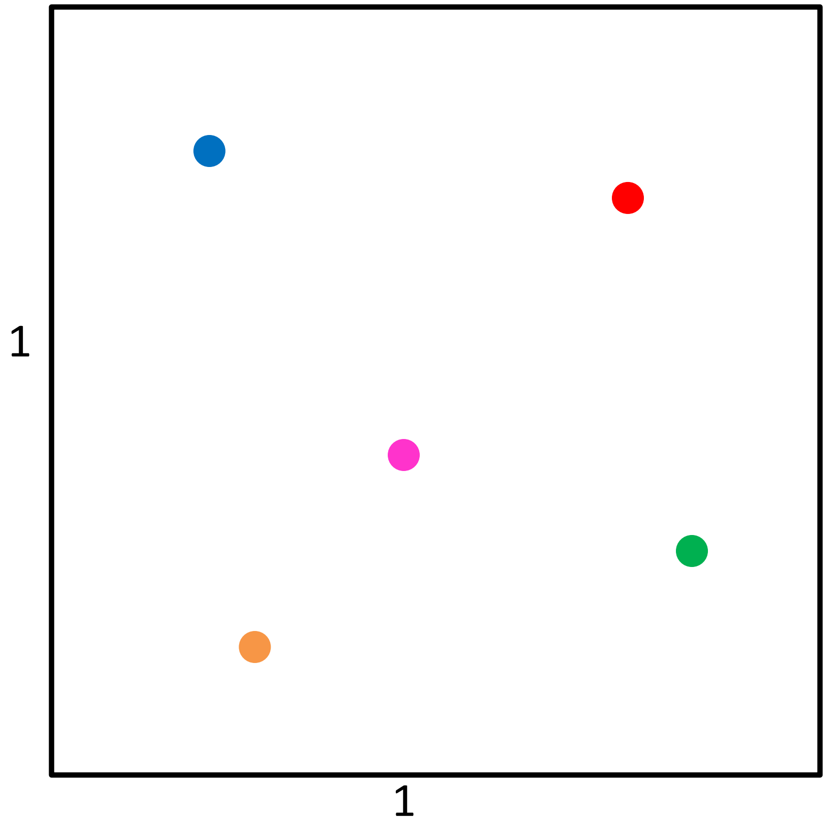
CS4102 Algorithms
Fall 2018

Warm up

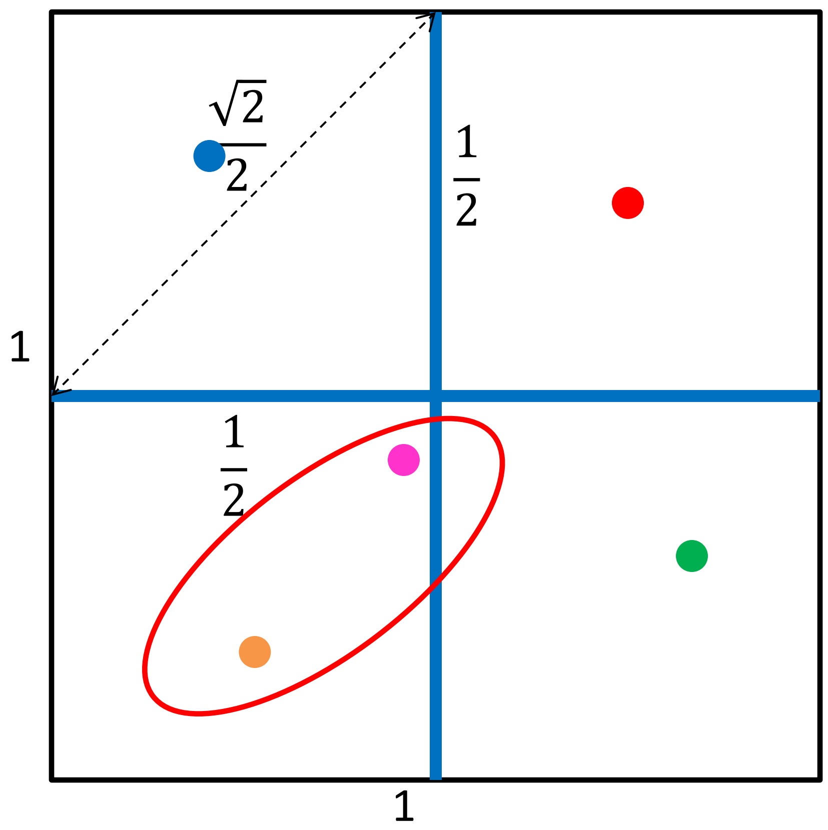
Given any 5 points on the unit square, show there's always a pair distance $\leq \frac{\sqrt{2}}{2}$ apart



If points p_1, p_2 in same quadrant, then $\delta(p_1, p_2) \leq \frac{\sqrt{2}}{2}$

Given 5 points, two must share the same quadrant

Pigeonhole Principle!



Today’s Keywords

- Solving recurrences
- Cookbook Method
- Master Theorem
- Substitution Method

CLRS Readings

- Chapter 4

Homework

- Hw1 due 11pm Wednesday, Sept 12
  – Written (use Latex!)
  – Asymptotic notation
  – Recurrences
  – Divide and conquer
- Hw2 released Thursday, Sept 13
  – Programming assignment (Python or Java)
  – Divide and conquer

Homework

- Hw1 due 11pm ~~Wednesday, Sept 12~~ Friday, Sept 14
  - Written (use Latex!)
  - Asymptotic notation
  - Recurrences
  - Divide and conquer
- Hw2 released Thursday, Sept 13
  - Programming assignment (Python or Java)
  - Divide and conquer

Recurrence Solving Techniques

Tree

Guess/Check

(induction)

“Cookbook”

Substitution

Guess and Check Intuition

- To Prove: $T(n) = O(g(n))$
- Consider: $g_*(n) = O(g(n))$
- Goal: show $\exists n_0$ s.t. $\forall n > n_0, T(n) \leq g_*(n)$
  - (definition of big-O)
- Technique: Induction
  - Base cases:
    - show $T(1) \leq g_*(1), T(2) \leq g_*(2)$, … for a small number of cases
  - Hypothesis:
    - $\forall n \leq x_0, T(n) \leq g_*(n)$
  - Inductive step:
    - $T(x_0 + 1) \leq g_*(x_0 + 1)$

Karatsuba Guess and Check

$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

Goal: $T(n) \leq 24n^{\log_2 3} - 16n = O(n^{\log_2 3})$

Base cases: by inspection, holds for small n (at home)

Hypothesis: $\forall n \leq x_0, T(n) \leq 24n^{\log_2 3} - 16n$

Inductive step: $T(x_0 + 1) \leq 24(x_0 + 1)^{\log_2 3} - 16(x_0 + 1)$

What if we leave out the $-16n$?

$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

Goal: $T(n) \leq 24n^{\log_2 3} - 16n = O(n^{\log_2 3})$

Base cases: by inspection, holds for small n (at home)

Hypothesis: $\forall n \leq x_0, T(n) \leq 24n^{\log_2 3} - 16n$

Inductive step: $T(x_0 + 1) \leq 24(x_0 + 1)^{\log_2 3} - 16(x_0 + 1)$

What we wanted: $T(x_0 + 1) \leq 24(x_0 + 1)^{\log_2 3}$ Induction failed!

What we got: $T(x_0 + 1) \leq 24(x_0 + 1)^{\log_2 3} + 8(x_0 + 1)$

“Bad Mergesort” Guess and Check

$$T(n) = 2T(\frac{n}{2}) + 209n$$

Goal: $T(n) \leq 209n\log_2 n \ = O(n\log_2 n)$

Base cases: $T(1) = 0$

$T(2) = 518 \leq 209 \cdot 2\log_2 2$

… up to some small k

Hypothesis: $\forall n \leq x_0, T(n) \leq 209n\log_2 n$

Inductive step: $T(x_0 + 1) \leq 209(x_0 + 1)\log_2(x_0 + 1)$

Recurrence Solving Techniques


Tree


Guess/Check

"Cookbook"

Substitution

Observation

- **Divide**: $D(n)$ time,
- **Conquer**: recurse on small problems, size s
- **Combine**: $\mathrm{C}(n)$ time
- **Recurrence:**
  - $T(n) = D(n) + \sum T(s) + C(n)$

- Many D&C recurrences are of the form:
  - $T(n) = aT\left(\frac{n}{b}\right) + f(n),$ where $f(n) = D(n) + C(n)$

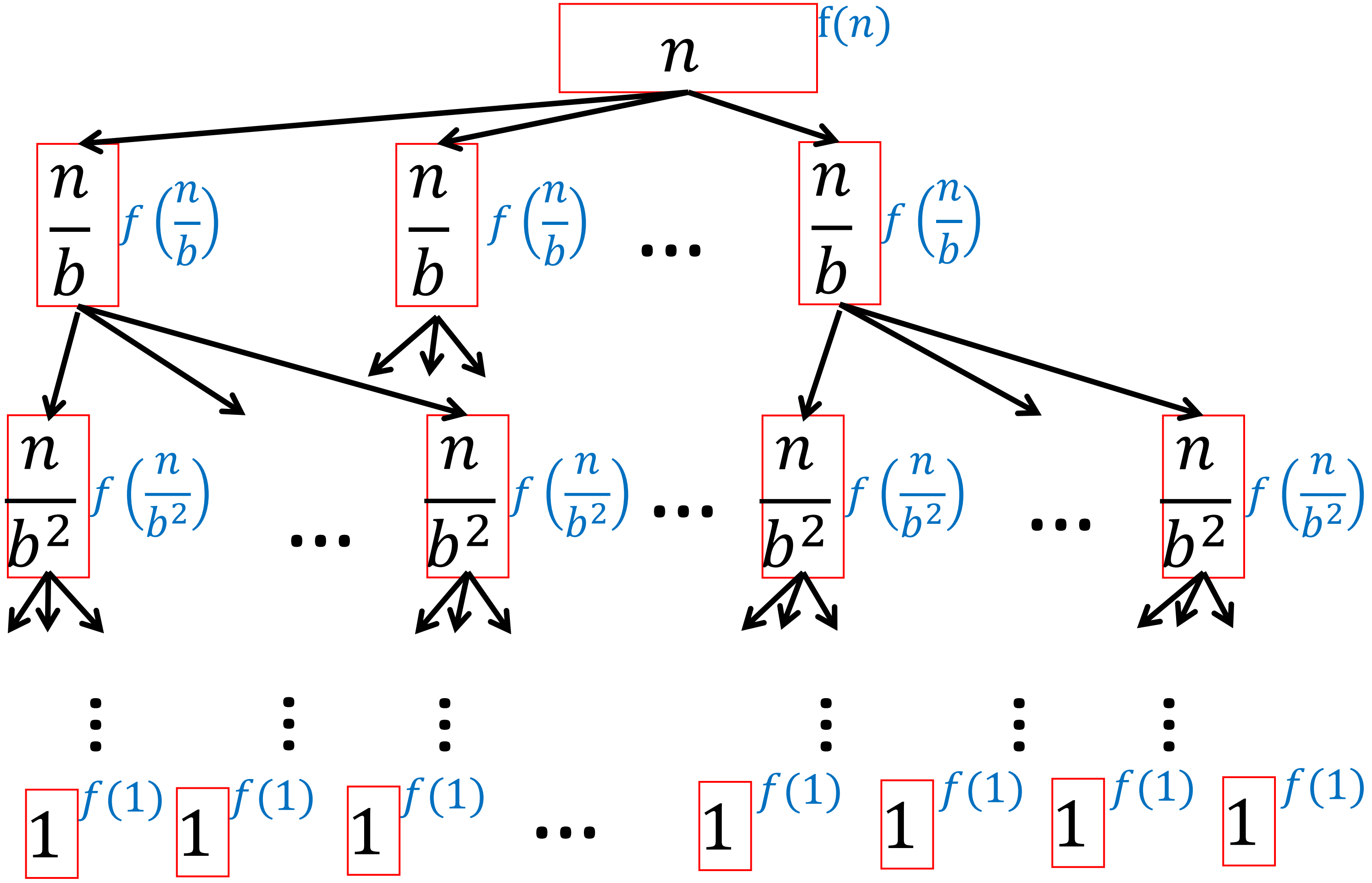
Remember…

- Better Attendance: $T(n) = T\left(\frac{n}{2}\right) + 2$
- MergeSort: $T(n) = 2\,T\left(\frac{n}{2}\right) + n$
- D&C Multiplication: $T(n) = 4T\left(\frac{n}{2}\right) + 5n$
- Karatsuba: $T(n) = 3T\left(\frac{n}{2}\right) + 8n$

General

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

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



$f(n)$

$af\left(\frac{n}{b}\right)$

$a^2 f\left(\frac{n}{b^2}\right)$

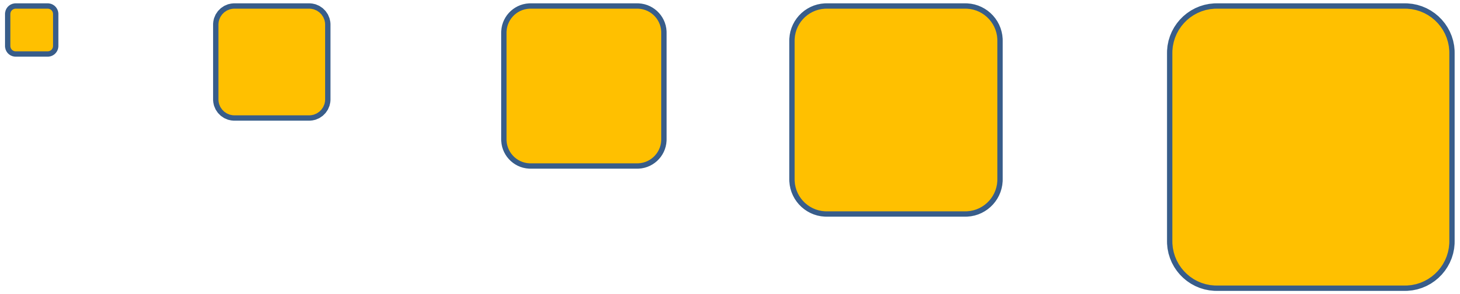
$\vdots$

$a^{\log_b n} f(1)$

3 Cases

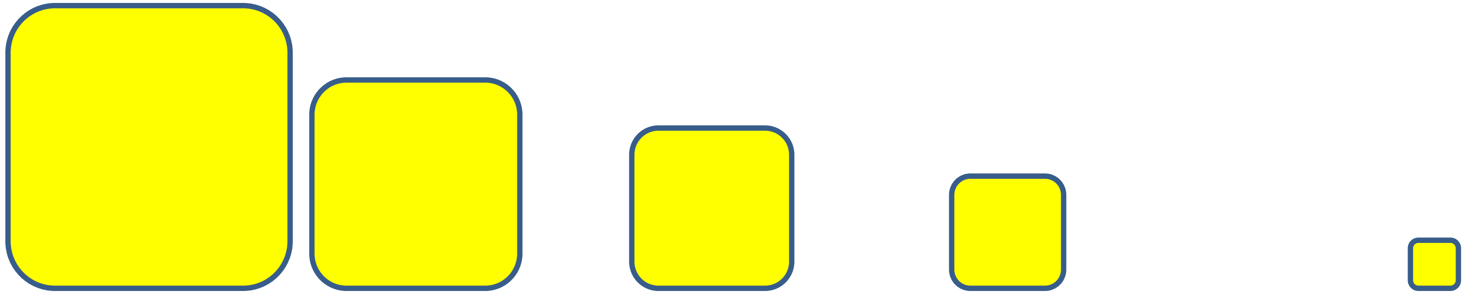
$$T(n) = f(n) + af\left(\frac{n}{b}\right) + a^2 f\left(\frac{n}{b^2}\right) + a^3 f\left(\frac{n}{b^3}\right) + \cdots + a^L f\left(\frac{n}{b^L}\right)$$

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

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

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $af\left(\frac{n}{b}\right) \le cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Proof of Case 1

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

$$f(n) = O\left(n^{\log_b a - \varepsilon}\right) \Rightarrow f(n) \leq c \cdot n^{\log_b a - \varepsilon}$$

Insert math here…

Conclusion: $T(n) = O(n^{\log_b a})$

Master Theorem Example 1

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

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

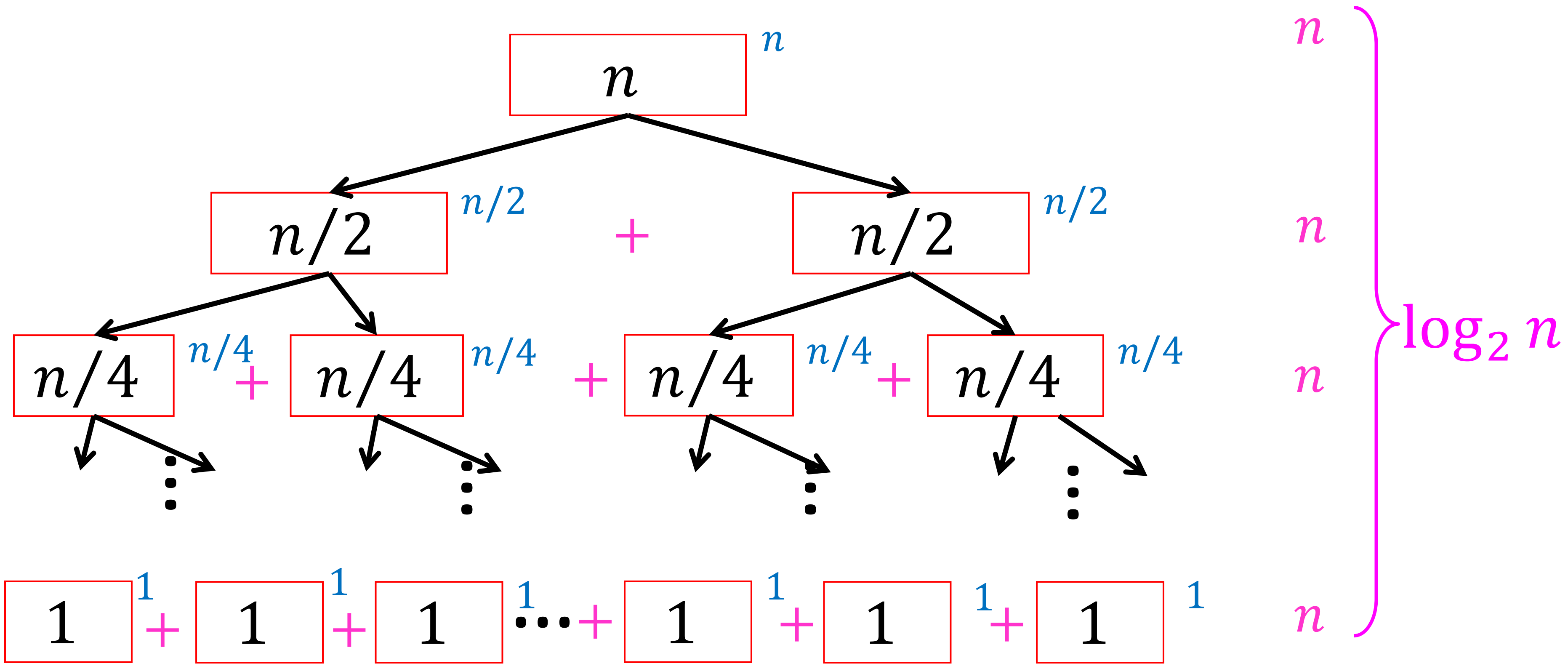
$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

Case 2

$$\Theta\left(n^{\log_2 2} \log n\right) = \Theta(n \log n)$$

Tree method

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$



Master Theorem Example 2

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

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a+\varepsilon})$ for some constant $\varepsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

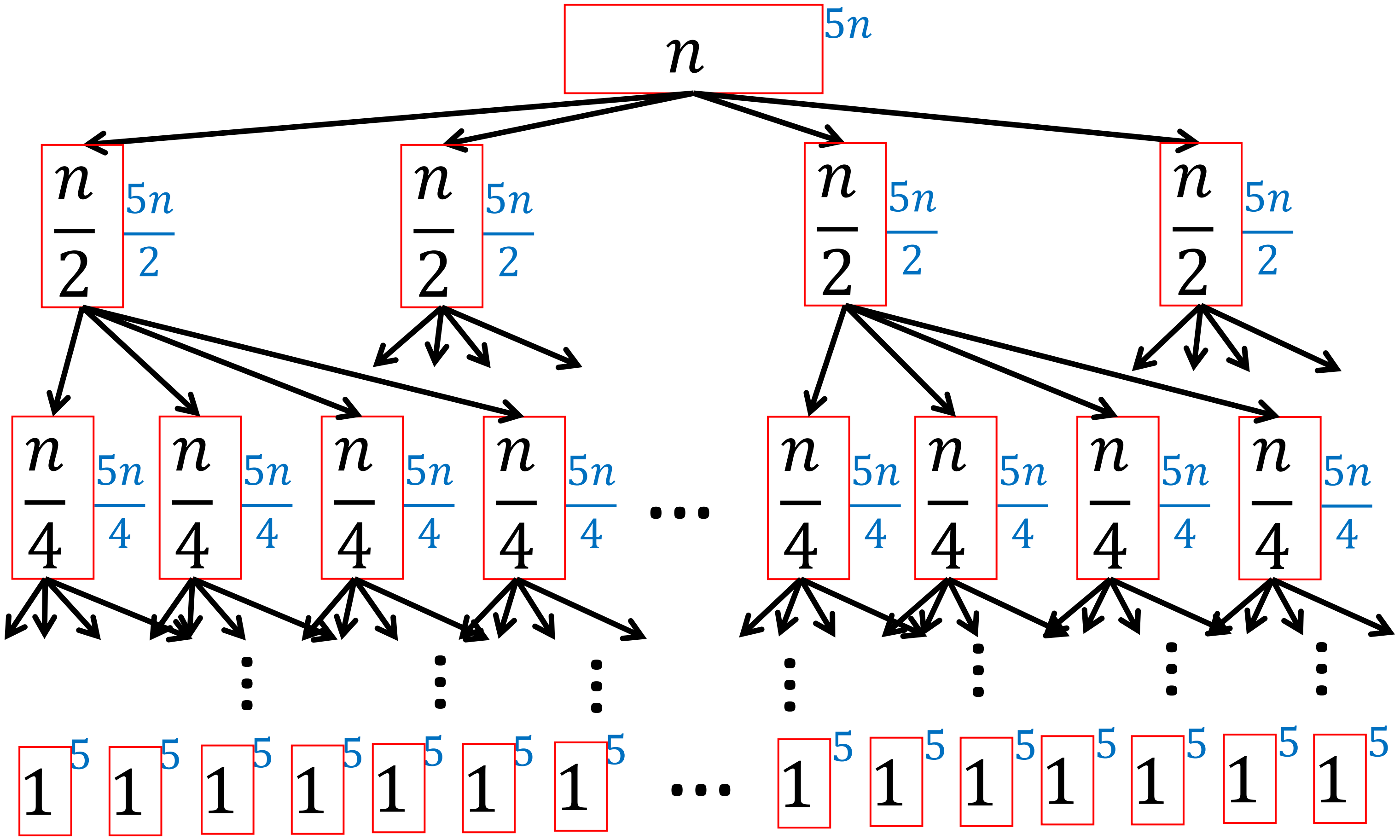
$$T(n) = 4T\left(\frac{n}{2}\right) + 5n$$

Case 1

$$\Theta\left(n^{\log_2 4}\right) = \Theta(n^2)$$

Tree method

$$T(n) = 4T\left(\frac{n}{2}\right) + 5n$$



$5n$

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

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

$\vdots$

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

Master Theorem Example 3

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

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

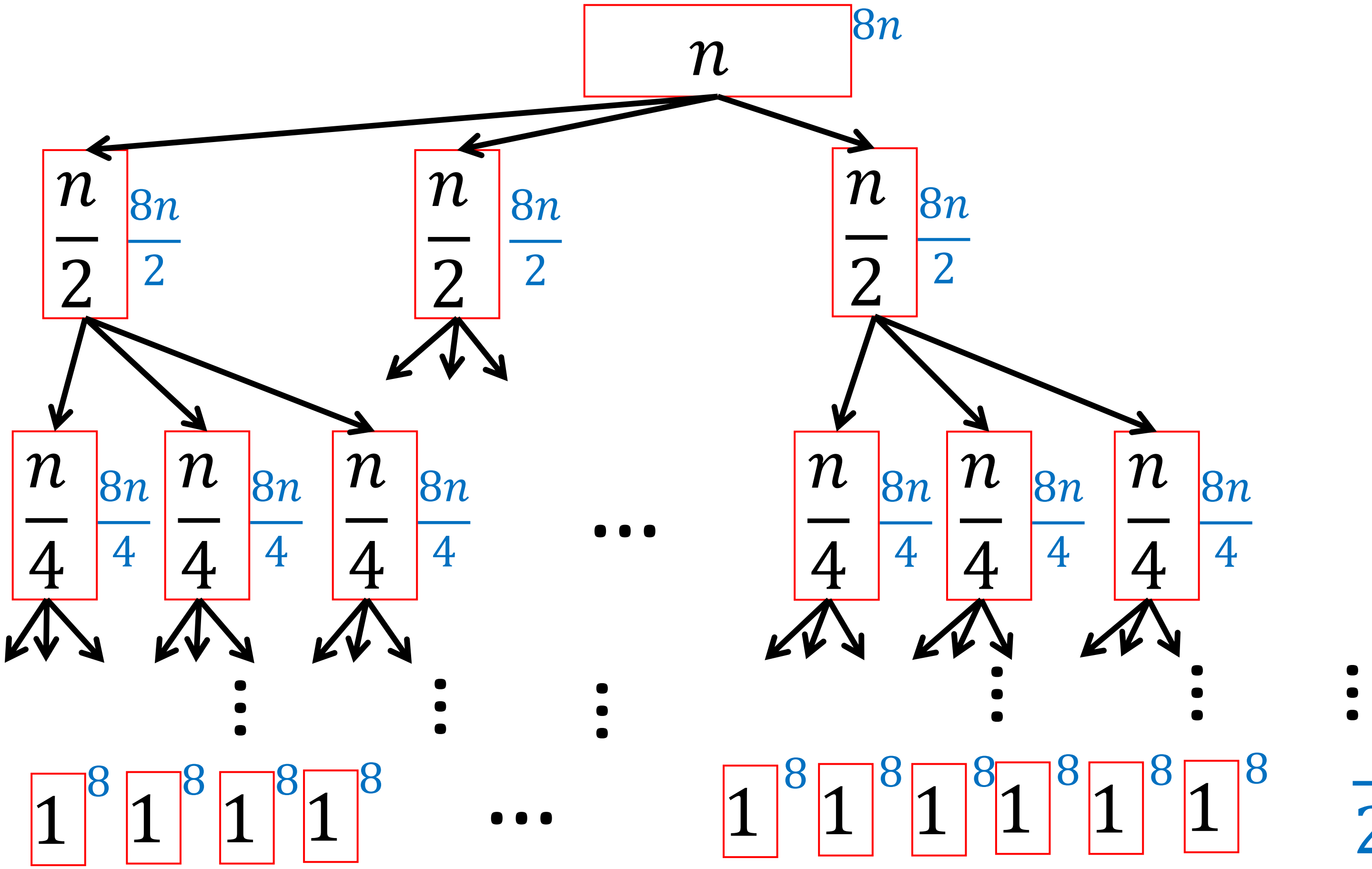
$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

Case 1

$$\Theta\left(n^{\log_2 3}\right) \approx \Theta(n^{1.5})$$

Karatsuba

$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$



$$8 \cdot 1n$$

$$\frac{8}{2} \cdot 3n$$

$$\frac{8}{4} \cdot 9n$$

$$\vdots$$

$$\frac{8}{2^{\log_2 n}} \cdot 3^{\log_2 n} n$$

Master Theorem Example 4

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

- Case 1: if $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
- Case 2: if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$
- Case 3: if $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

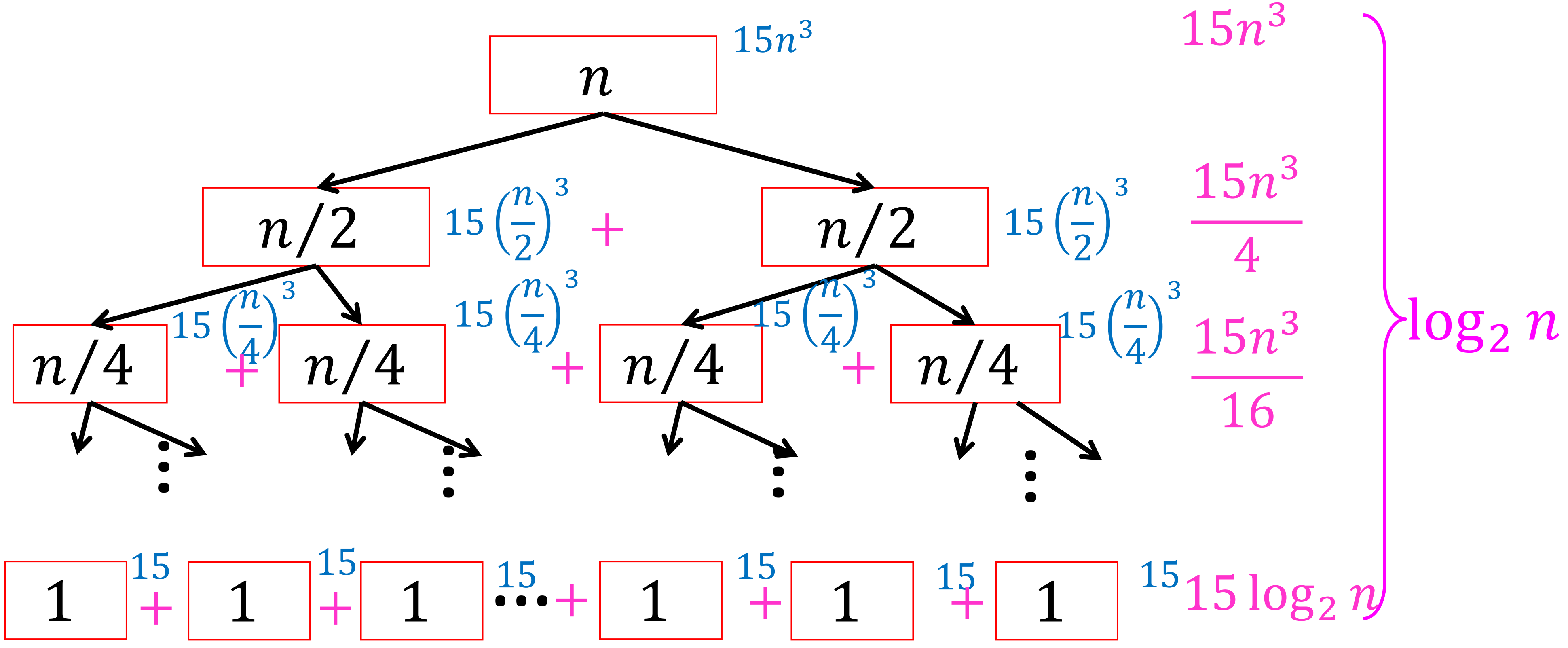
$$T(n) = 2T\left(\frac{n}{2}\right) + 15n^3$$

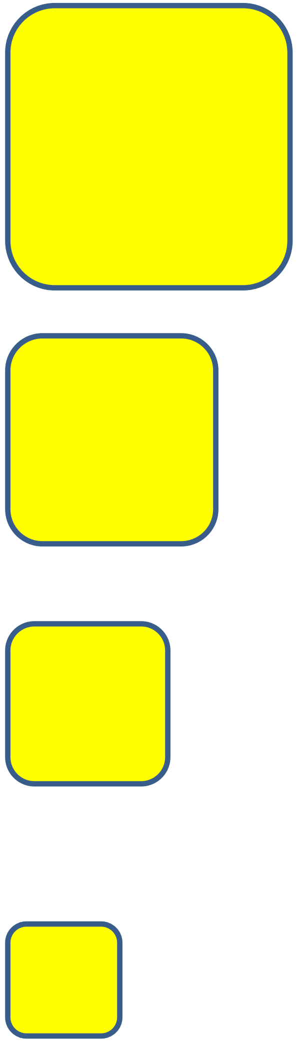
Case 3

$\Theta(n^3)$

Tree method

$$T(n) = 2T\left(\frac{n}{2}\right) + 15n^3$$





Recurrence Solving Techniques

Tree

Guess/Check

"Cookbook"

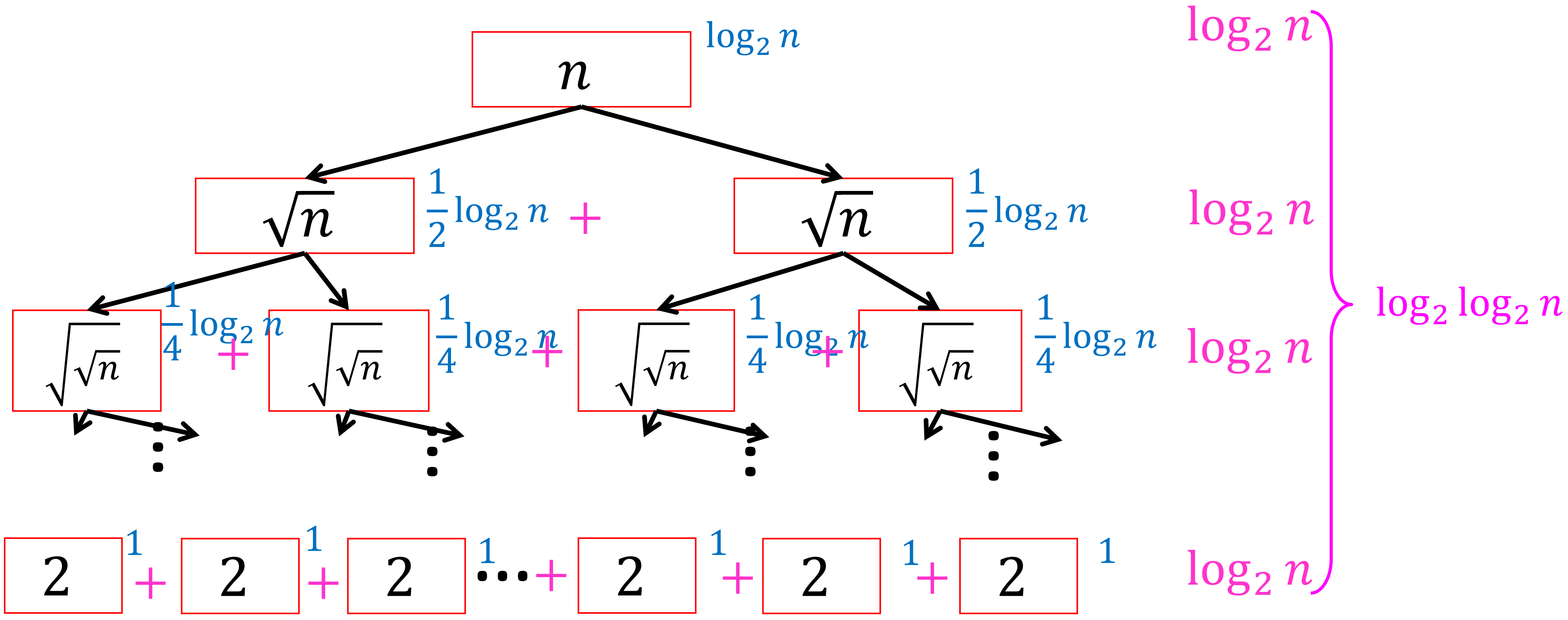
Substitution

Substitution Method

- Idea: take a “difficult” recurrence, re-express it such that one of our other methods applies.
- Example: $T(n) = 2T(\sqrt{n}) + \log_2 n$

Tree method

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



$$T(n) = O(\log_2 n \cdot \log_2 \log_2 n)$$

Substitution Method

- Idea: take a "difficult" recurrence, re-express it such that one of our other methods applies.
- Example:

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

Let $n = 2^m$, i.e. $m = \log_2 n$

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

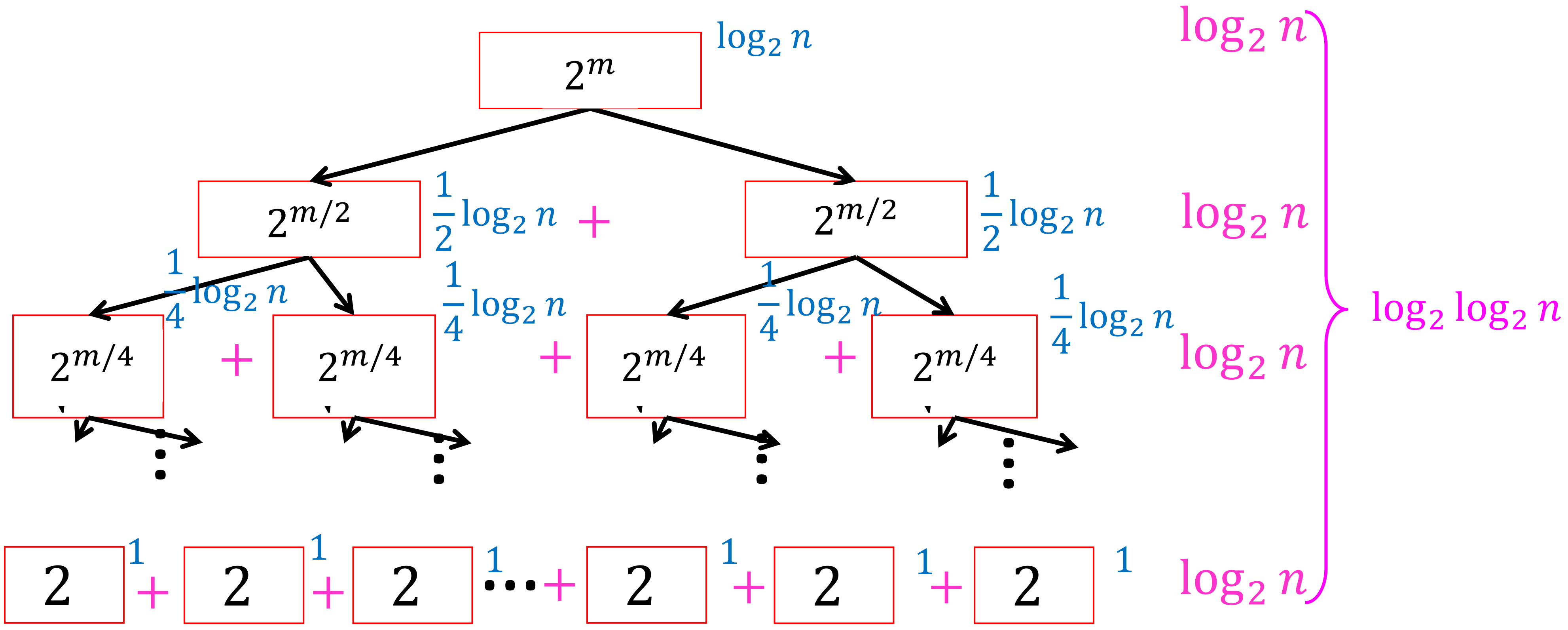
Let $S(m) = 2S\left(\frac{m}{2}\right) + m$ Case 2!

Let $S(m) = \Theta(m \log m)$ Substitute Back

Let $\mathrm{T}(n) = \Theta(\log n \log \log n)$

Tree method

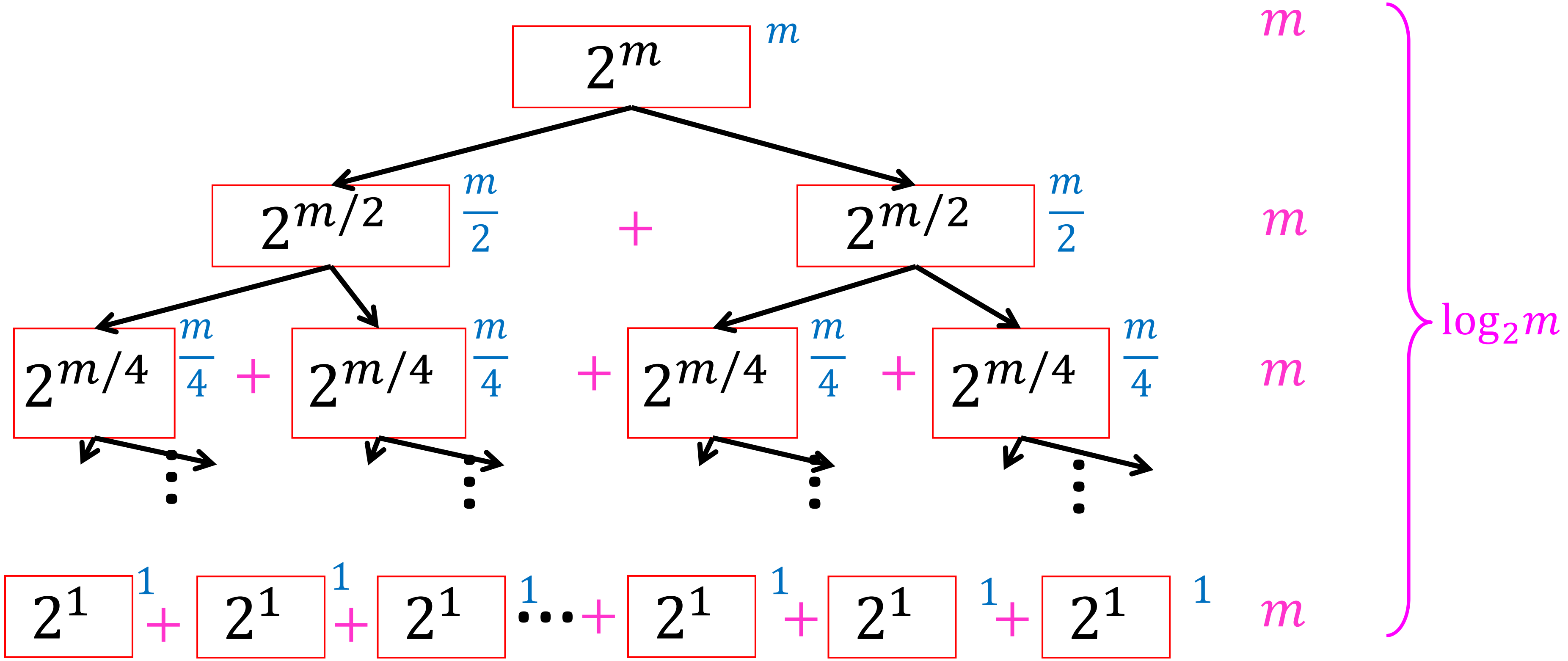
$n = 2^m \qquad T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m$



Tree method

$n = 2^m$

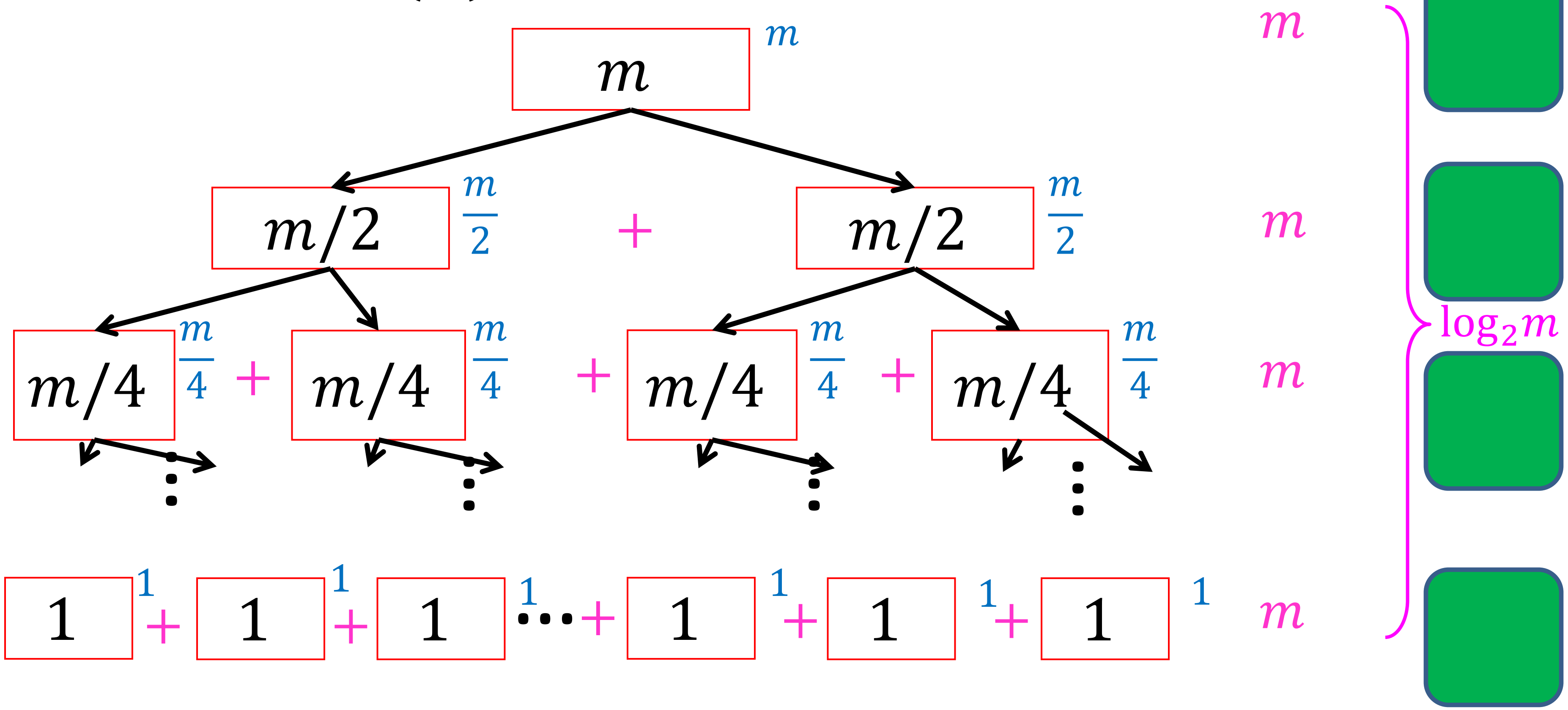
$T(2^m) = 2T(2^m) + \log_2 n$



Tree method

$$n = 2^m$$
$$T(2^m) = S(m)$$

$$S(m) = 2S\left(\frac{m}{2}\right) + m$$



$$T(n) = O(m \cdot \log_2 m) = O(\log_2 n \cdot \log_2 \log_2 n)$$