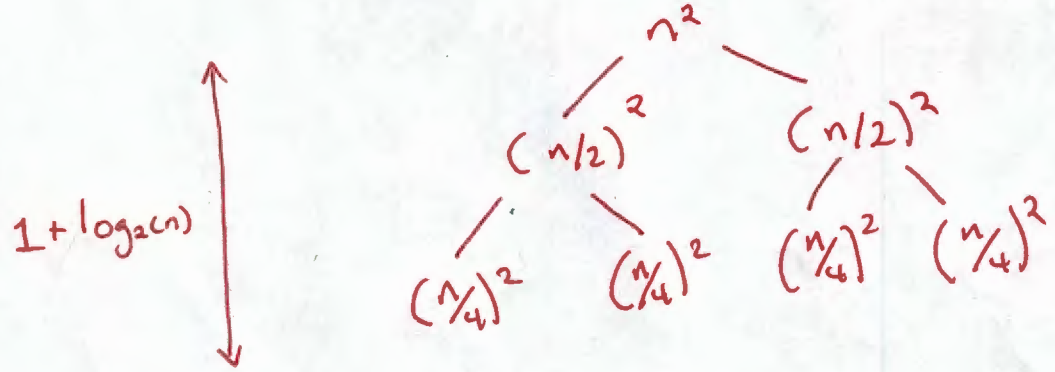
Master theorem / Lecture

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



$2(n/2)^2 = \frac{n^2}{2}$

$4\left(\frac{n}{4}\right)^2 \quad \frac{n^2}{4}$

$$\underbrace{\qquad\qquad}_{2^i\left(\frac{n}{2^i}\right)^2}$$

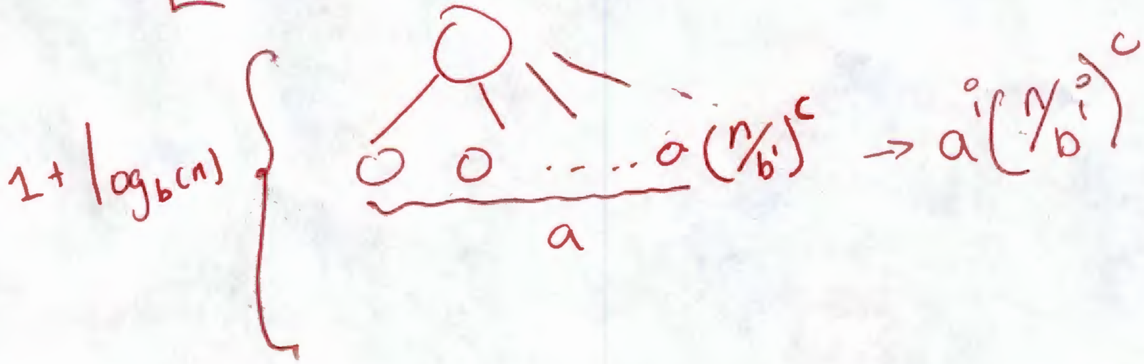
We can think of this more generally as

$\cancel{T(n) = aT(n/b) + f(n)}$ ← ~~The function was a square root function.~~

$$T(n) = \begin{cases} aT(n/b) + n^c & \text{if } n>1 \\ d & \text{if } n=1 \end{cases}$$

Include, Important for later

[Draw generic tree]

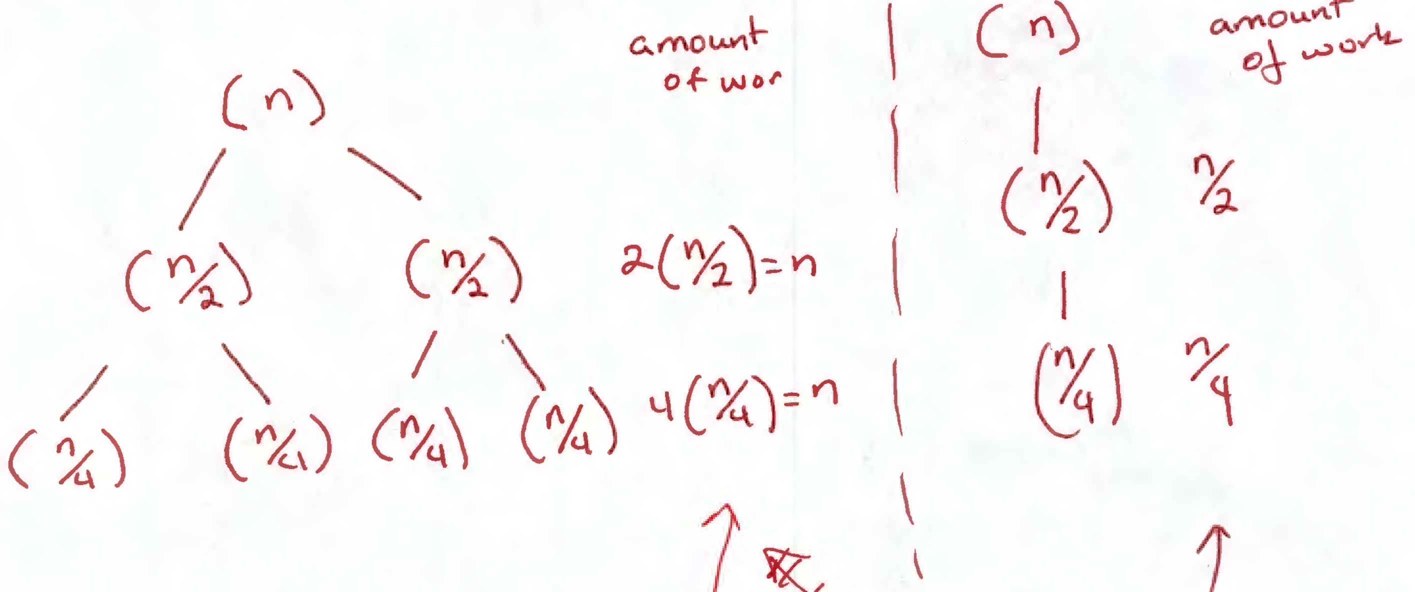


Let's compare the behaviour from the previous recurrence from last class

① $T(n) = 2T(n/2) + n$ ② $T(n) = T(n/2) + n$ ③ $T(n) = 4T(n/2) + n$

All tree have depth $\log_2(n) + 1$ because of the size of the subproblem & the assumption that @ $T(1) = 1$

Let's begin by comparing recurrence 1 & 2



Same amount of work @ each level

The amount of work decrease

Let focus on the second recurrence.

$$\overbrace{n + \frac{n}{2} + \frac{n}{4} \cdots\cdots 2 + 1}^{\log(n)+1}$$

$$n\underbrace{\left(1 + \frac{1}{2} + \frac{1}{4} \cdots\cdots \left(\frac{1}{2}\right)^{\log_2 n}\right)}_{\text{which is a constant.}}$$

Geometric series

Most work at leve

Let quick sanity check 5

~~5 + 4~~

$\cancel{5 + 5/2 + 5/4}$

$\log_2(8) = 3$

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

$8 + 4 + 2 + 1 = 8\left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right)$ ✓

So $n\underbrace{\left(1 + \frac{1}{2^1} + \frac{1}{2^2} \cdots \left(\frac{1}{2}\right)^{\log_2(n)}\right)}_{\text{~~a constant,~~ ~~because so small.~~}} = O(n)$

Dominated by largest term

~~Let~~ So first case

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

same amount of work @ each level

$n \log_2(n)$ ← merge sort

Second case

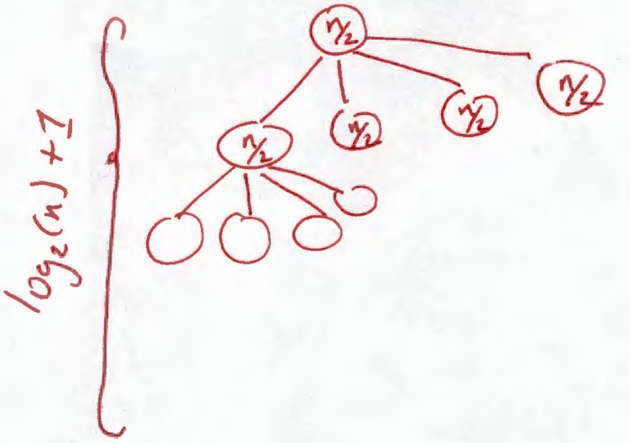
$T(n) = T(n/2) + n$

← Amount of work decreases @ each level

Let's consider the third case.

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

$$T(n) = 4\,T(n/2) + n$$



$$4^i \frac{n}{2^i} = 2^i n$$

Note as the problem size halves the number of node quadruples.

If we some the work done @ level 0 to $\log_2(n)$

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

$$\boxed{\sum_{i=0}^{n} 2^i = 2^{i+1} - 1}$$

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

$$= (n 2^1 - 1)\, n$$

$$= 2n^2 - n$$

$$\sim 2n^2$$

$$= O(n^2)$$

$T\cancel{(n) = O(n)^2}$

Let's consider the general form of the

$$T(n) = a\,T(n/2) + n$$

1. if $a < 2 \Rightarrow T(n) = O(n)$
2. $a = 2 \Rightarrow T(n) = O(n \log n)$ ← merge sort analysis
3. $a > 2 \Rightarrow T(n) = O(n^{\log_2 a})$

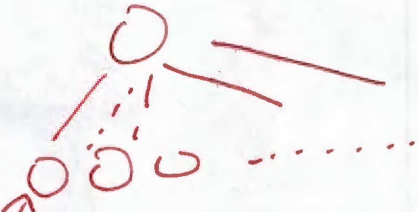
wait a second.

I get the n^2 case

but. - - - - - - -

Let's discuss

$$a\,T(n/2) + n$$



a^i node at each level.

$n/2^i$ problem siz.

Total amount of work @ each level

$$\underbrace{a^i}_{\text{number of node}} \times \left(n/2^i\right) = n \times \left(\frac{a^i}{2^i}\right) = n\left(a/2\right)^i$$

← problem size.

$$n \sum_{0}^{\log_2(n)} \left(a/2\right)^i$$

$$n\left(\frac{a}{2}\right)^{\log_2(n)}$$

Dominate

by the largest

why is largest term the laston

$$n\left(\frac{a}{2}\right)^{0} = n$$

$$n\left(\frac{a}{2}\right)^{0} < n\left(\frac{a}{2}\right)^{\log_2(n)}$$

$$n\left(\frac{a}{2}\right)^{\log_2(n)} = \frac{n\, a^{\log_2(n)}}{2^{\log_2 n}}$$ Distribute the logs

$$= \frac{\cancel{n}\, a^{\log_2(n)}}{\cancel{n}}$$

$$= a^{\log_2(n)}$$ substitute. $\leftarrow$ $\left[a = 2^{\log_2 a}\right]$

$$= \left(2^{\log_2 a}\right)^{\log_2(n)}$$ erpon is multipcation

$$= 2^{\log_2 a \times \log_2(n)}$$

$$= 2^{\log_2(n)\log_2(a)}$$

$$= n^{\log_2(a)}$$

$$T(n) = a\,T(n/2) + n$$

$a<2$ $T(n) = O(n)$

$a=2$ $T(n) = O(n\log(n))$

$a>2$ $T(n)$ $O(n^{\log_2 a})$

The Master Theorem

$$T(n) = \begin{cases} aT(n/b) + n^c & \text{if } n > 1 \\ d & \text{if } n = 1 \end{cases}$$

1. $\log_b a < c \qquad T(n) = \Theta(n^c)$
2. $\log_b a = c \qquad T(n) = \Theta(n^c \log(n))$
3. $\log_b a > c \qquad T(n) = \Theta(n^{\log_b a})$

Let's prove <u>case 1</u>

$$aT(n/b) + n^c$$

There will be $\log_b n$ levels

and the sub problems will be multiplied by $\boxed{a^i}$

and the problem size will be $\boxed{n/b^i}$

and we will have $(n/b^i)^c$ additional work.

$$a^i\left(\frac{n}{b^i}\right)^c = n^c\left(\frac{a^i}{b^i}\right)$$

$$= n^c\left(\frac{a}{b}\right)^i$$

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

$$= n^c \underbrace{\sum_{i=0}^{\log_b N}\left(\frac{a^i}{b^i}\right)}_{\text{another geometric series}}$$

$$= \Theta(n^c)$$

Apply the Master theorems.

Case ~~1~~ 3

$\log_b a > c$ $\quad T(n) = \Theta(n^{\log_b a})$

Consider this recurrence.

$$T(n) = 8T(n/2) + 1000n^2$$

$a = 8 \quad b = 2 \quad f(n) = 1000n^2$

$f(n) = O(n^c)$ where $c = 2$

$\log_b a = \log_2 8 = 3 > c \quad [3 > 2]$

$T(n) = \Theta(n^{\log_b a}) = O(n^3)$

Turn out that the exact solution

$\Theta(n^3) = 2000n^3 - 1000n^2$ assuming $T(1) = 1$

So, it works

not for f(x).

Case 2

$$T(n) = 2T(n/2) + \cancel{\Theta(n)}\ 10n$$

$a=2 \quad b=2 \quad c=1$

$\log_b a = c \qquad T(n) = \Theta(n^c \log(n))$

$\log_2 2 = 1 \iff c=1$

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

Solving this recurence

$$T(n) = n + 10n\log_2(n)$$
$$= \Theta(n\log_2(n))$$

Case 1

$\log_b a > c \qquad T(n) = \Theta(n^{\log_b a})$

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

$a = 2 \qquad b = 2 \qquad f(n) = n^2$

$f(n) = (n^c)$

$c = 2$

$\log_b a = \log_2 2 = 1$

$T(n) = \Theta(n^{\log_2 2})$

$= \Theta(n^2)$

$T(n) = 2n^2 - n$

~~Quiz~~

4100 + 02

4102

1800 + 19

1819

Karatsuba Multiplication >

multiply two number together.

$$
\begin{array}{r}
4102 \\
\times\ 1819 \\
\hline
36918 \\
4102 \\
32816 \\
4102 \\
\hline
7461538
\end{array}
$$

n multiples } n levels $\Rightarrow O(n^2)$

Can we do better.

$$
\begin{array}{r}
\overset{a}{\boxed{41}}\,\overset{b}{\boxed{02}} \\
\times\ c\ \boxed{18}\,\underset{d}{\boxed{19}} \\
\hline
\end{array}
$$

$= 10^2 \overset{a}{\boxed{41}} + \overset{b}{\boxed{02}}$

$= 10^2 \underset{c}{\boxed{18}} + \boxed{19}$ -d

equivalent.

$$(10^2 a + b) \times (10^2 c + d)$$

let n represent the number of digits.

expand $(10^{n/2} a + b) \times (10^{n/2} c + d)$

n was 4

$n/2 = 2$

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

Divide:

① Break n-digit numbers into
4 number $n/2$ digits each.

② Conquer.

if $n > 1$
Recursively compute ac, ad, bc, bd.

If $n = 1$
Compute ac, ad, bc, bd.
(base case) → fail recursion.

③ $10^n(ac) + 10^{n/2}(ad+bc) + bd$ an
↑ 5

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

4 recursive calls

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

Solve using Master theorem*

$a=4 \quad b=2 \quad c=1$

$\log_2 4 > 1$
$2 > 1$
Case 3 $\log_b a > c$
$T(n) = \Theta(n^{\log_b a})$

$T(n) = O(n^{\log_b a}) = \Theta(n^2)$ ← no better !!

What if we could decrease the number of multiplication

a	b
c	D

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

what if we write these so that ~~Share~~ uses the result computed in other steps

clever trick $\left\{ \underbrace{ad + bc}_{\text{2 multiplicate} \downarrow \text{additional}} = \underbrace{(a+b)(c+d)}_{\text{one multiplication}} - \underbrace{ac - bd}_{\text{already computed}} \right.$

New recurrence

$$T(n) = 3T(n/2) + 8n$$ $\boxed{\text{4 exta}}$

$a = 3 \quad b = 2 \quad c = 1$

$\log_2 3 > 1$

Case 3
$\log_b a > c$
$T(n) = \Theta(n^{\log_b a})$
$= \Theta(n^{\log_2 3})$