CS 3100
Data Structures and Algorithms 2
Lecture 8: Divide and Conquer

Co-instructors: Robbie Hott and Tom Horton

Fall 2023

Readings in CLRS 4th edition:

- Section 4.1-4.4

Announcements

- Upcoming dates
  - PA1 due Sept 17 (Sunday) at 11:59pm
  - PS2 available
  - PA2 available
- Course email (comes to both professors and head TAs):

cs3100@cshelpdesk.atlassian.net

Divide and Conquer

[CLRS Chapter 4]

Divide:

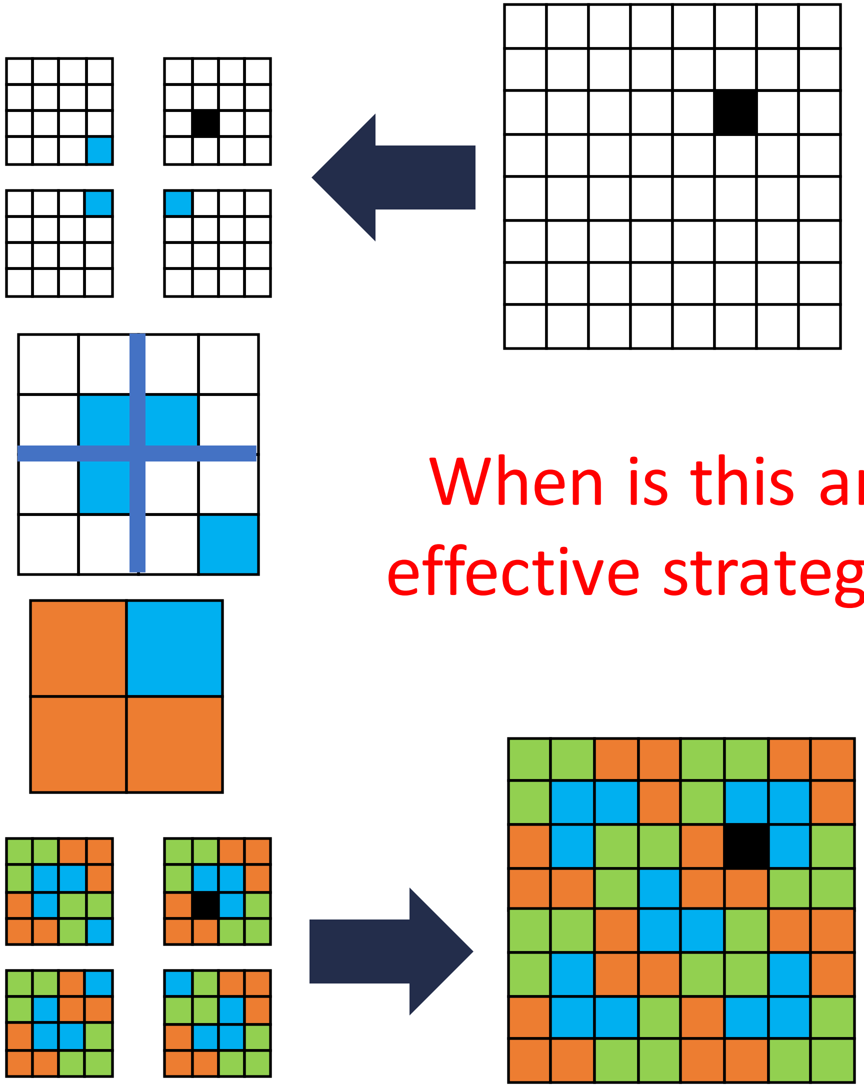
- Break the problem into multiple subproblems, each smaller instances of the original

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?

Merge Sort

Divide:

- Break n-element list into two lists of $^n/_2$ elements

Conquer:

- If $n > 1$:
  - Sort each sublist recursively
- If $n = 1$:
  - List is already sorted (base case)

Combine:

- Merge together sorted sublists into one sorted list

Merge

Combine: Merge sorted sublists into one sorted list

Inputs:

- 2 sorted lists (L_1, L_2)
- 1 output list (L_{out})

While (L_1 and L_2 not empty):

  If $L_1[0] \leq L_2[0]$:

    L_{out}.append(L_1.pop())

  Else:

    L_{out}.append(L_2.pop())

L_{out}.append(L_1)

L_{out}.append(L_2)

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:

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

Analyzing Merge Sort

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

Divide: 0 comparisons

Conquer: recurse on 2 small problems, size $\frac{n}{2}$

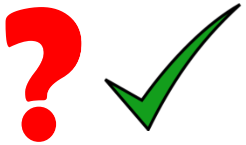
Combine: n comparisons

Recurrence:

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

Recurrence Solving Techniques

Tree



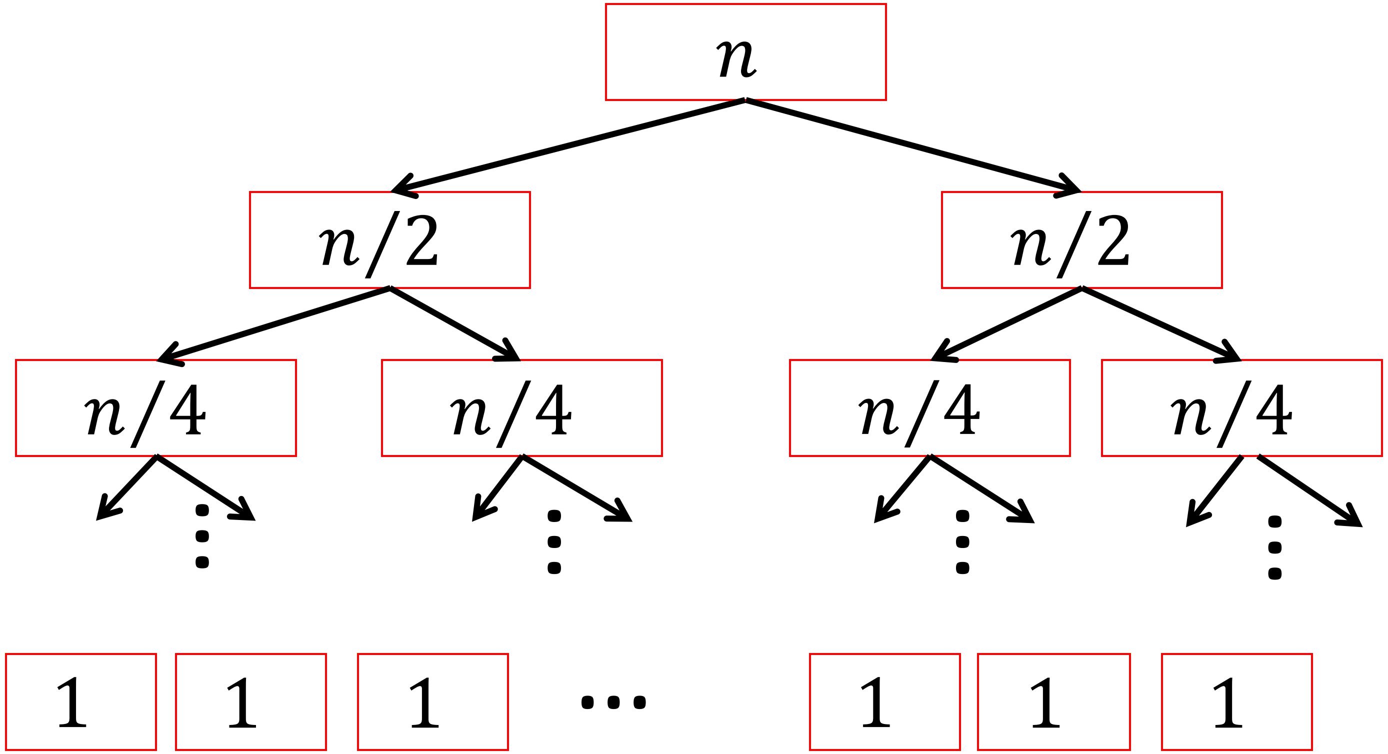
Guess/Check

“Cookbook”

Substitution

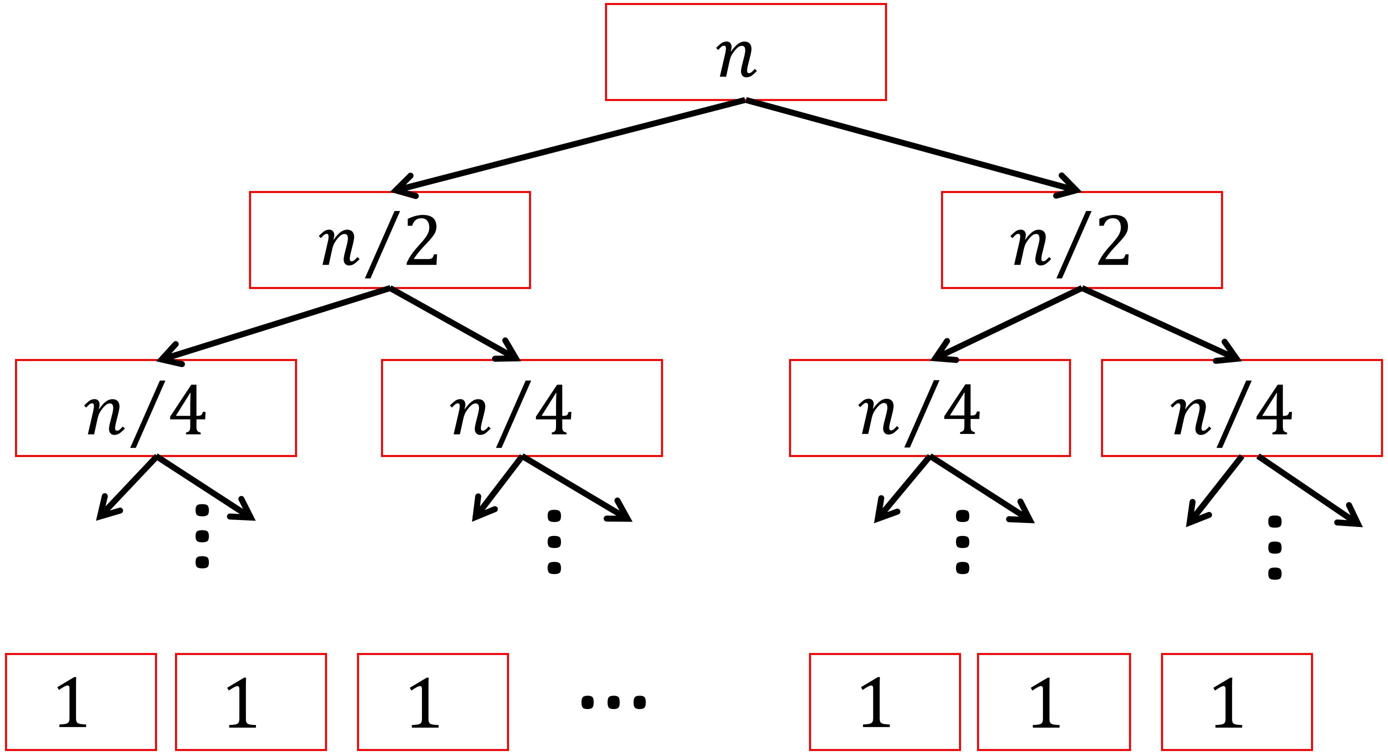
Tree Method

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



Tree Method

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

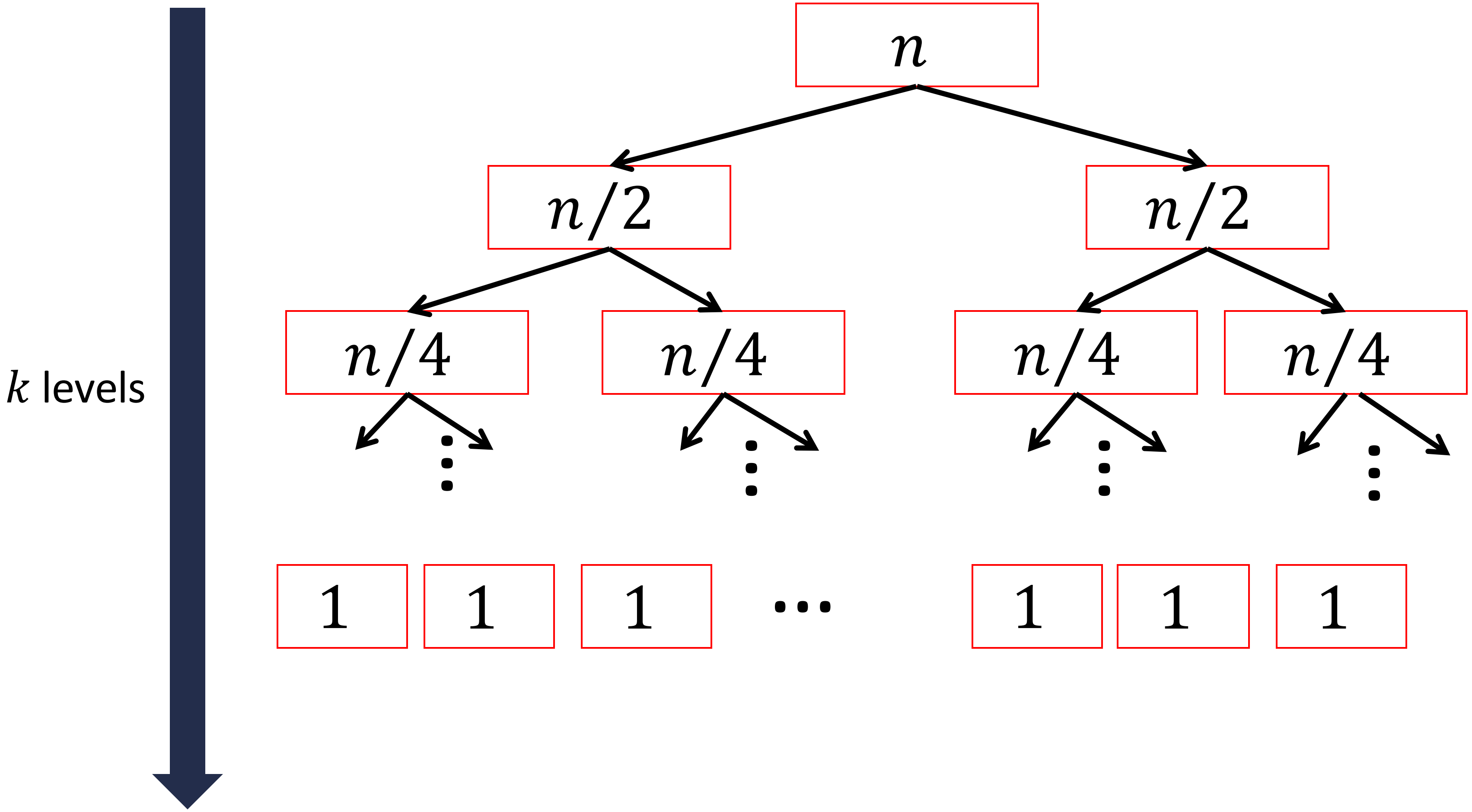


Number of subproblems

1

Tree Method

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



Number of subproblems	Cost of subproblem
1	n
2	$n/2$
4	$n/4$
2^k	$\frac{n}{2^k} = 1$

Tree Method

3. Use asymptotic notation to simplify

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

How many levels?

Problem size at k^{th} level: $\frac{n}{2^k}$

Base case: $n = 1$

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

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

Number of subproblems	Cost of subproblem
1	n
2	$n/2$
4	$n/4$
2^k	$\frac{n}{2^k} = 1$

Tree Method

3. Use asymptotic notation to simplify

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

$$k = \log_2 n$$

What is the cost?

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

Total cost: $T(n) = \sum_{i=0}^{\log_2 n} n = n \sum_{i=0}^{\log_2 n} 1 = n \log_2 n$

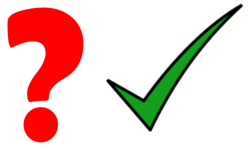
$= \Theta(n \log n)$

Number of subproblems	Cost of subproblem
1	n
2	$n/2$
4	$n/4$
2^k	$\frac{n}{2^k} = 1$

Recurrence Solving Techniques

Tree

get a picture of recursion



Guess/Check

guess and use induction to prove

"Cookbook"

MAGIC!

Substitution

substitute in to simplify

Multiplication

Want to multiply large numbers together

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

n-digit numbers

How do we measure input size?

number of digits

What do we "count" for run time?

number of <u>elementary</u> operations
(single-digit multiplications)

"Schoolbook" Multiplication

How many multiplications?

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

n-digit numbers

$$\left.\begin{array}{l} n \text{ mults} \\ n \text{ mults} \\ n \text{ mults} \\ n \text{ mults} \end{array}\right\} n \text{ levels}$$

$\Rightarrow \Theta(n^2)$

What about cost of additions?

$\Theta(n^2)$

"Schoolbook" Multiplication

Can we do better?

How many multiplications?

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

n-digit numbers

$$\left.\begin{array}{l} n \text{ mults} \\ n \text{ mults} \\ n \text{ mults} \\ n \text{ mults} \end{array}\right\} n \text{ levels} \Rightarrow \Theta(n^2)$$

What about cost of additions?

$\Theta(n^2)$

Divide and Conquer Multiplication

1. Break into smaller subproblems

$$\boxed{a}\boxed{b} = 10^{\frac{n}{2}}\boxed{a} + \boxed{b}$$

$$\times\ \boxed{c}\boxed{d} = 10^{\frac{n}{2}}\boxed{c} + \boxed{d}$$

$$\begin{aligned} = \quad & 10^{n}(\boxed{a} \times \boxed{c}) + \\ & 10^{\frac{n}{2}}(\boxed{a} \times \boxed{d} + \boxed{b} \times \boxed{c}) + \\ & (\boxed{b} \times \boxed{d}) \end{aligned}$$

Divide and Conquer Multiplication

Divide:

- Break n-digit numbers into four numbers of $n/2$ digits each (call them a, b, c, d)

Conquer:

- If $n > 1$:
  - Recursively compute ac, ad, bc, bd
- If $n = 1$: (i.e. one digit each)
  - Compute ac, ad, bc, bd directly (base case)

Combine:

- $10^n(ac) + 10^{n/2}(ad + bc) + bd$

For simplicity, assume that $n = 2^k$ is a power of 2

Divide and Conquer Multiplication

2. Use recurrence relation to express recursive running time

$$10^n(ac) + 10^{n/2}(ad + bc) + bd$$

Recursively solve

$$T(n)$$

Divide and Conquer Multiplication

2. Use recurrence relation to express recursive running time

$$10^n(ac) + 10^{n/2}(ad + bc) + bd$$

Recursively solve

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

Need to compute 4 multiplications, each of size $n/2$

Divide and Conquer Multiplication

2. Use recurrence relation to express recursive running time

$$10^n(ac) + 10^{n/2}(ad + bc) + bd$$

Recursively solve

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

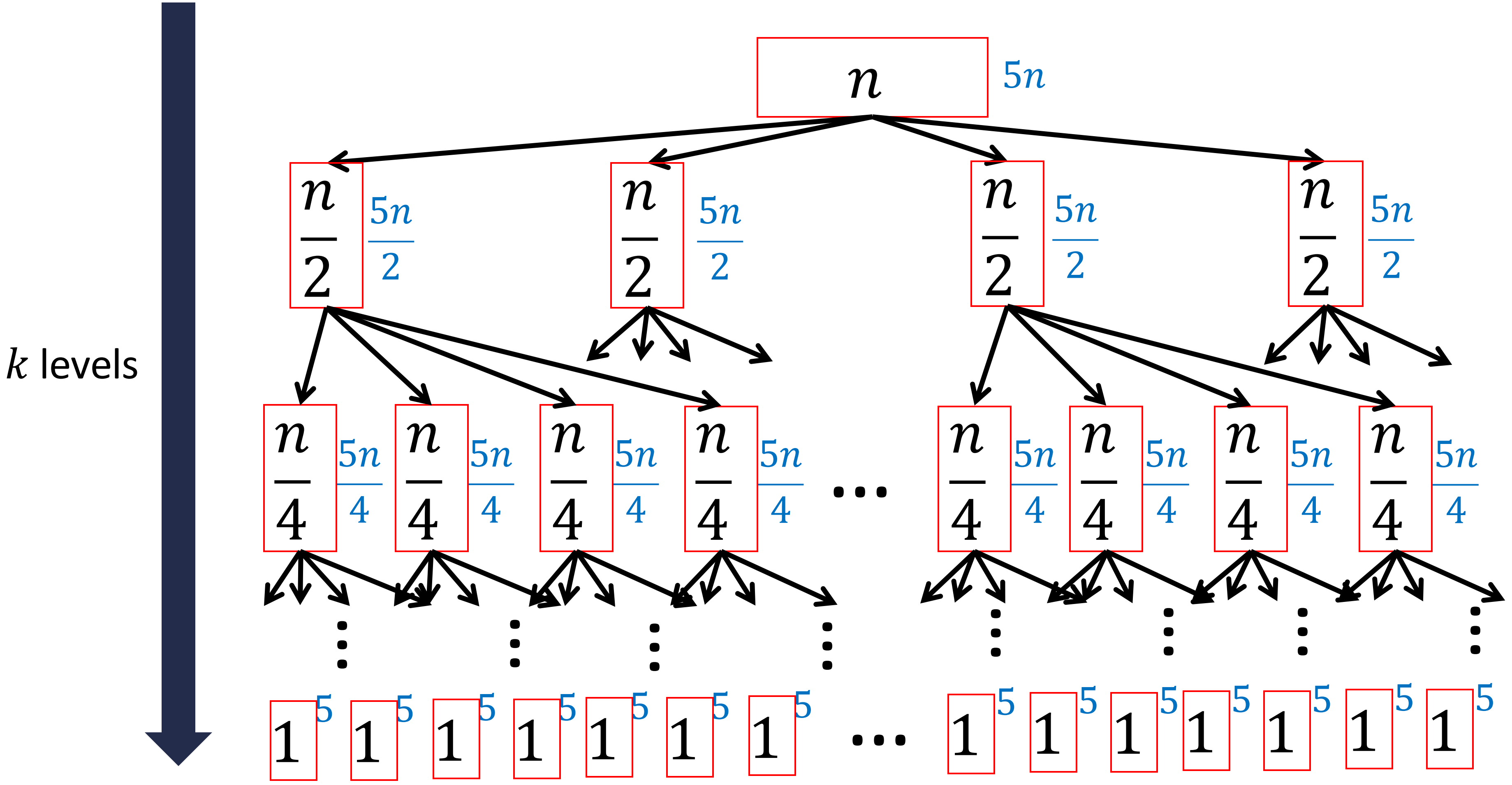
Need to compute 4 multiplications, each of size $n/2$

2 shifts and 3 additions on n-bit values

Divide and Conquer Multiplication

3. Use asymptotic notation to simplify

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



Number of subproblems	Cost of subproblem
1	$5n$
4	$\frac{5n}{2}$
16	$\frac{5n}{4}$
4^k	$\frac{5n}{2^k} = 5$

Divide and Conquer Multiplication

3. Use asymptotic notation to simplify

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

How many levels?

Problem size at k^{th} level: $\frac{n}{2^k}$

Base case: $n = 1$

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

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

Number of subproblems	Cost of subproblem
1	$5n$
4	$\frac{5n}{2}$
16	$\frac{5n}{4}$
4^k	$\frac{5n}{2^k} = 5$

Divide and Conquer Multiplication

3. Use asymptotic notation to simplify

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

$$k = \log_2 n$$

What is the cost?

Cost at level i: $4^i \cdot \frac{5n}{2^i} = 2^i \cdot 5n$

Total cost: $T(n) = \sum_{i=0}^{\log_2 n} 2^i \cdot 5n = 5n \sum_{i=0}^{\log_2 n} 2^i$

Number of subproblems	Cost of subproblem
1	$5n$
4	$\frac{5n}{2}$
16	$\frac{5n}{4}$
4^k	$\frac{5n}{2^k} = 5$

Divide and Conquer Multiplication

3. Use asymptotic notation to simplify

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

$$= 5n \sum_{i=0}^{\log_2 n} 2^i$$

$$= 5n \cdot \frac{2^{\log_2 n+1} - 1}{2 - 1}$$

$$= 5n(2n - 1) = \Theta(n^2)$$

$$\sum_{i=0}^{L} a^i = \frac{a^{L+1} - 1}{a - 1}$$

No better than the schoolbook method!

Divide and Conquer Multiplication

3. Use asymptotic notation to simplify

$$\sum_{i=0}^{L} a^i = \frac{a^{L+1} - 1}{a - 1}$$

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

$$= 5n \sum_{i=0}^{\log_2 n} 2^i$$

$$= 5n \cdot \frac{2^{\log_2 n + 1} - 1}{2 - 1}$$

$$= 5n(2n - 1) = \Theta(n^2)$$

Is there a $o(n^2)$ algorithm for multiplication?

Karatsuba Multiplication

1. Break into smaller subproblems

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

$$
\begin{aligned}
= \;& 10^{n}\left(\boxed{a} \times \boxed{c}\right) + \\
& 10^{\frac{n}{2}}\left(\boxed{a} \times \boxed{d} + \boxed{b} \times \boxed{c}\right) + \\
& \left(\boxed{b} \times \boxed{d}\right)
\end{aligned}
$$

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

Karatsuba Multiplication

$$10^{n}(\boxed{ac}) + 10^{\frac{n}{2}}(\boxed{ad + bc}) + \boxed{bd}$$

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

Can't avoid these

This can be simplified!

$$(a + b)(c + d) =$$

$$\boxed{ac} + \boxed{ad + bc} + \boxed{bd}$$

$$\boxed{ad + bc} = \boxed{(a + b)(c + d) - \boxed{ac} - \boxed{bd}}$$

Two multiplications

One multiplication

Karatsuba Multiplication

2. Use recurrence relation to express recursive running time

	a	b
×	c	d

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

Recursively solve

$$T(n) =$$

Karatsuba Multiplication

2. Use recurrence relation to express recursive running time

	a	b
×	c	d

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

Recursively solve

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

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

Karatsuba Multiplication

2. Use recurrence relation to express recursive running time

	a	b
$\times$	c	d

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

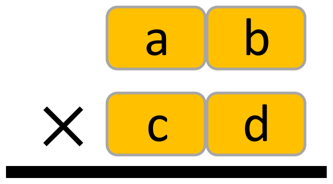
Recursively solve

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

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

2 shifts and 6 additions on n-bit values

Karatsuba Multiplication



Divide:

- Break n-digit numbers into four numbers of $^n/_2$ digits each (call them a, b, c, d)

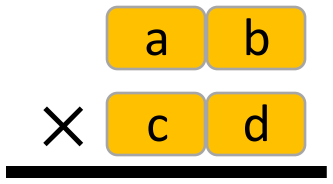
Conquer:

- If $n > 1$:
  - Recursively compute ac, bd, $(a+b)(c+d)$
- If $n = 1$:
  - Compute ac, bd, $(a+b)(c+d)$ directly (base case)

Combine:

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

Karatsuba Multiplication



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

Pseudocode:

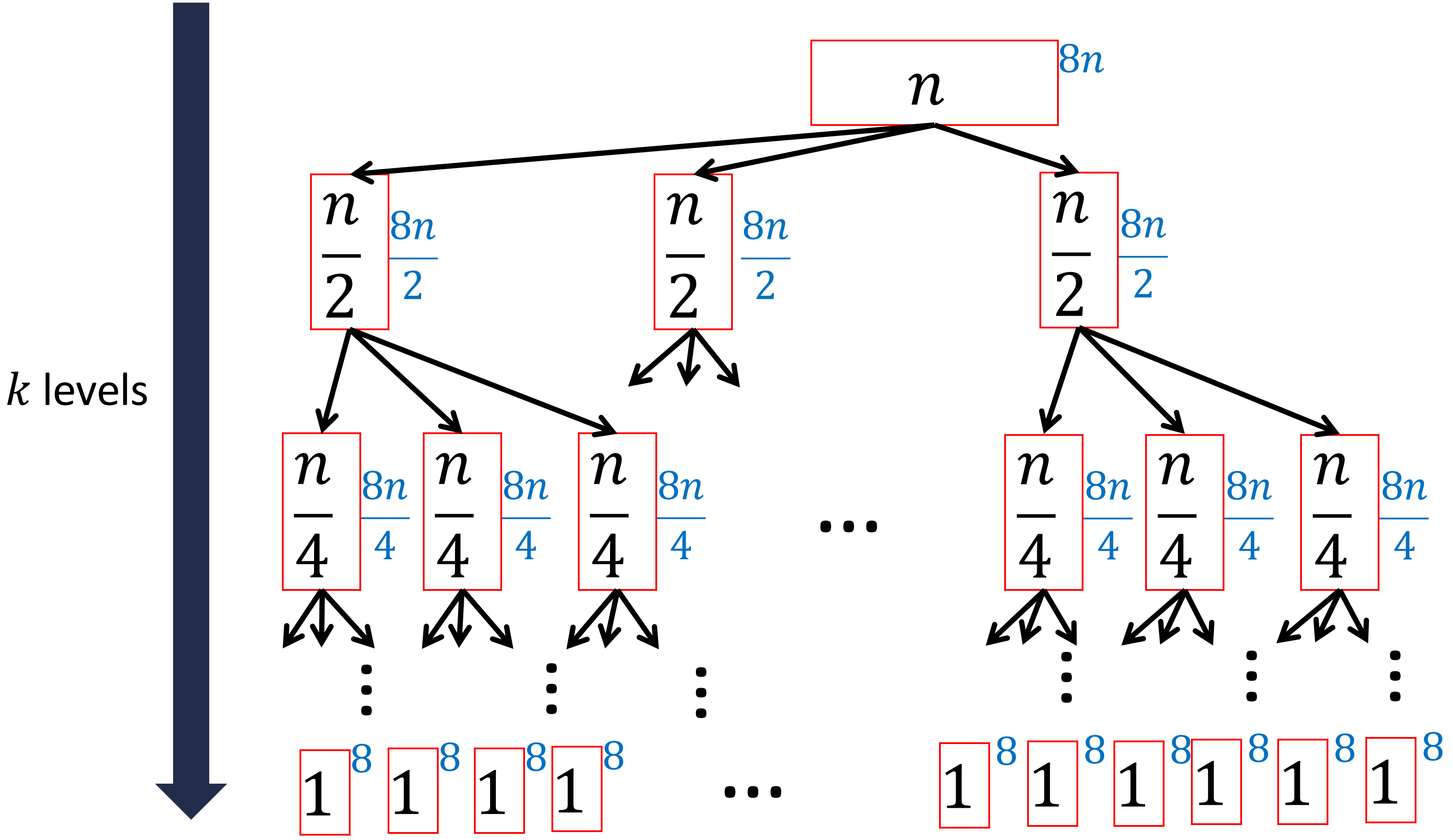
1. $x \leftarrow \text{Karatsuba}(a, c)$
2. $y \leftarrow \text{Karatsuba}(a, d)$
3. $z \leftarrow \text{Karatsuba}(a+b, c+d) - x - y$
4. Return $10^n x + 10^{n/2} z + y$

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

Karatsuba Multiplication

3. Use asymptotic notation to simplify

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



Number of subproblems	Cost of subproblem
1	$8n$
3	$\frac{8n}{2}$
9	$\frac{8n}{4}$
3^k	$\frac{8n}{2^k} = 8$

Karatsuba Multiplication

3. Use asymptotic notation to simplify

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

How many levels?

Problem size at k^{th} level: $\dfrac{n}{2^k}$

Base case: $n = 1$

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

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

Number of subproblems	Cost of subproblem
1	$8n$
3	$\dfrac{8n}{2}$
9	$\dfrac{8n}{4}$
3^k	$\dfrac{8n}{2^k} = 8$

Karatsuba Multiplication

3. Use asymptotic notation to simplify

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

$$k = \log_2 n$$

What is the cost?

Cost at level i: $3^i \cdot \frac{8n}{2^i} = \left(\frac{3}{2}\right)^i \cdot 8n$

Total cost: $T(n) = \sum_{i=0}^{\log_2 n} \left(\frac{3}{2}\right)^i \cdot 8n = 8n \sum_{i=0}^{\log_2 n} \left(\frac{3}{2}\right)^i$

Number of subproblems	Cost of subproblem
1	$8n$
3	$\frac{8n}{2}$
9	$\frac{8n}{4}$
3^k	$\frac{8n}{2^k} = 8$

Karatsuba Multiplication

3. Use asymptotic notation to simplify

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

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

$$= 8n \frac{({}^3/_2)^{\log_2 n+1} - 1}{{}^3/_2 - 1}$$

$$\sum_{i=0}^{L} a^i = \frac{a^{L+1} - 1}{a - 1}$$

Karatsuba Multiplication

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

How to simplify this
(using asymptotic notation)?

Drop constant multiples

Karatsuba Multiplication

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

How to simplify this (using asymptotic notation)?

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

Drop constant multiples

$$= \Theta\left(\frac{3}{2}n \cdot \left({}^{3}/_{2}\right)^{\log_2 n} - n\right)$$

Distribute terms

Karatsuba Multiplication

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

How to simplify this (using asymptotic notation)?

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

Drop constant multiples

$$= \Theta\left(\frac{3}{2}n \cdot \left({}^{3}/_{2}\right)^{\log_2 n} - n\right)$$

Distribute terms

$$= \Theta\left(n \cdot \left({}^{3}/_{2}\right)^{\log_2 n}\right)$$

Drop constants and low-order terms

Karatsuba Multiplication

$$T(n) = \Theta\left(n \cdot \left(3/2\right)^{\log_2 n}\right)$$

How to simplify this (using asymptotic notation)?

Properties of logarithms:

$$a^{bc} = \left(a^b\right)^c$$

$$2^{\log_2 n} = n$$

$$3^{\log_2 n} = 2^{\log_2\left(3^{\log_2 n}\right)} = 2^{(\log_2 n)(\log_2 3)} = \left(2^{\log_2 n}\right)^{\log_2 3} = n^{\log_2 3}$$

$$2^{\log_2 n} = n$$

$$\log a^b = b \log a$$

$$2^{\log_2 n} = n$$

Karatsuba Multiplication

$$T(n) = \Theta\left(n \cdot \left({}^{3}/_{2}\right)^{\log_2 n}\right)$$

$$= \Theta\left(n \cdot \left(\frac{3^{\log_2 n}}{2^{\log_2 n}}\right)\right)$$

$$= \Theta\left(n \cdot \left(\frac{n^{\log_2 3}}{n}\right)\right)$$

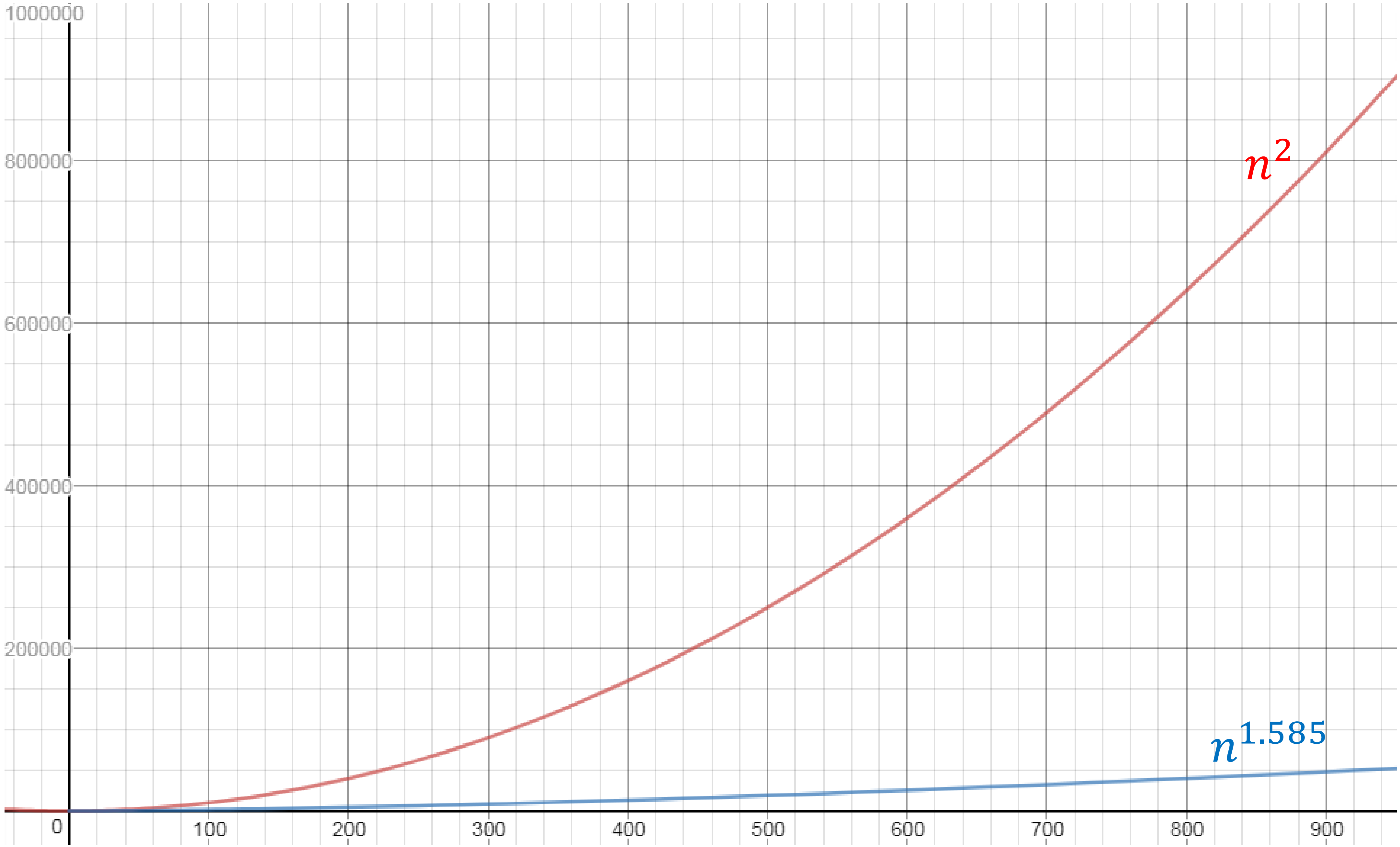
$$= \Theta\left(n^{\log_2 3}\right) \approx \Theta\left(n^{1.585}\right)$$

How to simplify this (using asymptotic notation)?

$$2^{\log_2 n} = n$$

$$3^{\log_2 n} = n^{\log_2 3}$$

Strictly better than schoolbook method!


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