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 2017

A Useful Recurrence

Consider the following 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 from previous lectures.

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

Strassen's 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

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

How the Recurrence Unfolds

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

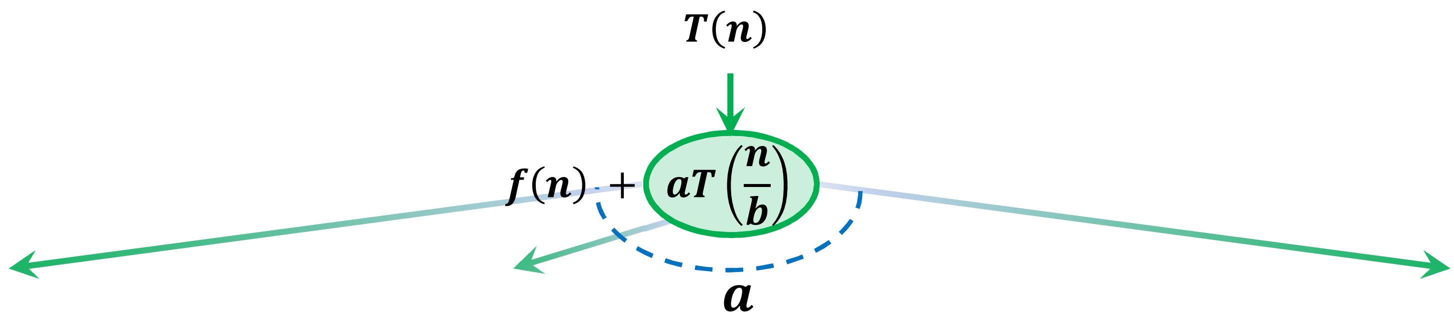
$$\boldsymbol{T(n)}$$

$$\downarrow$$

$$\boldsymbol{f(n)} \ + \ \boldsymbol{aT\left(\frac{n}{b}\right)}$$

How the Recurrence Unfolds

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



How the Recurrence Unfolds

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

$$\boldsymbol{T(n)}$$

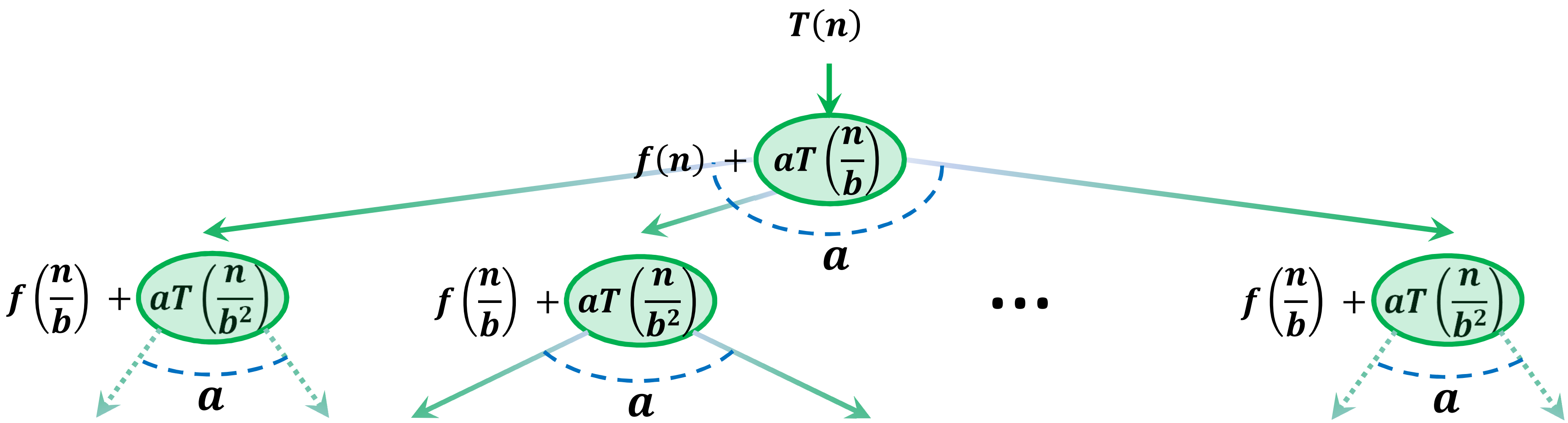
$$\boldsymbol{f(n) + aT\left(\frac{n}{b}\right)}$$

$$\boldsymbol{a}$$

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

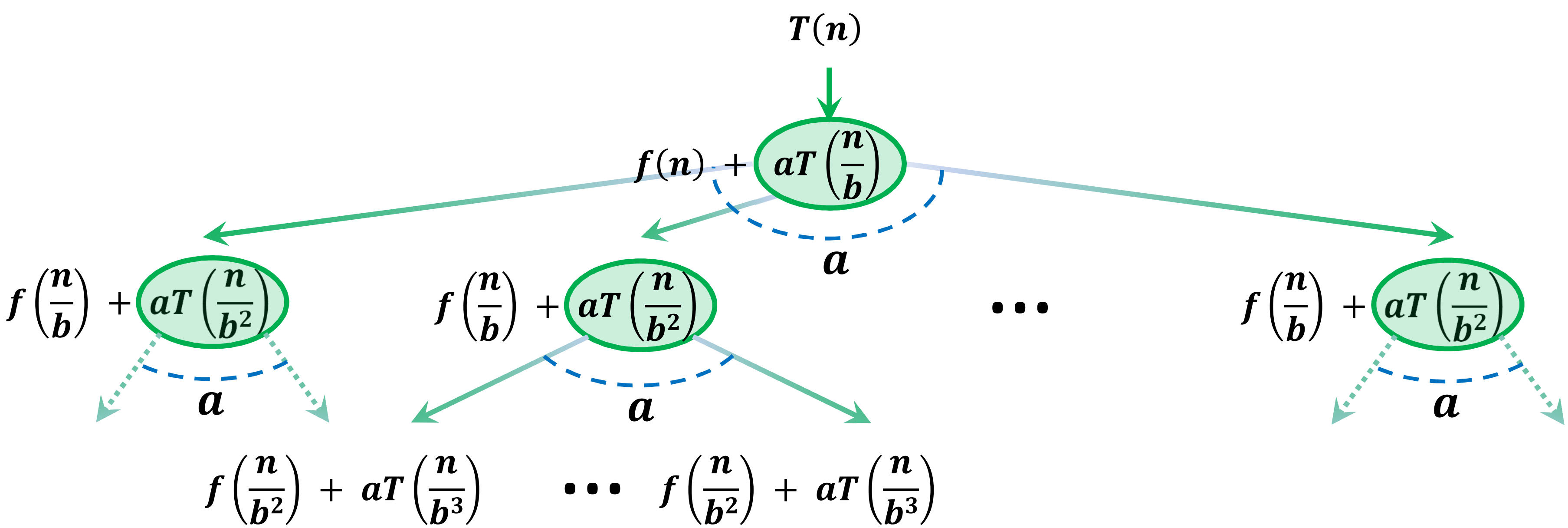
How the Recurrence Unfolds

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



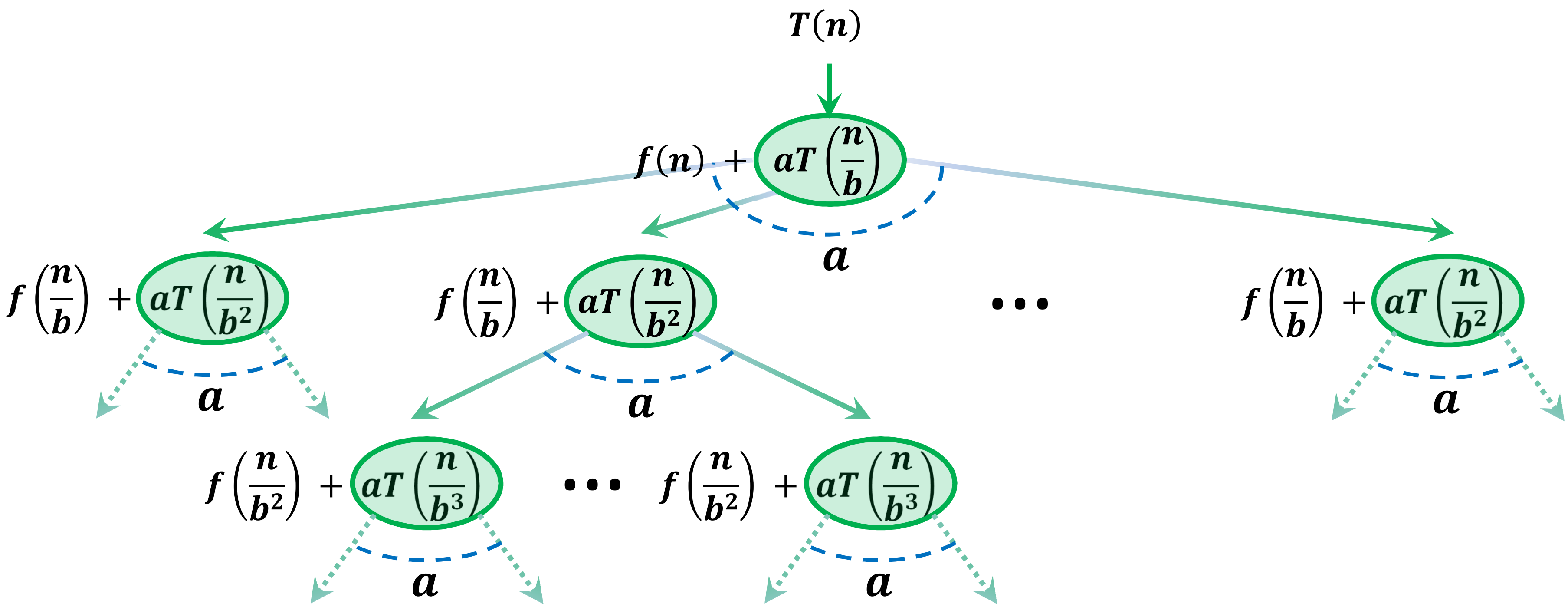
How the Recurrence Unfolds

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



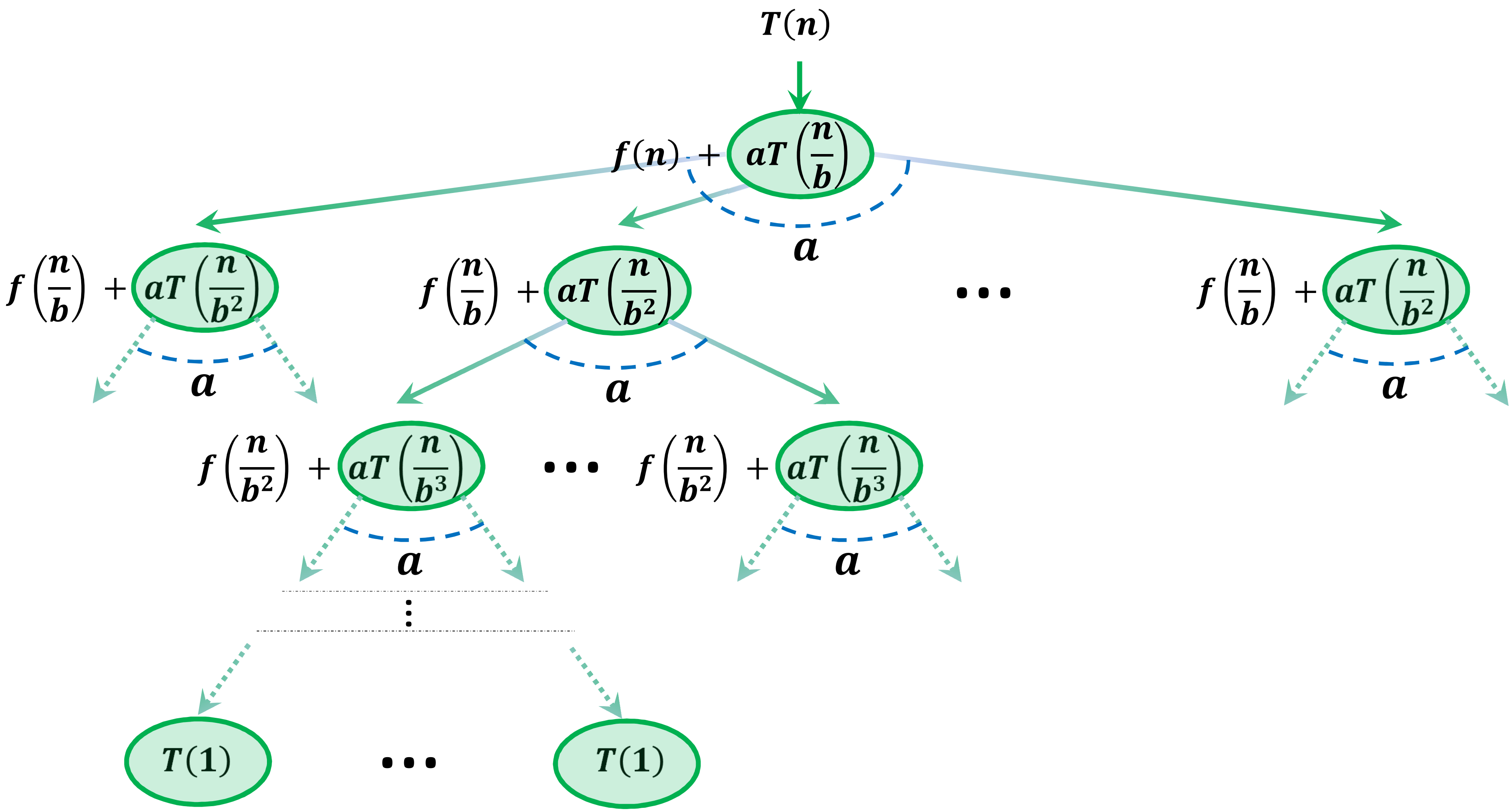
How the Recurrence Unfolds

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



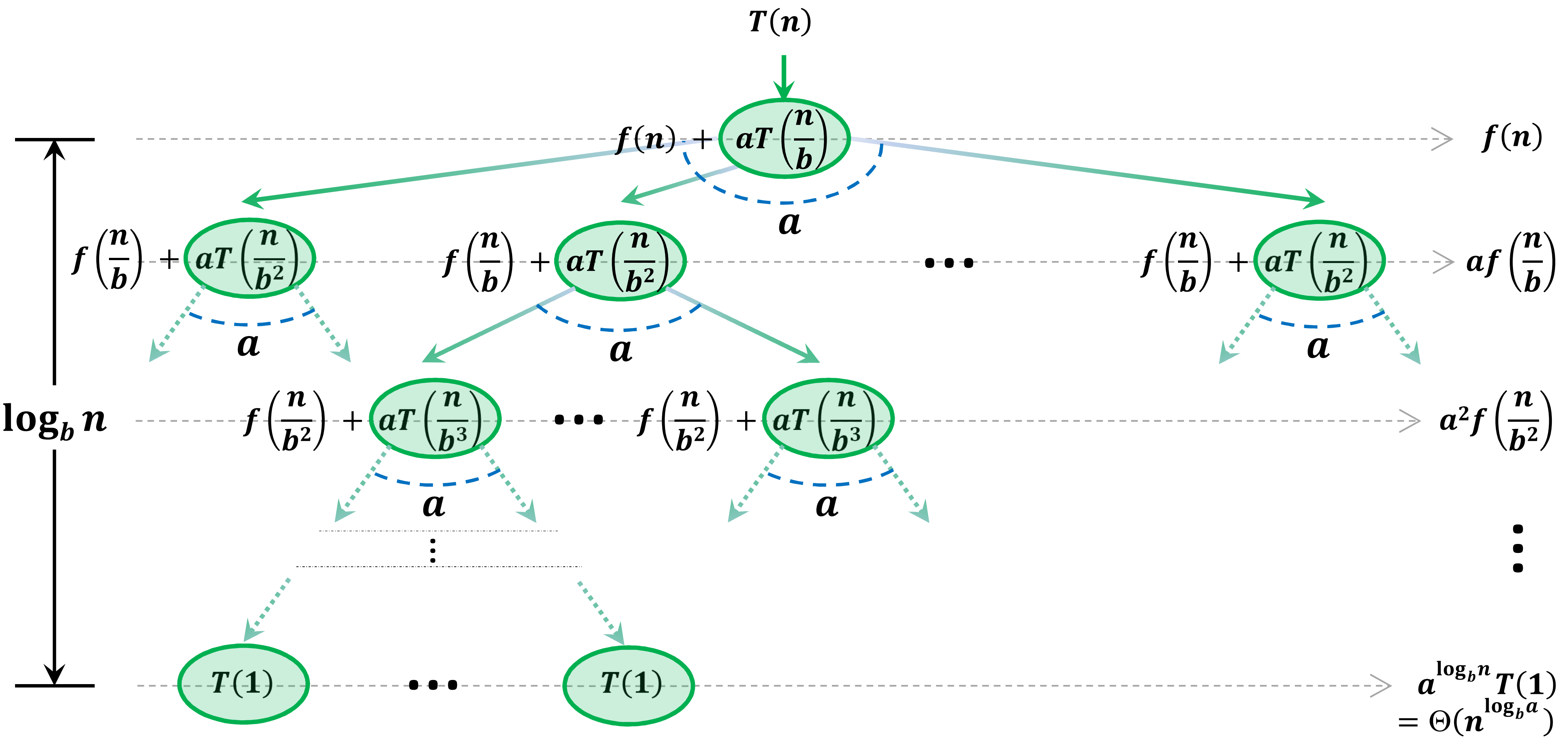
How the Recurrence Unfolds

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



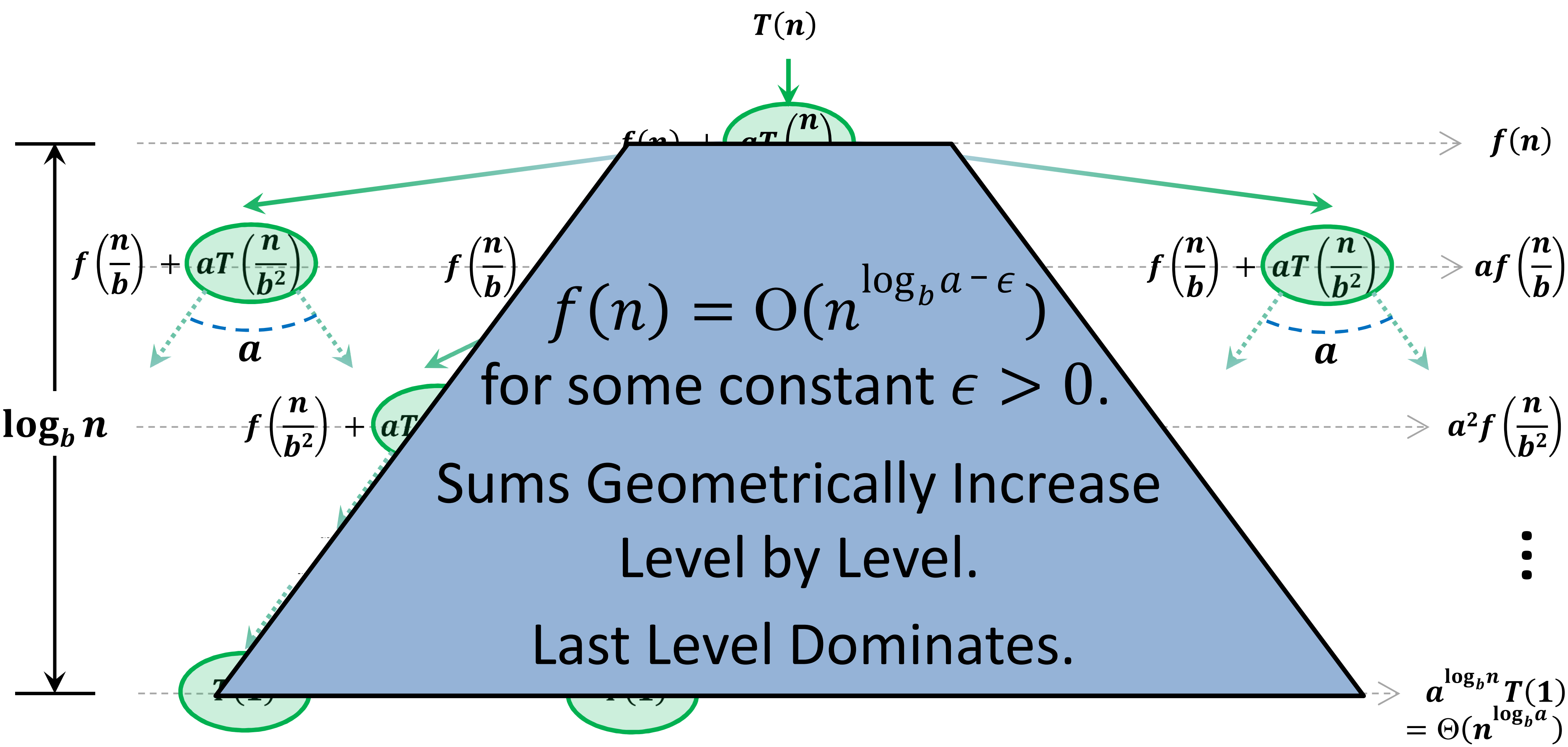
How the Recurrence Unfolds

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



How the Recurrence Unfolds: Case 1

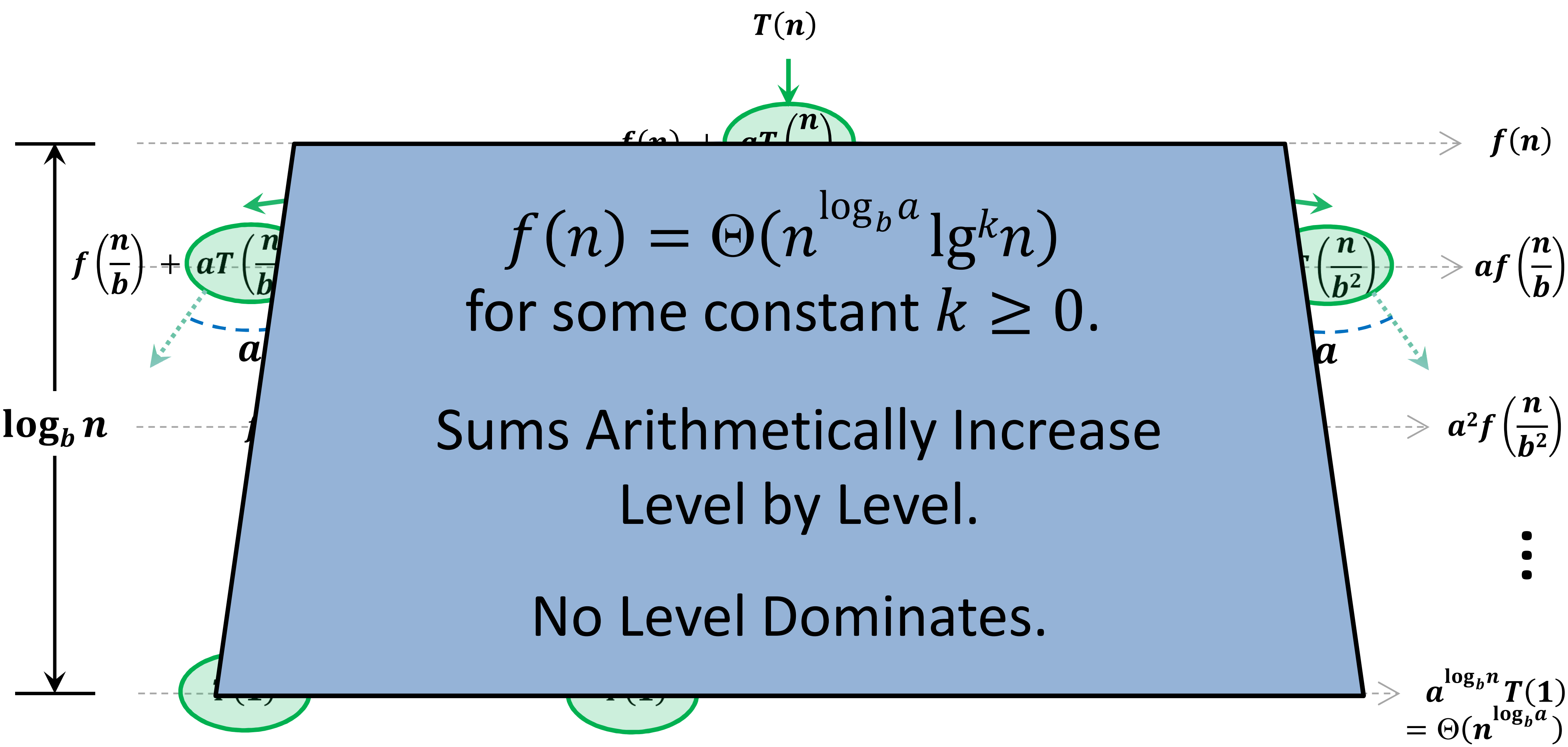
$$T(n) = \begin{cases} \Theta(1), & if\ n \le 1, \\ aT\left(\frac{n}{b}\right) + f(n), & otherwise. \end{cases}$$



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

How the Recurrence Unfolds: Case 2

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



$f(n) = \Theta(n^{\log_b a} \lg^k n)$
for some constant $k \ge 0$.

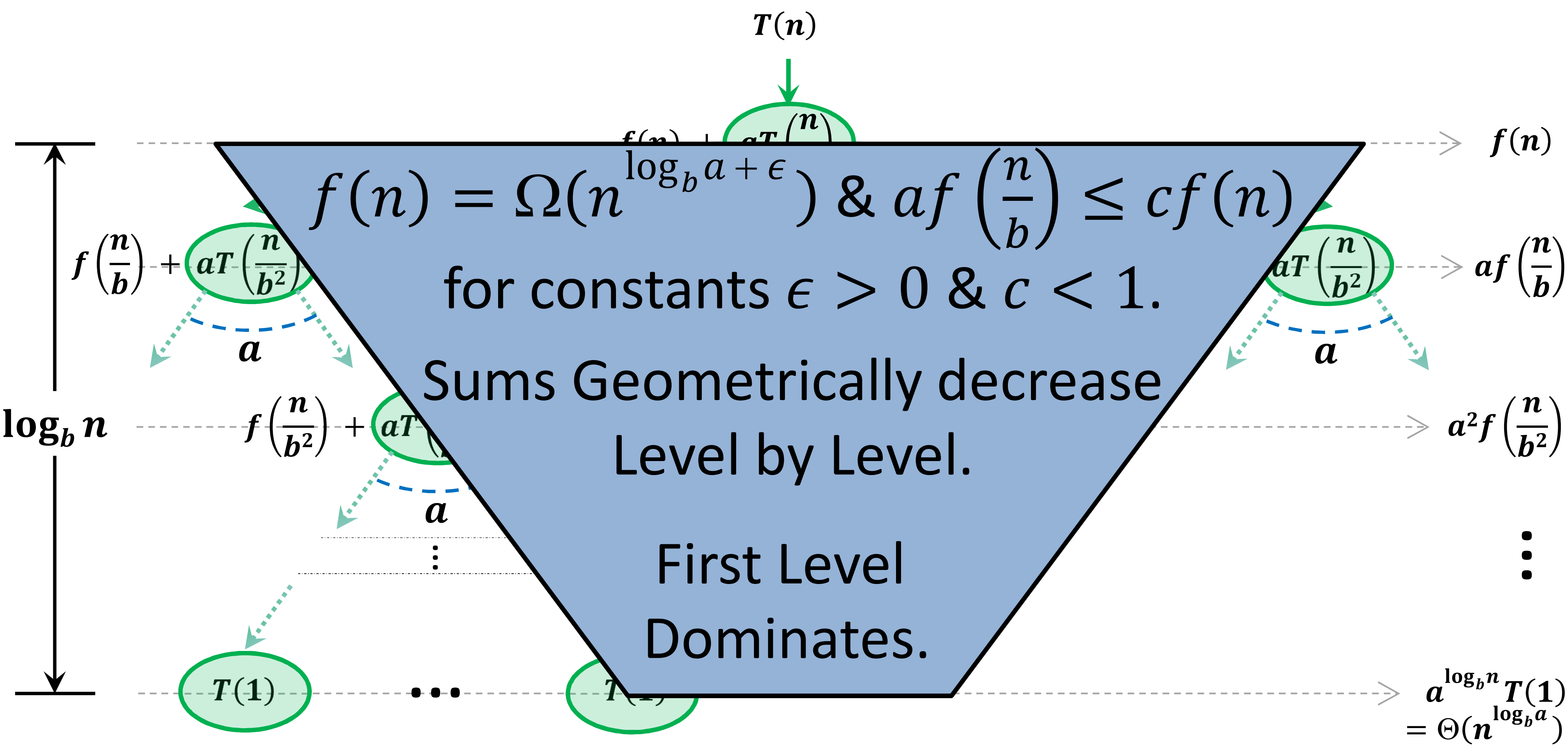
Sums Arithmetically Increase Level by Level.

No Level Dominates.

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

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

Sums Geometrically decrease Level by Level.

First Level Dominates.

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

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 $\epsilon > 0$

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

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

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

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 each recursive call 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: $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\log n}$

$f(n) = O\left(n^{\log_b a}\right)$, but $\neq O\left(n^{\log_b a-\epsilon}\right)$ 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