CS4102 Algorithms

Spring 2020 (Horton's lecture slides)

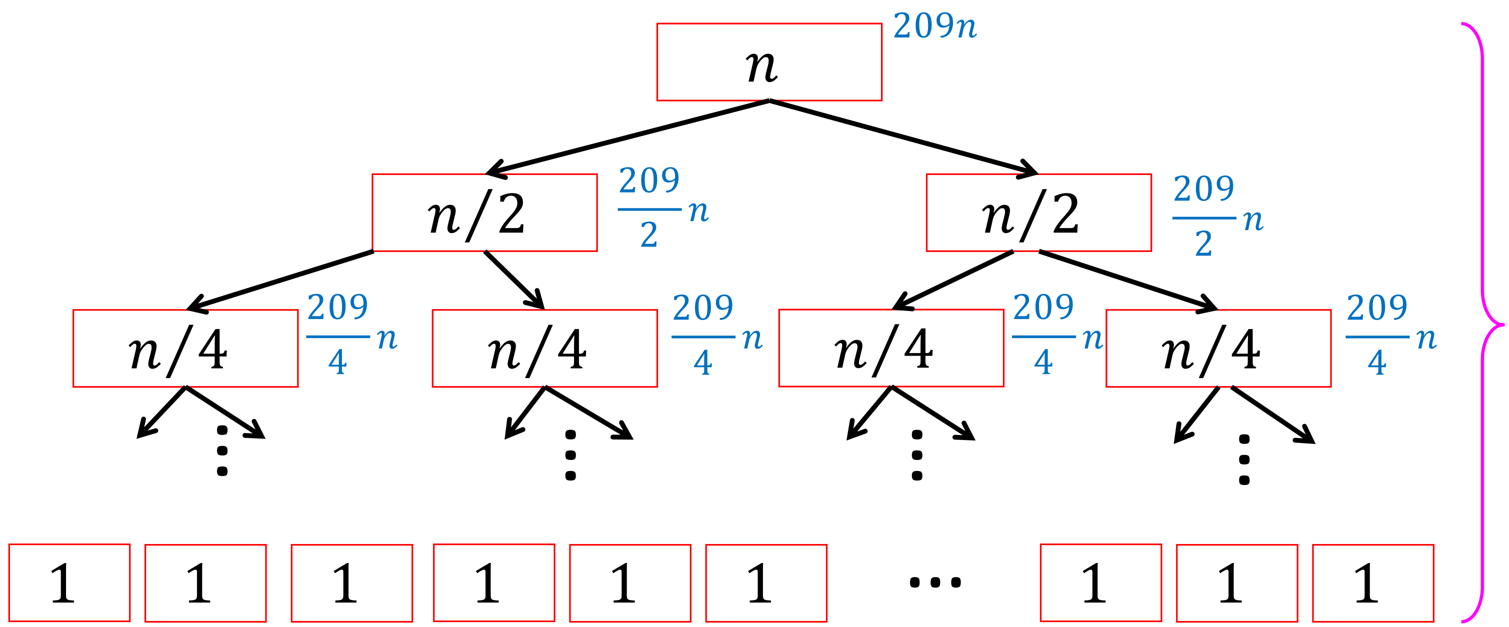
Warm Up

What is the asymptotic run time of MergeSort if its recurrence is

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

Tree Method

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



$\Rightarrow$ $209n$ per level

$\log_2 n + 1$ levels of recursion

$$T(n) = 209n \sum_{i=0}^{\log n} 1 = 209n \log_2 n$$

Tree Method

$$T(n) = 2T(n/2) + 209n$$

	Number of subproblems	Cost of subproblem
	1	$209n$
	2	$209n/2$
	4	$209n/4$
	2^k	$209n/2^k$

What is the cost?

Cost at level i: $2^i \cdot \frac{209n}{2^i} = 209n$

Total cost: $T(n) = \sum_{i=0}^{\log_2 n} 209n$

$$= 209n \sum_{i=0}^{\log_2 n} 1 \quad = 209n(\log_2 n + 1)$$

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

Today's Keywords

- Karatsuba (finishing up)
- Guess and Check Method
- Induction
- Master Theorem

CLRS Readings

- Chapter 4

Homeworks

- Hw1 due Thursday, January 30 at 11pm
  - Start early!
  - Written (use Latex!) – Submit BOTH pdf and zip!
  - Asymptotic notation
  - Recurrences
  - Divide and Conquer

Karatsuba Multiplication

1. Break into smaller subproblems

$$
\begin{array}{r}
\boxed{a}\,\boxed{b} \\
\times\ \boxed{c}\,\boxed{d} \\
\hline
\end{array}
\quad
\begin{array}{l}
= 10^{\frac{n}{2}}\,\boxed{a} + \boxed{b} \\
= 10^{\frac{n}{2}}\,\boxed{c} + \boxed{d}
\end{array}
$$

$$
10^{n}\left(\boxed{a}\times\boxed{c}\right) + \\
10^{\frac{n}{2}}\left(\boxed{a}\times\boxed{d} + \boxed{b}\times\boxed{c}\right) + \\
\left(\boxed{b}\times\boxed{d}\right)
$$

Recall: previous divide-and-conquer recursively computed ac, ad, bc, bd

Karatsuba lets us reuse ac, bd to compute $(ad + bc)$ in one multiply

Karatsuba Multiplication

2. Use recurrence relation to express recursive running time

$$\begin{array}{rcc} & a & b \\ \times & c & d \\ \hline \end{array}$$

$$10^n(ac) + 10^{n/2}\big((a+b)(c+d) - ac - bd\big) + bd$$

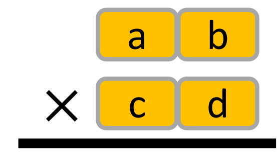
Recursively solve

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

Need to compute **3** multiplications, each of size $n/2$: ac, bd, $(a+b)(c+d)$

Karatsuba Multiplication

2. Use recurrence relation to express recursive running time



$$10^n(ac) + 10^{n/2}\big((a+b)(c+d) - ac - bd\big) + bd$$

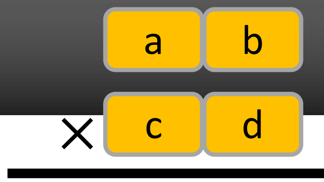
Recursively solve

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

Need to compute **3** multiplications, each of size $n/2$: ac, bd, $(a+b)(c+d)$

2 shifts and 6 additions on n-bit values

Karatsuba Algorithm



1. Recursively compute: $ac, bd, (a+b)(c+d)$
2. $(ad+bc) = (a+b)(c+d) - ac - bd$
3. Return $10^n(ac) + 10^{\frac{n}{2}}(ad+bc) + bd$

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

<u>Pseudo-code</u>

1. $x \leftarrow$ Karatsuba(a, c)
2. $y \leftarrow$ Karatsuba(b, d)
3. $z \leftarrow$ Karatsuba$(a+b, c+d) - x - y$
4. Return $10^n x + 10^{n/2} z + y$

Karatsuba Example

$$n = 4$$

$$\begin{array}{r} 4\,1\,0\,2 \\ \times\,1\,8\,1\,9 \\ \hline \end{array}$$

$$\left.\begin{array}{r} a = 41 \\ b = 02 \\ c = 18 \\ d = 19 \end{array}\right\}$$ Constant time divide $\Theta(1)$

$$\left.\begin{array}{r} a + b = 43 \\ c + d = 37 \end{array}\right\}$$ 2 preliminary additions $\Theta(2n)$

$$\left.\begin{array}{r} ac = 41 \times 18 = 738 \\ bd = 02 \times 19 = 38 \\ (a+b)(c+d) = 43 \times 37 = 1591 \end{array}\right\}$$ 3 recursive Karatsuba calls each size $^n/_2 = 2$ $3\mathrm{T}(n/2)$

Karatsuba Example

$$n = 4$$

$$\left.\begin{aligned} ac &= 41\times 18 = 738 \\ bd &= 02\times 19 = 38 \\ (a+b)(c+d) &= 43\times 37 = 1591 \end{aligned}\right\}$$

3 recursive Karatsuba calls

each size $^{n}/_{2} = 2$

$3\mathrm{T}(n/2)$

$$10^n(ac) + 10^{\frac{n}{2}}\big((a+b)(c+d) - ac - bd\big) + bd$$

$$10^4(ac) + 10^{\frac{4}{2}}\big((a+b)(c+d) - ac - bd\big) + bd$$

$$10000(ac) + 100\big((a+b)(c+d) - ac - bd\big) + bd$$

$$10000(738) + 100(1591 - 738 - 38) + 38$$

$$10000(738) + 100(815) + 38$$

Combine step

Karatsuba Example

$$10000(738) + 100(815) + 38$$

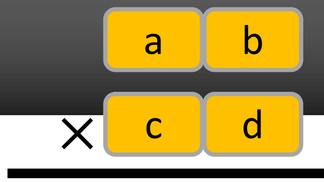
$$7380000 + 81500 + 38$$

$$7461538$$

Combine step $\Theta(6n)$

$n = 4$

$$\begin{array}{r} 4\,1\,0\,2 \\ \times\,1\,8\,1\,9 \\ \hline 7\,4\,6\,1\,5\,3\,8 \end{array}$$



Karatsuba Algorithm

1. Recursively compute: $ac, bd, (a+b)(c+d)$
2. $(ad+bc) = (a+b)(c+d) - ac - bd$
3. Return $10^n(ac) + 10^{\frac{n}{2}}(ad+bc) + bd$

<u>Pseudo-code</u>

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

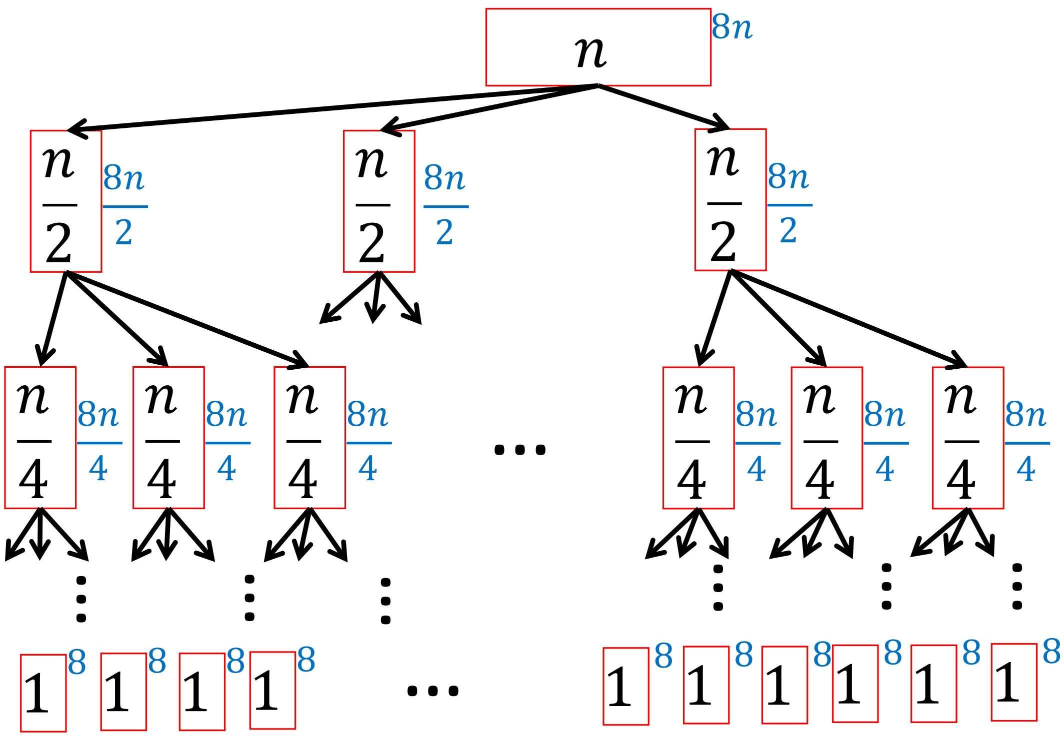
1. $x \leftarrow$ Karatsuba(a, c)
2. $y \leftarrow$ Karatsuba(b, d)
3. $z \leftarrow$ Karatsuba$(a+b, c+d) - x - y$
4. Return $10^n x + 10^{n/2} z + y$

Karatsuba

3. Use asymptotic notation to simplify

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

$$T(n) = 8n \sum_{i=0}^{\log_2 n} \left({}^3/_2\right)^i$$



$$8n \cdot 1$$

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

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

$$\vdots$$

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

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 supplement)

Karatsuba

Karatsuba

Karatsuba

3. Use asymptotic notation to simplify

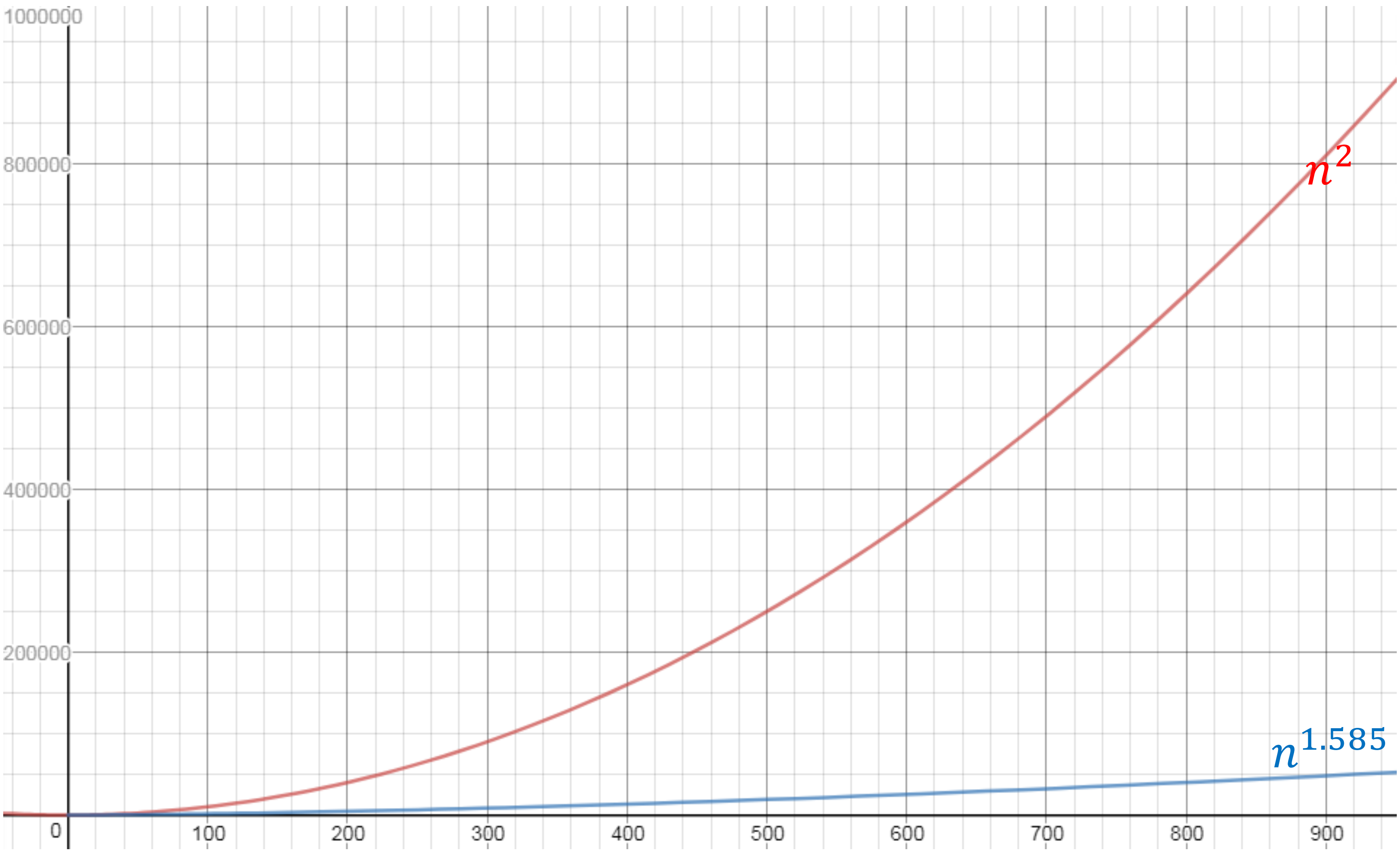
$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

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

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

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

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



1000000
800000
600000
400000
200000
0
100
200
300
400
500
600
700
800
900
n^2
$n^{1.585}$

Recurrence Solving Techniques


Tree

Guess/Check

(induction)

“Cookbook”

Substitution

Induction (review)

Goal: $\forall k \in \mathbb{N}, P(k)$ holds

Base case(s): $P(1)$ holds

Technically, called *strong induction*

Hypothesis: $\forall x \leq x_0, P(x)$ holds

Inductive step: show $P(1), \dots, P(x_0) \Rightarrow P(x_0 + 1)$

Guess and Check Intuition

- **Show:** $T(n) \in O(g(n))$
- **Consider:** $g_*(n) = c \cdot g(n)$ for some constant c, i.e. pick $g_*(n) \in O(g(n))$
- **Goal:** show $\exists n_0$ such that $\forall n > n_0, T(n) \leq g_*(n)$
  - (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 (may need additional base cases)
  - Hypothesis:
    - $\forall n \leq x_0, T(n) \leq g_*(n)$
  - Inductive step:
    - Show $T(x_0 + 1) \leq g_*(x_0 + 1)$

Need to ensure that in inductive step, can either appeal to a base case or to the inductive hypothesis

Karatsuba Guess and Check (Loose)

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

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

Base cases: $T(1) = 8 \leq 3000$
$T(2) = 3(8) + 16 = 40 \leq 3000 \cdot 2^{1.6}$
… up to some small k

Hypothesis: $\forall n \leq x_0, T(n) \leq 3000n^{1.6}$

Inductive step: Show that $T(x_0 + 1) \leq 3000(x_0 + 1)^{1.6}$

Karatsuba Guess and Check (Loose)

Karatsuba Guess and Check (Loose)

Mergesort Guess and Check

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

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

Base cases: $T(1) = 0$

$T(2) = 2 \leq 2 \log_2 2$

… up to some small k

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

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

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

Mergesort Guess and Check

Mergesort Guess and Check

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

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

Karatsuba Guess and Check

Karatsuba Guess and Check

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

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

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

Base cases: $T(1) = 0$

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

… up to some small k

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

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

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:**

$$T(n) = D(n) + \sum T(s) + C(n)$$

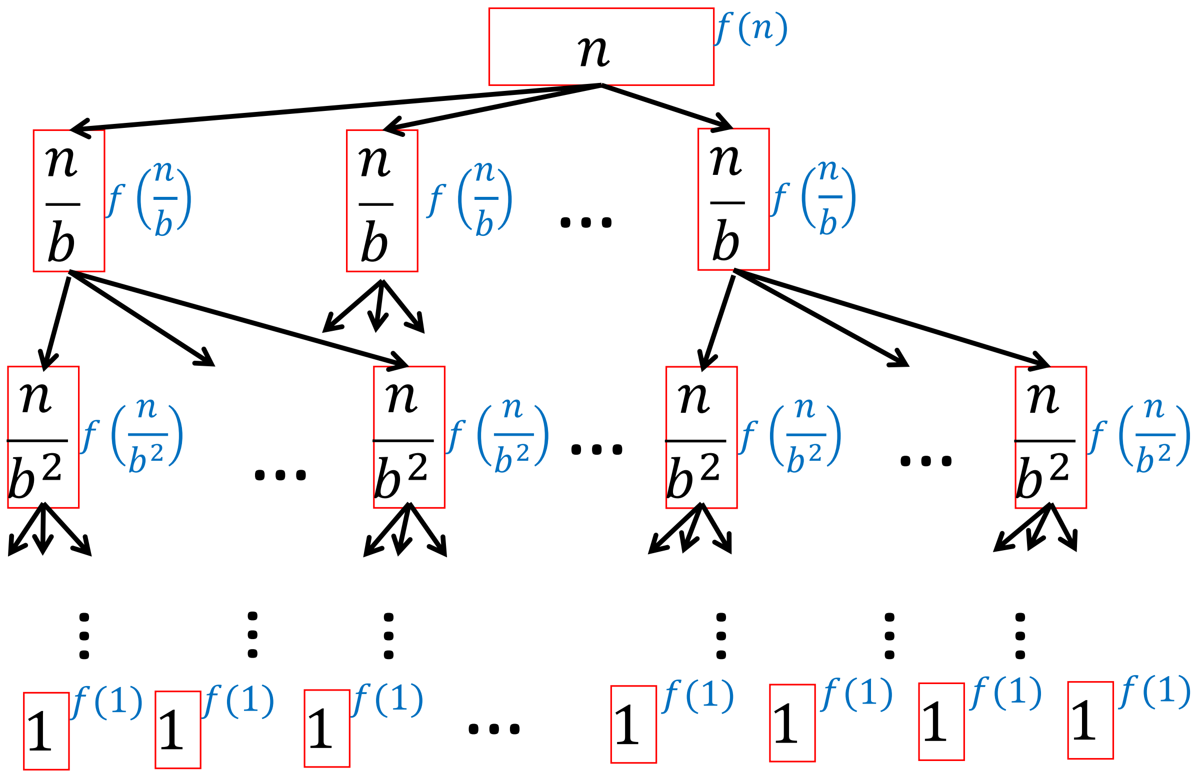
- Many D&C recurrences are of form:

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

General

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

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



$f(n)$

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

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

$\vdots$

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

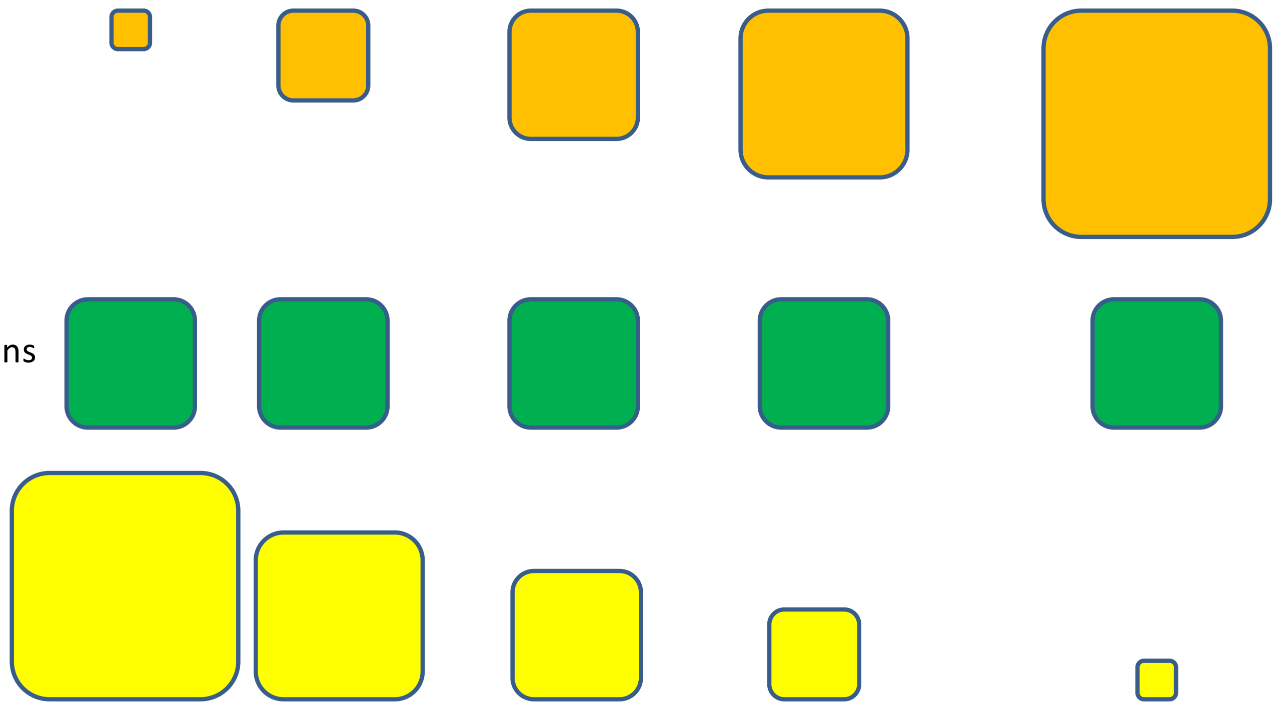
3 Cases

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

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

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

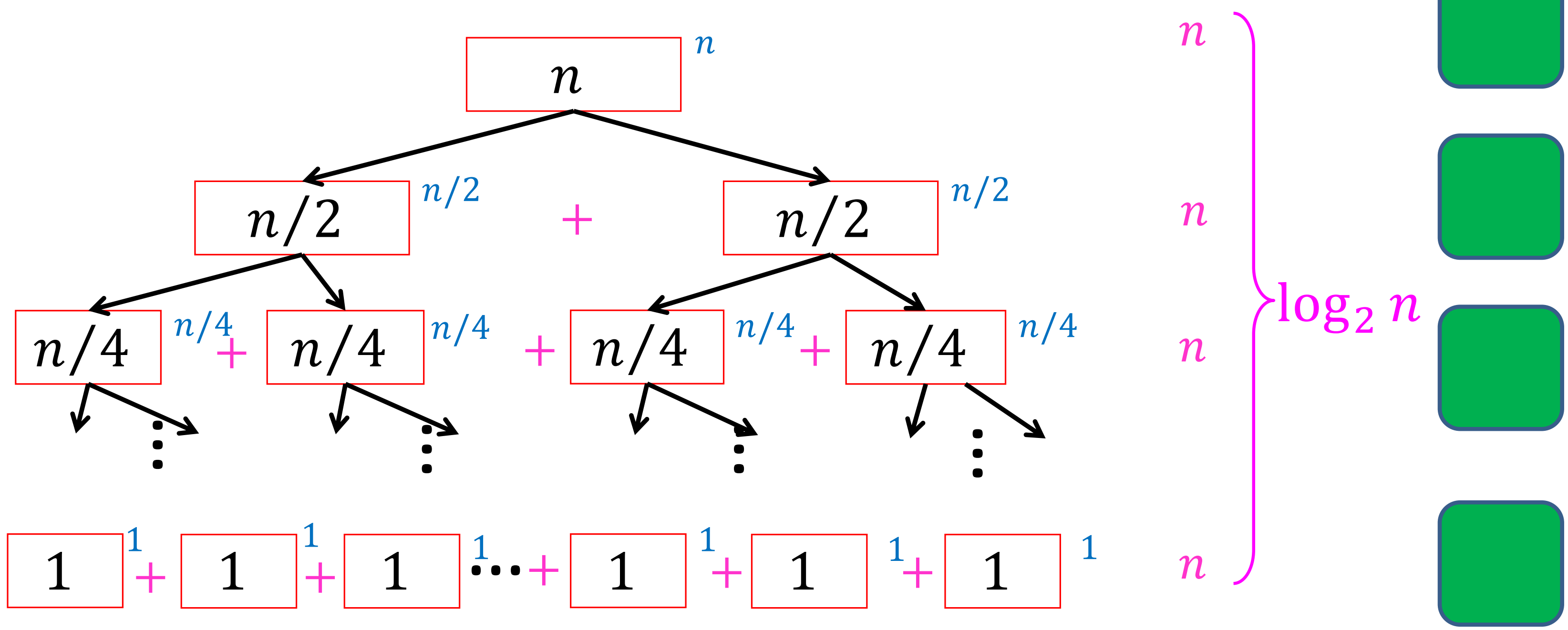
$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

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

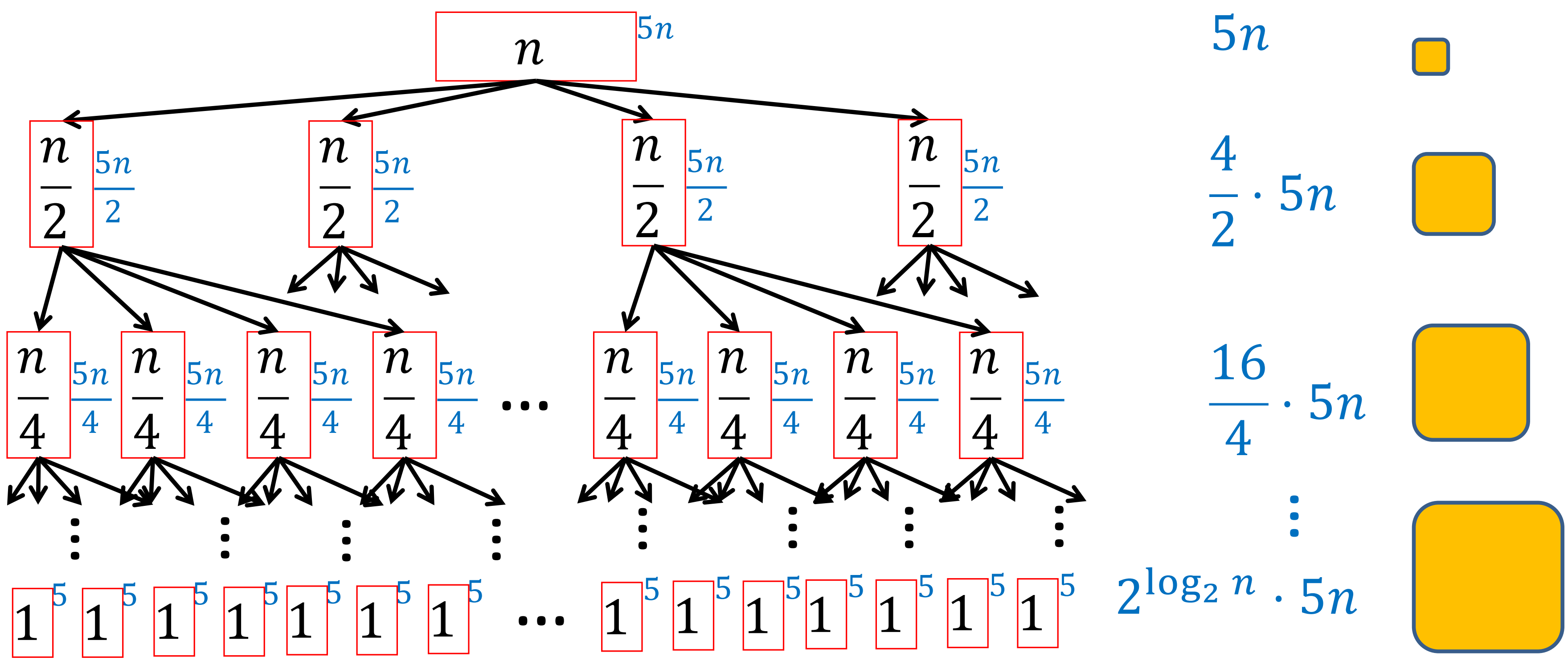
$$T(n) = 4T\left(\frac{n}{2}\right) + 5n$$

Case 1

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

Tree method

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



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

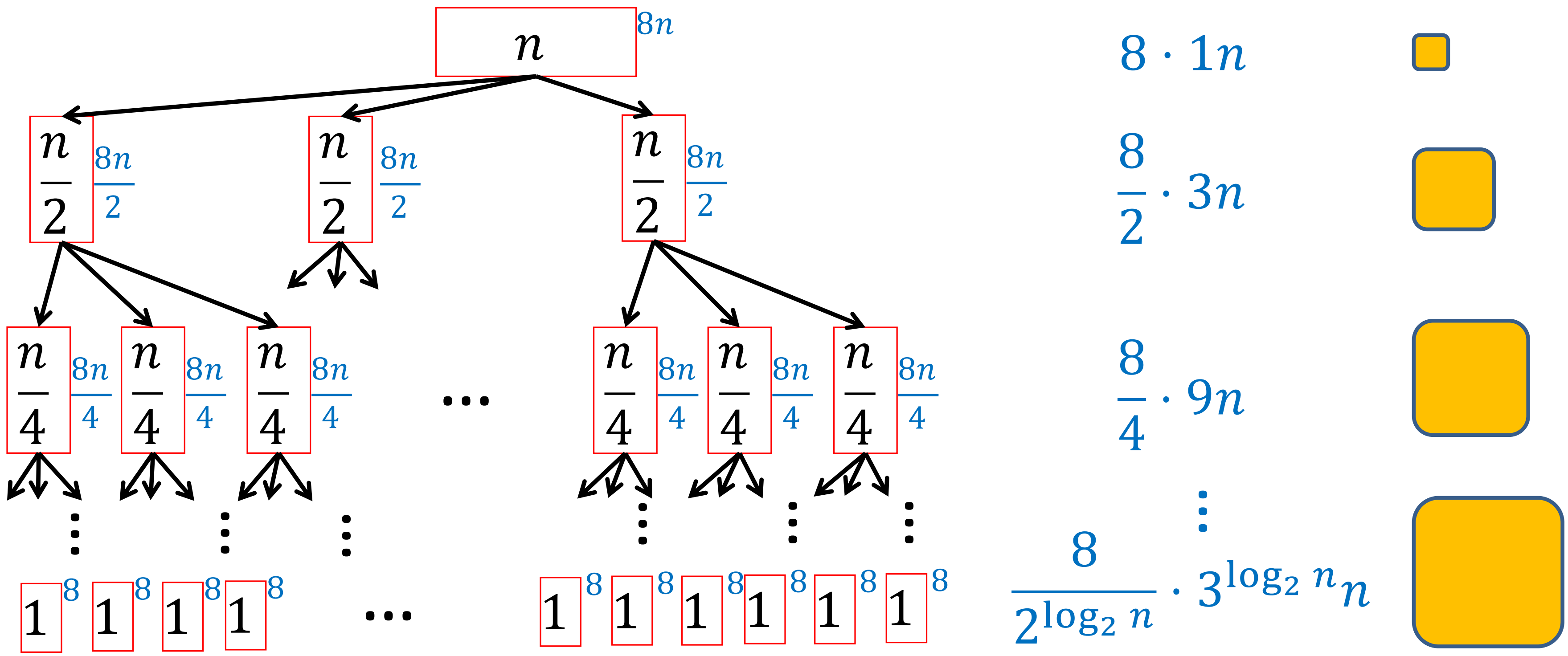
$$T(n) = 3T\left(\frac{n}{2}\right) + 8n$$

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

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