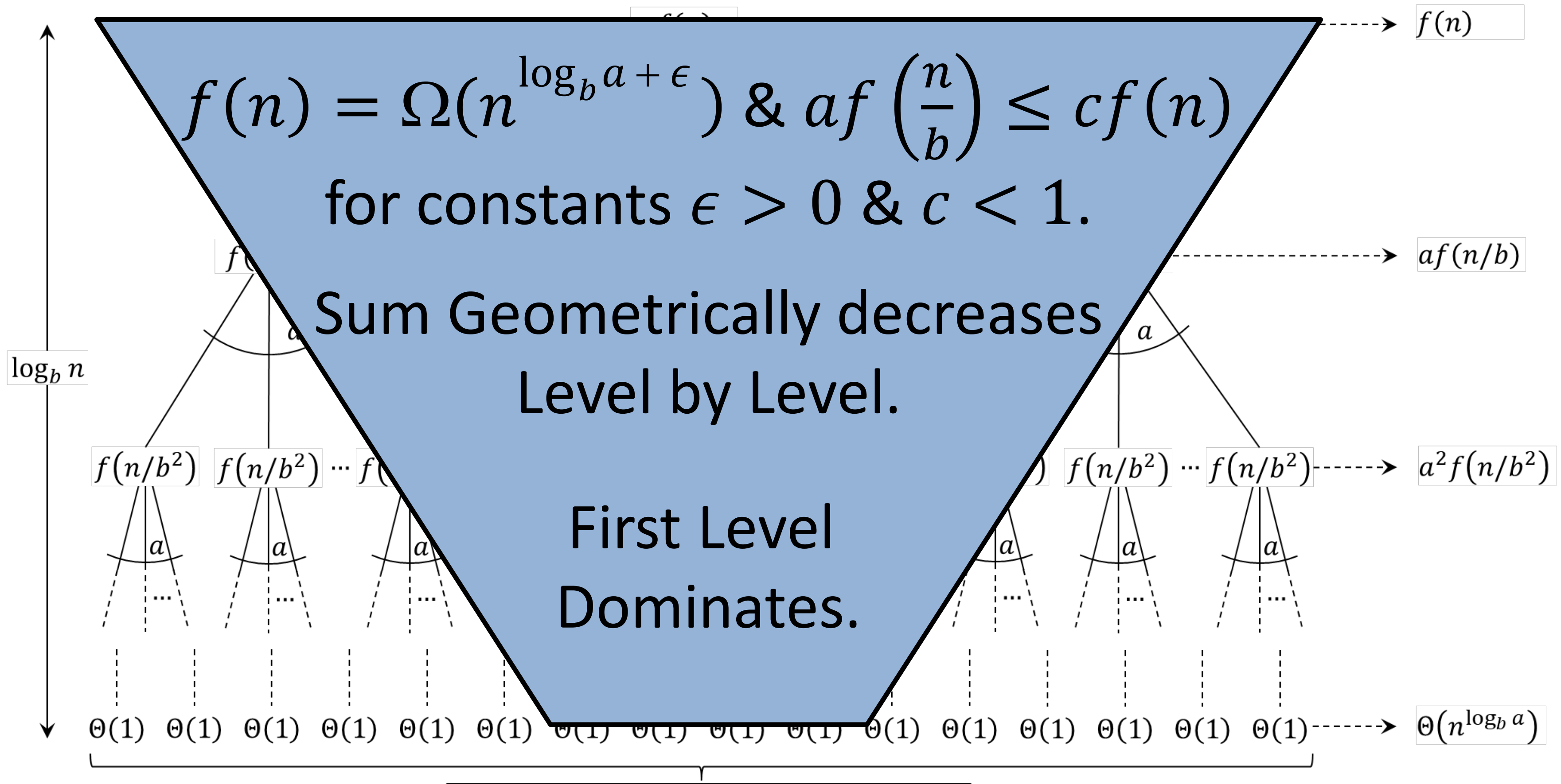
Sum Changes by at Most a Constant Factor from Root to Leaves.

No Level Dominates.

$$\boxed{T(n) = \Theta(n^{\log_b a} \log n)}$$

How the Recurrence Unfolds: Case 3

$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



$f(n) = \Omega(n^{\log_b a + \epsilon})$ & $af\left(\frac{n}{b}\right) \le cf(n)$

for constants $\epsilon > 0$ & $c < 1$.

Sum Geometrically decreases Level by Level.

First Level Dominates.

$$\boxed{T(n) = \Theta(f(n))}$$

The Master Theorem

$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise\ (a \geq 1, b > 1). \end{cases}$$

Case 1: $f(n) = \mathrm{O}(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$

$$\boxed{T(n) = \Theta(n^{\log_b a})}$$

Case 2: $f(n) = \Theta(n^{\log_b a})$

$$\boxed{T(n) = \Theta\left(n^{\log_b a} \log n\right)}$$

Case 3: $f(n) = \Omega(n^{\log_b a + \epsilon})$ and $af\left(\frac{n}{b}\right) \leq cf(n)$
for constants $\epsilon > 0$ and $c < 1$.

$$\boxed{T(n) = \Theta\left(f(n)\right)}$$

The Master Theorem

$$T(n) = \begin{cases} \Theta(1), & if\ n \leq 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise\ (a \geq 1, b > 1). \end{cases}$$

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

$$\boxed{T(n) = \Theta(n^{\log_b a})}$$

extended

Case 2: $f(n) = \Theta\left(n^{\log_b a} \lg^k n\right)$ for some constant $k \geq 0$.

$$\boxed{T(n) = \Theta\left(n^{\log_b a} \log^{k+1} n\right)}$$

Case 3: $f(n) = \Omega(n^{\log_b a + \epsilon})$ and $af\left(\frac{n}{b}\right) \leq cf(n)$

for constants $\epsilon > 0$ and $c < 1$.

$$\boxed{T(n) = \Theta\left(f(n)\right)}$$

Example Applications of Master Theorem

Example 1: $T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n)$

Master Theorem Case 1: $T(n) = \Theta\left(n^{\log_2 3}\right)$

Example 2: $T(n) = 7T\left(\frac{n}{2}\right) + \Theta\left(n^2\right)$

Master Theorem Case 1: $T(n) = \Theta\left(n^{\log_2 7}\right)$

Example 3: $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$

Master Theorem Case 2: $T(n) = \Theta(n\log n)$

Assuming that we have an infinite number of processors, and all recursive calls in example 2 above can be executed in parallel:

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

Master Theorem Case 3: $T(n) = \Theta\left(n^2\right)$

Recurrences not Solvable using the Master Theorem

Example 1: $T(n) = \sqrt{n}\, T\left(\frac{n}{2}\right) + n$

$a = \sqrt{n}$ is not a constant

Example 2: $T(n) = 2T\left(\frac{n}{\log n}\right) + n^2$

$b = \log n$ is not a constant

Example 3: $T(n) = \frac{1}{2} T\left(\frac{n}{2}\right) + n^2$

$a = \frac{1}{2}$ is not ≥ 1

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

$b = \frac{3}{4}$ is not > 1.

Recurrences not Solvable using the Master Theorem

Example 5: $T(n) = 3T\left(\frac{n}{2}\right) - n$

$f(n) = -n$ is not positive

Example 6: $T(n) = 2\,T\left(\frac{n}{2}\right) + n^2 \sin n$

violates regularity condition of case 3

Example 7: $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log n}$

$f(n) = \mathrm{O}\left(n^{\log_b a}\right)$, but $\neq \mathrm{O}\left(n^{\log_b a - \epsilon}\right)$ for any constant $\epsilon > 0$

Example 8: $T(n) = T\left(\frac{n}{2}\right) + 2T\left(\frac{n}{4}\right) + n$

a and b are not fixed

Proof of
The Master Theorem

Proof of the Master Theorem for Exact Powers of b

LEMMA 1: Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. Define $T(n)$ on exact powers of b by the recurrence

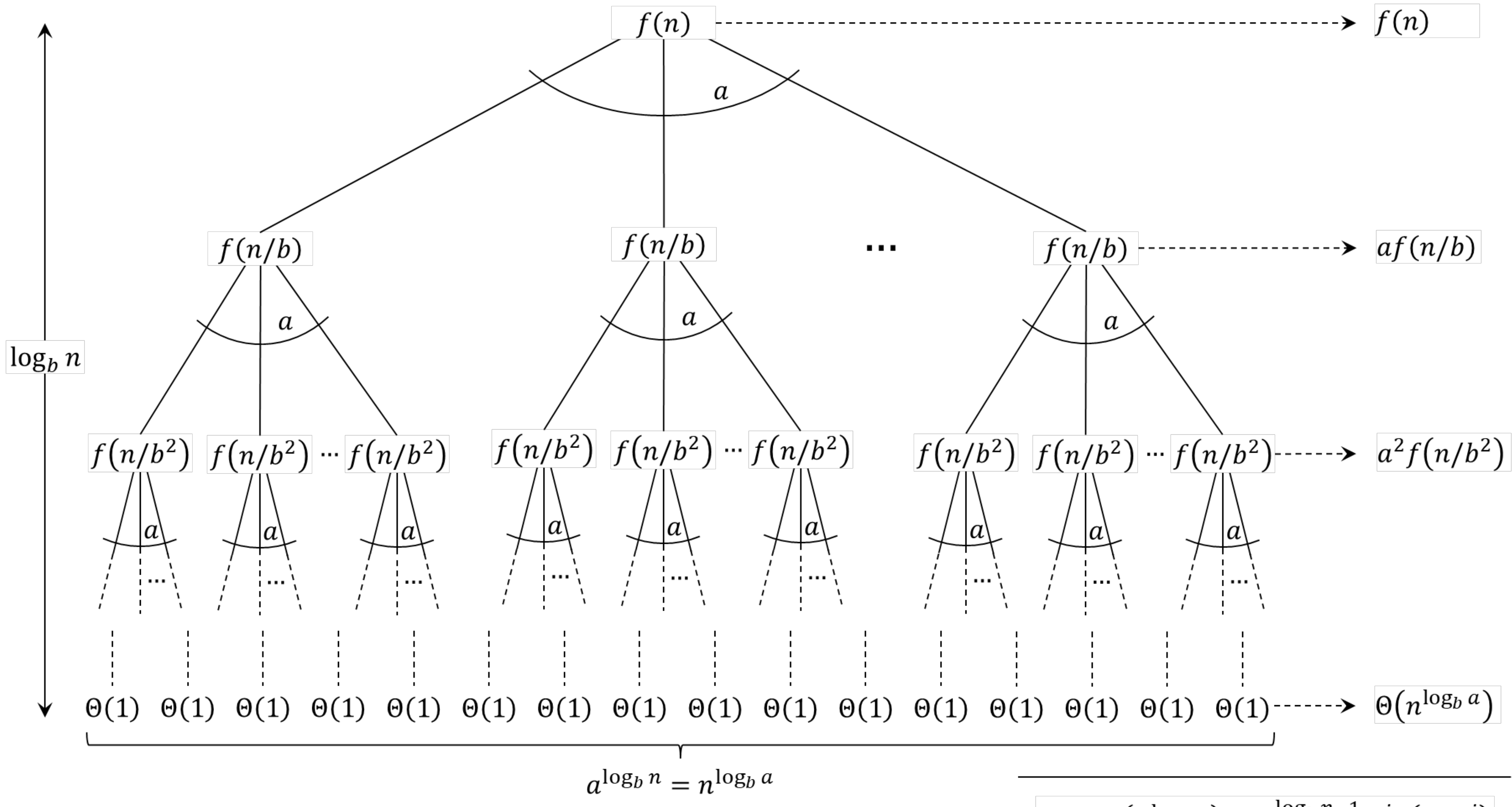
$$T(n) = \begin{cases} \Theta(1), & if\ n = 1, \\ aT\left(\frac{n}{b}\right) + f(n), & if\ n = b^i, \end{cases}$$

where i is a positive integer.

Then

$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\log_b n - 1} a^j f\left(\frac{n}{b^j}\right).$$

Proof of the Master Theorem for Exact Powers of *b*



Total: $\Theta(n^{\log_b a}) + \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j)$

Proof of the Master Theorem for Exact Powers of *b*

LEMMA 2: Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. A function $g(n)$ defined over exact powers of b by

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f\left(\frac{n}{b^j}\right)$$

has the following asymptotic bounds for exact powers of b:

1. If $f(n) = O\left(n^{\log_b a - \epsilon}\right)$ for some constant $\epsilon > 0$, then
$$g(n) = O\left(n^{\log_b a}\right).$$
2. If $f(n) = \Theta\left(n^{\log_b a}\right)$, then
$$g(n) = \Theta\left(n^{\log_b a} \log n\right).$$
3. If $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then
$$g(n) = \Theta\left(f(n)\right).$$

Proof of the Master Theorem for Exact Powers of b

PROOF OF LEMMA 2:

Case 1: We have:

$$f(n) = O\left(n^{\log_b a-\epsilon}\right) \Rightarrow f\left(n/b^j\right) = O\left(\left(n/b^j\right)^{\log_b a-\epsilon}\right).$$

Substituting: $g(n) = O\left(\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a-\epsilon}\right).$

Now, $\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a-\epsilon}$

$$\begin{aligned}
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{ab^\epsilon}{b^{\log_b a}}\right)^j \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} (b^\epsilon)^j \\
&= n^{\log_b a-\epsilon} \left(\frac{b^{\epsilon \log_b n}-1}{b^\epsilon-1}\right) \\
&= n^{\log_b a-\epsilon} \left(\frac{n^\epsilon-1}{b^\epsilon-1}\right) \\
&= n^{\log_b a-\epsilon} O(n^\epsilon) = O\left(n^{\log_b a}\right)
\end{aligned}$$

Hence, $g(n) = O\left(n^{\log_b a}\right)$.

Proof of the Master Theorem for Exact Powers of b

PROOF OF LEMMA 2:

Case 2: We have:

$$f(n) = \Theta\left(n^{\log_b a}\right) \Rightarrow f\left(n/b^j\right) = \Theta\left(\left(n/b^j\right)^{\log_b a}\right).$$

Substituting: $g(n) = \Theta\left(\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a}\right)$.

Now, $\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a}$

$$\begin{aligned} &= n^{\log_b a} \sum_{j=0}^{\log_b n-1} \left(\frac{a}{b^{\log_b a}}\right)^j \\ &= n^{\log_b a} \sum_{j=0}^{\log_b n-1} 1 \\ &= n^{\log_b a} \log_b n \end{aligned}$$

Hence, $g(n) = \Theta\left(n^{\log_b a} \log_b n\right) = \Theta\left(n^{\log_b a} \log n\right)$.

Proof of the Master Theorem for Exact Powers of b

Proof of Lemma 2:

Case 3: Since $f(n)$ appears in the definition of $g(n)$ and all terms of $g(n)$ are nonnegative, we conclude that for exact powers of b:

$$g(n) = \Omega\big(f(n)\big).$$

Given that for some constant $c < 1$ and all sufficiently large n:

$$\begin{aligned} af\left(\frac{n}{b}\right) &\le cf(n) \\ \Rightarrow f\left(\frac{n}{b}\right) &\le \left(\frac{c}{a}\right) f(n) \\ \Rightarrow f\left(\frac{n}{b^j}\right) &\le \left(\frac{c}{a}\right)^j f(n) \\ \Rightarrow a^j f\left(\frac{n}{b^j}\right) &\le c^j f(n), \end{aligned}$$

where we assume that the values we iterate on are sufficiently large. Since the last, and smallest such value is $\frac{n}{b^{j-1}}$, it is enough to assume that $\frac{n}{b^{j-1}}$ is sufficiently large.

Proof of the Master Theorem for Exact Powers of b

PROOF OF LEMMA 2:

Case 3 (continued): Substituting into the expression for $g(n)$, and adding an $O(1)$ term to capture the terms that are not covered by our assumption that n is sufficiently large, we get:

$$
\begin{aligned}
g(n) &= \sum_{j=0}^{\log_b n-1} a^j f\left(\frac{n}{b^j}\right) \\
&\le \sum_{j=0}^{\log_b n-1} c^j f(n) + O(1) \\
&\le f(n) \sum_{j=0}^{\infty} c^j + O(1) \\
&= f(n)\left(\frac{1}{1-c}\right) + O(1) \\
&= O\big(f(n)\big)
\end{aligned}
$$

Hence, for exact powers of b: $g(n) = \Theta\big(f(n)\big)$.

Proof of the Master Theorem for Exact Powers of *b*

LEMMA 3: Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. Define $T(n)$ on exact powers of b by the recurrence

$$T(n) = \begin{cases} \Theta(1), & if\ n = 1, \\ aT\left(\frac{n}{b}\right) + f(n), & if\ n = b^i, \end{cases}$$

where $i > 0$ is an integer.

Then $T(n)$ has the asymptotic bounds below for exact powers of b, and some constants $\epsilon > 0$ and $c < 0$:

1. If $f(n) = O\left(n^{\log_b a - \epsilon}\right)$, then $T(n) = \Theta\left(n^{\log_b a}\right)$.
2. If $f(n) = \Theta\left(n^{\log_b a}\right)$, then $T(n) = \Theta\left(n^{\log_b a} \log n\right)$.
3. If $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right)$, and if $af(n/b) \leq cf(n)$ for all sufficiently large n, then $T(n) = \Theta\big(f(n)\big)$.

PROOF OF LEMMA 3: Follows from Lemma 1 and Lemma 2.

Extending the Master Theorem to All Integral n

We need to extend our analysis to allow situations in which floors and ceilings appear in the Master recurrence:

$$T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + f(n) \qquad \cdots\cdots (1)$$

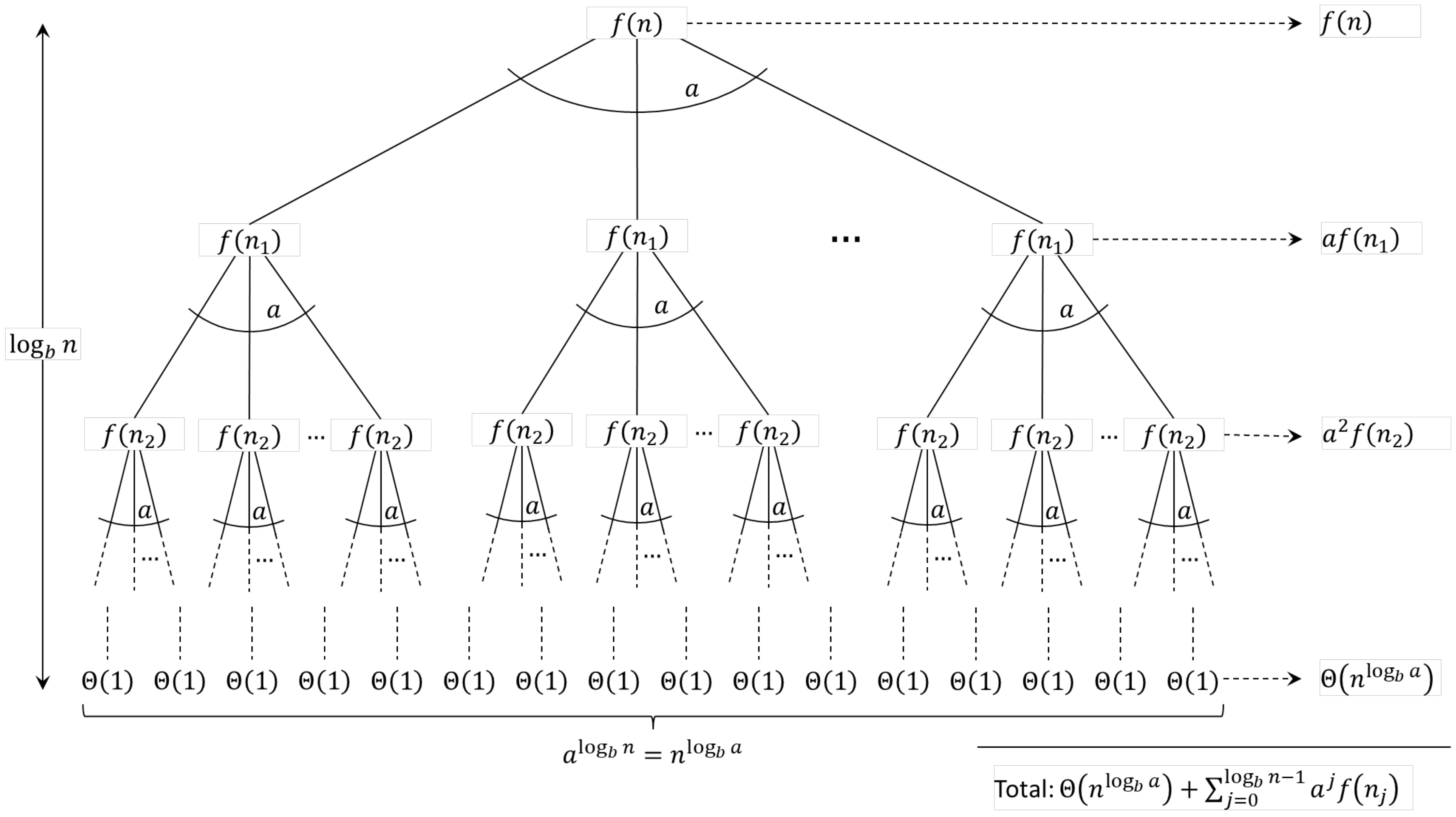
$$\text{and } T(n) = aT\left(\left\lfloor \frac{n}{b} \right\rfloor\right) + f(n) \qquad \cdots\cdots (2)$$

Obtaining a lower bound on recurrence (1) and an upper bound on recurrence (2) are not difficult because we can use $\left\lceil \frac{n}{b} \right\rceil \geq \frac{n}{b}$ in the first case and $\left\lfloor \frac{n}{b} \right\rfloor \leq \frac{n}{b}$ in the second case.

Upper bounding recurrence (1) and lower bounding recurrence (2) require more effort, but they use similar techniques.

So, we will only try to prove an upper bound on recurrence (1).

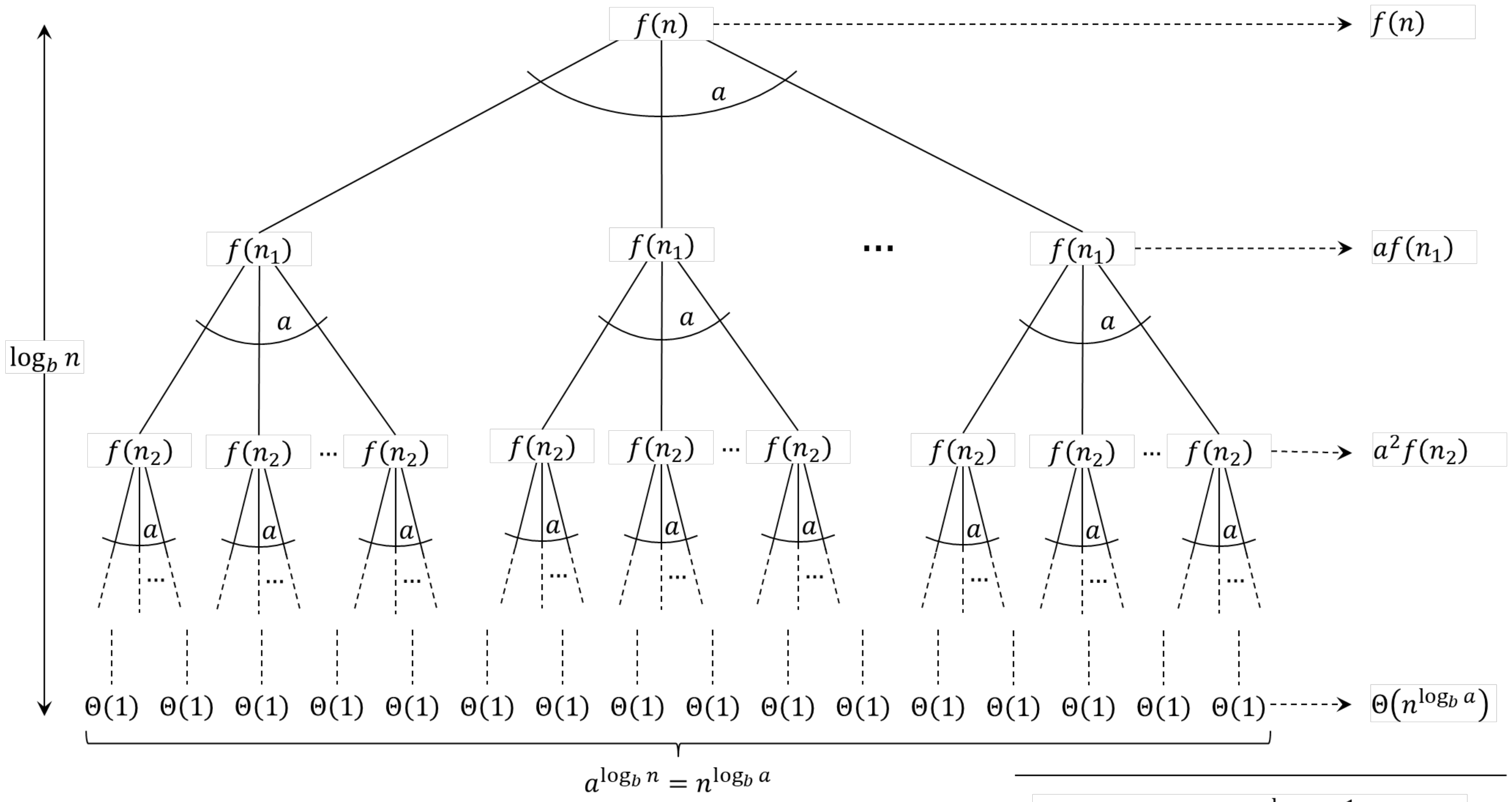
Upper Bounding $T(n) = aT\lceil n/b \rceil + f(n)$



As we go down the recursion tree we encounter a sequence of recursive invocations on the arguments:

$$n, \quad \lceil n/b \rceil, \quad \left\lceil \lceil n/b \rceil / b \right\rceil, \quad \left\lceil \left\lceil \lceil n/b \rceil / b \right\rceil / b \right\rceil, \quad \ldots\ldots\ldots$$

Upper Bounding $T(n) = aT\lceil n/b \rceil + f(n)$



Total: $\Theta(n^{\log_b a}) + \sum_{j=0}^{\log_b n - 1} a^j f(n_j)$

Let's define the j^{th} element in the sequence by n_j, where

$$n_j = \begin{cases} n, & if\ j = 0, \\ \lceil n_{j-1}/b \rceil, & if\ j > 0. \end{cases}$$

Upper Bounding $T(n) = aT\lceil n/b \rceil + f(n)$

Let's first determine a depth h such that n_h is a constant.

Using the inequality $\lceil x \rceil \leq x + 1$, we obtain:

$$n_0 \leq n,$$

$$n_1 \leq \frac{n}{b} + 1,$$

$$n_2 \leq \frac{n}{b^2} + \frac{1}{b} + 1,$$

$$n_3 \leq \frac{n}{b^3} + \frac{1}{b^2} + \frac{1}{b} + 1,$$

and so on.

In general,

$$\begin{aligned} n_j &\leq \frac{n}{b^j} + \textstyle\sum_{i=0}^{j-1} \frac{1}{b^i} \\ &< \frac{n}{b^j} + \textstyle\sum_{i=0}^{\infty} \frac{1}{b^i} \\ &= \frac{n}{b^j} + \frac{b}{b-1}. \end{aligned}$$

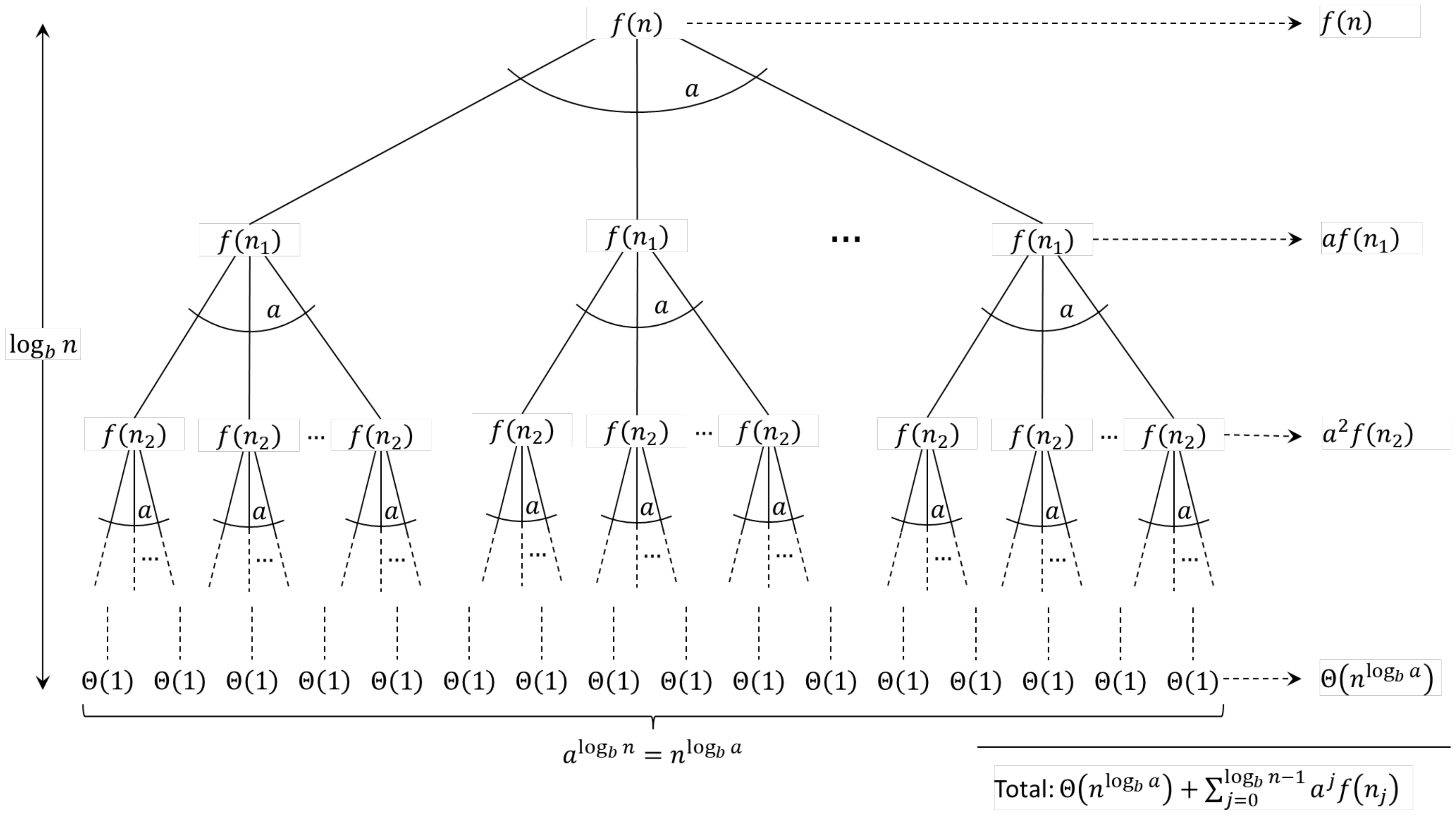
Upper Bounding $\boldsymbol{T(n) = aT\lceil n/b \rceil + f(n)}$

Letting $h = \lfloor \log_b n \rfloor$ we obtain:

$$\begin{aligned} n_{\lfloor \log_b n \rfloor} &< \frac{n}{b^{\lfloor \log_b n \rfloor}} + \frac{b}{b-1} \\ &< \frac{n}{b^{\log_b n - 1}} + \frac{b}{b-1} \\ &< \frac{n}{n/b} + \frac{b}{b-1} \\ &= b + \frac{b}{b-1} \\ &= O(1) \end{aligned}$$

Hence, at depth $h = \lfloor \log_b n \rfloor$ the problem size is at most a constant.

Upper Bounding $T(n) = aT\lceil n/b \rceil + f(n)$



From the figure above we get:

$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\log_b n - 1} a^j f\left(n_j\right)$$

Upper Bounding $T(n) = aT[n/b] + f(n)$

We have:

$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\log_b n - 1} a^j f\left(n_j\right)$$

We will have to evaluate the following sum:

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f\left(n_j\right)$$

Upper Bounding $T(n) = aT\lceil n/b \rceil + f(n)$

We will evaluate the following sum:

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n_j)$$

Case 2: We have: $f(n) = \Theta\left(n^{\log_b a}\right)$.

If we can show that $f(n_j) = O\left(\left(\frac{n}{b^j}\right)^{\log_b a}\right)$, then case 2 of

Lemma 2 will go through.

Observe that $j \leq \lfloor \log_b n \rfloor \Rightarrow \frac{b^j}{n} \leq 1$.

Also, $f(n) = O\left(n^{\log_b a}\right)$ implies that there exists a constant $c' > 0$ such that for all sufficiently large n_j the following holds:

$$f(n_j) \leq c' \left(\frac{n}{b^j} + \frac{b}{b-1}\right)^{\log_b a}.$$

Upper Bounding $T(n) = aT[n/b] + f(n)$

Case 2 (continued): We have:

$$
\begin{aligned}
f(n_j) &\le c'\left(\frac{n}{b^j} + \frac{b}{b-1}\right)^{\log_b a} \\
&= c'\left(\frac{n}{b^j}\left(1 + \frac{b^j}{n}\frac{b}{b-1}\right)\right)^{\log_b a} . \\
&= c'\left(\frac{n^{\log_b a}}{a^j}\right)\left(1 + \frac{b^j}{n}\frac{b}{b-1}\right)^{\log_b a} . \\
&\le c'\left(\frac{n^{\log_b a}}{a^j}\right)\left(1 + \frac{b}{b-1}\right)^{\log_b a} . \\
&= O\left(\frac{n^{\log_b a}}{a^j}\right) \\
&= O\left(\left(\frac{n}{b^j}\right)^{\log_b a}\right)
\end{aligned}
$$

Upper Bounding $T(n) = aT[n/b] + f(n)$

Case 1: The proof is similar to that of case 2. The key is to prove the bound $f(n_j) = O\left(\left(\frac{n}{b^j}\right)^{\log_b a - \epsilon}\right)$ which is similar to what we did in case 2 though the algebra is more intricate.

Upper Bounding $\boldsymbol{T(n) = aT\lceil n/b \rceil + f(n)}$

Case 3: If $af\left(\left\lceil \frac{n}{b} \right\rceil\right) \leq cf(n)$ for $n > b + \frac{b}{b-1}$, where $c < 1$ is a constant then it follows that $a^j f(n_j) \leq c^j f(n)$.

Therefore, we can evaluate $g(n)$ as in the proof of Lemma 2.