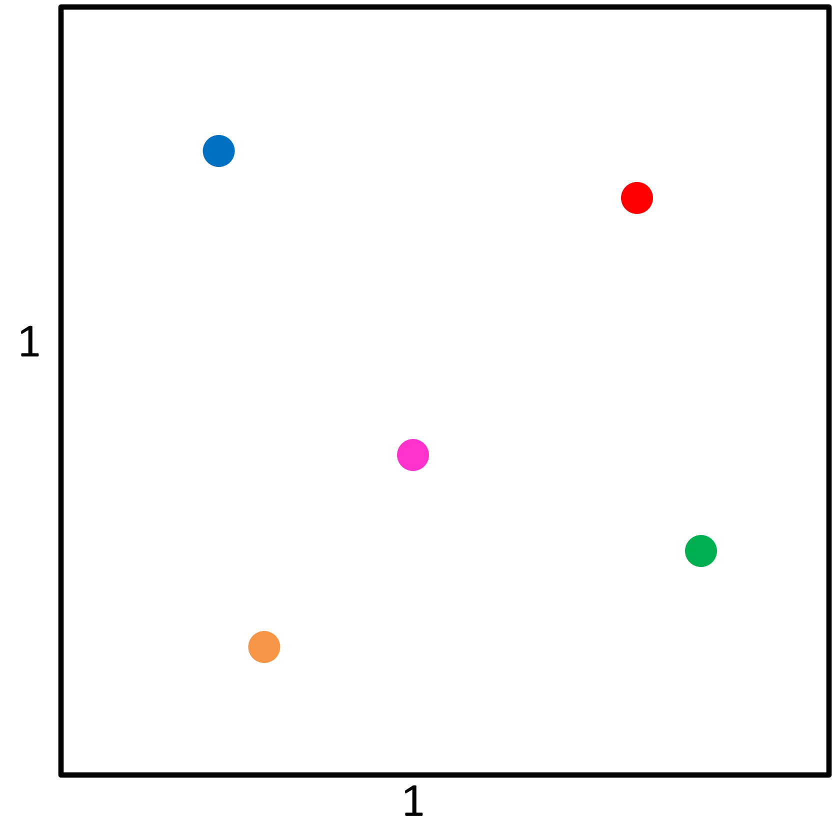
CS4102 Algorithms

Nate Brunelle

Spring 2018

Warm up

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



Today’s Keywords

- Karatsuba
- Guess and check Method
- Induction

CLRS Readings

- Chapter 4

Homeworks

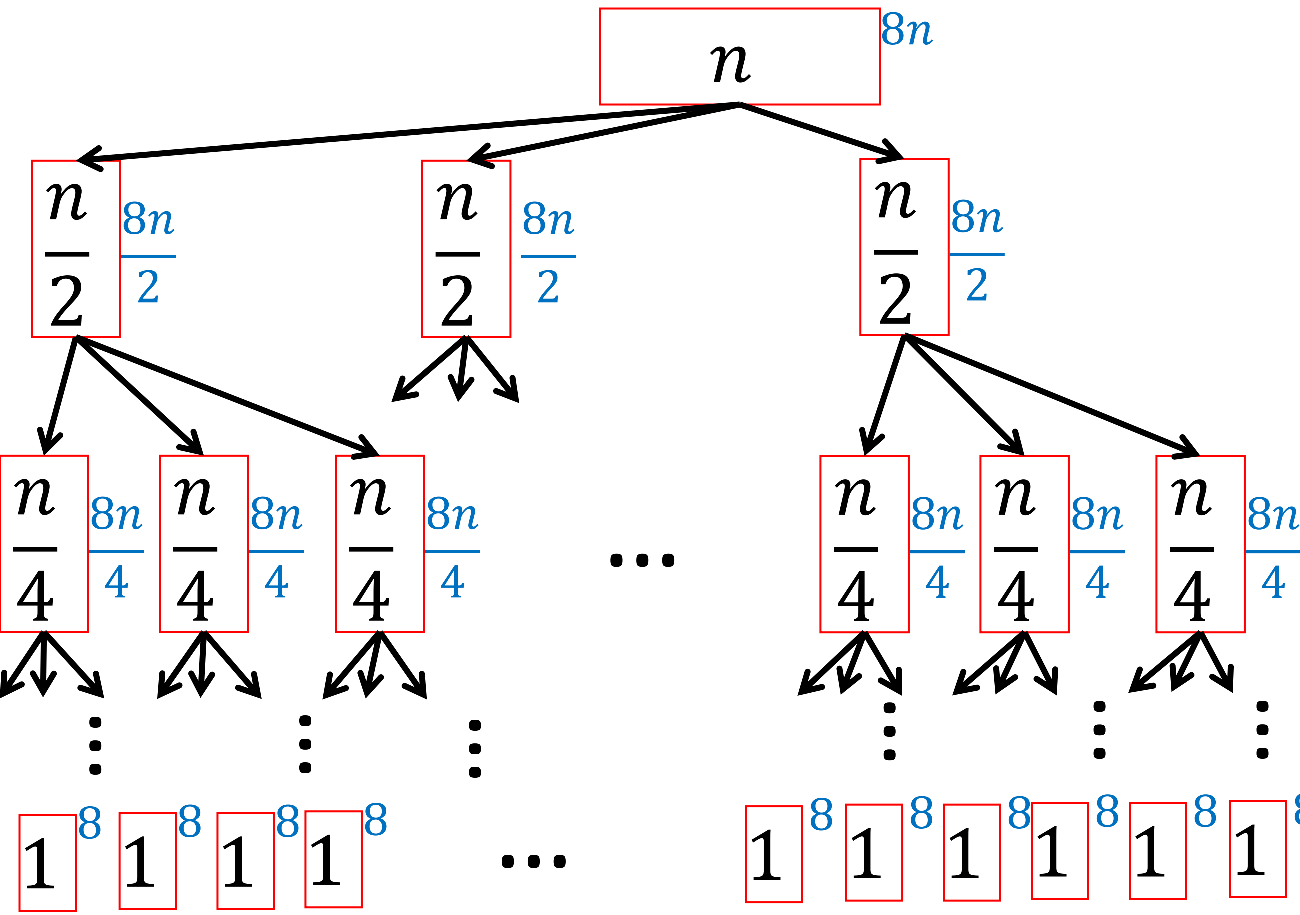
- Hw1 due 11pm Friday, February 2
  - Written (use Latex!)
  - Asymptotic notation
  - Recurrences
  - Divide and conquer

Karatsuba

$$T(n) = 8n \sum_{i=0}^{\log_2 n} \left({}^{3}/_{2}\right)^{i}$$

3. Use asymptotic notation to simplify

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



$$8 \cdot 1n$$

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

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

$$\vdots$$

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

Karatsuba

3. Use asymptotic notation to simplify

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

$$T(n) = 8n \sum_{i=0}^{\log_2 n} \left({}^{3}/{}_{2}\right)^{i}$$

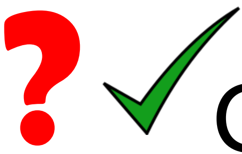
$$T(n) = 8n \frac{\left({}^{3}/{}_{2}\right)^{\log_2 n+1} - 1}{{}^{3}/{}_{2} - 1}$$

Math, math, and more math…(on board, see lecture supplemental)

$$T(n) = 24\left(n^{\log_2 3}\right) - 16n \;\; = \Theta\left(n^{\log_2 3}\right)$$
$$\approx \Theta\left(n^{1.585}\right)$$

Recurrence Solving Techniques


Tree


Guess/Check

(induction)

"Cookbook"

Substitution

Induction (review)

Goal: $\forall k, P(k)$ holds

Base cases: $P(1)$ holds

Hypothesis: $\forall n \leq n_0, P(n)$ holds

Inductive step: $P(n_0) \Rightarrow P(n_0 + 1)$

Guess and Check Intuition

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

Karatsuba Guess and Check (Loose)

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

Goal: $T(n) < 3000\, n^{1.6} = O(n^{1.6})$

Base cases: $T(1) = 8 < 3000$

$T(2) = 3(8) + 16 = 40 < 3000 \cdot 2^{1.6}$

… up to some small k

Hypothesis: $\forall n < n_0\; T(n) < 3000 n^{1.6}$

Inductive step: $T(n_0 + 1) < 3000(n_0 + 1)^{1.6}$

[see lecture supplemental]

Mergesort Guess and Check

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

Goal: $T(n) < 2n\log n \ = O(n\log n)$

Base cases: $T(1) = 0$

$T(2) = 2 < 4\log 2$

… up to some small k

Hypothesis: $\forall n < n_0\, T(n) < n\log n$

Inductive step: $T(n_0 + 1) < 2(n_0 + 1)\log(n_0 + 1)$

[see lecture supplemental]

Karatsuba Guess and Check

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

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

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

Hypothesis: $\forall n < n_0 \; T(n) < n^{\log_2 3} - 16n$

Inductive step: $T(n_0 + 1) < (n_0 + 1)^{\log_2 3} - 16(n_0 + 1)$

see lecture supplemental

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

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

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

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

Hypothesis: $\forall n < n_0\ T(n) < n^{\log_2 3} - 16n$

Inductive step: $T(n_0 + 1) < (n_0 + 1)^{\log_2 3} - 16(n_0 + 1)$

What we wanted: $T(n_0 + 1) < (n_0 + 1)^{\log_2 3}$ Induction failed!

What we got: $T(n_0 + 1) < (n_0 + 1)^{\log_2 3} + 8(n_0 + 1)$

Recurrence Solving Techniques

Tree

Guess/Check

"Cookbook"

Substitution

Observation

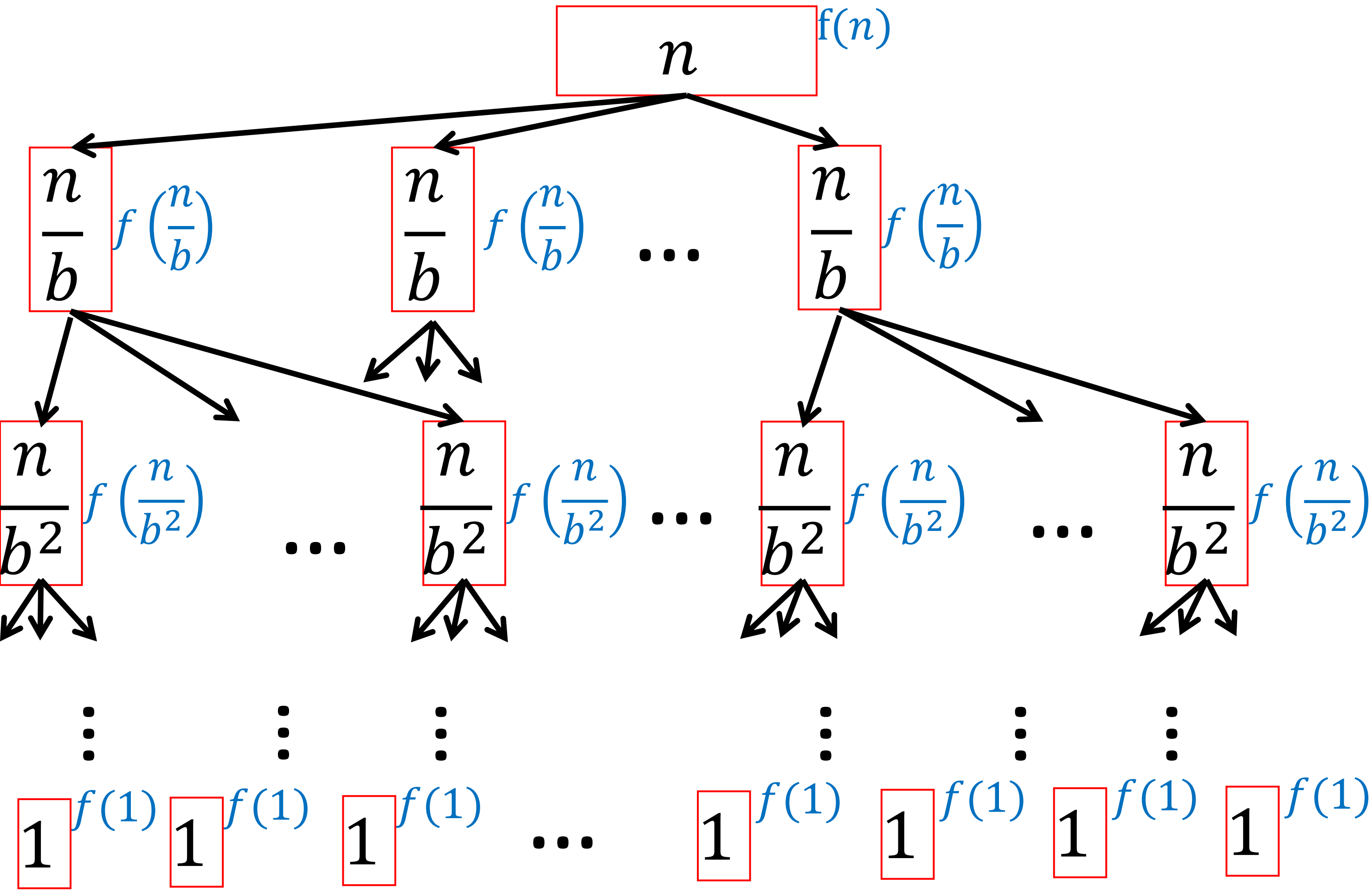
- **Divide**: $D(n)$ time,
- **Conquer**: recurse on small problems, size s
- **Combine**: $\mathrm{C}(n)$ time
- **Recurrence:**
  - $T(n) = D(n) + \sum T(s) + C(n)$

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

General

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

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



$$f(n)$$

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

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

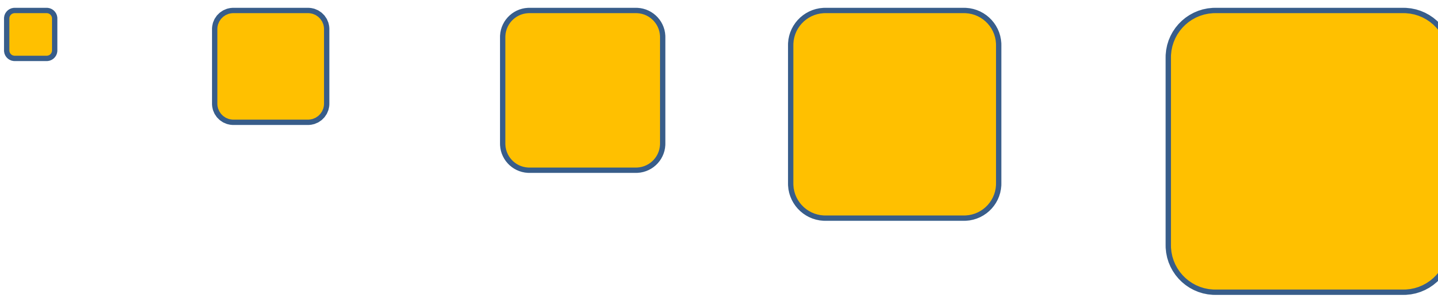
$$\vdots$$

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

3 Cases

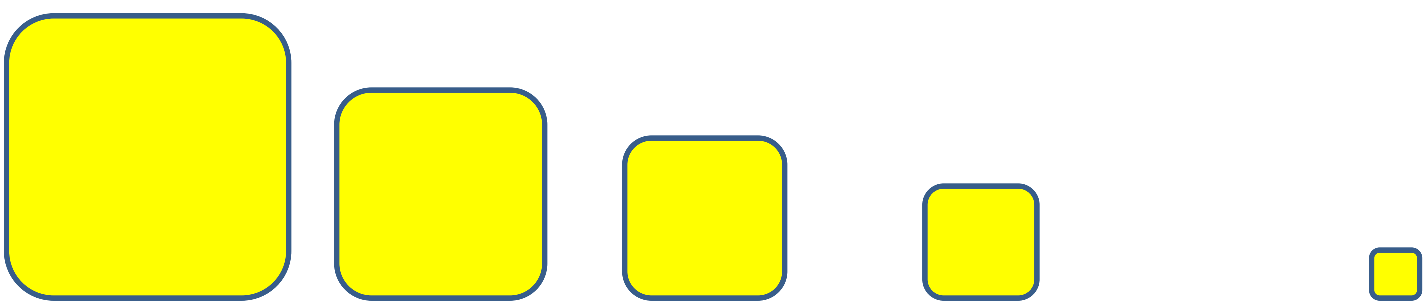
$$T(n) = f(n) + af\left(\frac{n}{b}\right) + a^2 f\left(\frac{n}{b^2}\right) + a^3 f\left(\frac{n}{b^3}\right) + \cdots + a^L f\left(\frac{n}{b^L}\right)$$

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 $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Proof of Case 1

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

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

Insert math here…

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

see lecture supplemental

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

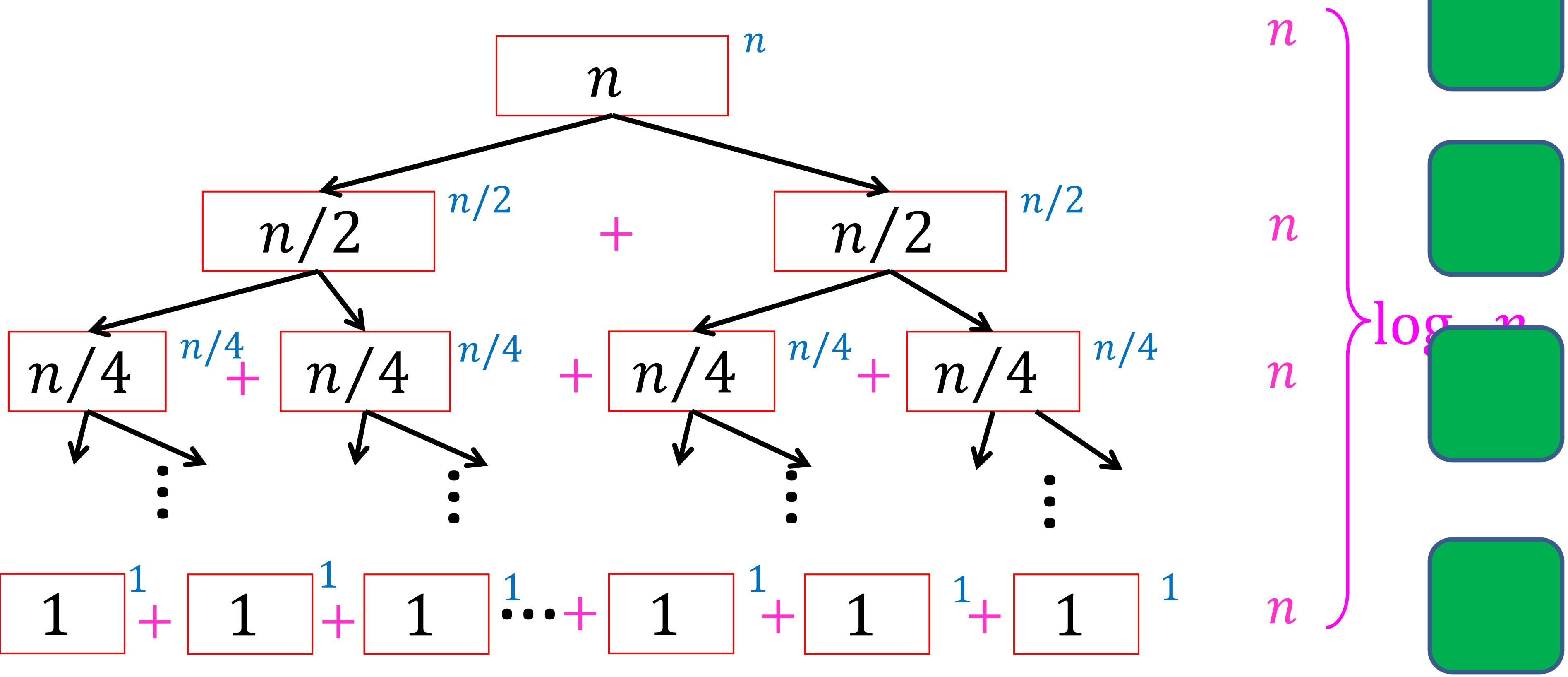
$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

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

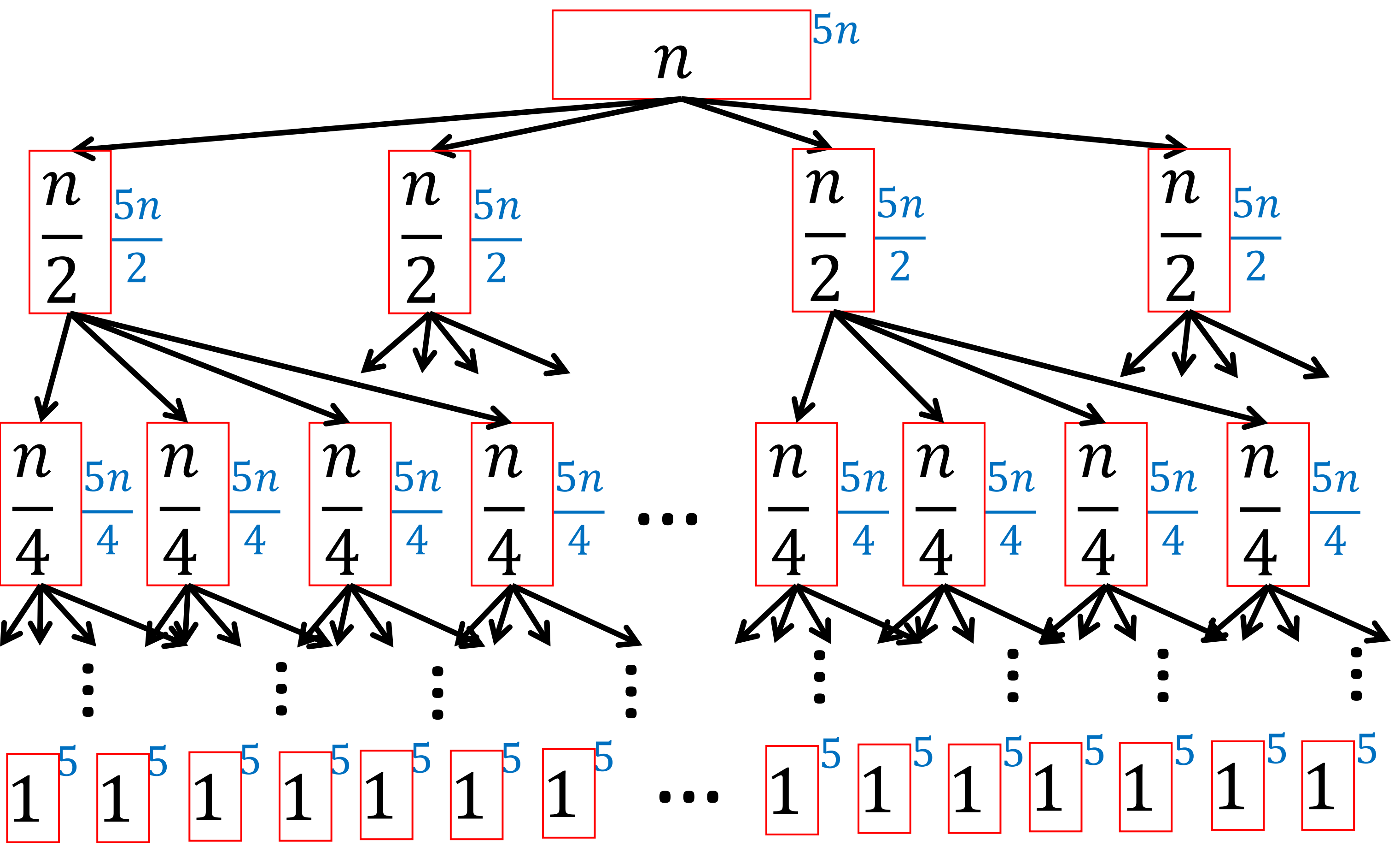
$$T(n) = 4T\left(\frac{n}{2}\right) + 5n$$

Case 1

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

Tree method

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



$5n$

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

$\frac{16}{4}$

⋮

$2^{\log_2}$

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

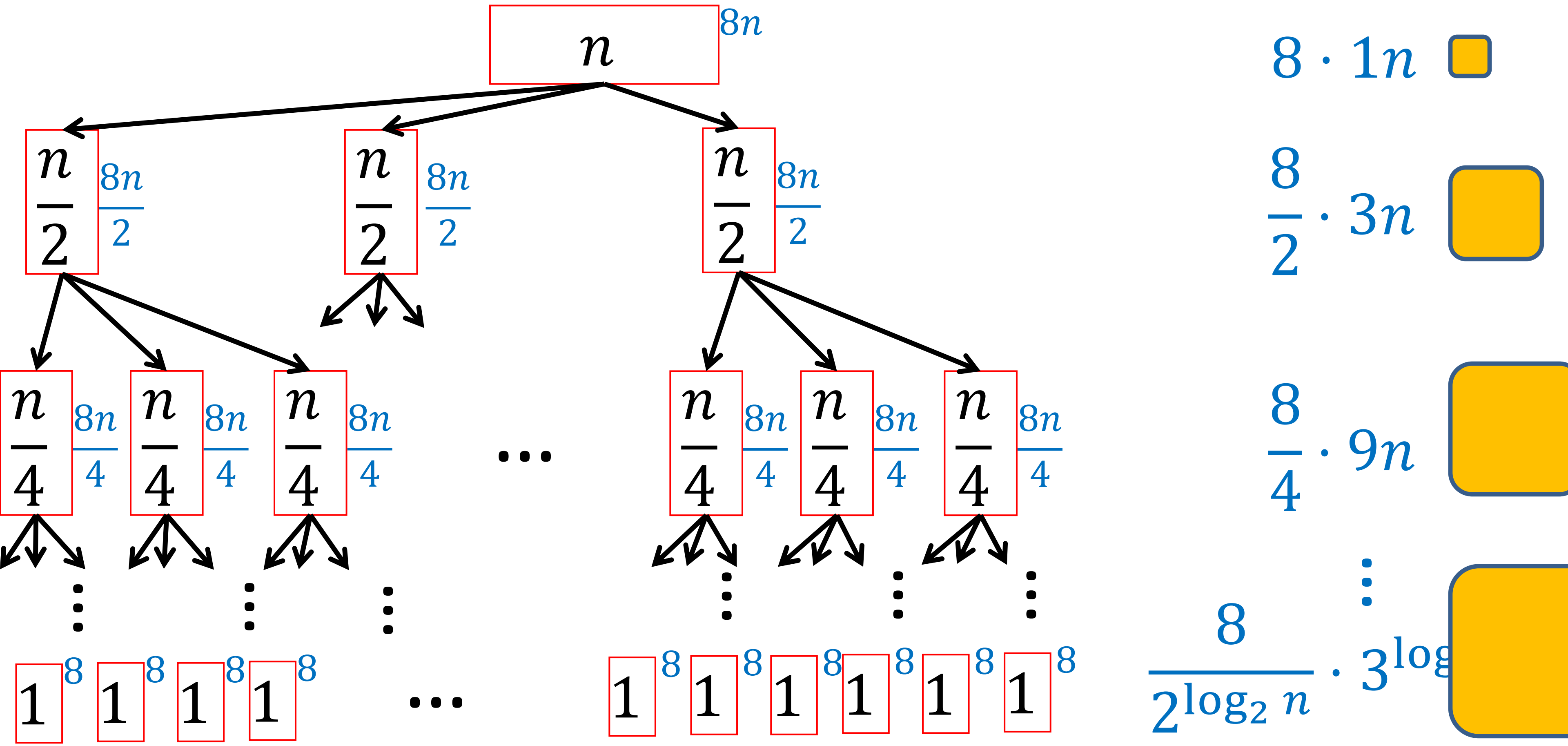
$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

Case 1

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

Karatsuba

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



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

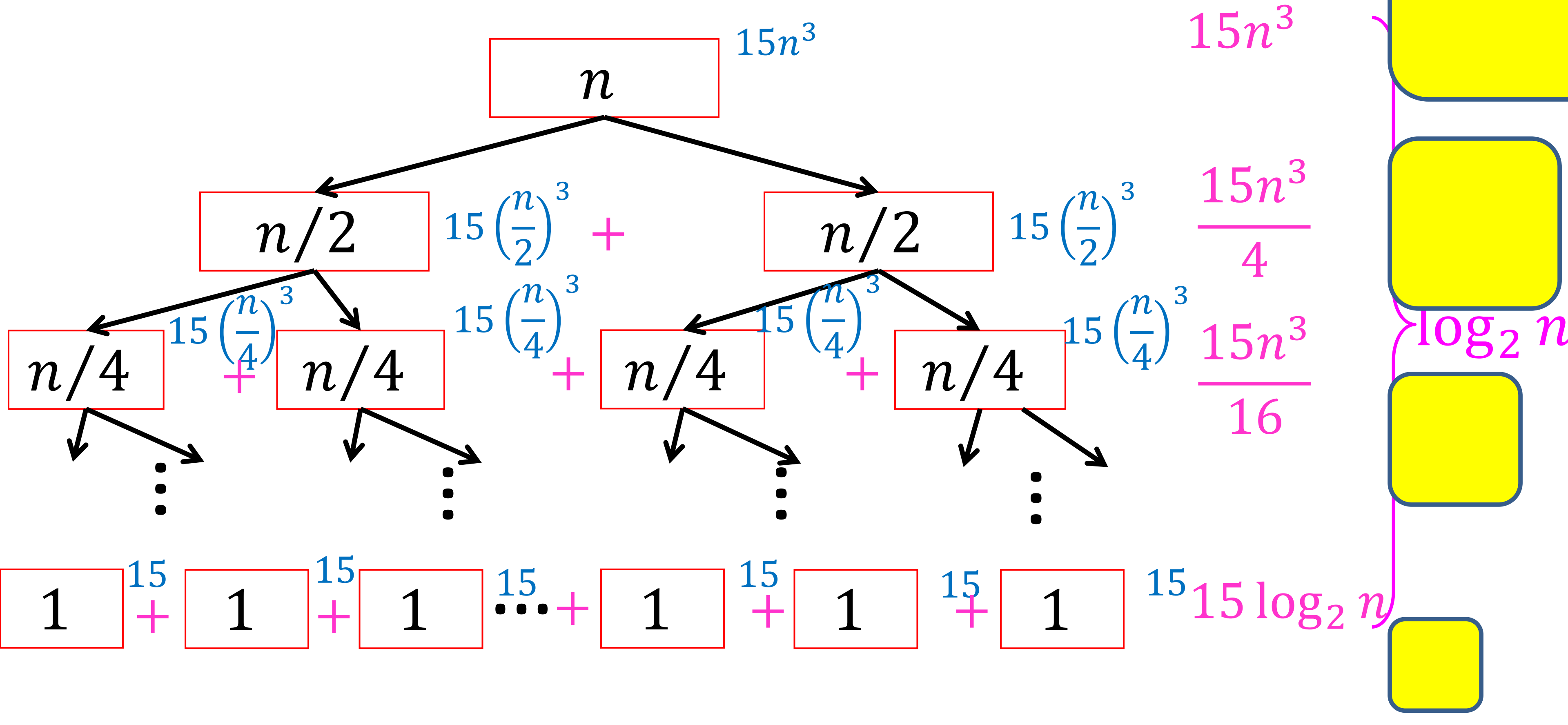
$$T(n) = 2T\left(\frac{n}{2}\right) + 15n^3$$

Case 3

$\Theta(n^3)$

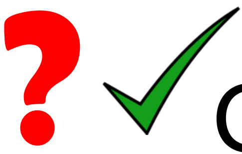
Tree method

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



Recurrence Solving Techniques

Tree



Guess/Check

“Cookbook”

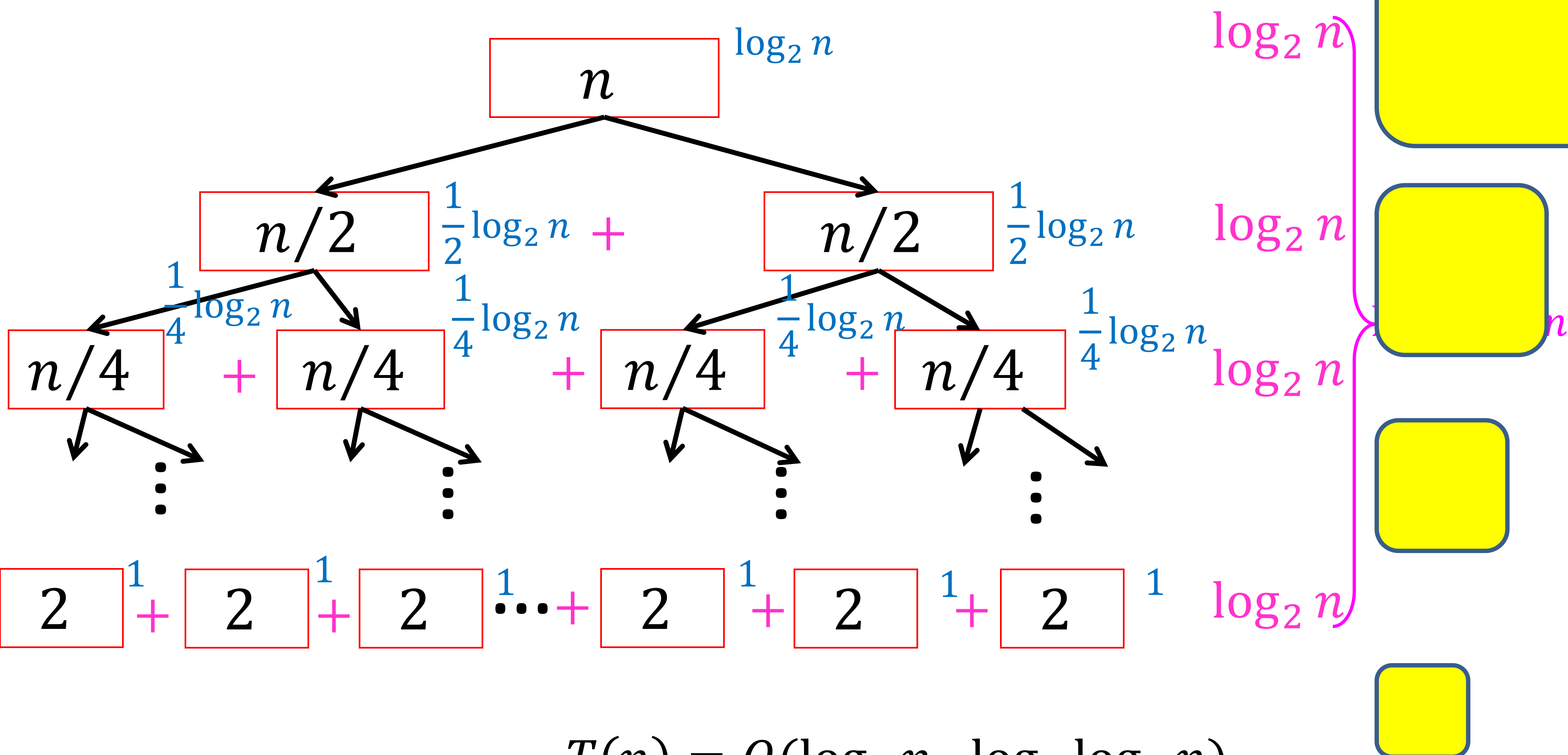
Substitution

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