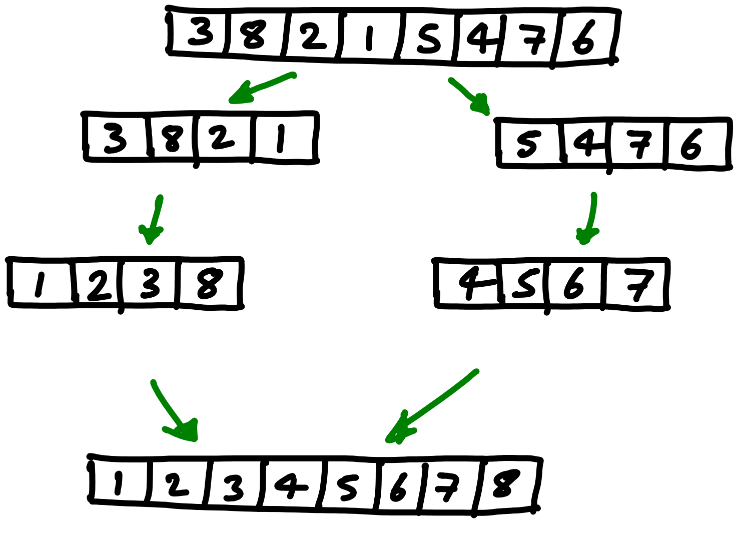
Back to Sorting : Merge Sort

a Divide-and-Conquer algorithm.

1) Divide the problem into 2 smaller instances.

2) "Conquer" = solve the smaller problems

3) Combine = merge the solutions



Merge two sorted arrays

A:									
1	2	3	5	8	9	14	16	17	18

B:									
4	6	7	10	11	12	13	15	19	20

Smallest # is at beginning of A or B : $\underset{A}{1}$ v. 4

Increment index in A $\underset{A}{2}$ v 4

Increment A $\underset{A}{3}$ v 4

Increment A 5 v $\underset{B}{4}$

Increment B $\underset{A}{5}$ v 6

etc

each time store smallest in a new array

$\Theta(n)$ time

Merge Sort time complexity $= T(n)$

1) Divide : $\Theta(1)$ time (identify index of split)

2) Conquer : $\Theta(1) + 2T(\frac{n}{2})$ to make 2 recursive calls

3) Merge : $\Theta(n)$

time = ?

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

Actually $T(n) = T\left(\left\lceil \frac{n}{2} \right\rceil\right) + T\left(n - \left\lceil \frac{n}{2} \right\rceil\right) + \Theta(n)$

this is just a detail

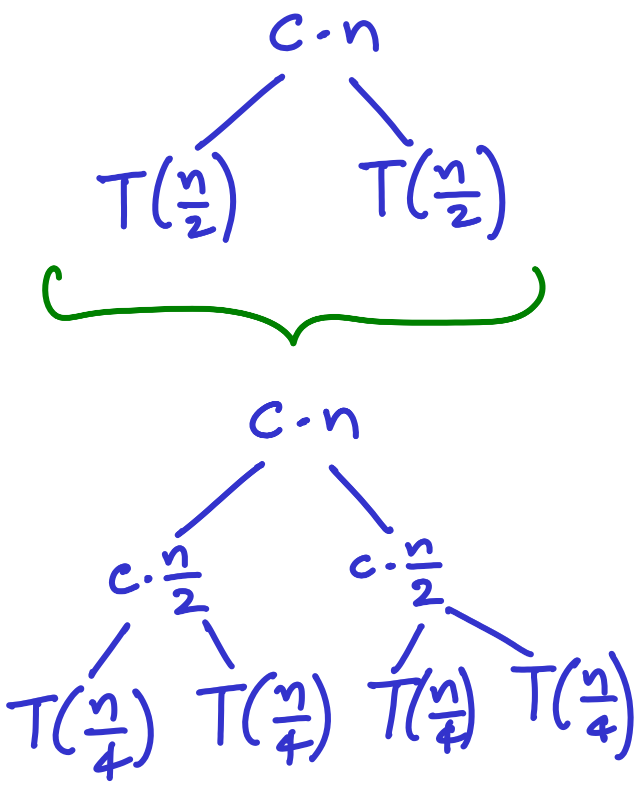
$$T(1) = \Theta(1)$$

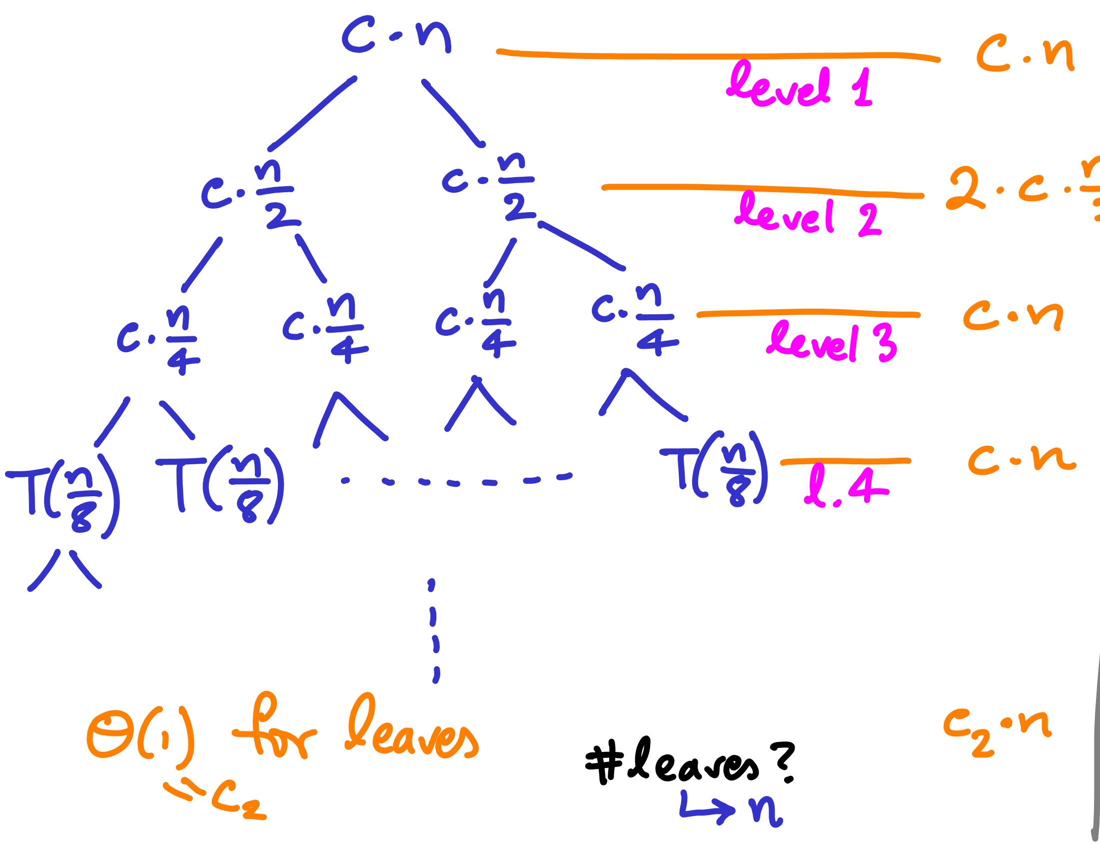
How to solve $T(n) = 2T(\frac{n}{2}) + \Theta(n)$ $[T(1) = \Theta(1)]$

The intuitive recursion tree:

first specify $\Theta(n) \to c \cdot n$

i.e. $T(n) = 2T(\frac{n}{2}) + c \cdot n$





$\underbrace{\#\text{levels} = \log_2 n}$

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

$cn\log n + c_2 n$

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

The more formal <u>substitution method</u>

Just focus on upper bound

Start by guessing the answer. Maybe $O(n\log n)$?

Use induction: assume that for $k<n$ $T(k) \le c \cdot k\log k$.

Substitute: $T(n) \le 2 \cdot c \cdot \frac{n}{2} \log \frac{n}{2} + \Theta(n)$

$$\begin{aligned}
&\le c \cdot n \cdot \log \frac{n}{2} + d \cdot n \\
&= c \cdot n\log n - c \cdot n \log 2 + d \cdot n \\
&= c \cdot n\log n - (c \cdot n - dn) \\
&= c \cdot n\log n - (c-d)n
\end{aligned}$$

For $\underline{c > d}$ we get $T(n) \le cn\log n$

done

In this case, you can get a lower bound in a similar way.

That is often not the case

RECURRENCES - SUBSTITUTION METHOD (guessing)

$T(n) = 4T\left(\frac{n}{2}\right) + n$ — twice the input → four times the work (sort of)

$T(1) = \underline{\Theta(1)}$ — & suggestions? → guess $O(n^2)$

(try building up from T(1) to find a pattern)

Try $O(n^3)$ first: assume $T(k) \le c \cdot k^3$ for $k < n$

$$T(n) \le 4 \cdot c \cdot \left(\frac{n}{2}\right)^3 + n = \frac{1}{2} \cdot c \cdot n^3 + n = c \cdot n^3 - \frac{1}{2} c n^3 + n = cn^3 - \underbrace{\left(\frac{1}{2} cn^3 - n\right)}_{\geq 0 \text{ if } c \geq 2}$$

Notice c depends on base case

... assume $T(1) \le c \cdot 1^3$

Pause for a minute: why assume $T(k) \leq c \cdot k^3$ instead of $T(k) = O(k^3)$

$\hookrightarrow T(n) \leq 4T\left(\frac{n}{2}\right) + n \leq 4 \cdot O\left(\left(\frac{n}{2}\right)^3\right) + n = O(n^3) + n = O(n^3)$!?

What about $T(n) = n$? Prove it's $O(1)$: Base case: $T(1) = \text{const} = O(1)$ ✓

Assume $T(k) = O(1)$

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

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

THIS IS INCORRECT

Don't use Big-O within induction proof

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

Assume $T(k) \leq c \cdot k^2$, so $T(n) \leq 4 \cdot c \cdot \left(\frac{n}{2}\right)^2 + n = c \cdot n^2 + \underbrace{n}_{\text{never negative!}}$

$cn^2 + n$ is $O(n^2)$ but we have commited to a constant, c.

$T(n) = c \cdot n^2 + n < c \cdot n^2 + n^2 = (c+1) \cdot n^2$

Not good enough

Assume $T(k) \leq c_1 \cdot k^2 - c_2 \cdot k$

$$T(n) \leq 4 \cdot \left(c_1 \cdot \left(\frac{n}{2}\right)^2 - c_2 \cdot \frac{n}{2}\right) + n$$

$$= c_1 \cdot n^2 - 2c_2 \cdot n + n$$

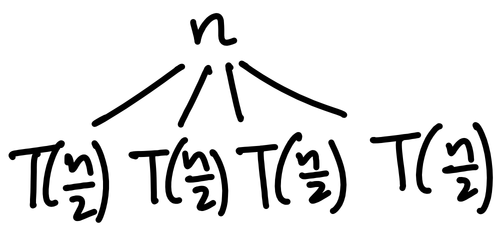
$$= \underbrace{c_1 \cdot n^2 - c_2 \cdot n}_{\text{same as } T(k)} \underbrace{- (c_2 - 1) \cdot n}_{<0 \text{ if } c_2 > 1}$$

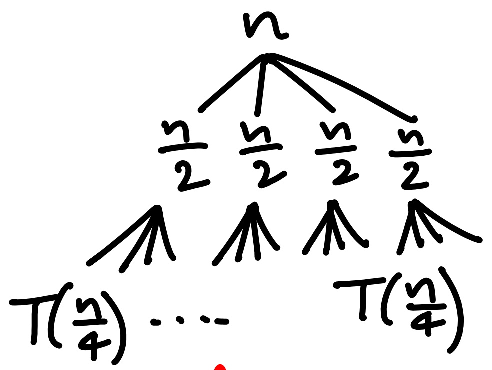
DONE

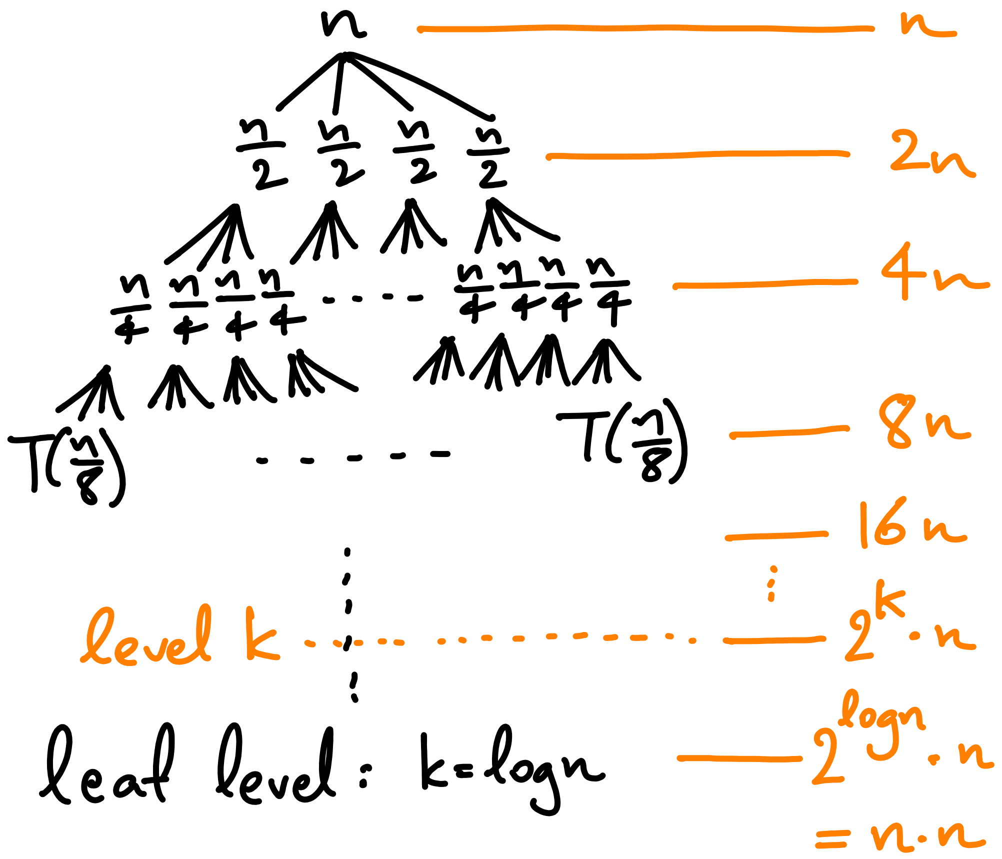
base case

$\underbrace{T(1) \leq c_1 - c_2}_{\text{forces a bound on } c_1}$

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



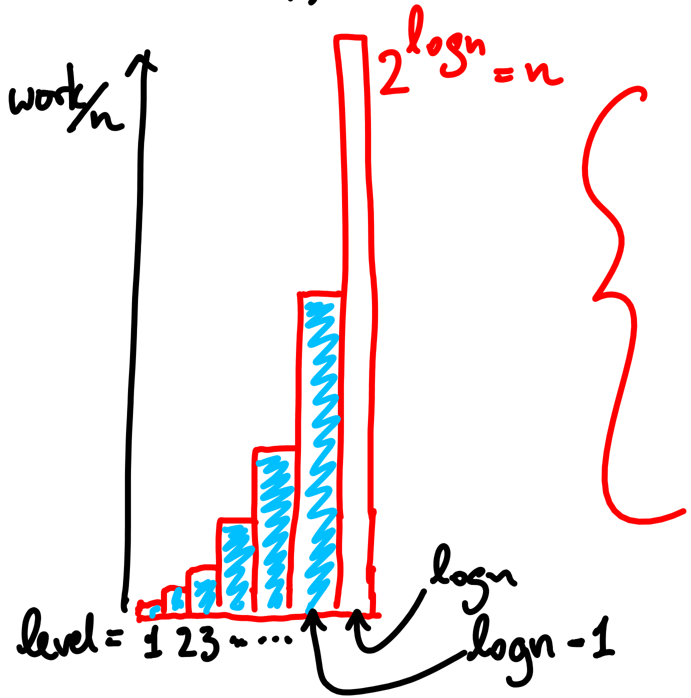




Doubling series

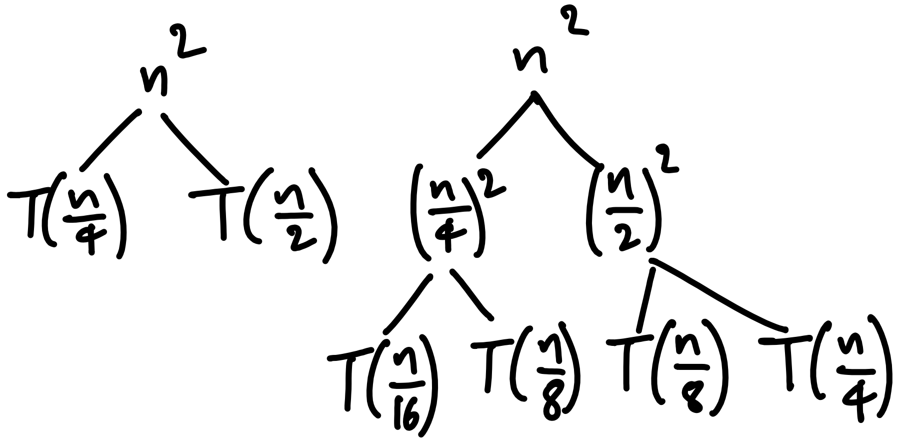
↓

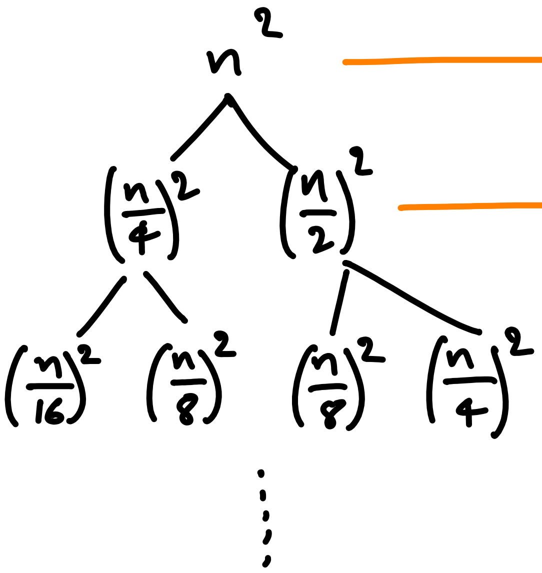
last term dominates



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

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





n^2

$\frac{5}{16} n^2$

$\frac{25}{256} n^2$

$\vdots$

$\left(\frac{5}{16}\right)^k \cdot n^2$

leaf level: asymmetric

deepest on right side... $< \left(\frac{5}{16}\right)^{\log n} \cdot n^2$

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

$$< n^2 \cdot \left[1 + \frac{1}{2} + \frac{1}{4} + \cdots + \left(\frac{1}{2}\right)^k + \cdots\right]$$

$$= n^2 \cdot 2 = O(n^2)$$

Verify w/ substitution