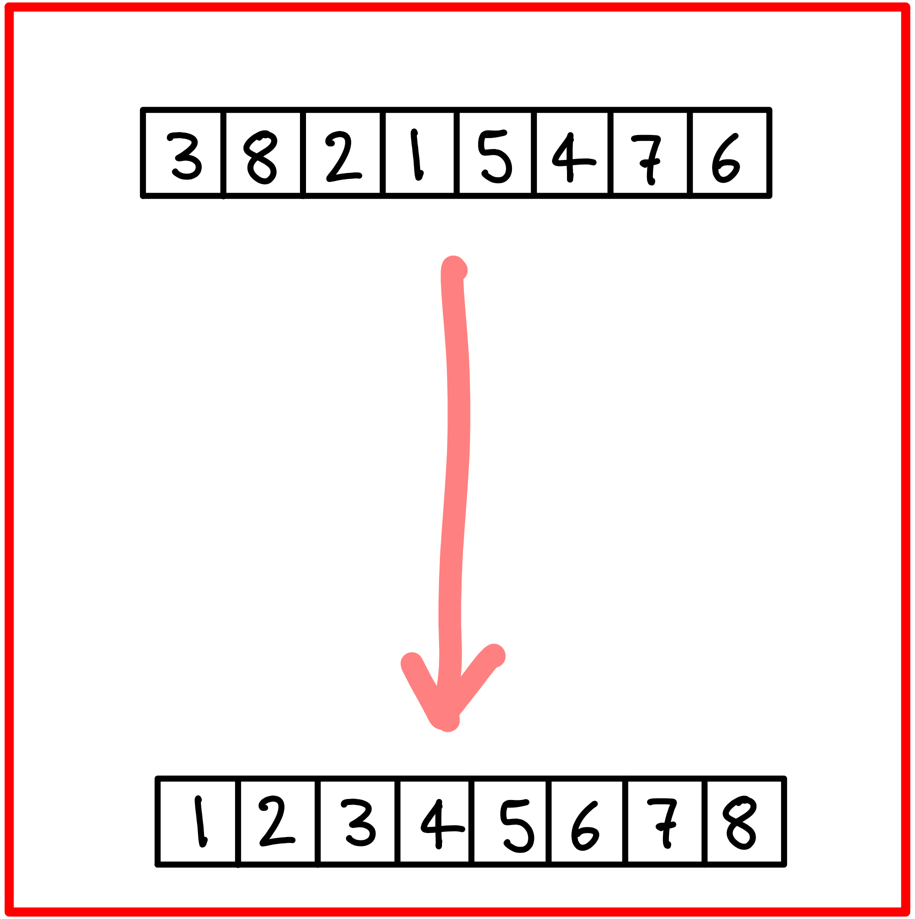
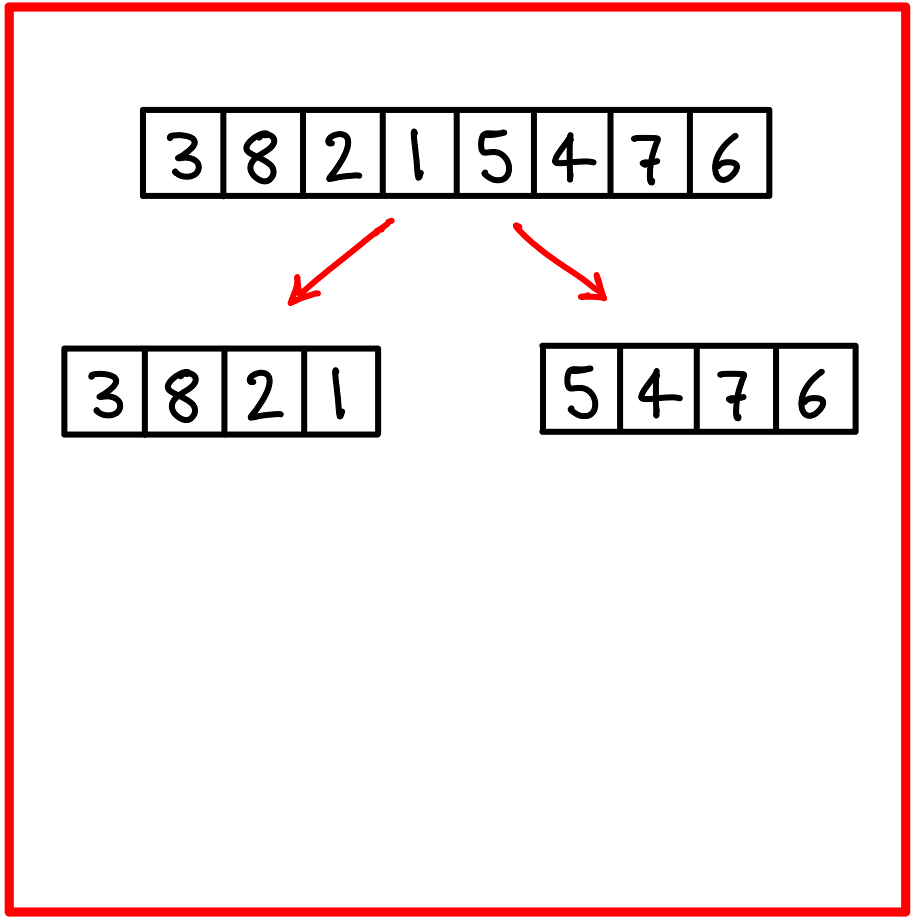
MERGE SORT, Divide & Conquer, dealing with recurrence relations

MERGE SORT, Divide & Conquer, dealing with recurrence relations



MERGE SORT, Divide & Conquer, dealing with recurrence relations

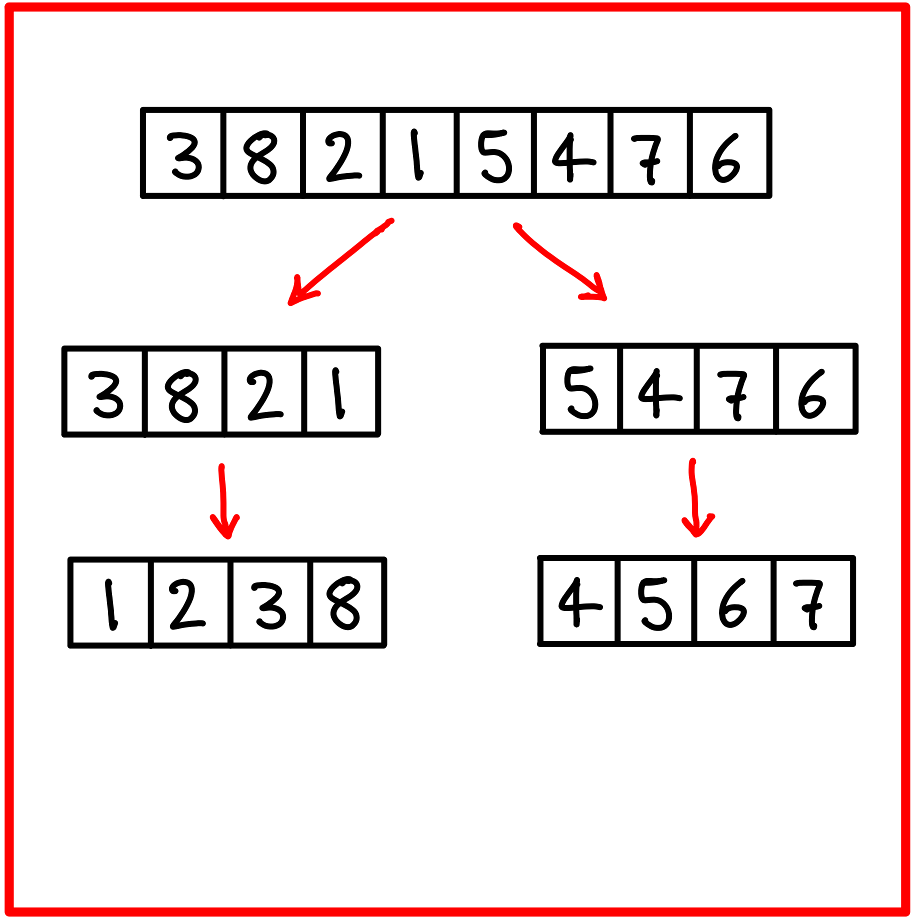
Divide problem into 2 smaller instances



MERGE SORT, Divide & Conquer, dealing with recurrence relations

Divide problem into 2 smaller instances

Conquer (= solve) the smaller problems

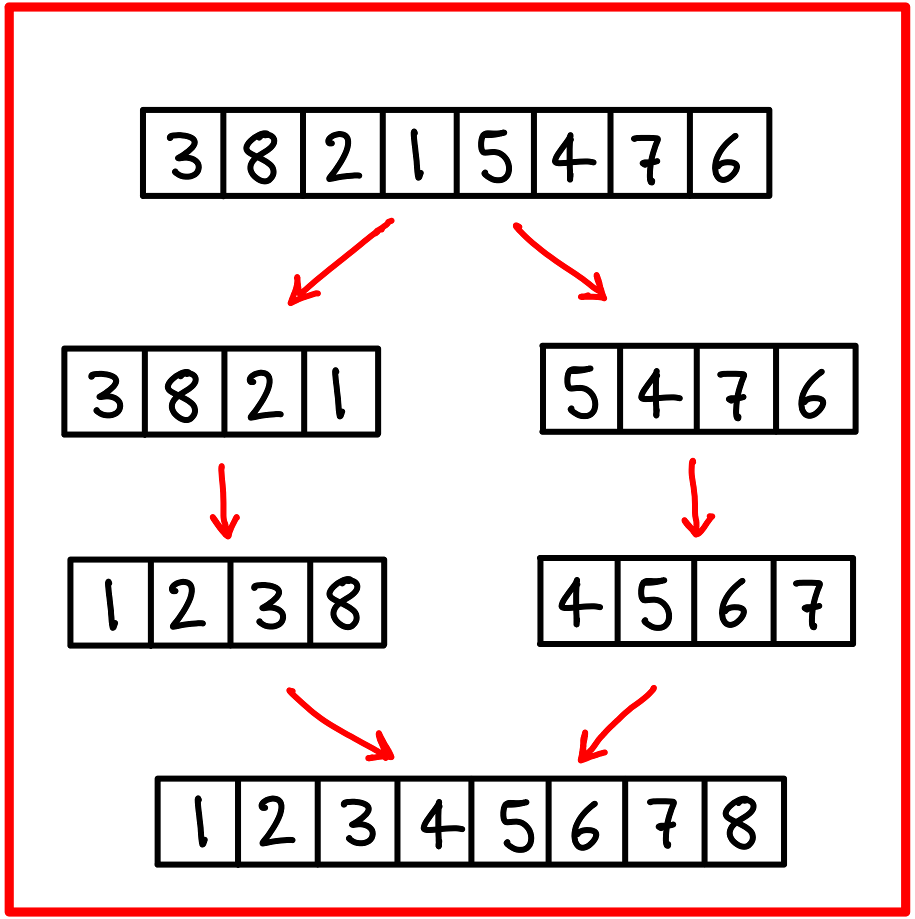


MERGE SORT, Divide & Conquer, dealing with recurrence relations

Divide problem into 2 smaller instances

Conquer (= solve) the smaller problems

Combine (= merge) the solutions

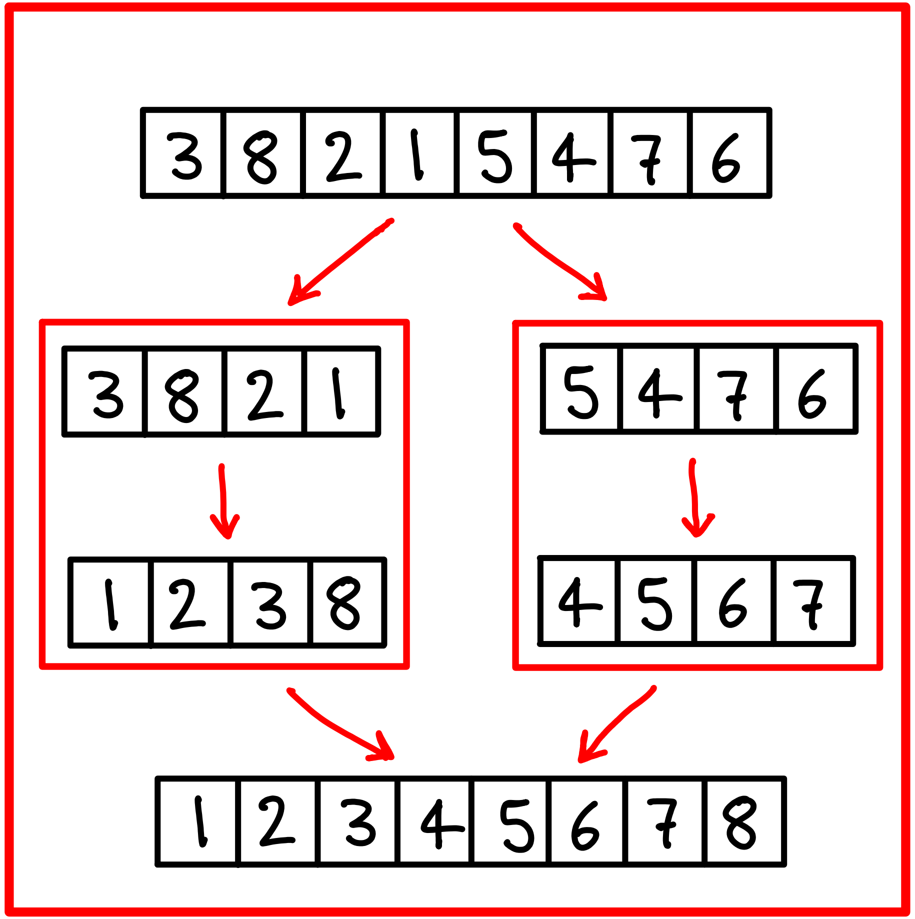


MERGE SORT, Divide & Conquer, dealing with recurrence relations

Divide problem into 2 smaller instances

Conquer (= solve) the smaller problems
(Mergesort)

Combine (= merge) the solutions



Merging 2 sorted arrays:

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

A:	1	2	3	5	9	14	16	20
B:	4	6	10	11	15	19	25	31

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

copy to output

A:	1	2	3	5	9	14	16	20
B:	4	6	10	11	15	19	25	31

output:	1															

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

copy to output

A:

1	2	3	5	9	14	16	20

Increment index of A

B:

4	6	10	11	15	19	25	31

output:

1															

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

Increment index of A

B:

4	6	10	11	15	19	25	31

output:

1															

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

Increment index of A

B:

4	6	10	11	15	19	25	31

output:

1	2														

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

Increment index of A

Increment A

B:

4	6	10	11	15	19	25	31

output:

1	2														

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Increment index of A

Increment A

output:

1	2														

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Increment index of A

Increment A

output:

1	2	3													

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Increment index of A

Increment A

Increment A

output:

1	2	3													

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Increment index of A

Increment A

Increment A

output:

1	2	3													

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

Increment index of A
Increment A
Increment A

B:

4	6	10	11	15	19	25	31

output:

1	2	3	4												

Merging 2 sorted arrays:

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

4	6	10	11	15	19	25	31

Increment index of A

Increment A

Increment A

Increment B

output:

1	2	3	4												

Merging 2 sorted arrays:

$\boxed{\Theta(n) \text{ time}}$

Smallest element is at leftmost position of A or B.

In general: copy to output, increment, smallest remaining element found with 1 comparison.

A:

1	2	3	5	9	14	16	20

B:

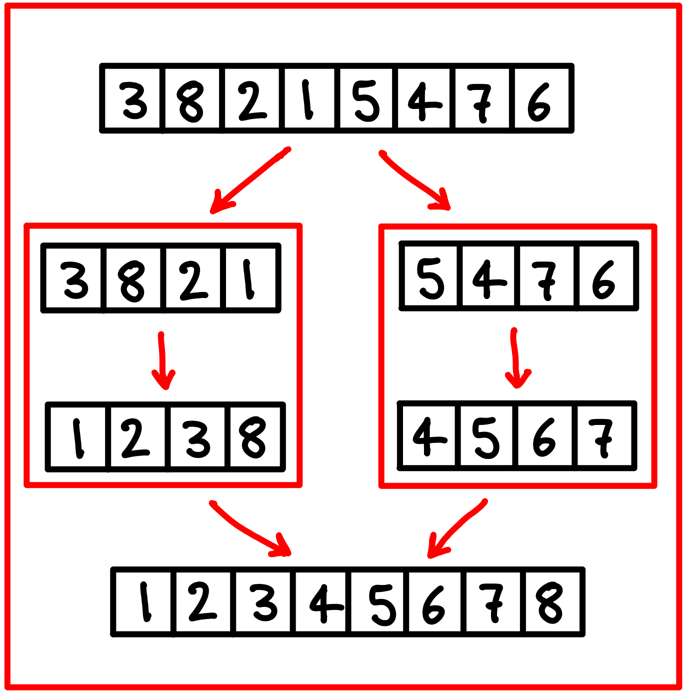
4	6	10	11	15	19	25	31

Increment index of A
Increment A
Increment A
Increment B
etc

output:

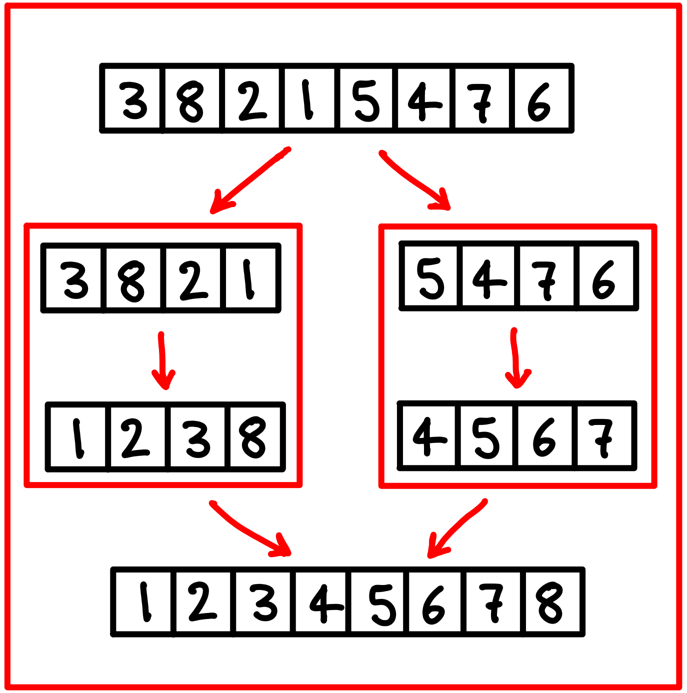
1	2	3	4												

Mergesort time for n elements: $T(n)$



Mergesort time for n elements: $T(n)$

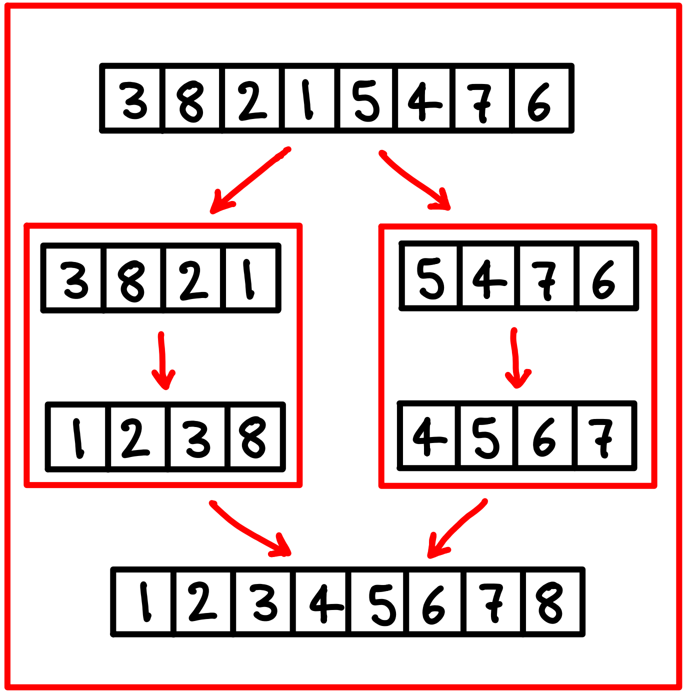
1) Divide $\Theta(1)$



Mergesort time for n elements: $T(n)$

1) Divide $\Theta(1)$

3) Merge $\Theta(n)$

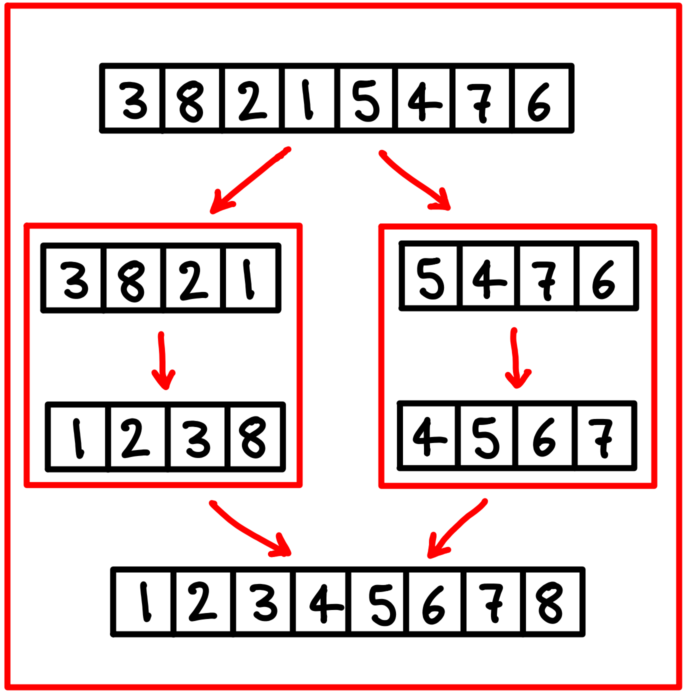


Mergesort time for n elements: $T(n)$

1) Divide $\Theta(1)$

2) Conquer $\Theta(1) + 2 \cdot T(\frac{n}{2})$

3) Merge $\Theta(n)$



Mergesort time for n elements: $T(n)$

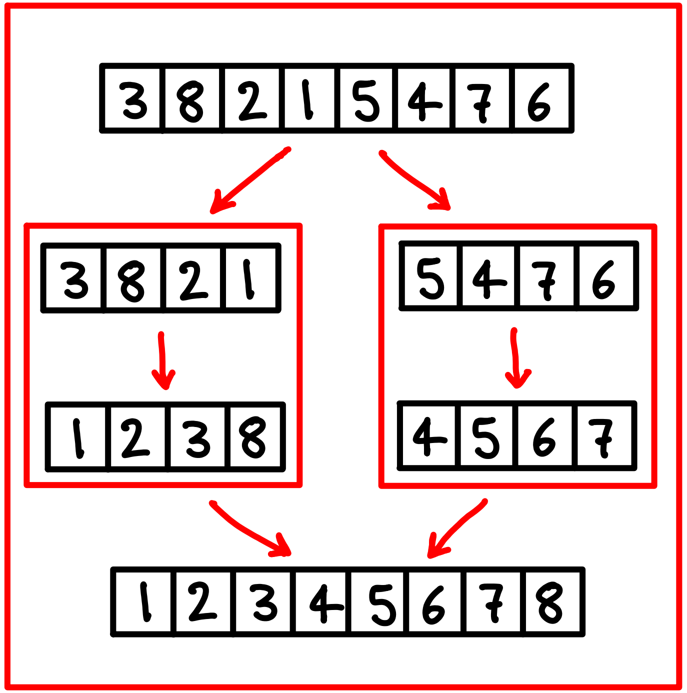
1) Divide $\Theta(1)$

2) Conquer $\Theta(1) + 2 \cdot T\left(\frac{n}{2}\right)$

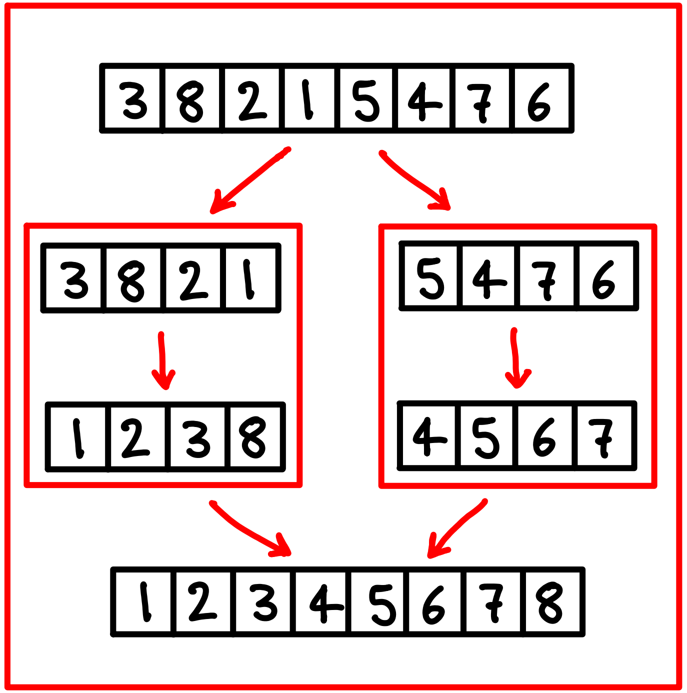
3) Merge $\Theta(n)$

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

$$T(1) = \Theta(1)$$



Mergesort time for n elements: $T(n)$



1) Divide $\Theta(1)$

2) Conquer $\Theta(1) + 2 \cdot T\left(\frac{n}{2}\right)$

3) Merge $\Theta(n)$

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

$$T(1) = \Theta(1)$$

Actually $T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n)$ → easy to deal with

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n)$ $\quad T(1) = \Theta(1)$

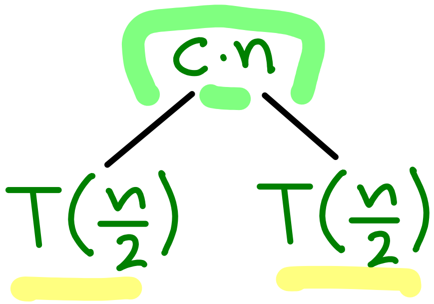
How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ $\qquad T(1) = \Theta(1)$

$T(n) = 2 \cdot T(\frac{n}{2}) + c \cdot n$ ← must do this → $= c_2$

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \Theta(n)$ $\qquad T(1) = \Theta(1)$

$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$ ← must do this → $= c_2$

Recursion tree:

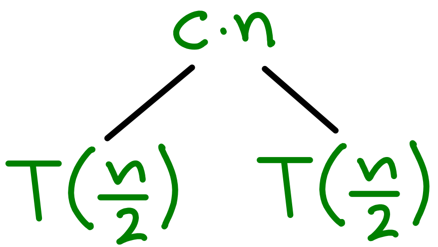


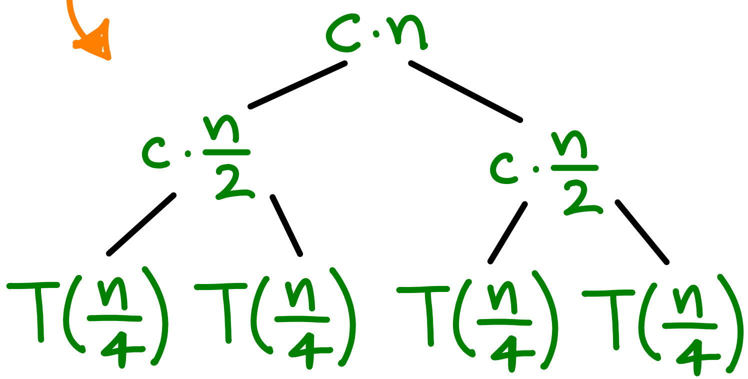
How to solve $T(n) = 2\cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$

$$T(n) = 2\cdot T\left(\frac{n}{2}\right) + c\cdot n$$ ← must do this →

$$T(1) = \Theta(1) = c_2$$

Recursion tree:





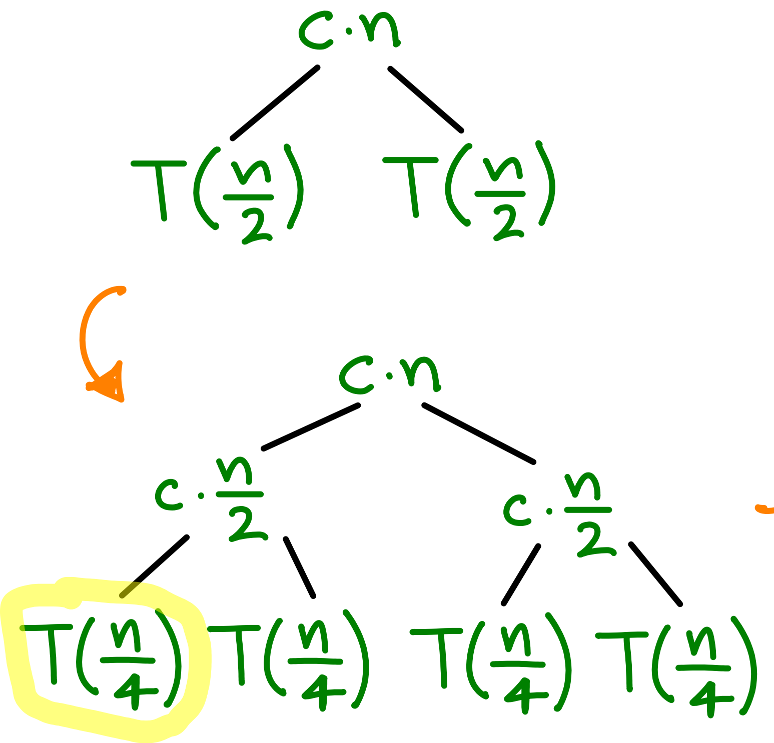
$$T\left(\frac{n}{2}\right) = 2\cdot T\left(\frac{n}{4}\right) + c\cdot\frac{n}{2}$$

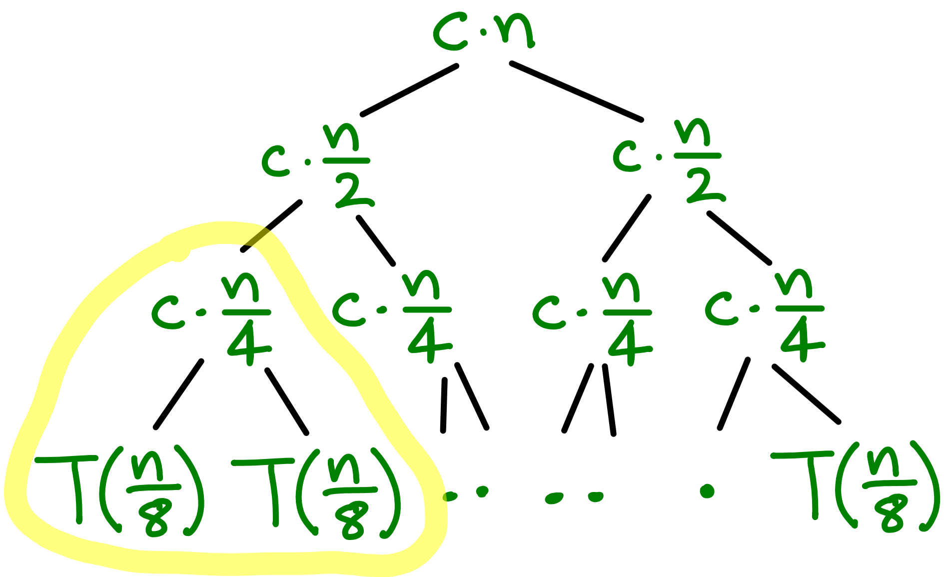
How to solve $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$

$T(1) = \Theta(1)$

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + c \cdot n$ ← must do this → $= c_2$

Recursion tree:



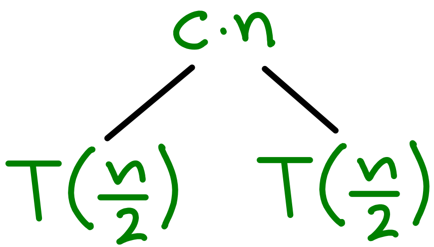


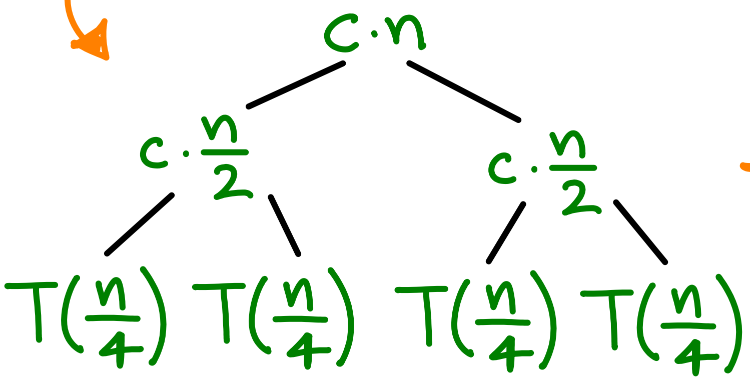
How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$

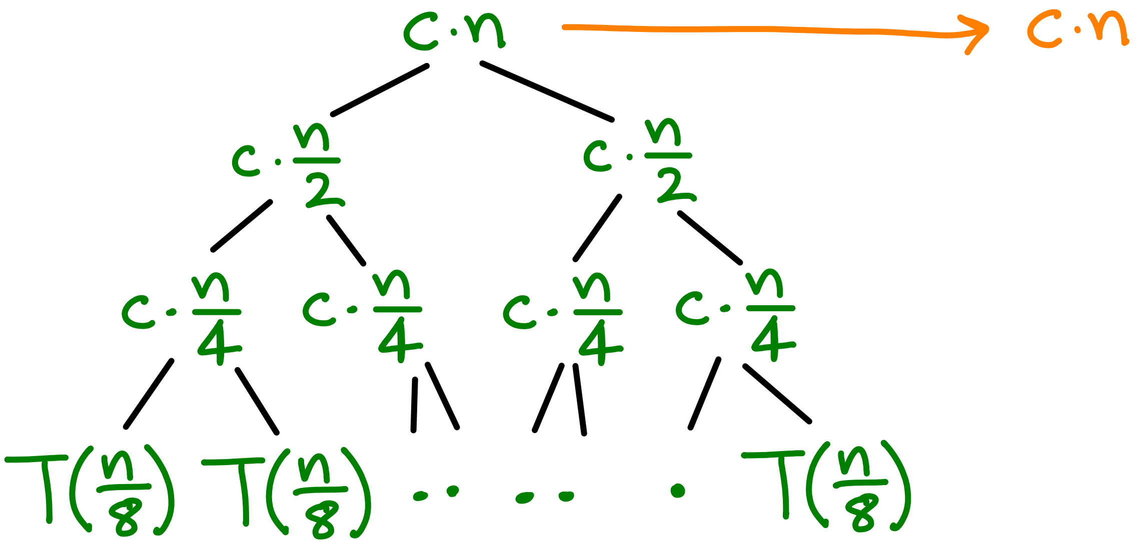
$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$ ← must do this →

$T(1) = \Theta(1) = c_2$

Recursion tree:



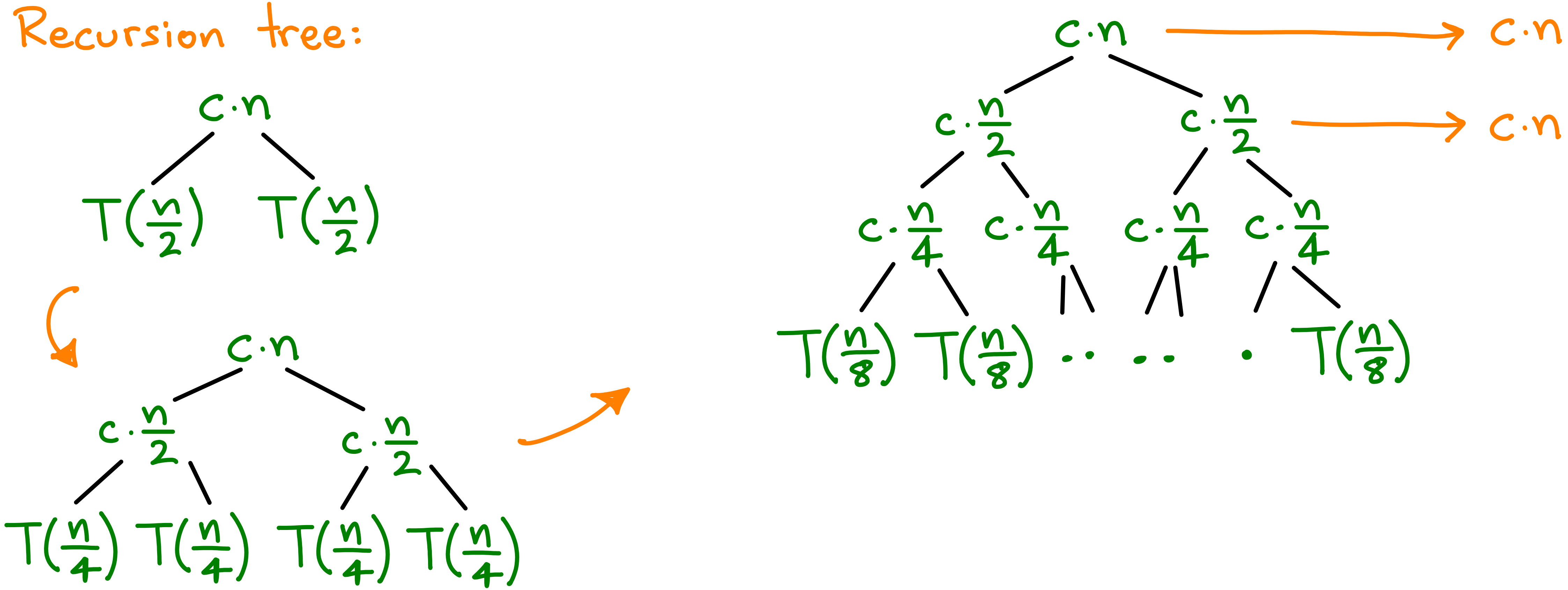




How to solve $T(n) = 2\cdot T\left(\frac{n}{2}\right) + \Theta(n)$ $\qquad T(1) = \Theta(1)$

$T(n) = 2\cdot T\left(\frac{n}{2}\right) + c\cdot n$ ← must do this → $= c_2$

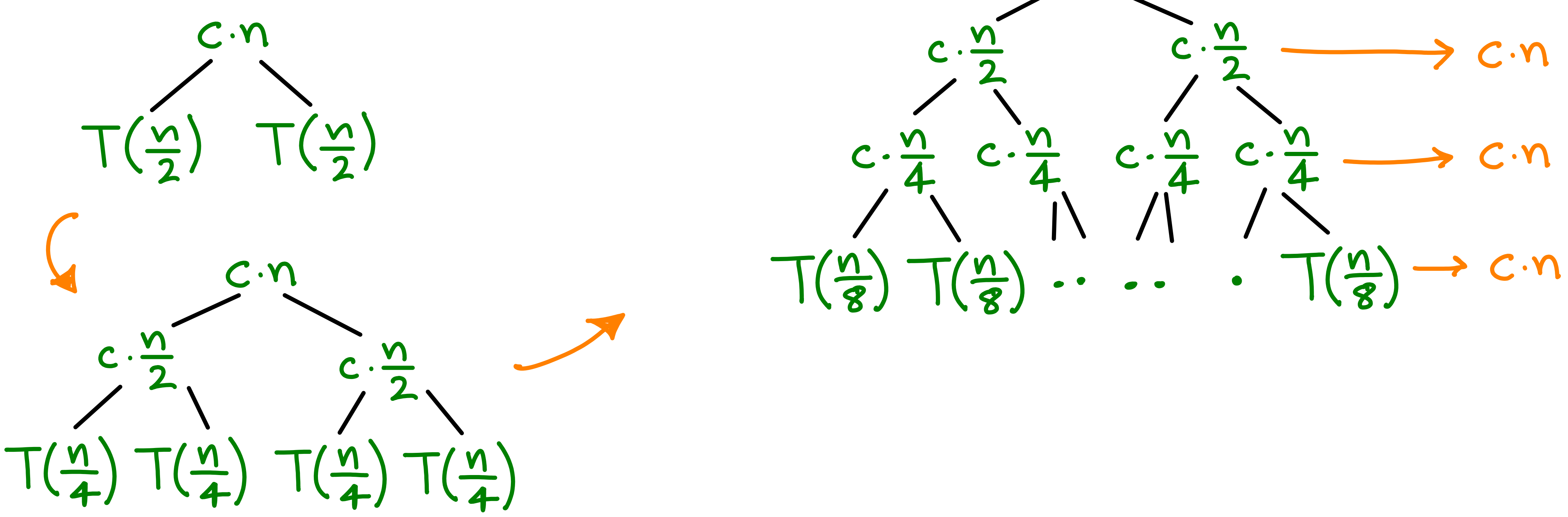
Recursion tree:



How to solve $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$ $\qquad T(1) = \Theta(1)$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + c \cdot n \quad \leftarrow \text{must do this} \rightarrow \quad = c_2$$

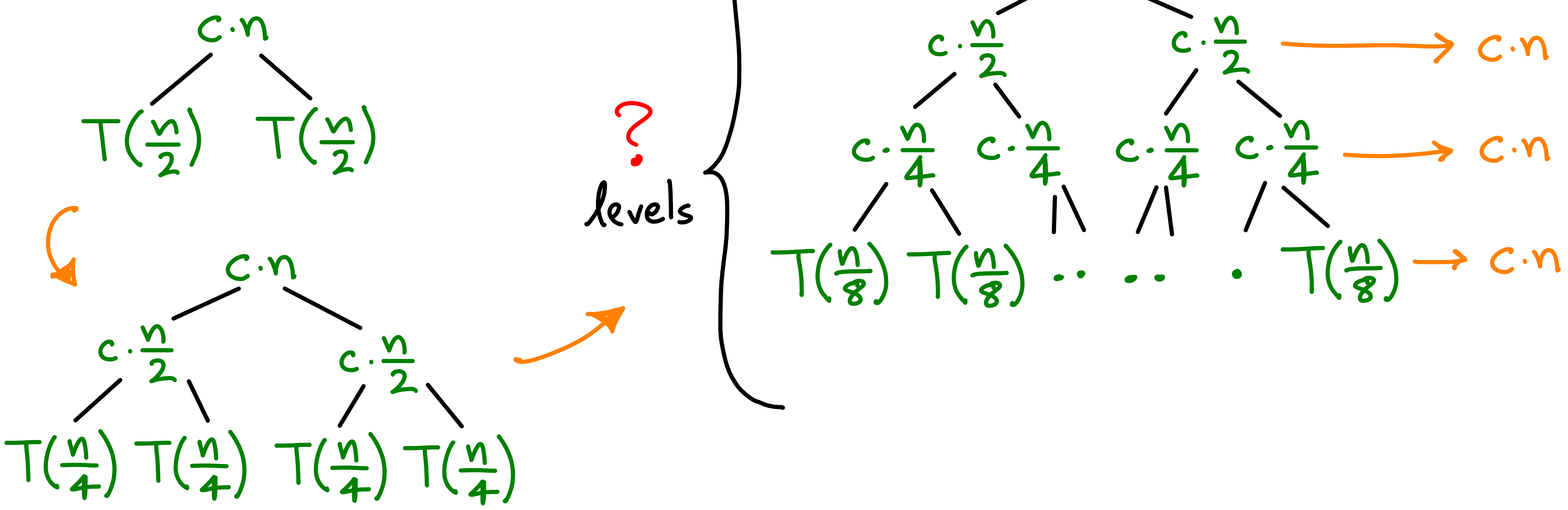
Recursion tree:



How to solve $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$ $T(1) = \Theta(1)$

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + c \cdot n$ ← must do this → $= c_2$

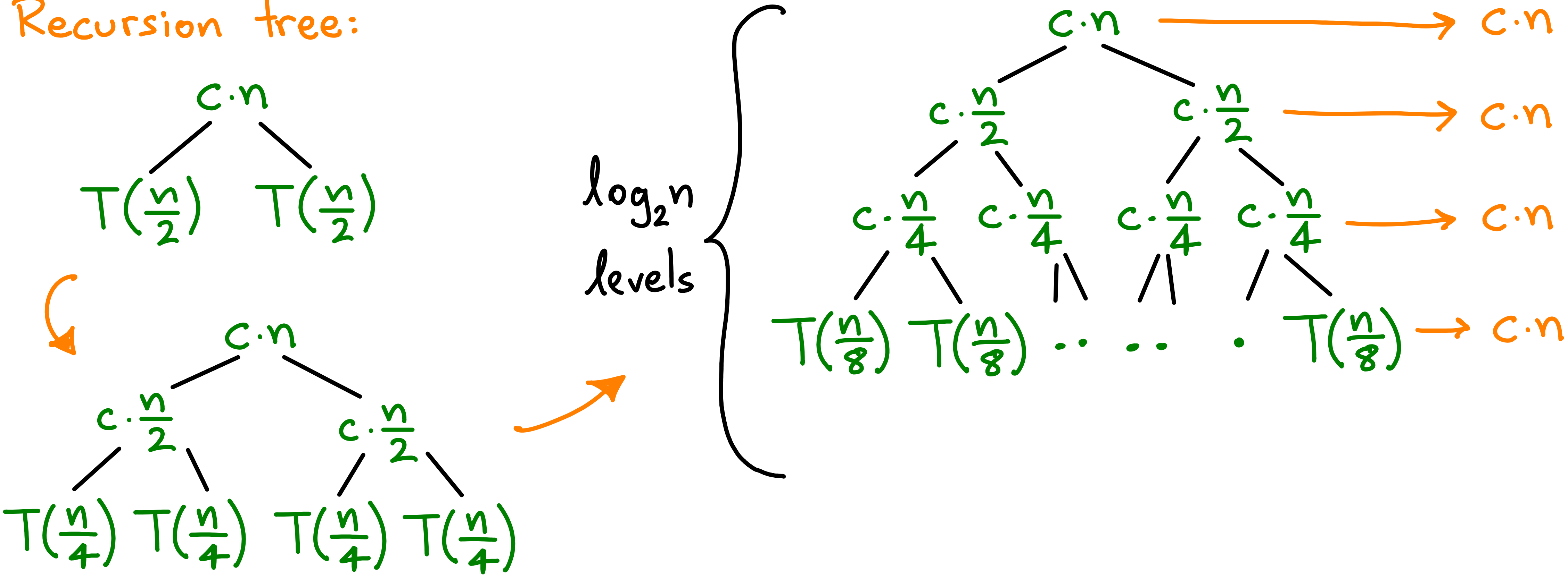
Recursion tree:



How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \Theta(n)$ $\qquad T(1) = \Theta(1)$

$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$ ← must do this → $= c_2$

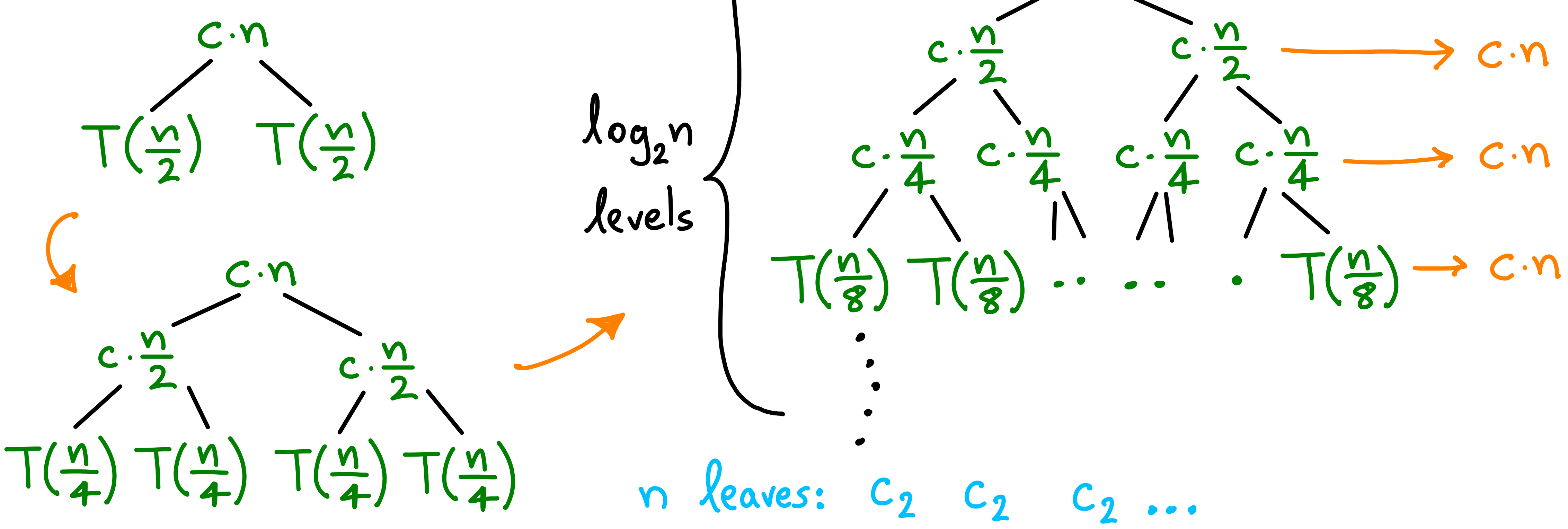
Recursion tree:



How to solve $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$ $\qquad T(1) = \Theta(1)$

$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + c \cdot n$ ← must do this → $= c_2$

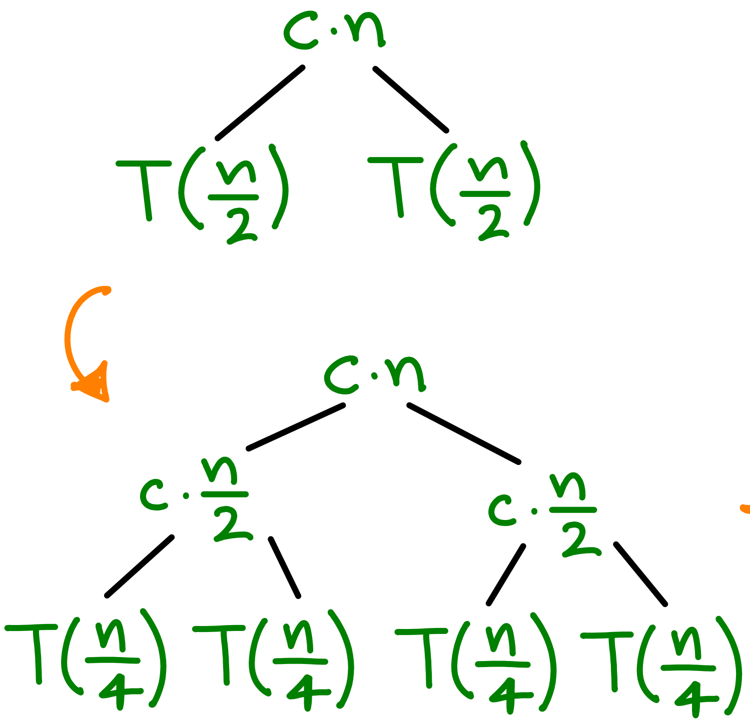
Recursion tree:

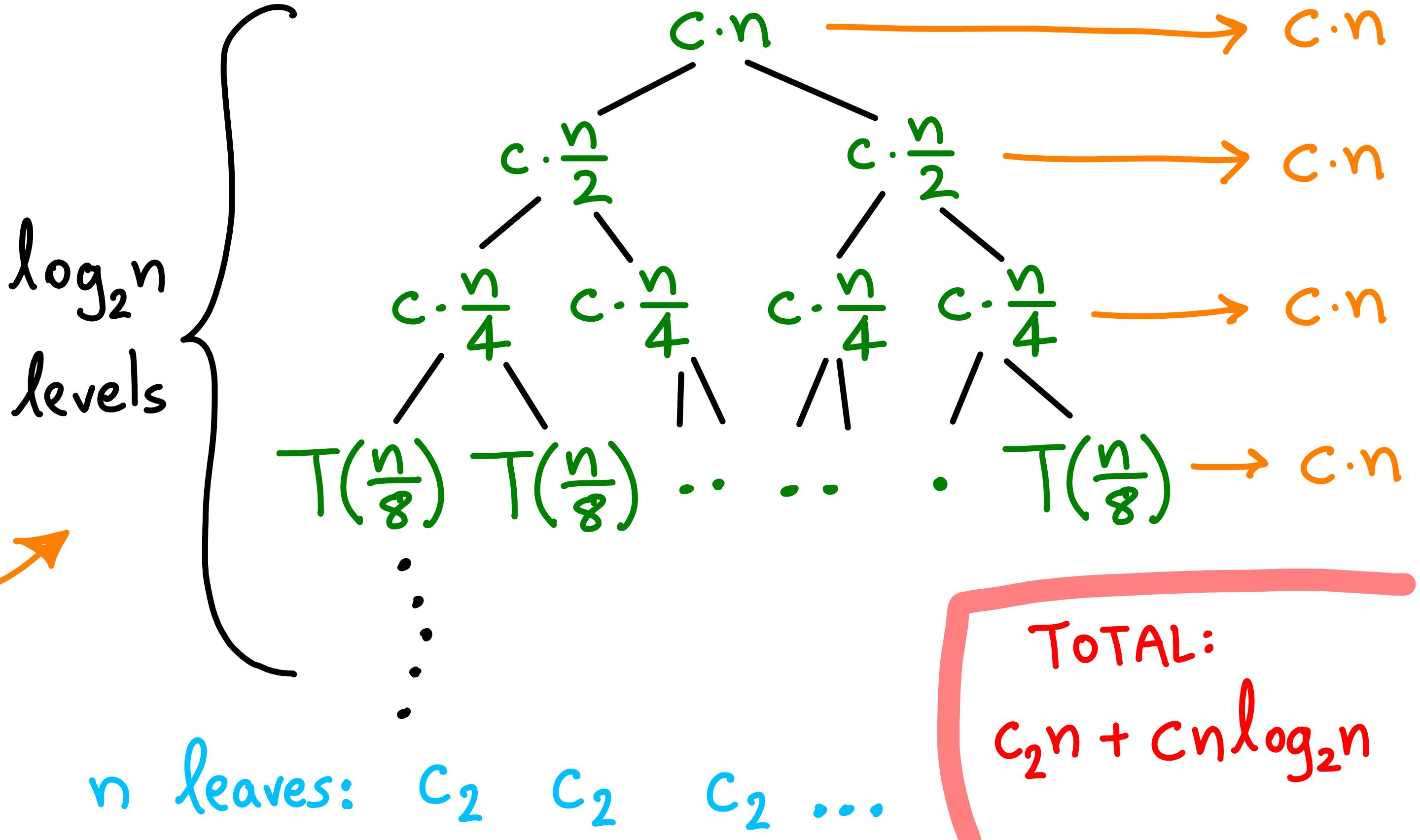


How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ $\qquad T(1) = \Theta(1)$

$T(n) = 2 \cdot T(\frac{n}{2}) + c \cdot n$ ← must do this → $= c_2$

Recursion tree:





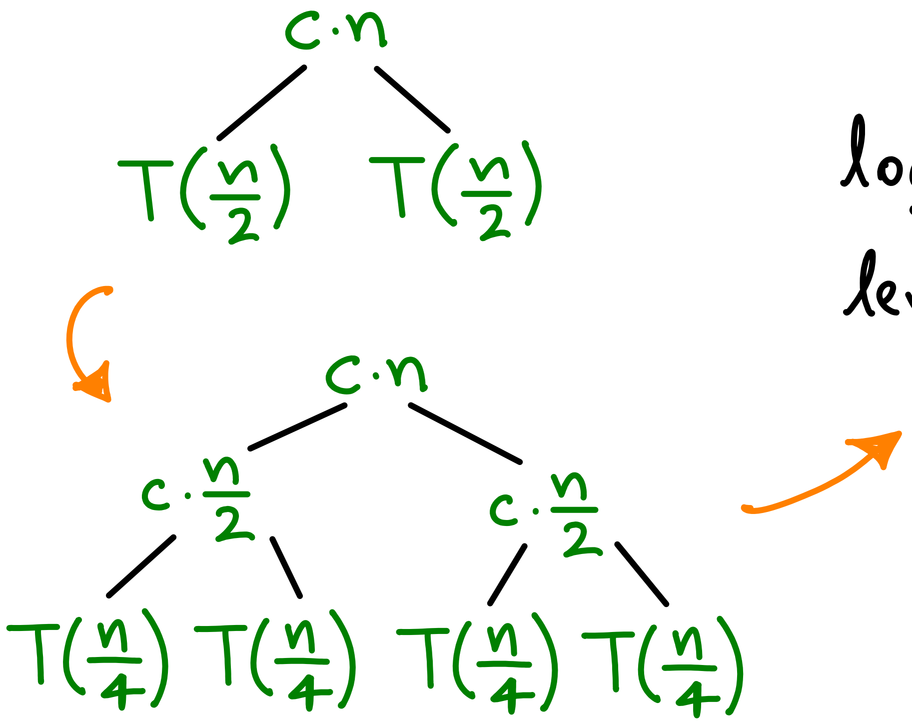
TOTAL:

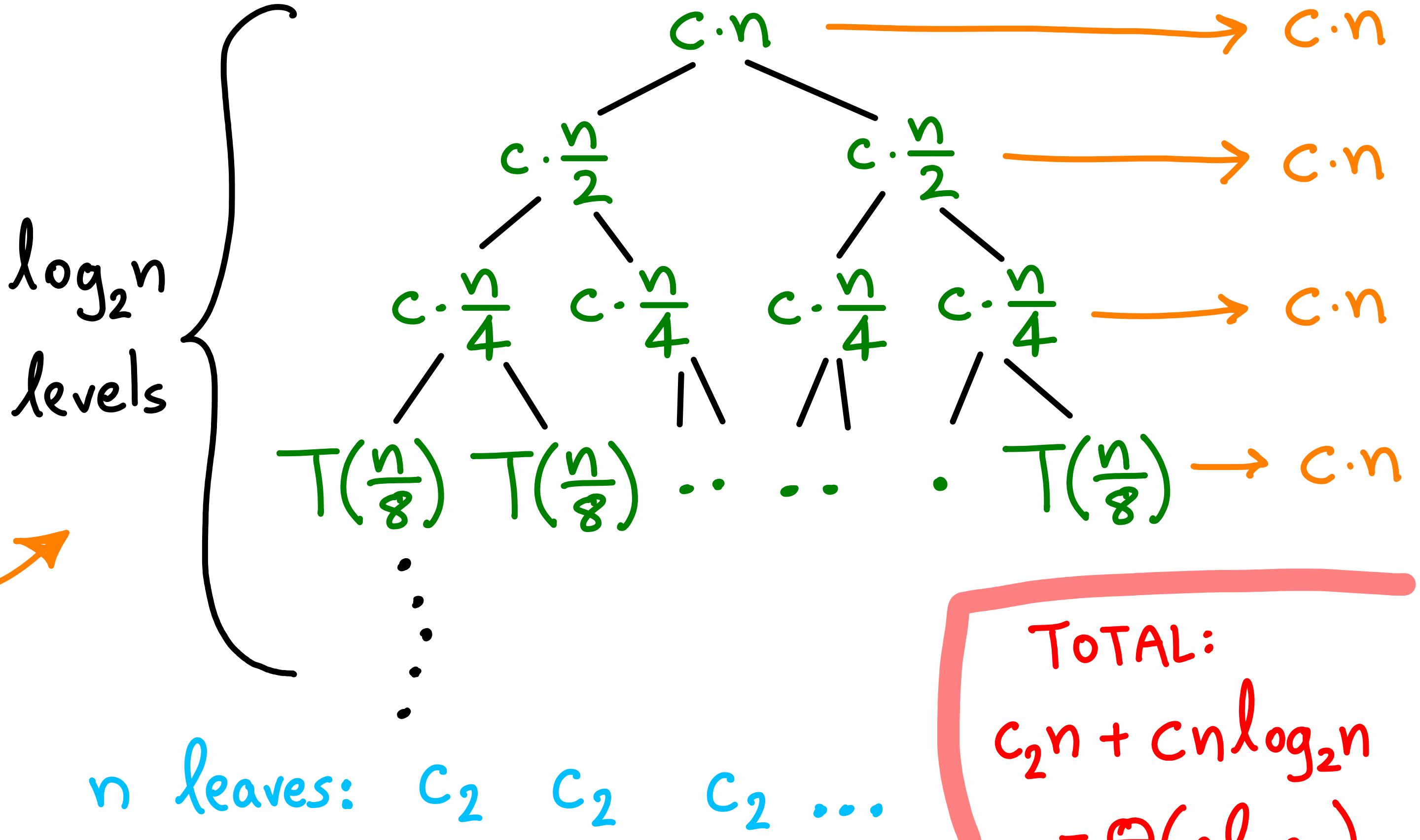
$c_2 n + c n \log_2 n$

How to solve $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{\Theta(n)}$ $\quad T(1) = \Theta(1)$

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + c \cdot n \quad \leftarrow \text{must do this} \rightarrow \quad = c_2$$

Recursion tree:





TOTAL:

$$c_2 n + c n \log_2 n = \Theta(n \log n)$$

The recursion tree method relies on noticing a pattern and for it to be formal one must prove that the pattern holds not just for a few levels.

If that's not easy to do, use substitution and induction...

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n)$ by substitution (induction)

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2 \cdot T(\tfrac{n}{2}) + c \cdot n$$ ← must do this

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2 \cdot T(\tfrac{n}{2}) + c \cdot n$$ ← must do this

o) You need to have a guess for the answer.

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2\cdot T(\tfrac{n}{2}) + c\cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$$ ← must do this

0) You need to have a guess for the answer.
↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \leq d\cdot k\log k$

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2 \cdot T(\tfrac{n}{2}) + c \cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d \cdot n \log n$? → $O(n \log n)$

1) Inductive hypothesis: for all $k < n$, $T(k) \leq d \cdot k \log k$

2) Substitute: $T(n)$ ... ?

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n)$ by substitution (induction)

$$T(n) = 2 \cdot T(\tfrac{n}{2}) + c \cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d \cdot n \log n$? → $O(n \log n)$

1) Inductive hypothesis: for all $k < n$, $T(k) \leq d \cdot k \log k$

2) Substitute: $T(n)$ (using $k = \frac{n}{2}$)

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2\cdot T(\tfrac{n}{2}) + c\cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \le d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \le d\cdot k\log k$

2) Substitute: $T(n) \le 2\cdot d\,\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \leq d\cdot k\log k$

2) Substitute: $T(n) \leq 2\cdot d\,\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

3) Algebra:

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2\cdot T(\tfrac{n}{2}) + c\cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \le d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \le d\cdot k\log k$

2) Substitute: $T(n) \le 2\cdot d\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

$$= dn\log n - dn\log 2 + cn$$

3) Algebra:

How to solve $T(n) = 2 \cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2 \cdot T(\tfrac{n}{2}) + c \cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \le d \cdot n\log n\,?$ → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \le d \cdot k\log k$

2) Substitute: $T(n) \le 2 \cdot d\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

$$= dn\log n - dn\log 2 + cn$$

3) Algebra: $$= dn\log n - (dn - cn)$$

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$T(n) = 2\cdot T(\frac{n}{2}) + c\cdot n$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \leq d\cdot k\log k$

2) Substitute: $T(n) \leq 2\cdot d\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

$$= dn\log n - dn\log 2 + cn$$

3) Algebra:

$$= dn\log n - (dn - cn)$$: desired form − leftovers

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \underbrace{\Theta(n)}$ by substitution (induction)

$$T(n) = 2\cdot T(\tfrac{n}{2}) + c\cdot n$$ ← must do this

0) You need to have a guess for the answer.

↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \leq d\cdot k\log k$

2) Substitute: $T(n) \leq 2\cdot d\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

3) Algebra:
$$\begin{cases} = dn\log n - dn\log 2 + cn \\ = dn\log n - (dn - cn) & \text{: desired form – leftovers} \\ \leq dn\log n \quad \ldots \text{ if } d \geq c \quad \square \end{cases}$$

(base case omitted)

How to solve $T(n) = 2\cdot T(\frac{n}{2}) + \Theta(n)$ by substitution (induction)

$$T(n) = 2\cdot T(\tfrac{n}{2}) + c\cdot n$$ ← must do this

part of input → no control

0) You need to have a guess for the answer. we control this

↳ focus on upper bound: $T(n) \leq d\cdot n\log n$? → $O(n\log n)$

1) Inductive hypothesis: for all $k<n$, $T(k) \leq d\cdot k\log k$

2) Substitute: $T(n) \leq 2\cdot d\frac{n}{2}\log\frac{n}{2} + cn$ (using $k=\frac{n}{2}$)

3) Algebra:
$$\begin{cases} = dn\log n - dn\log 2 + cn \\ = dn\log n - (dn - cn) \\ \leq dn\log n \;\ldots \text{ if } d \geq c \end{cases}$$
: desired form − leftovers

$\square$ (base case omitted)

More observations:

- For an upper bound, you <u>must</u> get the desired form or less.

  (exactly: with the same leading constant)

More observations:

- For an upper bound, you <u>must</u> get the desired form or less.
  (exactly: with the same leading constant)

- Like any inductive proof,
  if it doesn't work, that doesn't imply it's not true.

More observations:

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

- For an upper bound, you must get the desired form or less.

  (exactly: with the same leading constant)

- Like any inductive proof,

  if it doesn't work, that doesn't imply it's not true.

- For mergesort specifically, the constant c comes from the merge step

More observations:

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

- For an upper bound, you <u>must</u> get the desired form or less.

  (exactly: with the same leading constant)

- Like any inductive proof,

  if it doesn't work, that doesn't imply it's not true.

- For mergesort specifically, the constant c comes from the merge step

  ...and this ends up as the leading constant. $T(n) \leq c\, n \log n$

  (Speedup of mergesort is directly proportional to speedup of merge)

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

What should we guess?

$$T(1) = 1$$

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

What should we guess?

$T(1) = 1$

$T(2) = 4T(1) + 2$

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

What should we guess?

$$T(1) = 1$$

$$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$$

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

What should we guess?

$T(1) = 1$

$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$

$T(4) = 4T(2) + 4$

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

What should we guess?

$T(1) = 1$

$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$

$T(4) = 4T(2) + 4 = 4 \cdot 6 + 4 = 28$

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

What should we guess?

$$T(1) = 1$$

$$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$$

$$T(4) = 4T(2) + 4 = 4 \cdot 6 + 4 = 28$$

$$T(8) = 4T(4) + 8 = 4 \cdot 28 + 8 = 120$$

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

What should we guess?

$$T(1) = 1$$

$$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$$

$$T(4) = 4T(2) + 4 = 4 \cdot 6 + 4 = 28$$

$$T(8) = 4T(4) + 8 = 4 \cdot 28 + 8 = 120$$

$$T(16) = 4T(8) + 16 = 4 \cdot 120 + 16 = 496$$

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

What should we guess?

$$T(1) = 1$$

$$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$$

$$T(4) = 4T(2) + 4 = 4 \cdot 6 + 4 = 28$$

$$T(8) = 4T(4) + 8 = 4 \cdot 28 + 8 = 120$$

$$T(16) = 4T(8) + 16 = 4 \cdot 120 + 16 = 496$$

$$T(32) = 4T(16) + 32 = \underbrace{4 \cdot 496 + 32}_{\text{2nd term less and less significant}} = 2016$$

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

What should we guess?

$T(1) = 1$

$$T(2) = 4T(1) + 2 = 4 \cdot 1 + 2 = 6$$

$$T(4) = 4T(2) + 4 = 4 \cdot 6 + 4 = 28$$

$$T(8) = 4T(4) + 8 = 4 \cdot 28 + 8 = 120$$

$$T(16) = 4T(8) + 16 = 4 \cdot 120 + 16 = 496$$

$$T(32) = 4T(16) + 32 = 4 \cdot 496 + 32 = 2016$$

: starts looking like $2n^2$

2nd term less and less significant

$T(n) = 4T(\frac{n}{2}) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though.

$T(n) = 4T(\frac{n}{2}) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $\quad T(1) = 1$ $\quad$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis : ?

$T(n) = 4T(\frac{n}{2}) + n$ $\quad T(1) = 1$ $\quad$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

$T(n) = 4T(\frac{n}{2}) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n)$ ... ?

$T(n) = 4T(\frac{n}{2}) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

$T(n) = 4T(\frac{n}{2}) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra: $= \frac{1}{2}cn^3 + n$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra:

$$= \frac{1}{2}cn^3 + n$$

$$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n$$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra:

$$= \frac{1}{2}cn^3 + n$$

$$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n = cn^3 - \left(\frac{1}{2}cn^3 - n\right)$$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $T(1) = 1$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra: $= \frac{1}{2}cn^3 + n$

$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n = cn^3 - \underbrace{\left(\frac{1}{2}cn^3 - n\right)}_{\geq 0 \text{ if } c \geq 2 \text{ for all } n \geq 0}$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $\quad T(1) = 1$ $\quad$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra:

$$= \frac{1}{2}cn^3 + n$$

$$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n = cn^3 - \overbrace{\underbrace{\left(\frac{1}{2}cn^3 - n\right)}_{\geq 0 \text{ if } c \geq 2 \text{ for all } n \geq 0}}^{\geq 0 \text{ if } c \geq 1 \text{ \& } n \geq 2}$$

but now you must handle a larger base case

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $\quad T(1) = 1$ $\quad$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra:

$$= \frac{1}{2}cn^3 + n$$

$$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n = cn^3 - \overbrace{\underbrace{\left(\frac{1}{2}cn^3 - n\right)}_{\substack{\geq 0 \text{ if } c \geq 2 \\ \text{for all } n \geq 0}}}^{\substack{\geq 0 \text{ if} \\ c \geq 1 \ \& \ n \geq 2}} \rightarrow T(n) \leq cn^3$$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ $\quad T(1) = 1$ $\quad$ What should we guess?

It looked quadratic. Not really convincing though. Let's be cautious.

Try $O(n^3)$: $T(n) \overset{?}{\leq} cn^3$

Hypothesis: $T(k) \leq ck^3$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n$

Algebra: $= \frac{1}{2}cn^3 + n$

$$= \underbrace{cn^3}_{\text{desired form}} - \frac{1}{2}cn^3 + n = cn^3 - \overbrace{\underbrace{\left(\frac{1}{2}cn^3 - n\right)}_{\geq 0 \text{ if } c \geq 2 \text{ for all } n \geq 0}}^{\geq 0 \text{ if } c \geq 1 \,\&\, n \geq 2} \rightarrow T(n) \leq cn^3$$

Base case: $T(1) = 1 \leq c \cdot 1^3$ ✓

$\square$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \leq ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$

Base case: $T(1) = O(1)$ ✓

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$

Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \leq ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$ Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

$$T(n-1) = n-1 = O(1)$$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \leq ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$

Let's show $T(n) = n$ is $O(1)$

Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

$\hookrightarrow T(n-1) = n-1 = O(1)$

$T(n) = n = n-1+1$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \le ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$ Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

$\hookrightarrow T(n-1) = n-1 = O(1)$

$T(n) = n = n-1+1 = O(1) + 1$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \leq ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$

Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

$\hookrightarrow T(n-1) = n-1 = O(1)$

$$T(n) = n = n-1+1 = O(1) + 1 = O(1)$$

We wanted to show $T(n) = 4T\left(\frac{n}{2}\right) + n = O(n^3)$.

Why use hypothesis $T(k) \leq ck^3$ instead of $T(k) = O(k^3)$?

e.g., $T(n) = 4T\left(\frac{n}{2}\right) + n$

$$= 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = 4 \cdot O\left(\frac{1}{8}n^3\right) + n = O(n^3)$$

Let's show $T(n) = n$ is $O(1)$

Base case: $T(1) = O(1)$ ✓

Hypothesis: $T(k) = O(1)$ for all $k < n$

$\hookrightarrow T(n-1) = n-1 = O(1)$

$T(n) = n = n-1+1 = O(1) + 1 = O(1)$

DON'T USE BIG-O DURING INDUCTION

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

Hypothesis: $T(k) \leq ck^2$ for $k < n$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

Hypothesis: $T(k) \leq ck^2$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^2 + n$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

Hypothesis: $T(k) \leq ck^2$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^2 + n$

Algebra: $= cn^2 + n$

...?

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

Hypothesis: $T(k) \leq ck^2$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^2 + n$

Algebra: $= cn^2 + n$ → failed to get $\leq$ desired form

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. Let's try $O(n^2)$

$$T(n) \overset{?}{\leq} cn^2$$

Hypothesis: $T(k) \leq ck^2$ for $k < n$

Substitute: $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^2 + n$

Algebra: $= cn^2 + n$ → failed to get $\leq$ desired form

$$= cn^2 + \frac{1}{n}n^2 = \left(c + \frac{1}{n}\right) \cdot n^2$$

so close to cn^2
but not good enough

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \leq ck^2$ failed.

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \leq ck^2$ failed. New hypothesis: $T(k) \leq ck^2 - dk$

Substitute: $T(n) \leq 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \leq ck^2$ failed. New hypothesis: $T(k) \leq ck^2 - dk$

Substitute: $T(n) \leq 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

Substitute: $T(n) \le 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n$$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

Substitute: $T(n) \le 4\cdot\left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n = cn^2 - dn - (dn - n)$$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

Substitute: $T(n) \le 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n = cn^2 - dn - \underbrace{(dn - n)}_{\ge 0 \text{ if } d \ge 1}$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

Substitute: $T(n) \le 4\cdot\left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n = cn^2 - dn - \underbrace{(dn - n)}_{\ge 0 \text{ if } d \ge 1}$$

So can c be anything?

(are we done?)

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \leq ck^2$ failed. New hypothesis: $T(k) \leq ck^2 - dk$

Substitute: $T(n) \leq 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n \quad = cn^2 - dn - \underbrace{(dn - n)}_{\geq 0 \text{ if } d \geq 1}$$

So can c be anything? No.

Base case: $T(1) = 1$

$T(n) = 4T\left(\frac{n}{2}\right) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \le ck^2$ failed. New hypothesis: $T(k) \le ck^2 - dk$

Substitute: $T(n) \le 4\cdot\left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n = cn^2 - dn - \underbrace{(dn - n)}_{\geq 0 \text{ if } d \geq 1}$$

So can c be anything? No.

Base case: $T(1) = 1 \le c\cdot 1^2 - d\cdot 1$ if $c > d$

$T(n) = 4T(\frac{n}{2}) + n$ / $T(1) = 1$ / We proved $O(n^3)$. We want $O(n^2)$

Hypothesis: $T(k) \leq ck^2$ failed. New hypothesis: $T(k) \leq ck^2 - dk$

Making the hypothesis stronger often works. See my notes on induction.

Substitute: $T(n) \leq 4 \cdot \left(c\left(\frac{n}{2}\right)^2 - d\frac{n}{2}\right) + n$

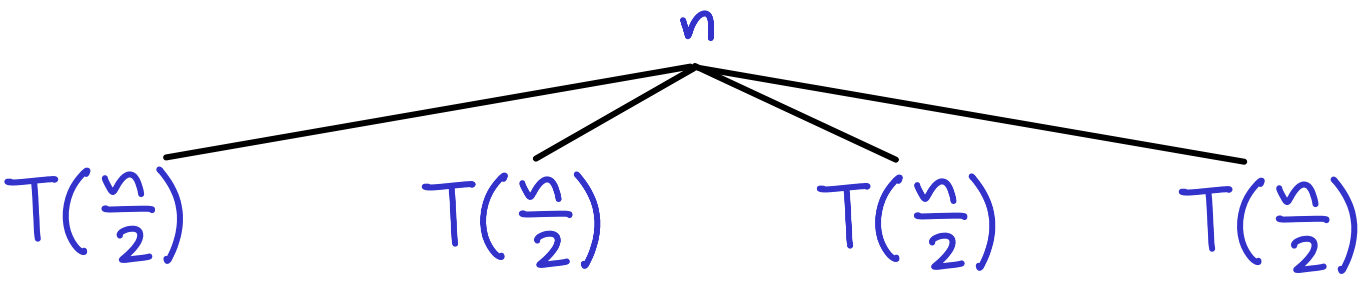
Algebra: $= cn^2 - 2dn + n$

$$= \underbrace{cn^2 - dn}_{\text{desired form}} - dn + n = cn^2 - dn - \underbrace{(dn - n)}_{\geq 0 \text{ if } d \geq 1}$$

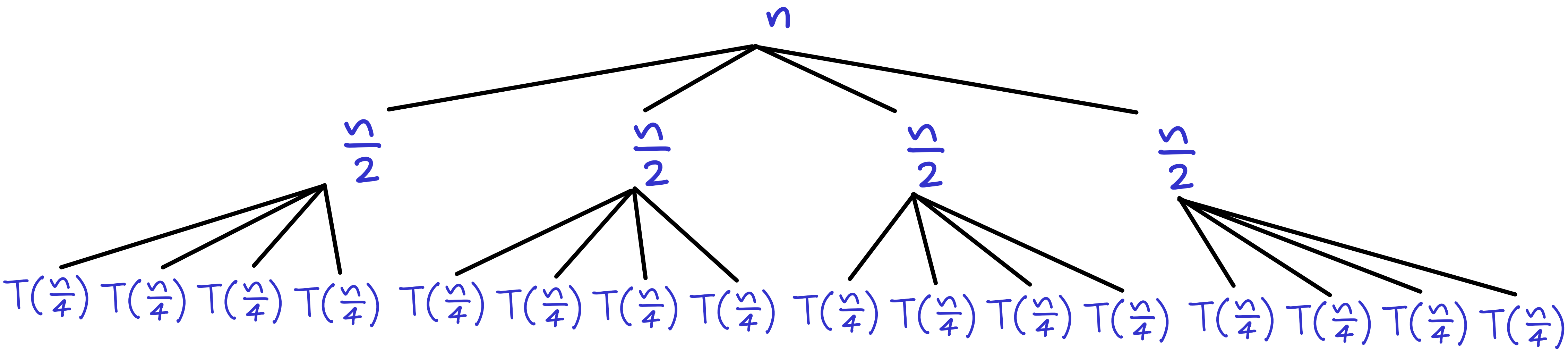
So can c be anything? No.

Base case: $T(1) = 1 \leq c \cdot 1^2 - d \cdot 1$ if $c > d$

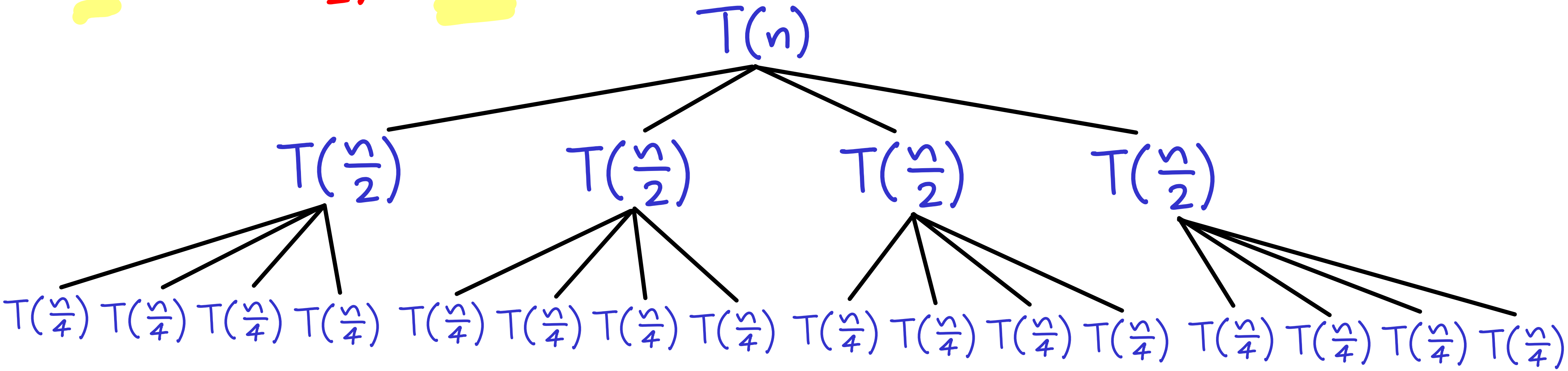
$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

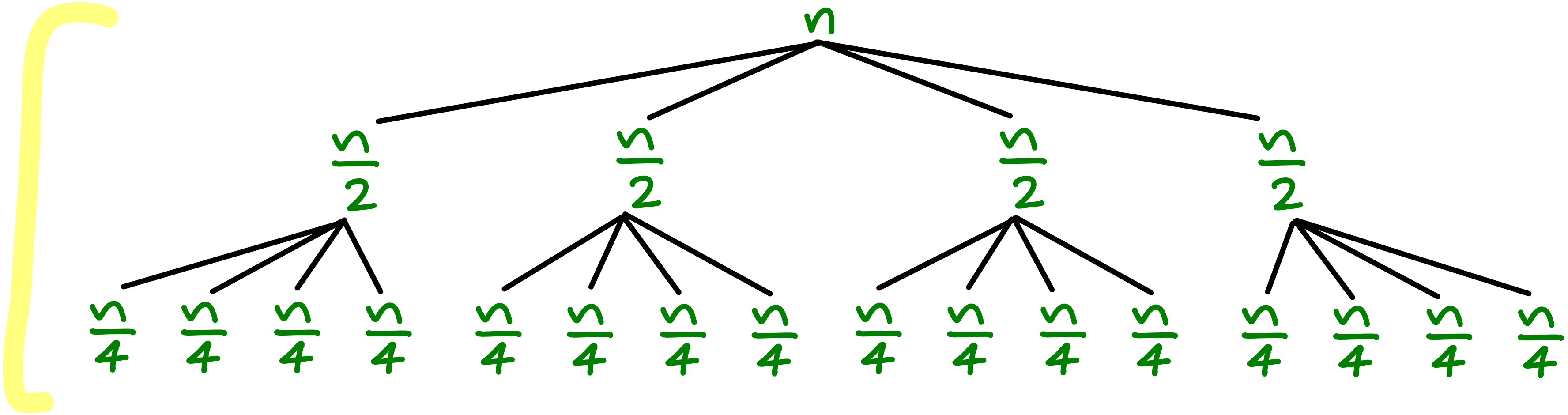


$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

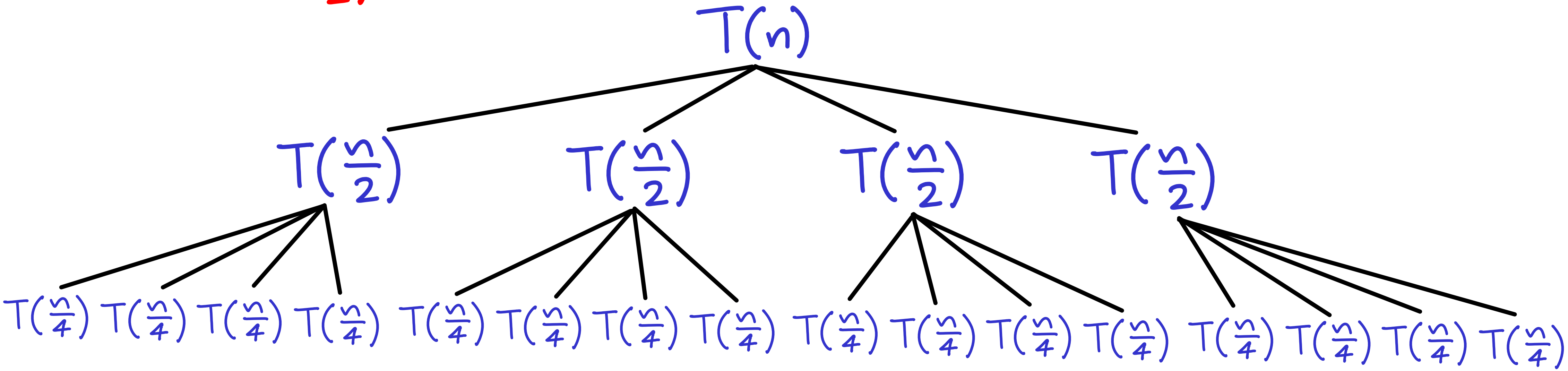


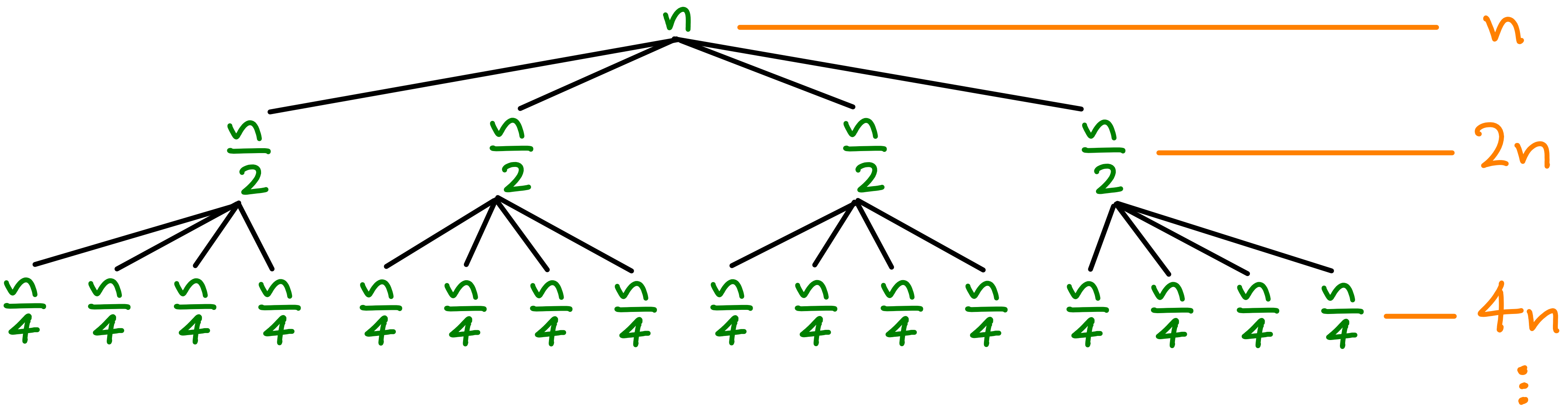
$$T(n) = 4T\left(\frac{n}{2}\right) + n$$



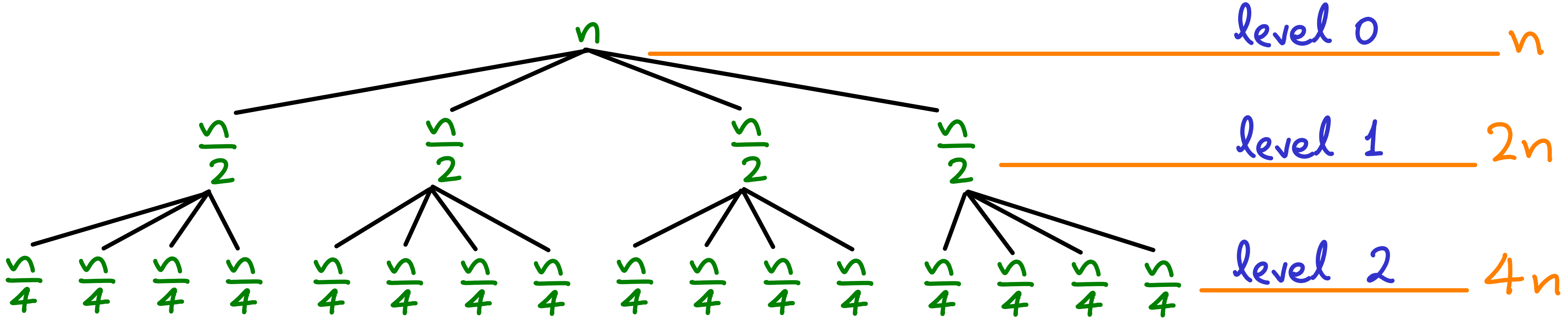


$$T(n) = 4T\left(\frac{n}{2}\right) + n$$

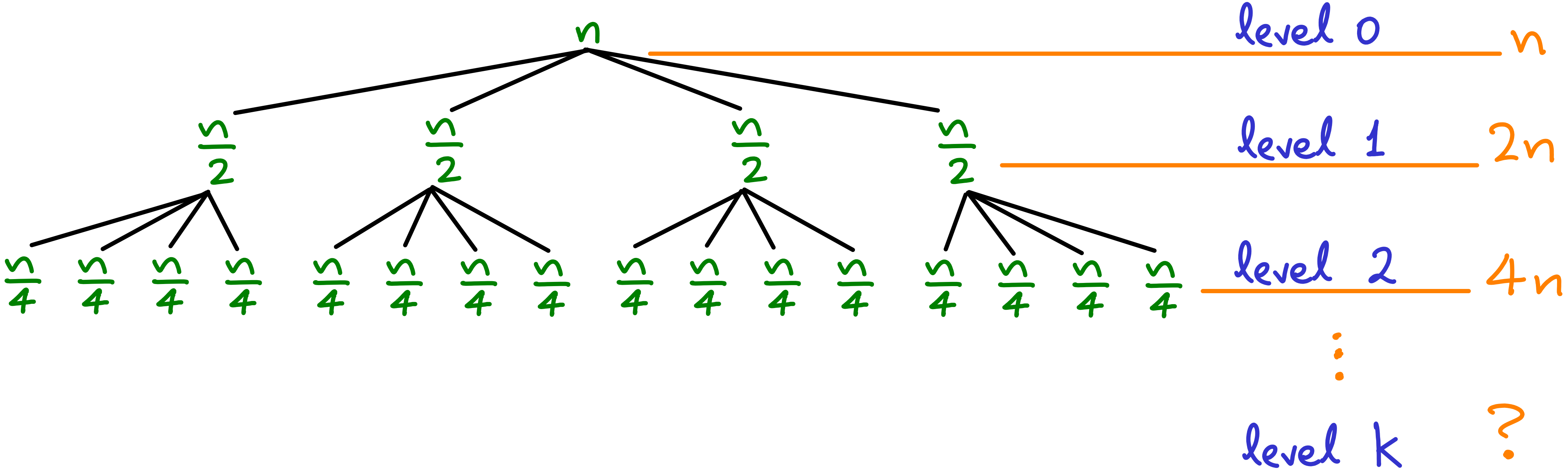




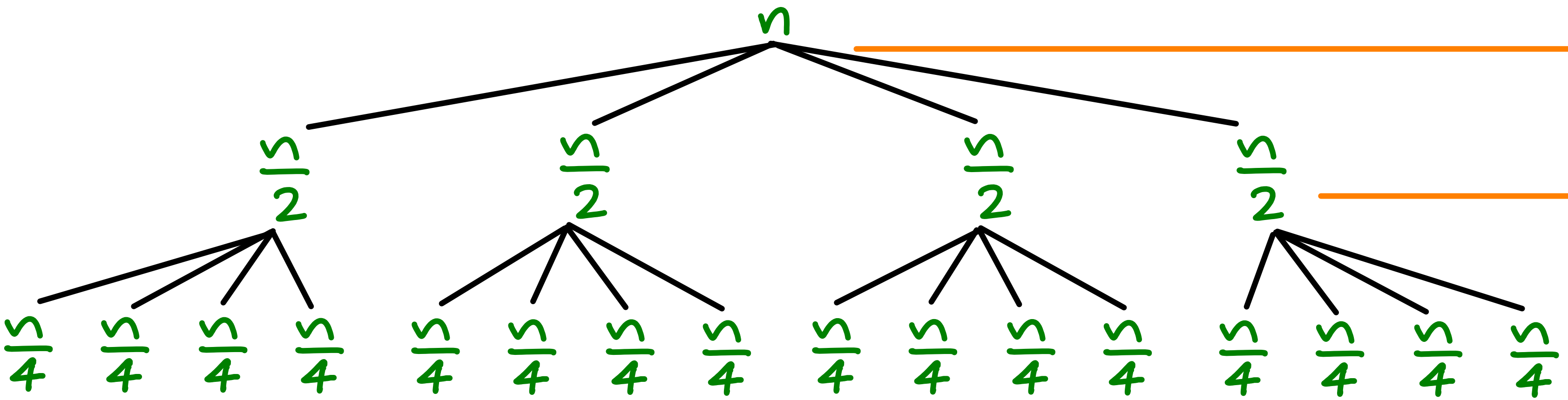
$$T(n) = 4T\left(\frac{n}{2}\right) + n$$



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



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



level 0 n

level 1 $2n$

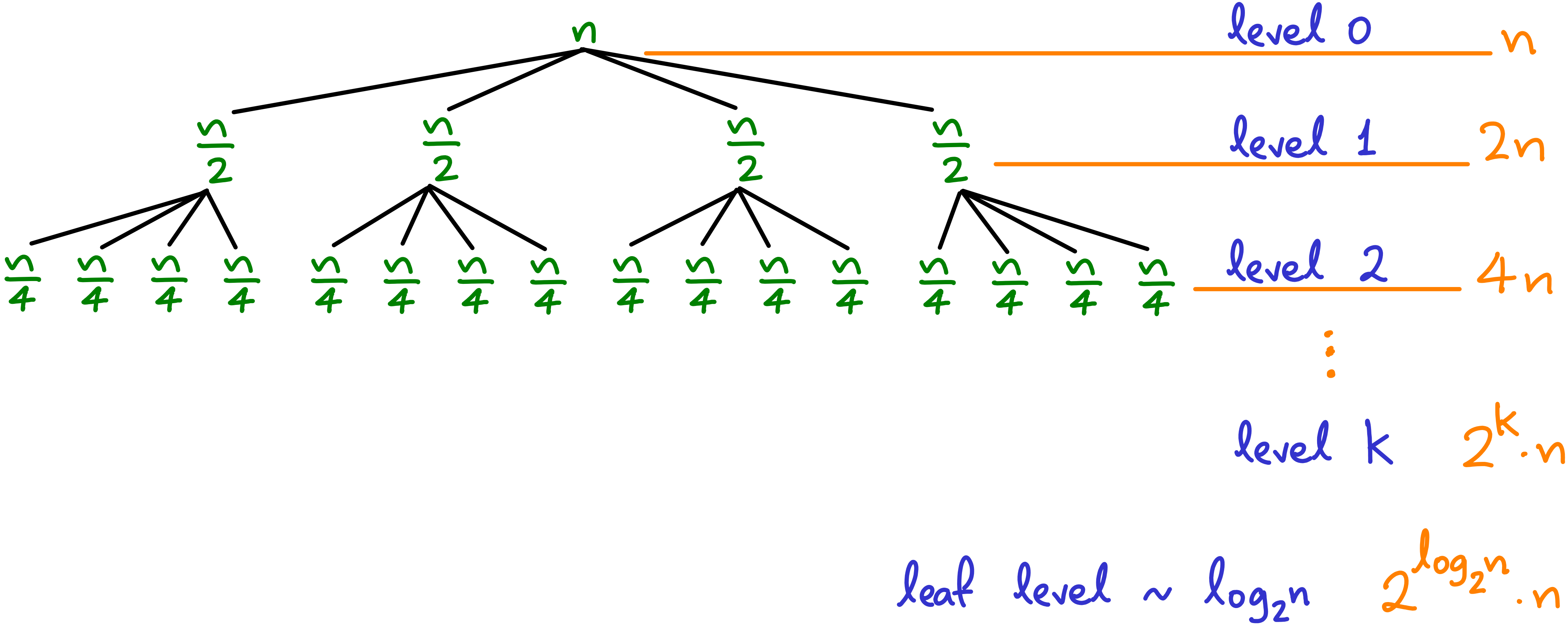
level 2 $4n$

$\vdots$

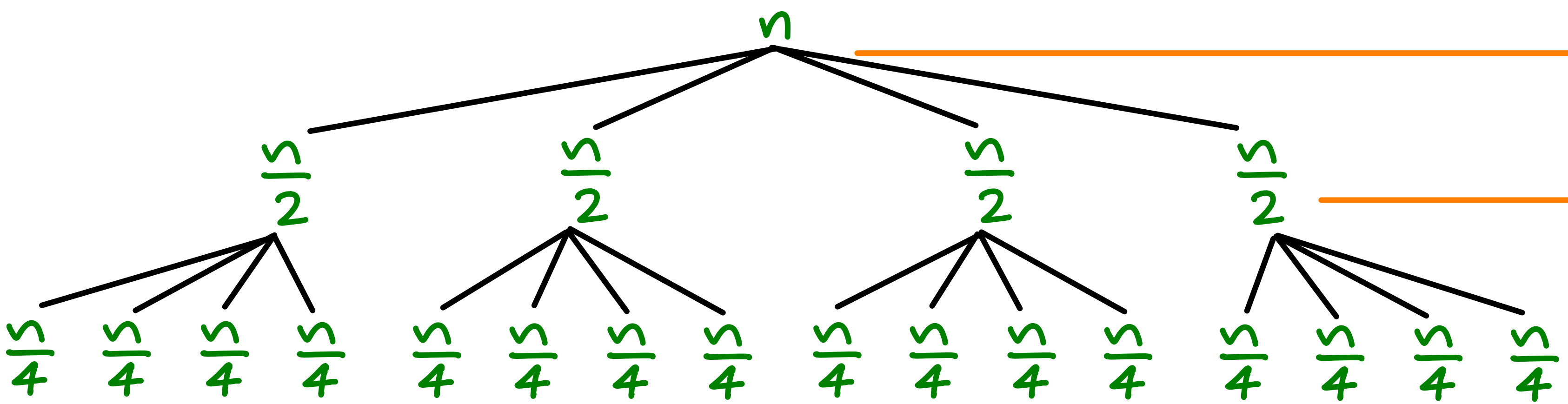
level k $2^k \cdot n$

leaf level ~ ?

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



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



level 0 n

level 1 $2n$

level 2 $4n$

$\vdots$

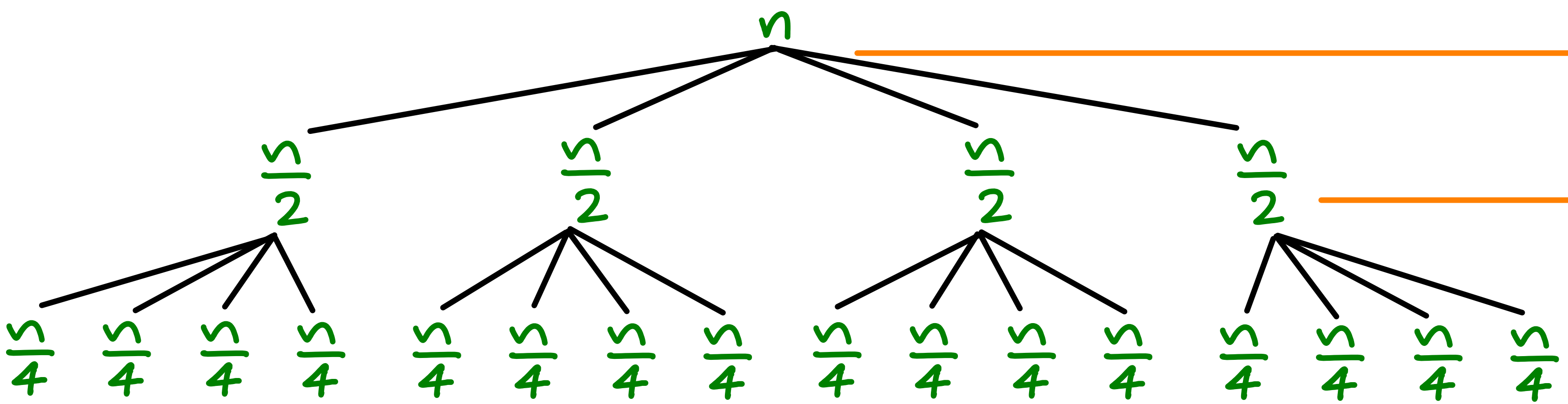
level k $2^k \cdot n$

leaf level ~ $\log_2 n$ $2^{\log_2 n} \cdot n$

$= n^2$

SUM = ?

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



level 0 $\quad n$

level 1 $\quad 2n$

level 2 $\quad 4n$

$\vdots$

level k $\quad 2^k \cdot n$

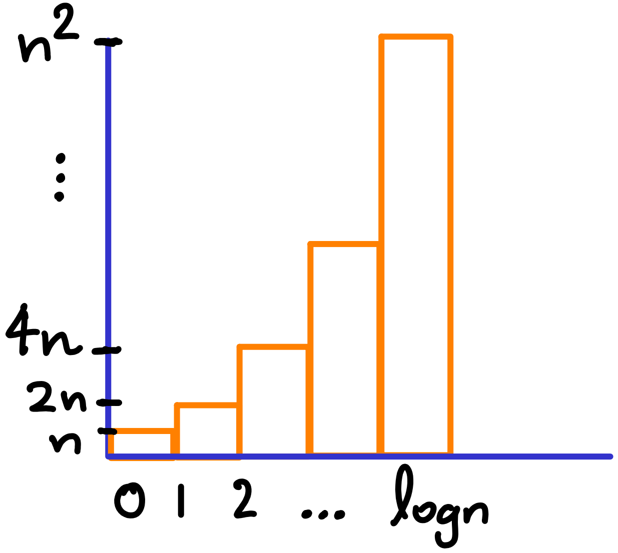
leaf level $\sim \log_2 n \quad 2^{\log_2 n} \cdot n$

$= n^2$

$\text{SUM} = O(n^2)$

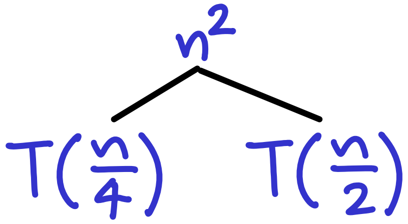
Geometric series

$\text{Sum} = 2n^2$

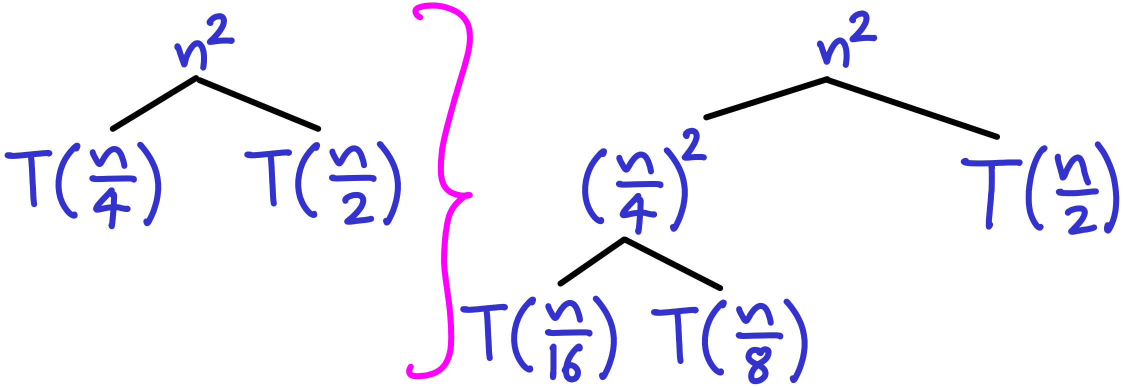


$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$

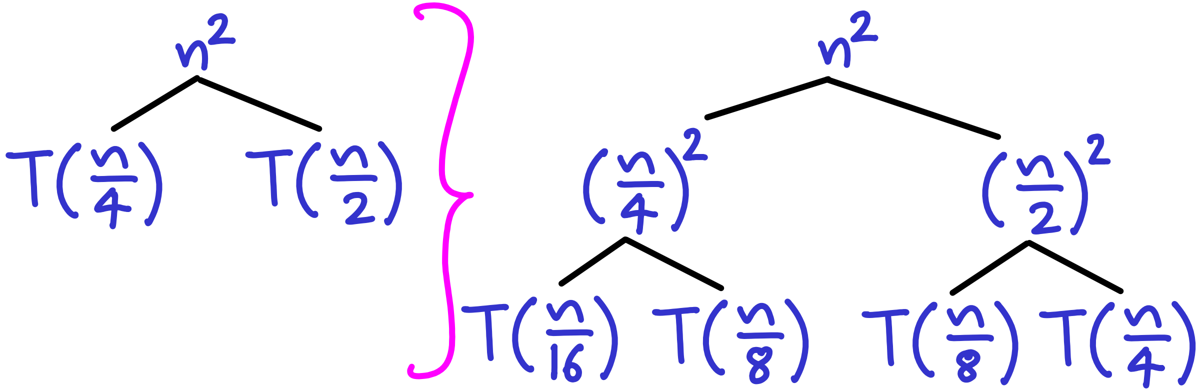
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



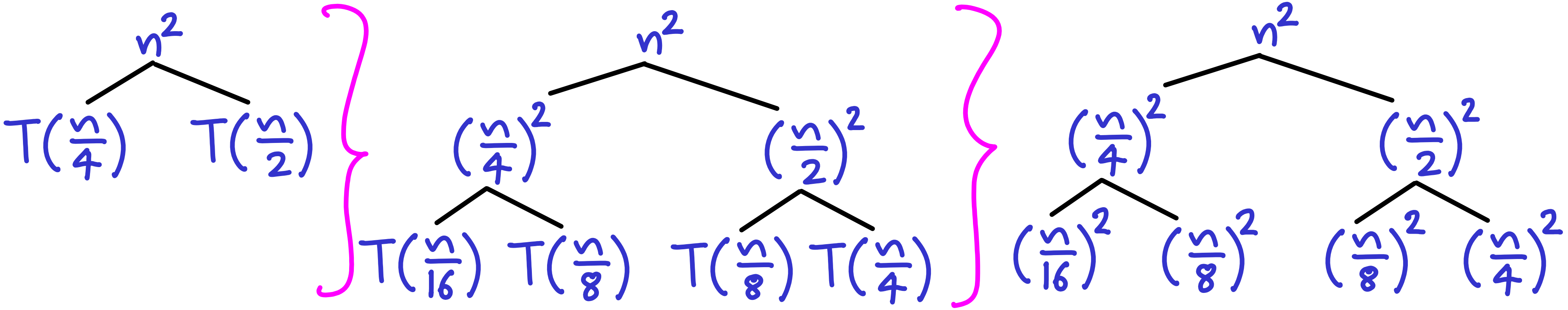
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



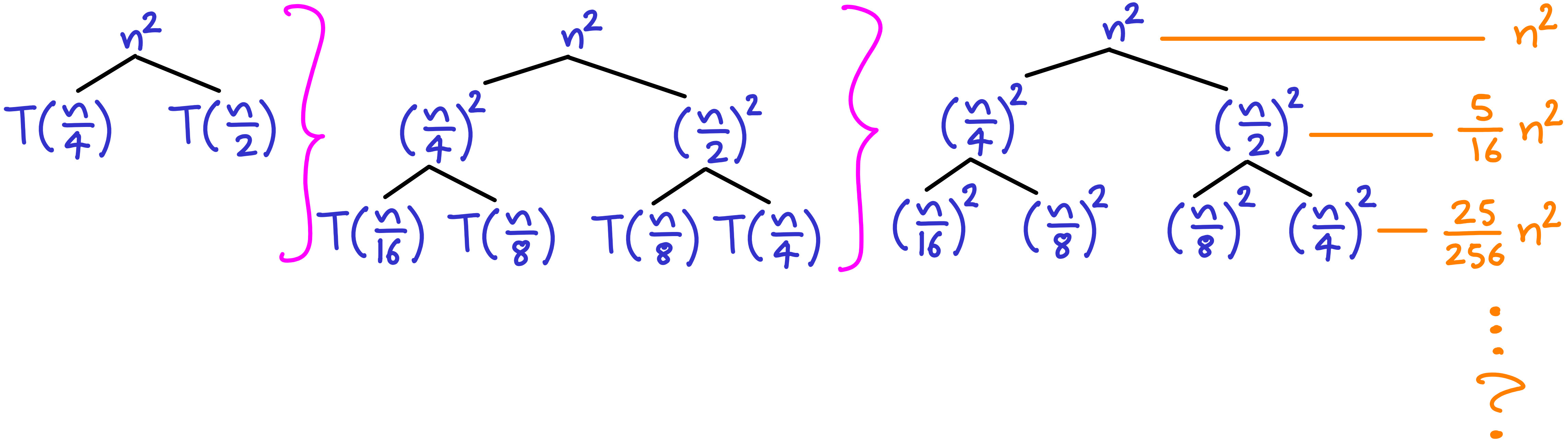
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



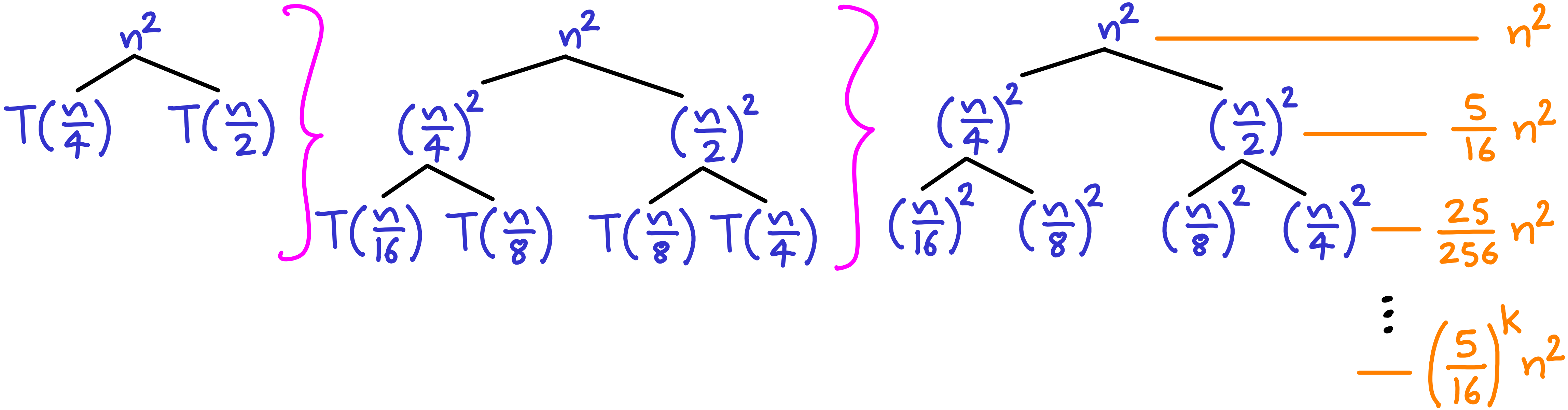
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



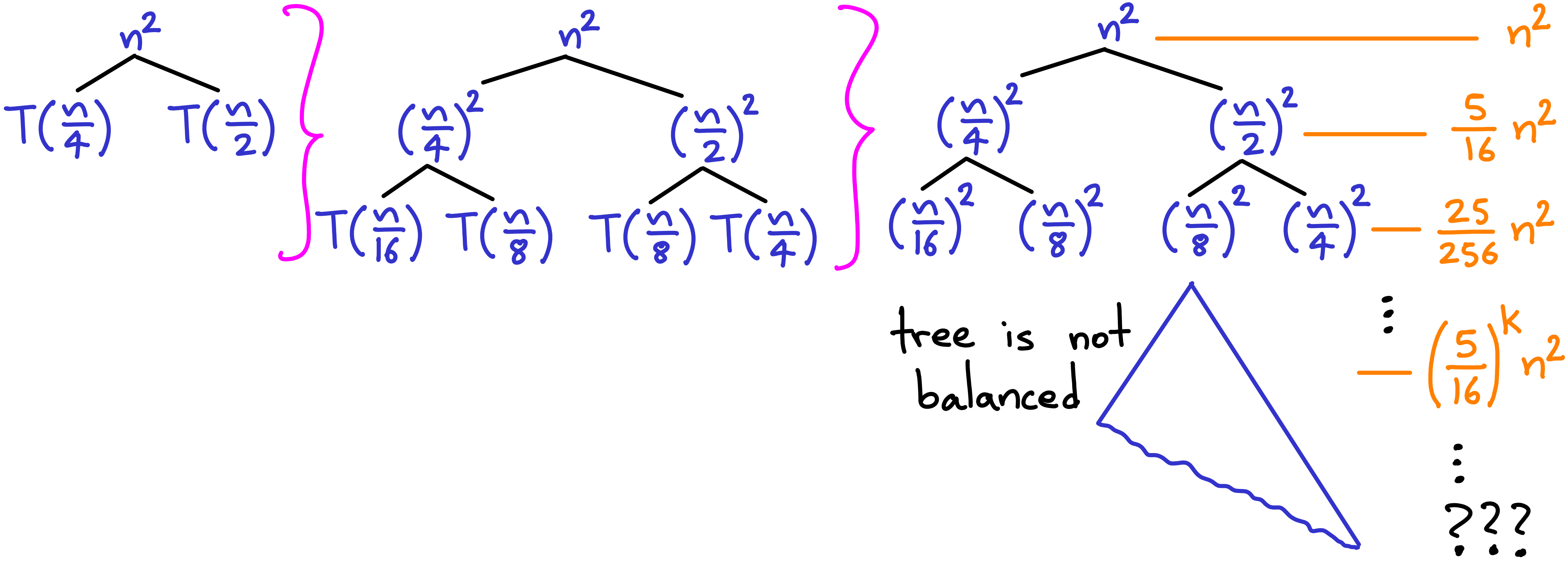
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



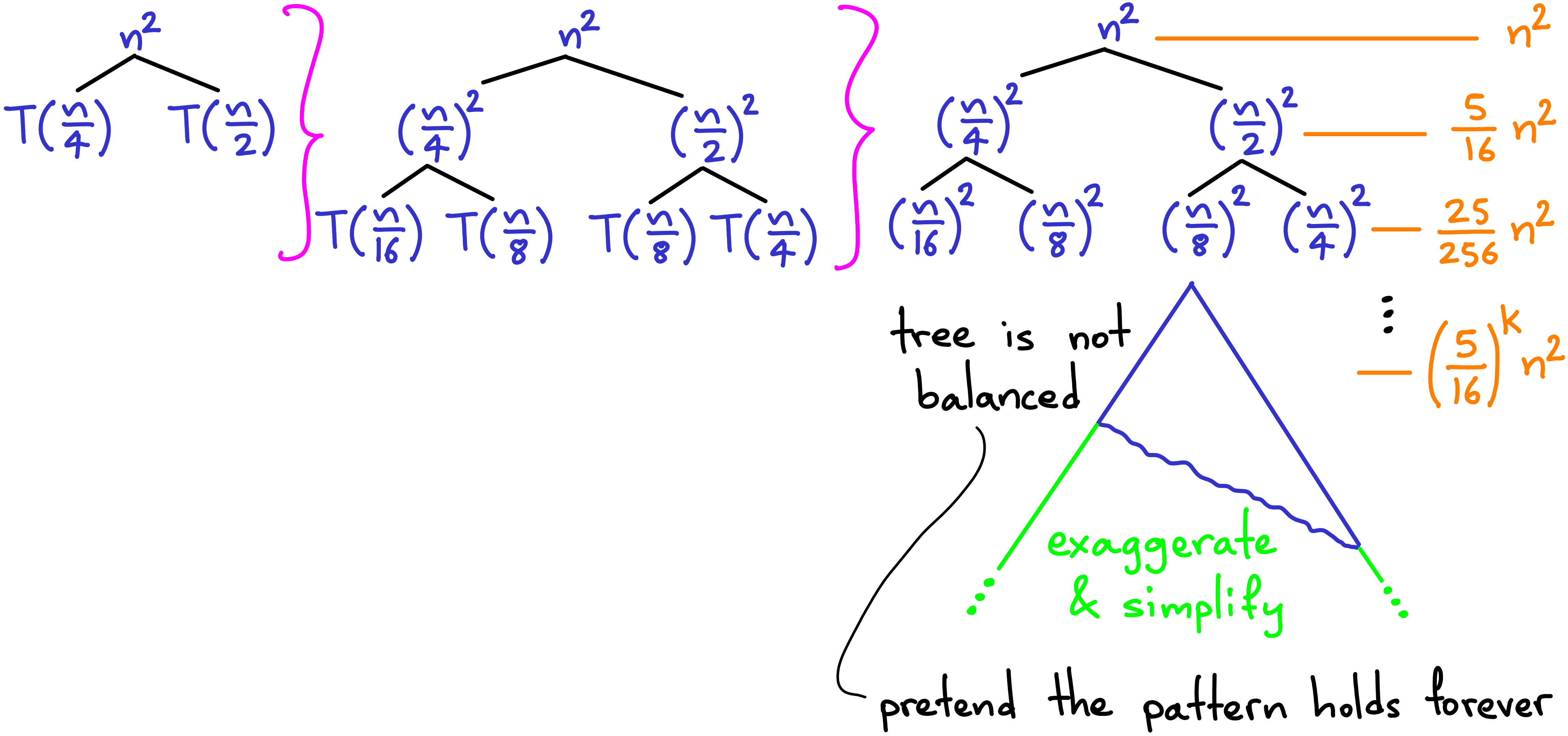
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



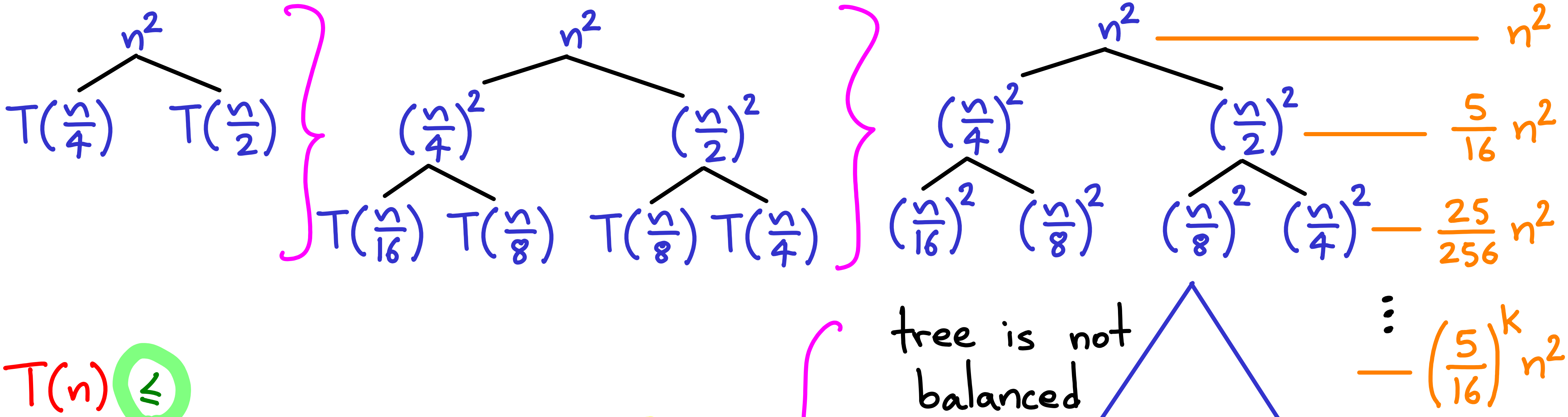
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



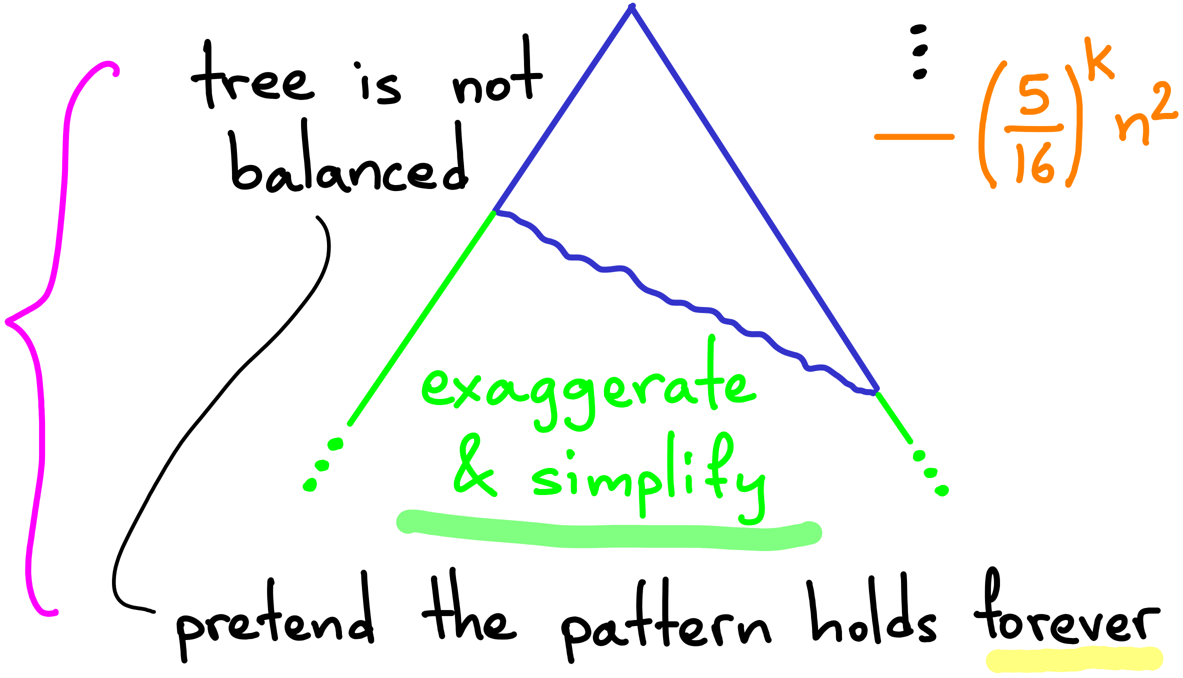
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



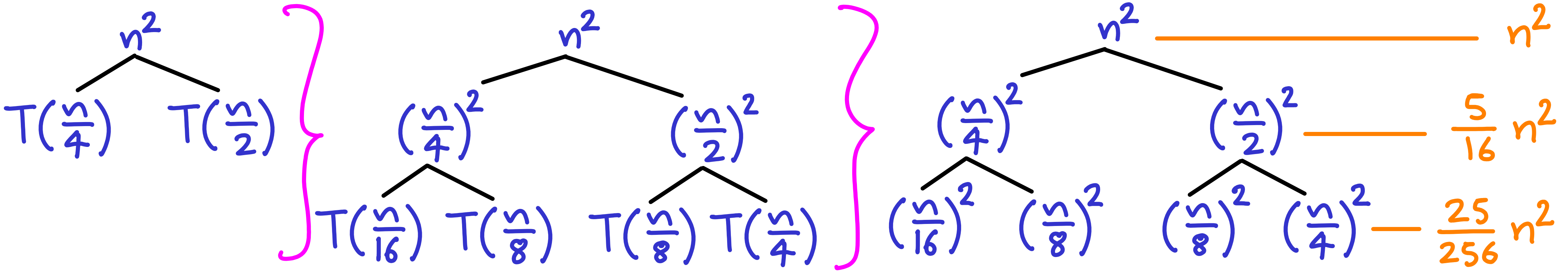
$$T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{n}{2}\right) + n^2$$



$$T(n) \le n^2 \cdot \left[1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots + \left(\frac{5}{16}\right)^\infty\right]$$



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



$$T(n) \leq n^2 \cdot \left[1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots + \left(\frac{5}{16}\right)^\infty\right]$$

$$< n^2 \cdot \left[1 + \frac{1}{2} + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^3 + \cdots\right] = 2n^2$$