CP312
Algorithm Design and Analysis I

LECTURE 5: RECURRENCES

Recurrences

- A **recurrence** is an equation or inequality that describes a function in terms of its value on **smaller inputs**.

- Recurrences give us a natural way to analyze the running times of **divide-and-conquer** algorithms.

Examples of Recurrences

- $T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T(n/2) + \Theta(n) & \text{if } n > 1 \end{cases}$

- $T(n) = T(2n/3) + T(n/3) + \Theta(n)$

- $T(n) \leq 4T(n/4) + \Theta(n^2)$

Unless otherwise stated you can always assume that the base case $T(1) = \Theta(1)$

Solving Recurrences

- We will study three Methods:

1. Substitution Method
2. Recursion Tree Method
3. The Master Method

The Substitution Method

1. **Guess** the form of the solution

2. **Verify** by induction

3. **Solve** for constants (n_0, c)

The Substitution Method

- Example: $T(n) = 4T(n/2) + n$
- We are given that $T(1) = \Theta(1) \leq d$ for some constant d

1. Guess $T(n) = O(n^3)$
2. Assume that $T(k) \leq ck^3$ for $k < n$

   Prove $T(n) \leq cn^3$ by induction

The Substitution Method

Inductive Hypothesis:
$T(k) \leq ck^3$ for $k < n$

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\leq 4c(n/2)^3 + n \\
&= (c/2)n^3 + n \\
&= (c/2)n^3 + n + (c/2)n^3 - (c/2)n^3 \\
&= cn^3 - \left((c/2)n^3 - n\right) \\
&\leq cn^3 \quad \text{whenever } \left((c/2)n^3 - n\right) \geq 0 \text{ which is when } c \geq 2
\end{aligned}
$$

The Substitution Method

Inductive Hypothesis:
$T(k) \leq ck^3$ for $k < n$

Base Case $(k = n_0)$: Let $n_0 = 1$

Recall we are given that:
$T(1) = \Theta(1) \leq d$

$T(n_0) = T(1) \leq d$

$\leq c(1)^3$ (by inductive hypothesis)

3. So the constants are $n_0 = 1$ and $c \geq d$

Thus, $T(n) = O(n^3)$

But is this upper bound tight?

The Substitution Method

- Example: $T(n) = 4T(n/2) + n$
- Assume that $T(1) = \Theta(1) \leq d$ for some constant d

1. Guess $T(n) = O\left(n^2\right)$
2. Assume that $T(k) \leq ck^2$ for $k < n$

   Prove $T(n) \leq cn^2$ by induction

The Substitution Method

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

$\leq cn^2$ for what value of c does this inequality hold?

Inductive Hypothesis:
$T(k) \leq ck^2$ for $k < n$

For **no** value of $c > 0$

The Substitution Method

- Example: $T(n) = 4T(n/2) + n$
- Assume that $T(1) = \Theta(1) \leq d$ for some constant d

1. Guess $T(n) = O(n^2)$
2. Assume that $T(k) \leq ck^2$ for $k < n$

   Prove $T(n) \leq cn^2$ by induction

Idea: strengthen the inductive hypothesis

The Substitution Method

- Example: $T(n) = 4T(n/2) + n$
- Assume that $T(1) = \Theta(1) \leq d$ for some constant d

1. Guess $T(n) = O\left(n^2\right)$
2. Assume that $T(k) \leq c_1 k^2 - c_2 k$ for $k < n$

   Prove $T(n) \leq c_1 n^2 - c_2 n$ by induction

Idea: strengthen the inductive hypothesis

The Substitution Method

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

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c_1(n/2)^2 - 4c_2(n/2) + n \\
&= c_1 n^2 - 2c_2 n + n \\
&= c_1 n^2 - c_2 n - (c_2 n - n) \\
&\le c_1 n^2 - c_2 n \quad \text{whenever } (c_2 n - n) \ge 0 \Rightarrow c_2 \ge 1
\end{aligned}
$$

3. Pick c_1 large enough to cover the base case

Thus, $T(n) = O\left(n^2\right)$

The Substitution Method

- Example of incorrect use:

Suppose we want to (incorrectly) prove that the recurrence

$T(n) = 2T(n/2) + n$ can be solved to be $T(n) = O(n)$

1. Guess $T(n) \leq cn$
2. So $T(n) \leq 2c(n/2) + n$

$$\leq cn + n = O(n)$$

Recursion Tree Method

- Needed summation rules:
  - $$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$$
  - $$\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1} = \frac{1-x^{n+1}}{1-x} \qquad \text{where } x \neq 1$$
  - $$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x} \qquad \text{for } |x| < 1$$

Example of Recursion Tree

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

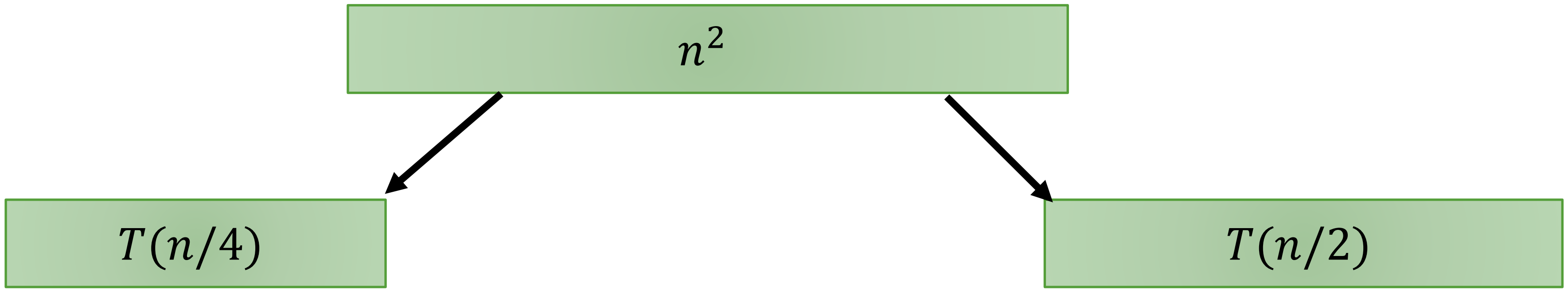
$$T(n) = T(n/4) + T(n/2) + n^2$$

Example of Recursion Tree

$T(n)$

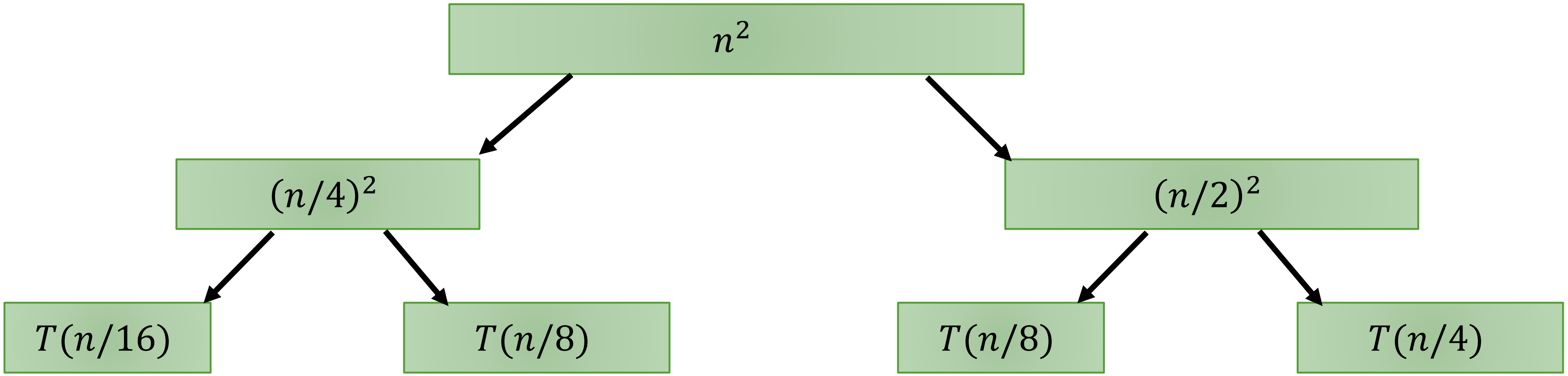
$$T(n) = T(n/4) + T(n/2) + n^2$$

Example of Recursion Tree



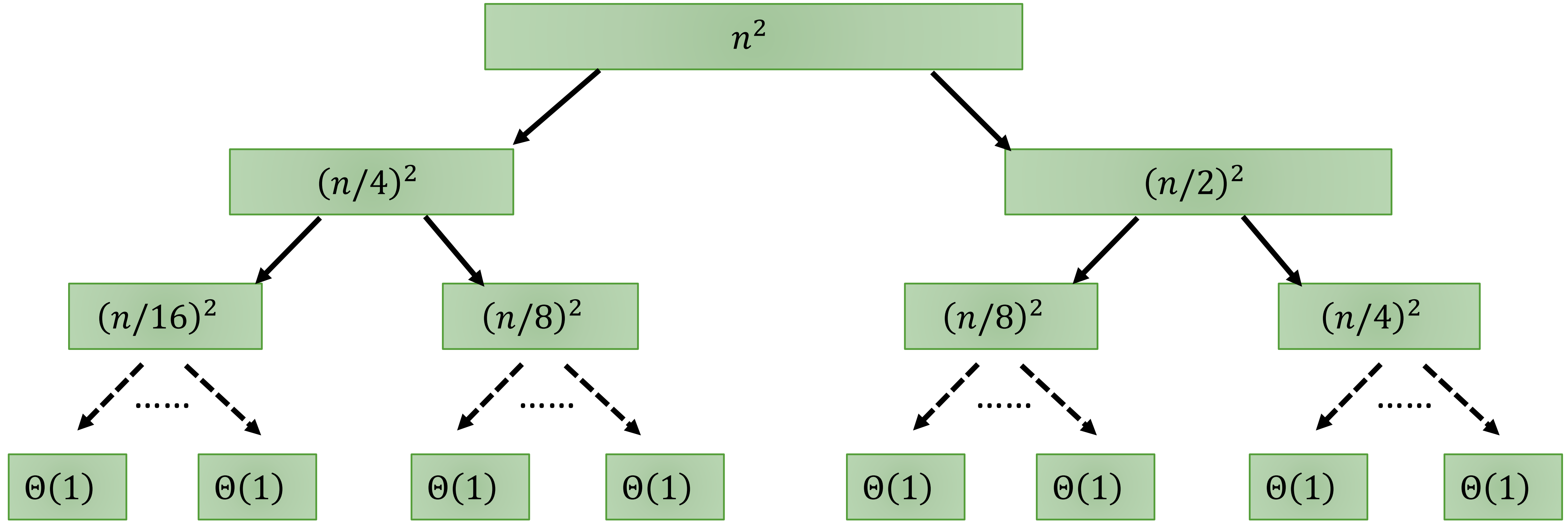
$$T(n) = T(n/4) + T(n/2) + n^2$$

Example of Recursion Tree



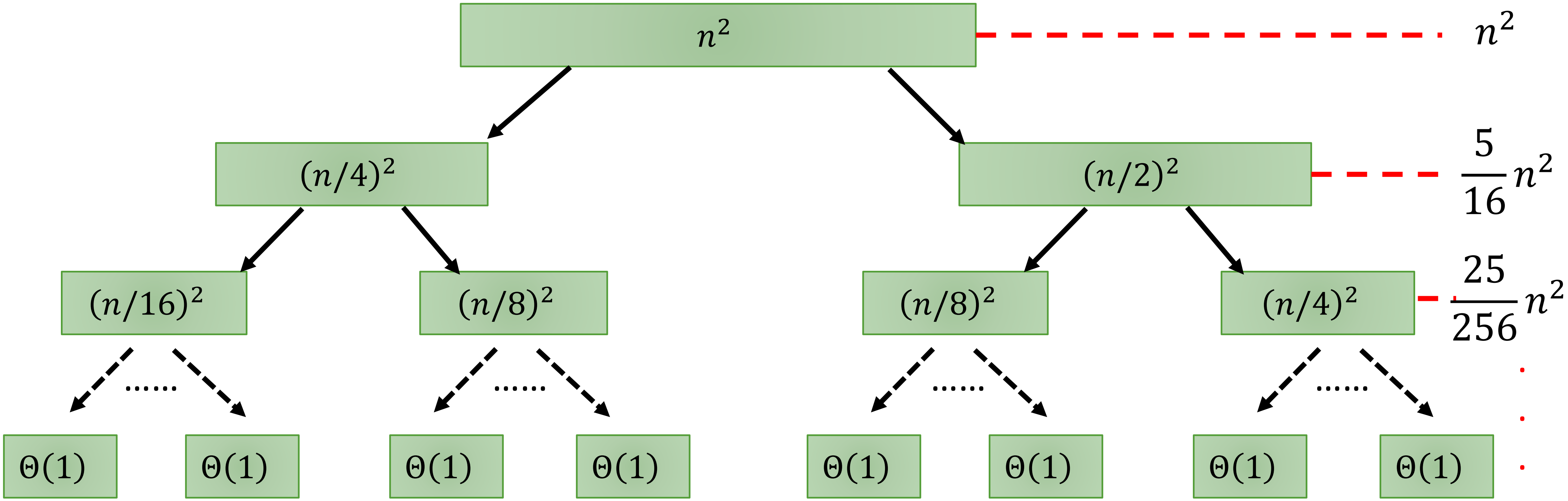
$$T(n) = T(n/4) + T(n/2) + n^2$$

Example of Recursion Tree



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

Example of Recursion Tree



$\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$ for $|x| < 1$

Example of Recursion Tree

- Solve $T(n) = T(n/4) + T(n/2) + n^2$
- Summing up the cost (time) in each level:
- $T(n) = n^2 + \frac{5}{16}n^2 + \left(\frac{5}{16}\right)^2 n^2 + \left(\frac{5}{16}\right)^3 n^2 + \cdots + \left(\frac{5}{16}\right)^h n^2$

$$\leq n^2 + \frac{5}{16}n^2 + \left(\frac{5}{16}\right)^2 n^2 + \left(\frac{5}{16}\right)^3 n^2 + \cdots$$

$$= n^2 \sum_{i=0}^{\infty} \left(\frac{5}{16}\right)^i$$

$$= \frac{16}{11} n^2 = O(n^2)$$

Example of Recursion Tree

- Solve $T(n) = T(n/4) + T(n/2) + n^2$
- Is it also $\Omega\left(n^2\right)$?
  - Yes!

- So it is $\Theta\left(n^2\right)$ and you can verify it using the substitution method!

The Master Method

- This method applies to recurrences of the form:

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

Where $a \geq 1, b > 1$ are constants and $f(n)$ is asymptotically positive

The Master Theorem

Given $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ where $a \geq 1, b > 1$ are constants

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

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

The Master Theorem

Given $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ where $a \geq 1, b > 1$ are constants

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

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

The Master Theorem

Given $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ where $a \geq 1, b > 1$ are constants

- **Case 3**: $f(n) = \Omega\left(n^{\log_b a+\epsilon}\right)$ for some constant $\epsilon > 0$
  - $f(n)$ grows polynomially faster than $n^{\log_b a}$ (by an n^ϵ factor)
  - $f(n)$ must satisfy the **regularity condition** that $af(n/b) \leq cf(n)$ for some constant $c < 1$

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

Examples: The Master Theorem

- Ex1: $T(n) = 4T(n/2) + n$ $a = 4, b = 2$

  $n^{\log_b a} = n^2$ and $f(n) = n$

  Case 1: $f(n) = O\left(n^{2-\epsilon}\right)$ for $\epsilon = 1$

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

Examples: The Master Theorem

- Ex2: $T(n) = 4T(n/2) + n^2$ $a = 4, b = 2$

  $n^{\log_b a} = n^2$ and $f(n) = n^2$

  Case 2: $f(n) = \Theta\left(n^2 \lg^0 n\right)$, that is $k = 0$

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

Examples: The Master Theorem

- Ex3: $T(n) = 4T(n/2) + n^3$ $a = 4, b = 2$

$n^{\log_b a} = n^2$ and $f(n) = n^3$

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

and $4(n/2)^3 \leq cn^3$ for $c = 1/2$ (regularity condition)

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

Examples: The Master Theorem

- Ex4: $T(n) = 4T(n/2) + \frac{n^2}{\lg n}$ $\qquad a = 4, b = 2$

  $n^{\log_b a} = n^2$ and $f(n) = n^2 / \lg n$

- None of the cases' conditions are fulfilled => Master Theorem cannot be applied here.