# CS 3100
# Data Structures and Algorithms 2
## Lecture 9: D&C, Master Theorem

**Co-instructors: Robbie Hott and Tom Horton**

**Fall 2023**

Readings in CLRS 4th edition:

- Section 4.5

# Announcements

- Upcoming dates
  - PS2 due September 29 (Friday) at 11:59pm
  - PA2 due October 8 (Sunday) at 11:59pm
  - Quiz 1 and 2, October 5 (in-class)
- Course email (comes to both professors and head TAs):

**cs3100@cshelpdesk.atlassian.net**

# Divide and Conquer

[CLRS Chapter 4]

**Divide**:

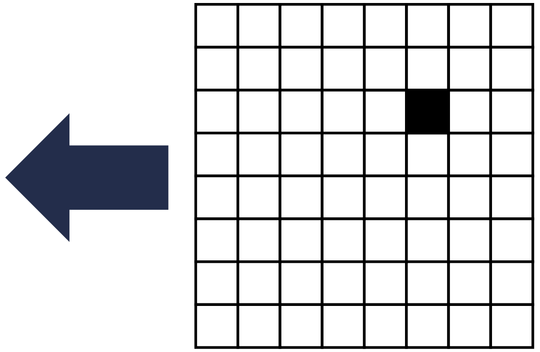
- Break the problem into multiple subproblems, each smaller instances of the original

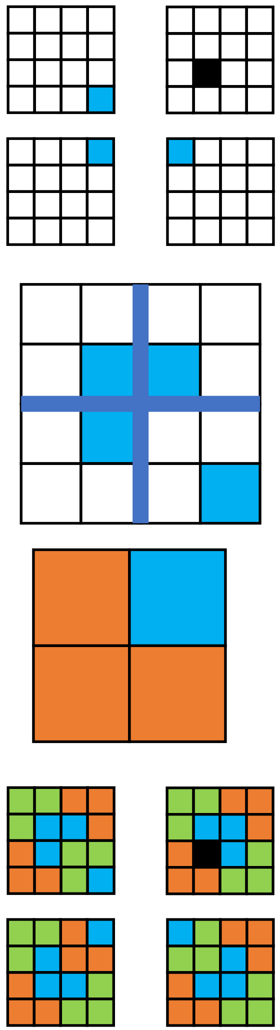
**Conquer**:

- If the suproblems are “large”:
  - Solve each subproblem recursively
- If the subproblems are “small”:
  - Solve them directly (base case)

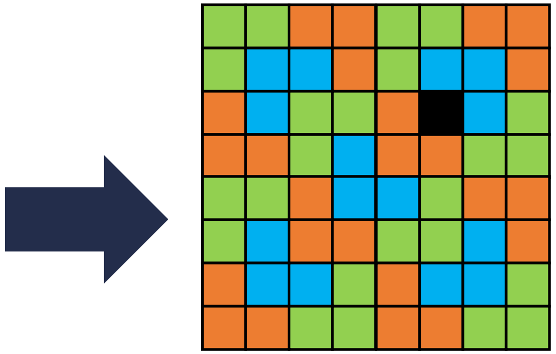
**Combine**:

- Merge solutions to subproblems to obtain solution for original problem





When is this an effective strategy?



# Analyzing Divide and Conquer

1. Break into smaller subproblems
2. Use recurrence relation to express recursive running time
3. Use asymptotic notation to simplify

**Divide**: $D(n)$ time

**Conquer**: Recurse on smaller problems of size $s_1, \dots, s_k$

**Combine**: $C(n)$ time

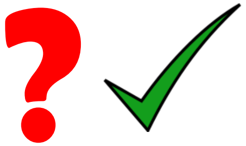
**Recurrence:**

- $T(n) = D(n) + \sum_{i \in [k]} T(s_i) + C(n)$

# Recurrence Solving Techniques

Tree

get a picture of recursion



Guess/Check

guess and use induction to prove


"Cookbook"

MAGIC!


Substitution

substitute in to simplify

# Observation

**Divide**: $D(n)$ time

**Conquer**: Recurse on smaller problems of size $s_1, \ldots, s_k$

**Combine**: $C(n)$ time

**Recurrence:**

- $T(n) = D(n) + \sum_{i \in [k]} T(s_i) + C(n)$

Many divide and conquer algorithms have recurrences are of form:

- $T(n) = a \cdot T(n/b) + f(n)$

$a$ and $b$ are constants

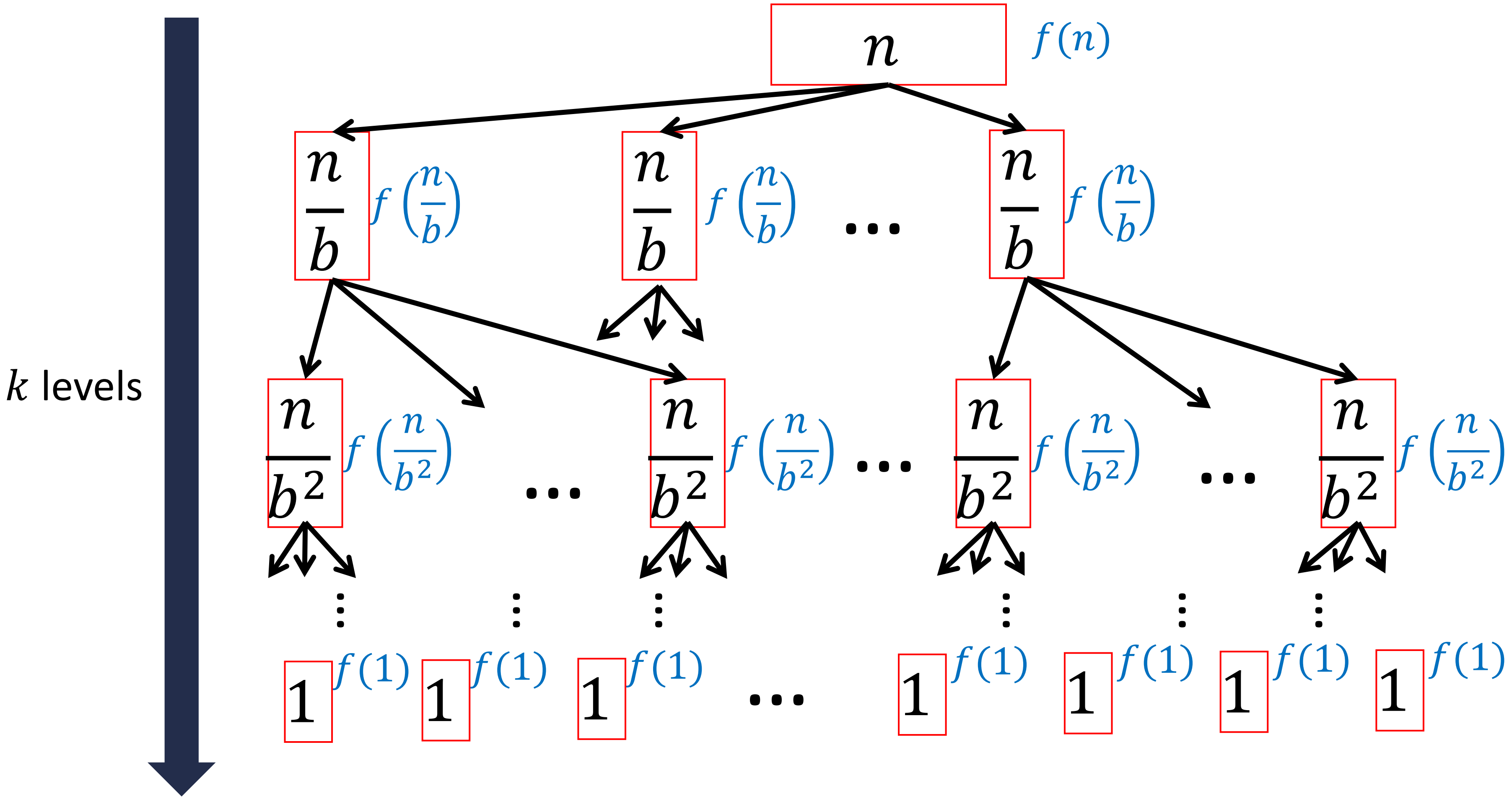
Mergesort: $T(n) = 2T(n/2) + n$

Divide and Conquer Multiplication: $T(n) = 4T(n/2) + 5n$

Karatsuba Multiplication: $T(n) = 3T(n/2) + 8n$

# General Recurrence

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




| Number of subproblems | Cost of subproblem |
|---|---|
| $1$ | $f(n)$ |
| $a$ | $f(n/b)$ |
| $a^2$ | $f(n/b^2)$ |
| $a^k$ | $f(n/b^k)$ |

# General Recurrence

3. Use asymptotic notation to simplify

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

| Number of subproblems | Cost of subproblem |
|---|---|
| $1$ | $f(n)$ |
| $a$ | $f(n/b)$ |
| $a^2$ | $f(n/b^2)$ |
| $a^k$ | $f(n/b^k)$ |

How many levels?

Problem size at $k^{\text{th}}$ level: $\frac{n}{b^k}$

Base case: $n = 1$

At level $k$, it should be the case that $\frac{n}{b^k} = 1$

$$n = b^k \Rightarrow k = \log_b n$$

# General Recurrence

3. Use asymptotic notation to simplify

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

$$k = \log_b n$$

What is the cost?

Cost at level $i$: $a^i \cdot f\left(\frac{n}{b^i}\right)$

Total cost: $T(n) = \sum_{i=0}^{\log_b n} a^i \cdot f\left(\frac{n}{b^i}\right)$

| Number of subproblems | Cost of subproblem |
|---|---|
| $1$ | $f(n)$ |
| $a$ | $f(n/b)$ |
| $a^2$ | $f(n/b^2)$ |
| $a^k$ | $f(n/b^k)$ |

# Three Cases

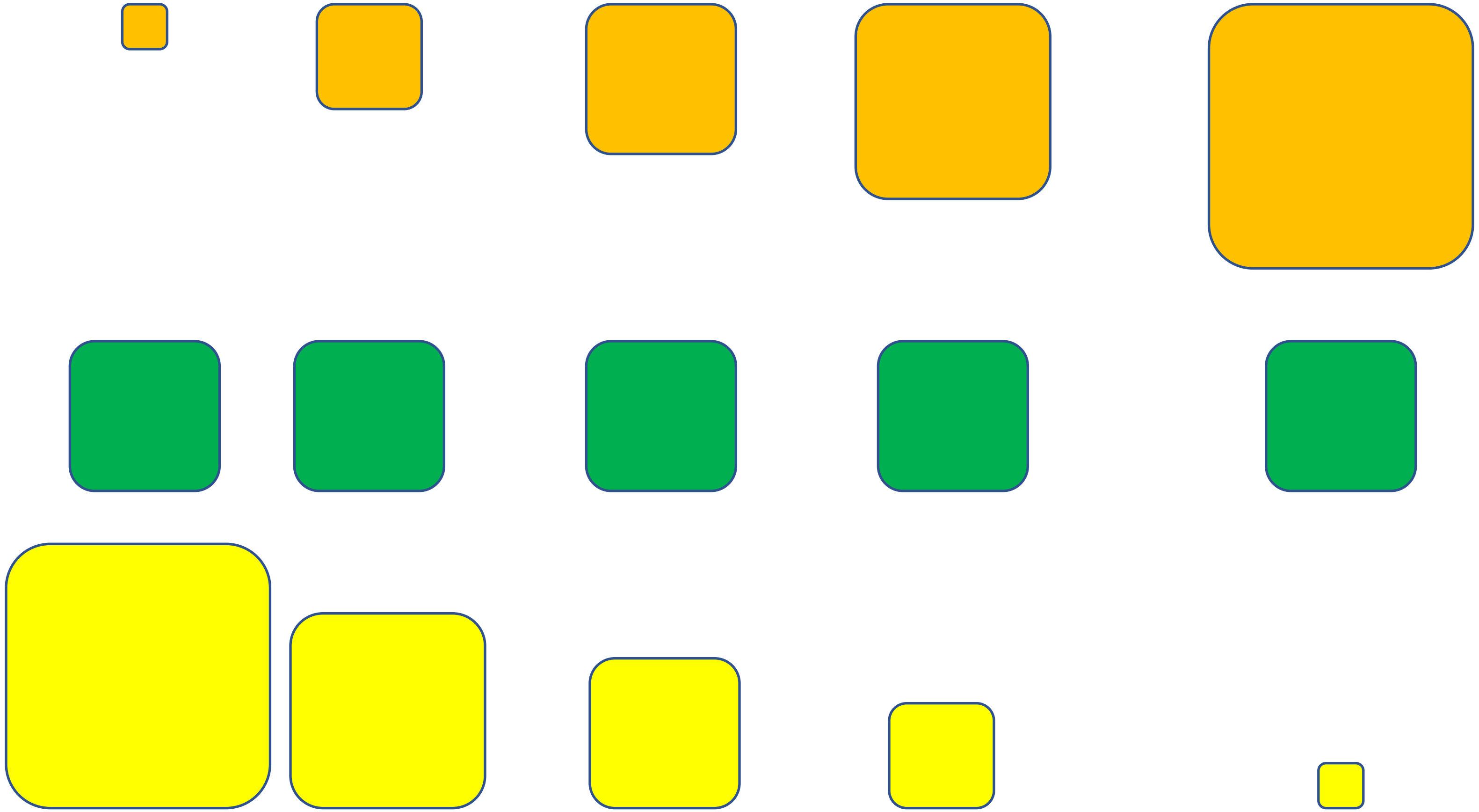
$$T(n) = f(n) + af\left(\frac{n}{b}\right) + a^2 f\left(\frac{n}{b^2}\right) + a^3 f\left(\frac{n}{b^3}\right) + \cdots + a^k f\left(\frac{n}{b^k}\right)$$

$$k = \log_b n$$

**Case 1:**
Most work happens
at the leaves

**Case 2:**
Work happens
consistently throughout

**Case 3:**
Most work happens
at top of tree



# Master Theorem

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

$$\delta = \log_b a$$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |

# Master Theorem

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

$$\delta = \log_b a$$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |

# Master Theorem

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

$$\delta = \log_b a$$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O(n^{\delta-\varepsilon})$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta(n^{\delta})$ |
| **Case 2** | $f(n) \in \Theta(n^{\delta})$ | $T(n) \in \Theta(n^{\delta} \log n)$ |
| **Case 3** | $f(n) \in \Omega(n^{\delta+\varepsilon})$ for some constant $\varepsilon > 0$ **AND** $af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta(f(n))$ |

# Master Theorem Example 1

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

[Merge Sort]

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

$$\delta = \log_b a$$

# Master Theorem Example 1

$$T(n) = 2T(n/2) + n$$ [Merge Sort]

**Step 1:** Compute $\delta = \log_b a = \log_2 2 = 1$

**Step 2:** Compare $n^\delta$ and $f(n)$

$$f(n) = n \in \Theta\left(n^\delta\right)$$

**Step 3:** Check table

$T(n) = aT(n/b) + f(n)$ $\delta = \log_b a$

# Master Theorem Example 1

$$T(n) = 2T(n/2) + n$$ [Merge Sort]

$\delta = 1$

$f(n) = n \in \Theta(n^{\delta})$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O(n^{\delta-\varepsilon})$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta(n^{\delta})$ |
| **Case 2** | $f(n) \in \Theta(n^{\delta})$ | $T(n) \in \Theta(n^{\delta} \log n)$ |
| **Case 3** | $f(n) \in \Omega(n^{\delta+\varepsilon})$ for some constant $\varepsilon > 0$<br>**AND**<br>$af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta(f(n))$ |

# Master Theorem Example 1

$T(n) = 2T(n/2) + n$ [Merge Sort]

$\delta = 1$

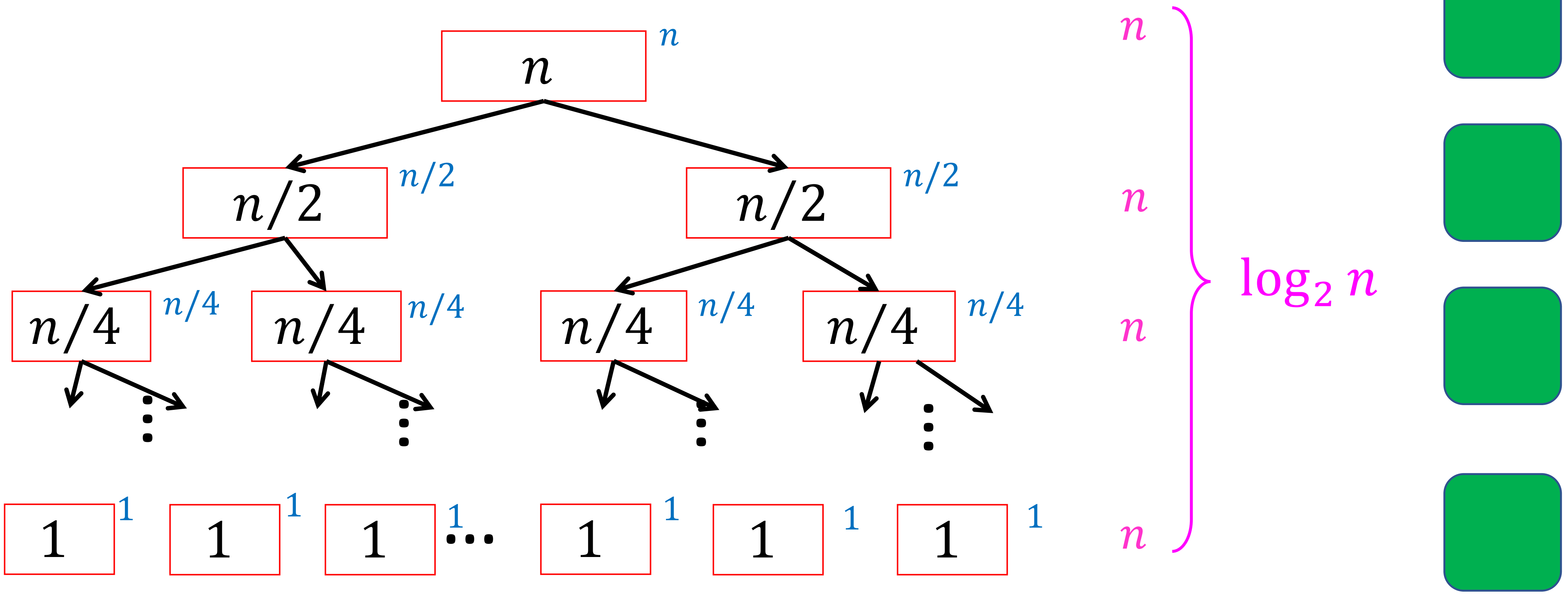
$f(n) = n \in \Theta(n^\delta)$

$T(n) = \Theta(n \log n)$

| | Requirement on $f$ | Implication |
|---|---|---|
| Case 1 | $f(n) \in O(n^{\delta-\varepsilon})$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta(n^\delta)$ |
| **Case 2** | $f(n) \in \Theta(n^\delta)$ | $T(n) \in \Theta(n^\delta \log n)$ |
| Case 3 | $f(n) \in \Omega(n^{\delta+\varepsilon})$ for some constant $\varepsilon > 0$<br>**AND**<br>$af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta(f(n))$ |

# Master Theorem Example 1 (Visually)

$$T(n) = 2T(n/2\,) + n$$




Cost is <u>consistent</u> across levels ⇒

Cost increases by log factor (≈ number of levels)

# Master Theorem Example 2

$$T(n) = 4T(n/2) + 5n$$

$T(n) = aT(n/b) + f(n)$ $\qquad$ $\delta = \log_b a$

# Master Theorem Example 2

$$T(n) = 4T(n/2) + 5n$$

**Step 1:** Compute $\delta = \log_b a = \log_2 4 = 2$

**Step 2:** Compare $n^\delta$ and $f(n)$

$$f(n) = 5n \in O(n^{2-1}) = O\left(n^{\delta-1}\right)$$

**Step 3:** Check table

$T(n) = aT(n/b) + f(n)$ $\qquad$ $\delta = \log_b a$

# Master Theorem Example 2

$$T(n) = 4T(n/2) + 5n$$

$\delta = 2$

$f(n) = 5n \in O\left(n^{\delta - 1}\right)$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$ **AND** $af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta\left(f(n)\right)$ |

# Master Theorem Example 2

$$T(n) = 4T(n/2) + 5n$$

$\delta = 2$

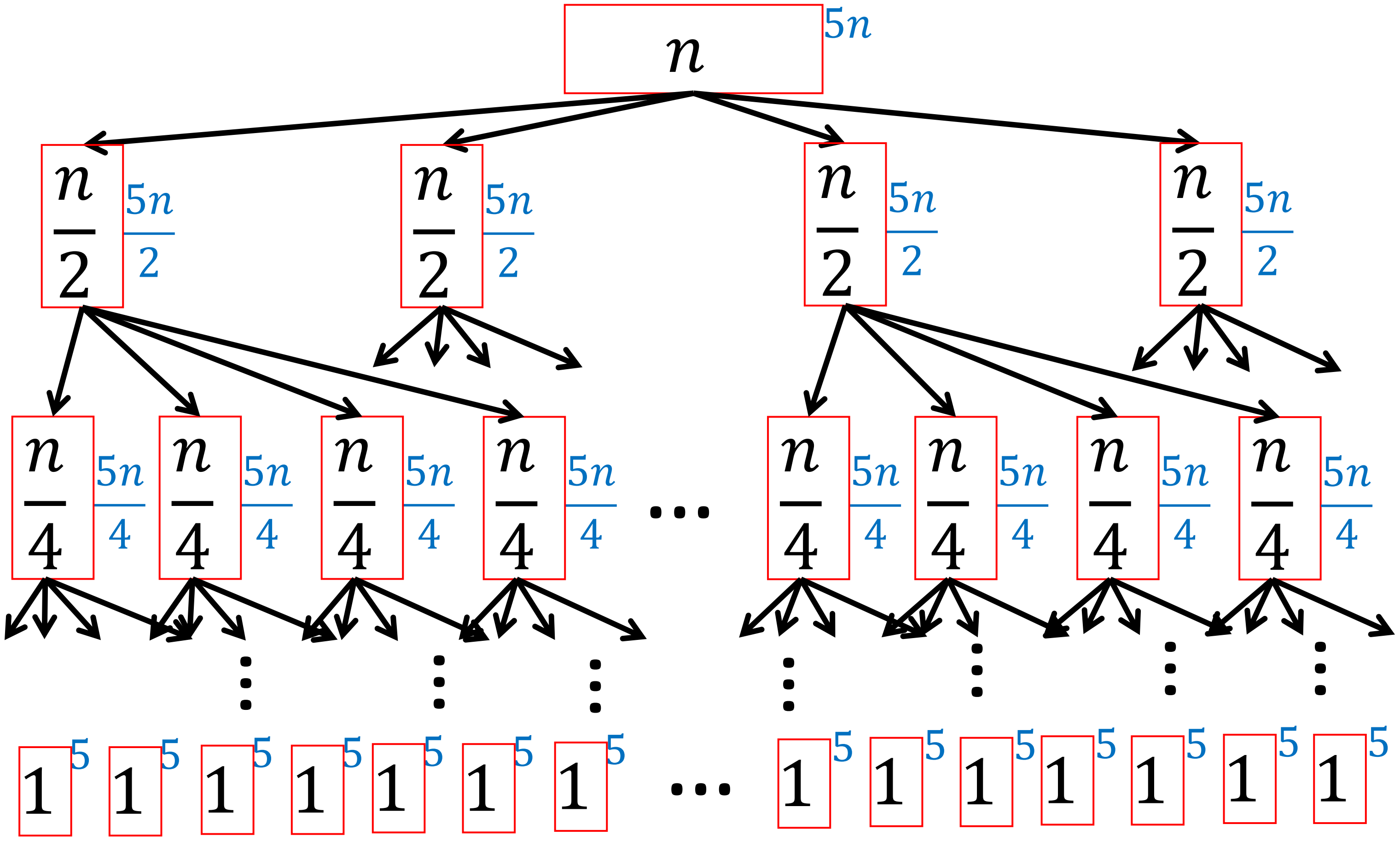
$f(n) = 5n \in O\left(n^{\delta-1}\right)$

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

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$<br>**AND**<br>$af\left(\frac{n}{b}\right) \le cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta(f(n))$ |

# Master Theorem Example 2 (Visually)

$$T(n) = 4T(n/2) + 5n$$



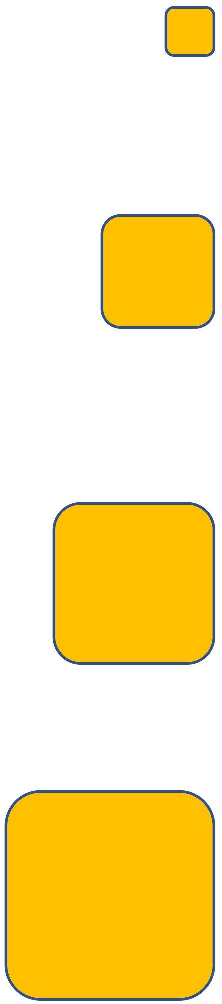

$$5n$$

$$\frac{4}{2} \cdot 5n$$

$$\frac{16}{4} \cdot 5n$$

$$\vdots$$

$$2^{\log_2 n} \cdot 5n$$



# Master Theorem Example 2 (Visually)

$$T(n) = 4T(n/2) + 5n$$

Cost is <u>increasing</u> with the recursion depth (due to large number of subproblems)

Most of the work happening in the leaves

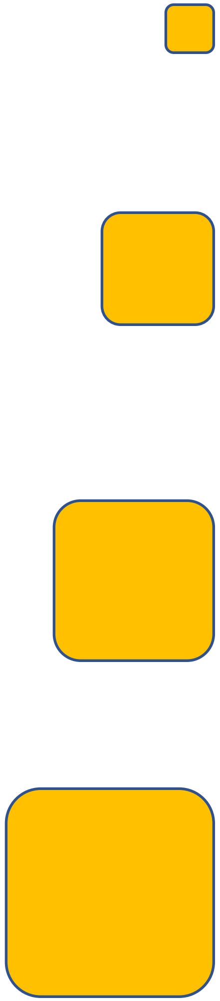
$$5n$$

$$\frac{4}{2} \cdot 5n$$

$$\frac{16}{4} \cdot 5n$$

$$\vdots$$

$$2^{\log_2 n} \cdot 5n$$



# Master Theorem Example 3

$$T(n) = 3T(n/2) + 8n$$

[Karatsuba]

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

$\delta = \log_b a$

# Master Theorem Example 3

$$T(n) = 3T(n/2) + 8n$$ [Karatsuba]

**Step 1:** Compute $\delta = \log_b a = \log_2 3$

**Step 2:** Compare $n^\delta$ and $f(n)$

$f(n) = 8n \in O\left(n^{\log_2 3 - \varepsilon}\right)$ for constant $\varepsilon > \log_2 3 - 1 > 0$

**Step 3:** Check table

$T(n) = aT(n/b) + f(n)$ $\delta = \log_b a$

# Master Theorem Example 3

$$T(n) = 3T(n/2) + 8n$$

[Karatsuba]

$$\delta = \log_2 3$$

$$f(n) = 5n \in O\left(n^{\delta-\varepsilon}\right)$$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$ **AND** $af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta\left(f(n)\right)$ |

# Master Theorem Example 3

$$T(n) = 3T(n/2) + 8n$$

[Karatsuba]

$$\delta = \log_2 3$$

$$f(n) = 5n \in O\left(n^{\delta-\varepsilon}\right)$$

$$T(n) = \Theta(n^{\log_2 3})$$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$ **AND** $af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta(f(n))$ |

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

$$T(n) = aT(n/b) + f(n) \qquad \delta = \log_b a$$

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

**Step 1:** Compute $\delta = \log_b a = \log_2 2 = 1$

**Step 2:** Compare $n^\delta$ and $f(n)$

$$f(n) = 15n^3 \in \Omega(n^{1+2}) = \Omega\left(n^{\delta+2}\right)$$

**Step 3:** Check table

$T(n) = aT(n/b) + f(n)$ $\quad\quad$ $\delta = \log_b a$

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

$\delta = 1$

$f(n) = 15n^3 \in \Omega\left(n^{\delta+2}\right)$

| | Requirement on $f$ | Implication |
|---|---|---|
| **Case 1** | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| **Case 2** | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$<br>**AND**<br>$af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta\left(f(n)\right)$ |

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

$\delta = 1$

$f(n) = 15n^3 \in \Omega\left(n^{\delta+2}\right)$

| | Requirement on $f$ | Implication |
|---|---|---|
| Case 1 | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| Case 2 | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta}\log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$<br>**AND**<br>$af\left(\frac{n}{b}\right) \le cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta\left(f(n)\right)$ |

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

$\delta = 1$

$f(n) = 15n^3 \in \Omega\left(n^{\delta+2}\right)$

**Important:** For Case 3, need to additionally check that $2f(n/2) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$

$$2f(n/2) = 30(n/2)^3 = \frac{30}{8}n^3 \leq \frac{1}{4}(15n^3)$$

# Master Theorem Example 4

$$T(n) = 2T(n/2) + 15n^3$$

$\delta = 1$

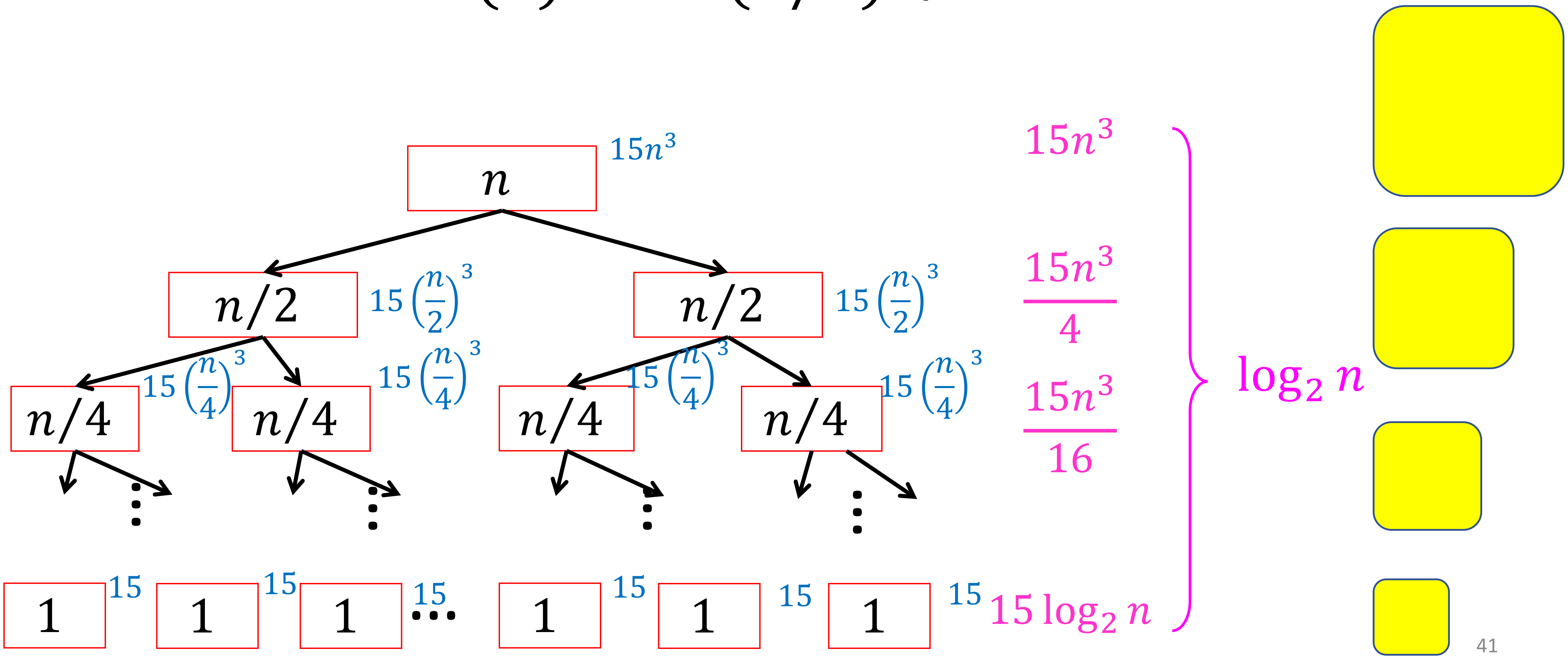
$f(n) = 15n^3 \in \Omega\left(n^{\delta+2}\right)$

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

| | Requirement on $f$ | Implication |
|---|---|---|
| Case 1 | $f(n) \in O\left(n^{\delta-\varepsilon}\right)$ for some constant $\varepsilon > 0$ | $T(n) \in \Theta\left(n^{\delta}\right)$ |
| Case 2 | $f(n) \in \Theta\left(n^{\delta}\right)$ | $T(n) \in \Theta\left(n^{\delta} \log n\right)$ |
| **Case 3** | $f(n) \in \Omega\left(n^{\delta+\varepsilon}\right)$ for some constant $\varepsilon > 0$ **AND** $af\left(\frac{n}{b}\right) \leq cf(n)$ for constant $c < 1$ and sufficiently large $n$ | $T(n) \in \Theta\left(f(n)\right)$ |

# Master Theorem Example 3 (Visually)

$$T(n) = 2T(n/2) + 15n^3$$



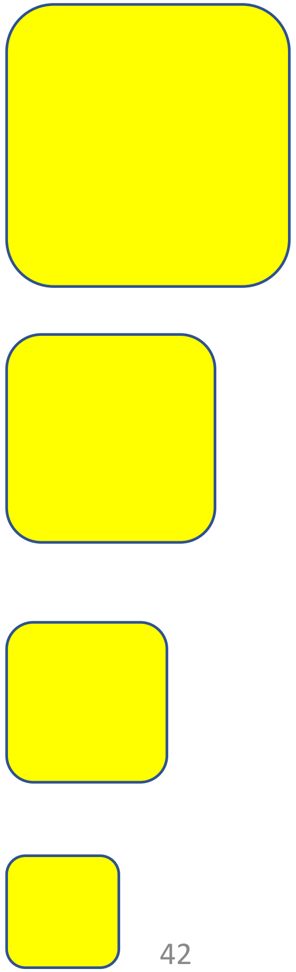
# Master Theorem Example 3 (Visually)

$$T(n) = 2T(n/2) + 15n^3$$

Cost is <u>decreasing</u> with the recursion depth (due to high *non-recursive* cost)

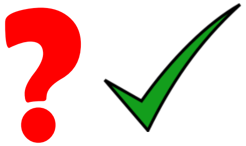
Most of the work happening at the top

$$\left.\begin{array}{c} 15n^3 \\ \\ \dfrac{15n^3}{4} \\ \\ \dfrac{15n^3}{16} \\ \\ 15\log_2 n \end{array}\right\} \log_2 n$$



# Recurrence Solving Techniques

Tree



Guess/Check


"Cookbook"


Substitution

# Substitution Method

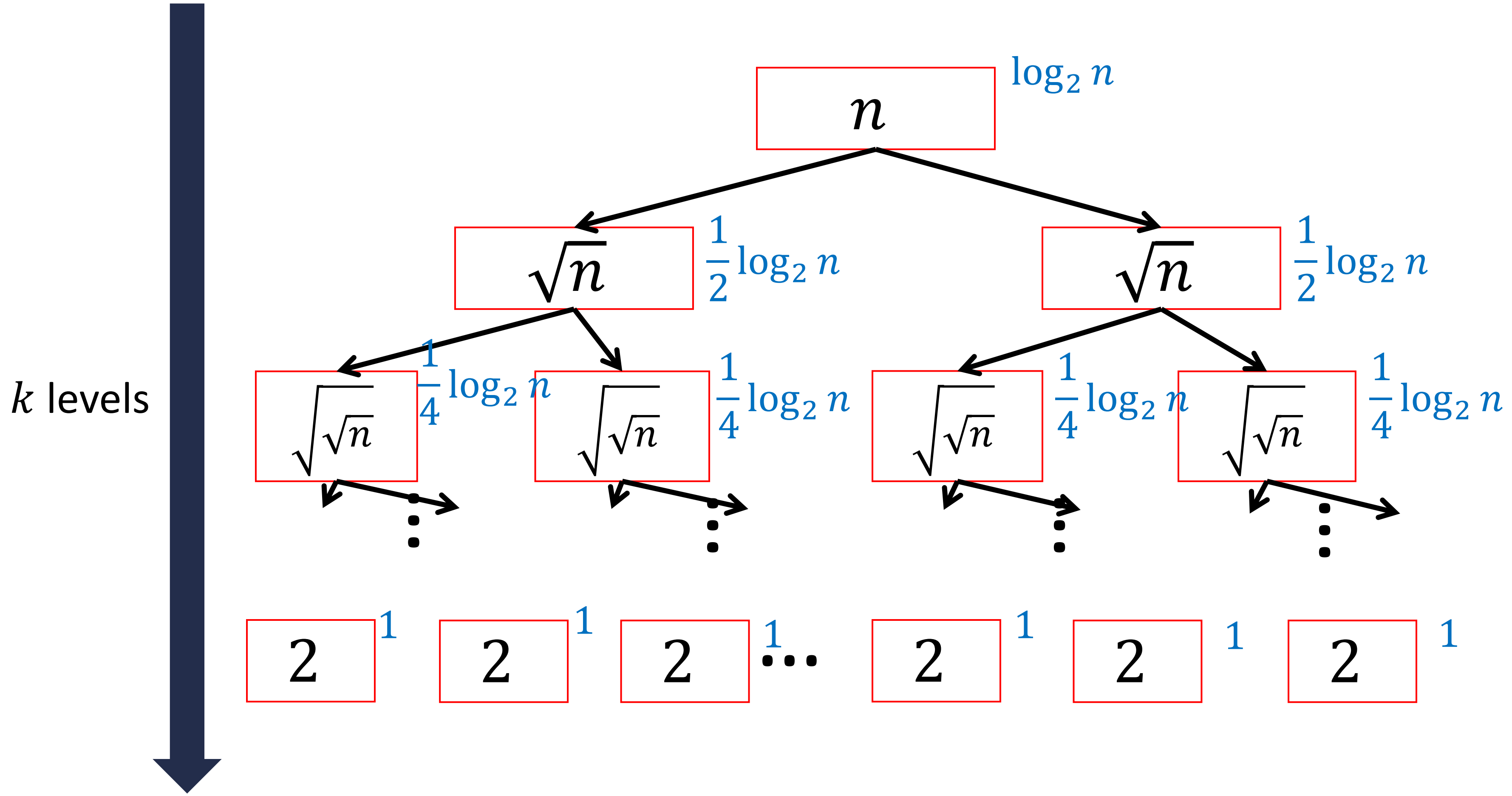
**Idea:** Take a “difficult” recurrence and re-express it such that one of our other methods applies

**Example:**

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$

# Tree Method

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$




| Number of subproblems | Cost of subproblem |
|---|---|
| $1$ | $\log_2 n$ |
| $2$ | $\frac{\log_2 n}{2}$ |
| $4$ | $\frac{\log_2 n}{4}$ |
| $2^k$ | $\frac{\log_2 n}{2^k}$ |

# Tree Method

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$

Each iteration, problem size goes from $n$ to $n^{1/2}$

| Number of subproblems | Cost of subproblem |
|---|---|
| $1$ | $\log_2 n$ |
| $2$ | $\frac{\log_2 n}{2}$ |
| $4$ | $\frac{\log_2 n}{4}$ |
| $2^k$ | $\frac{\log_2 n}{2^k}$ |

How many levels?

Problem size at $k^{\text{th}}$ level: $n^{1/2^k}$

Base case: $n = 2$

At level $k$, it should be the case that $n^{1/2^k} = 2$

$$n^{1/2^k} = 2 \Rightarrow \frac{1}{2^k} \log_2 n = 1$$

$$\Rightarrow 2^k = \log_2 n \Rightarrow k = \log_2 \log_2 n$$

# Tree Method

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$

| | Number of subproblems | Cost of subproblem |
|---|---|---|
| | $1$ | $\log_2 n$ |
| | $2$ | $\frac{\log_2 n}{2}$ |
| | $4$ | $\frac{\log_2 n}{4}$ |
| | $2^k$ | $\frac{\log_2 n}{2^k}$ |

$$k = \log_2 \log_2 n$$

What is the cost?

Cost at level $i$: $2^i \cdot \frac{\log_2 n}{2^i} = \log_2 n$

Total cost: $T(n) = \sum_{i=0}^{\log_2 \log_2 n} \log_2 n = (\log_2 n)(\log_2 \log_2 n)$

# Substitution Method

**Idea:** Take a "difficult" recurrence and re-express it such that one of our other methods applies

**Example:**

$$T(n) = 2T(\sqrt{n}) + \log_2 n$$

Consider the following substitution: let $n = 2^m$ (i.e., $m = \log_2 n$)

$T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m$ — Rewrite recurrence in terms of $m$

$S(m) = 2S\left(\frac{m}{2}\right) + m$ — Consider substitution $S(m) = T(2^m)$

$\Rightarrow S(m) = \Theta(m \log m)$ — Case 2 of Master Theorem

$\Rightarrow T(n) = \Theta(\log n \log \log n)$ — Substitute back for $T$ and $n$