5 March 2025

Mathematical Algorithms: Karatsuba Multiplication

Plan

* Multiplication
* Announcements
* Karatsuba
* Practice w/ Recurrences.

Multiplying Integers

$$p \times q = N$$

Given two integers p, q
Find their product N.

Multiplying Integers

$$p \times q = N$$

Given two integers p, q
Find their product N.

Questions.

* Isn't this just a math problem?
* Isn't this $O(1)$?

Grade school multiplication

$$4820 \times 905 = ?$$

$$\begin{array}{r} 4820 \\ \times\ \ 905 \\ \hline \end{array}$$

Grade school multiplication

$$4820 \times 905 = ?$$

```
   7 1
   4820
 x   915
 -------
  24100
 48200
4338000
-------
4410300
```

As the integers grow in magnitude, how many more basic operations do we use?

Grade school multiplication

$$4820 \times 905 = ?$$

```
   7 1
   4820
 *  915
 ------
  24100
 48200
4338000
-------
4410300
```

As the integers grow in magnitude, how many more basic operations do we use?

Basic Operations

* Multiplication of bits
* Addition of bits

Representing Natural Numbers

Binary Representation

$$4820 = 1001011010100$$

$$p = \sum_{i=0}^{n} 2^i \cdot p_i = p_n p_{n-1} \cdots p_2 p_1 p_0$$

for $p_i \in \{0, 1\}$

Representing Natural Numbers

Binary Representation

$$4820 = 1001011010100$$

$$p = \sum_{i=0}^{n} 2^i \cdot p_i = p_n p_{n-1} \cdots p_2 p_1 p_0$$

for $p_i \in \{0, 1\}$

Input size: length n of binary representation of p, q.

How does multiplication scale w/ input length?

$p_n p_{n-1} \cdots p_1 p_0$

$q_m q_{m-1} \cdots q_1 q_0$

Grade School Multiplication:

For every $j = 0, 1, \ldots, m$

  For every $i = 0, 1, \ldots, n$

    Record $q_j \cdot p_i$

Sum up totals

$\Theta(n^2)$ time for $m = \Theta(n)$.

Can we do better?

Announcements

* HW3 due
* HW4 released after lecture

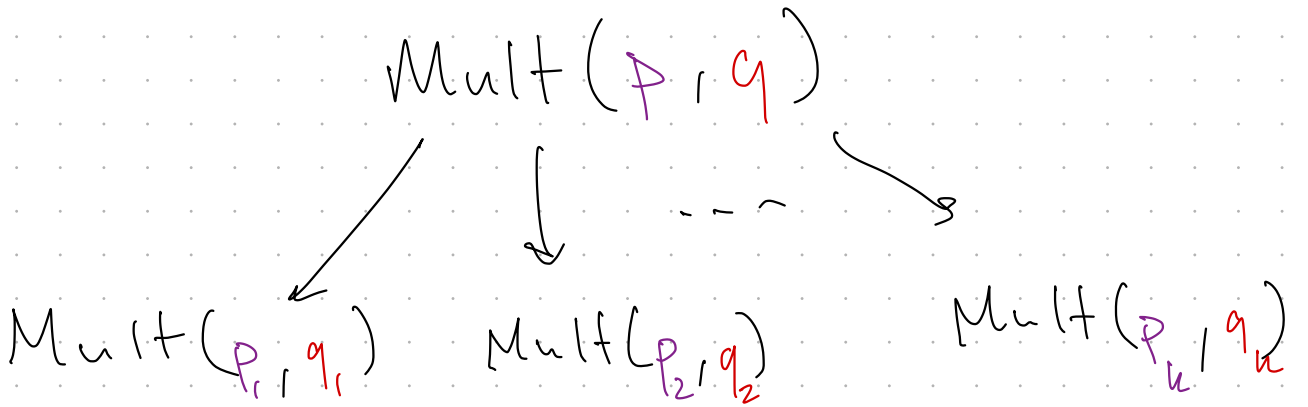
Divide & Recurse.

* To compute multiplication of n-bit numbers
  - ↳ Express product as sum of multiplication of smaller (fewer bits) numbers.

Divide & Recurse.

* To compute multiplication of n-bit numbers

  ↳ Express product as sum of multiplication of smaller (fewer bits) numbers.



where # bits in p_i q_i smaller!

Split numbers into high & low order bits

$$p = \underbrace{p_n p_{n-1} \cdots p_{n/2}}_{p_h} \underbrace{p_{n/2-1} \cdots p_1 p_0}_{p_\ell}$$

← Both represent numbers in $\{0, 1, \ldots, 2^{n/2}-1\}$

How many bits?

Split numbers into high & low order bits

$$p = p_n p_{n-1} \cdots p_{n/2} \; p_{n/2-1} \cdots p_1 p_0$$

p_h $\qquad$ p_ℓ

← Both represent numbers in $\{0, 1, \ldots, 2^{n/2}-1\}$

$$q = q_n q_{n-1} \cdots q_{n/2} \; q_{n/2-1} \cdots q_1 q_0$$

q_h $\qquad$ q_ℓ

Length drops by factor 2

Split numbers into high & low order bits

$$p = p_n p_{n-1} \cdots p_{n/2} \; p_{n/2-1} \cdots p_1 p_0$$

p_h $\qquad$ p_ℓ

← Both represent numbers in $\{0, 1, \ldots, 2^{n/2}-1\}$

$$q = q_n q_{n-1} \cdots q_{n/2} \; q_{n/2-1} \cdots q_1 q_0$$

q_h $\qquad$ q_ℓ

FOIL Identity $\qquad (ax + b)(cx + d)$

$$= acx^2 + (ad + bc)x + bd$$

Split numbers into high & low order bits

$$p = \underbrace{p_n p_{n-1} \cdots p_{n/2}}_{p_h} \underbrace{p_{n/2-1} \cdots p_1 p_0}_{p_\ell}$$

← Both represent numbers in $\{0, 1, \ldots, 2^{n/2}-1\}$

$$q = \underbrace{q_n q_{n-1} \cdots q_{n/2}}_{q_h} \underbrace{q_{n/2-1} \cdots q_1 q_0}_{q_\ell}$$

FOIL Identity $\quad (ax + b)(cx + d)$

$$= acx^2 + (ad + bc)x + bd$$

Consider $x = 2^{n/2}$

$$p = p_h 2^{n/2} + p_\ell \qquad q = q_h 2^{n/2} + q_\ell$$

$$p = p_n p_{n-1} \cdots p_{n/2} \; p_{n/2-1} \cdots p_1 p_0 \qquad q = q_n q_{n-1} \cdots q_{n/2} \; q_{n/2-1} \cdots q_1 q_0$$

p_h p_ℓ q_h q_ℓ

$$p \times q = \left(p_h 2^{n/2} + p_\ell\right)\left(q_h 2^{n/2} + q_\ell\right)$$

$$= p_h q_h 2^n + \left(p_h q_\ell + p_\ell q_h\right) 2^{n/2} + p_\ell q_\ell$$

$p = p_n p_{n-1} \cdots p_{n/2} \; p_{n/2-1} \cdots p_1 p_0$ $\qquad$ $q = q_n q_{n-1} \cdots q_{n/2} \; q_{n/2-1} \cdots q_1 q_0$

$p_h \qquad p_\ell \qquad\qquad q_h \qquad q_\ell$

$$p \cdot q = \left(p_h 2^{n/2} + p_\ell\right)\left(q_h 2^{n/2} + q_\ell\right)$$

$$= p_h q_h 2^n + \left(p_h q_\ell + p_\ell q_h\right) 2^{n/2} + p_\ell q_\ell$$

Note. Multiplication by 2^j for $j \in \mathbb{N}$ is bit shift

$$p \times q = \left(p_h 2^{n/2} + p_\ell\right)\left(q_h 2^{n/2} + q_\ell\right)$$

$$= p_h q_h 2^n + \left(p_h q_\ell + p_\ell q_h\right) 2^{n/2} + p_\ell q_\ell$$

Recursive-Multiply (p, q) // Base case: single bits.

* Split into $p_h p_\ell$, $q_h q_\ell$
* Return $RM(p_h, q_h) \cdot 2^n + \left(RM(p_h, q_\ell) + RM(p_\ell, q_h)\right) \cdot 2^{n/2} + RM(p_\ell, q_\ell)$

$$p \times q = \left(p_h 2^{n/2} + p_\ell\right)\left(q_h 2^{n/2} + q_\ell\right)$$

$$= p_h q_h 2^n + \left(p_h q_\ell + p_\ell q_h\right) 2^{n/2} + p_\ell q_\ell$$

Recursive-Multiply (p, q) // Base case: single bits.

* Split into $p_h p_\ell$, $q_h q_\ell$
* Return $RM(p_h, q_h) \cdot 2^n + \left(RM(p_h, q_\ell) + RM(p_\ell, q_h)\right) \cdot 2^{n/2} + RM(p_\ell, q_\ell)$

Running Time?

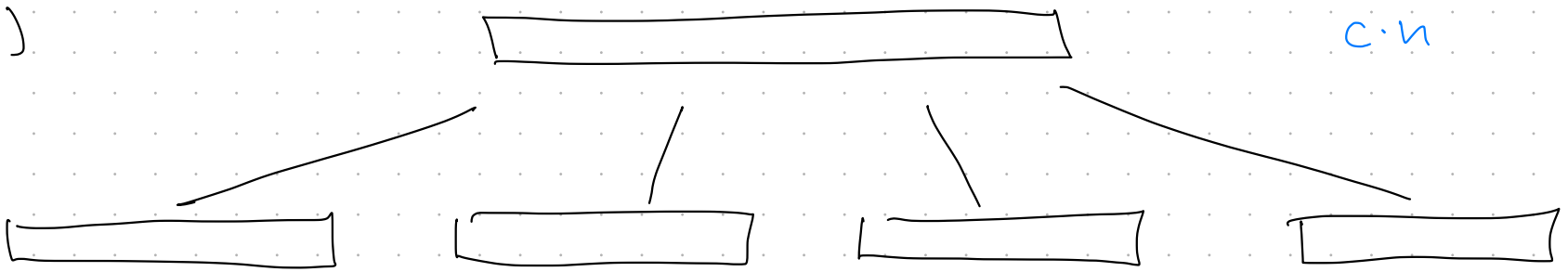
$$T(n) \leq 4 \cdot T(n/2) + c \cdot n$$

- 4: # subprobs.
- $n/2$: scale of subprobs.
- $c \cdot n$: bit shifts & additions of $2n$-bit #s

Recurrence Relations

* Tally up total work over Recursive Calls.

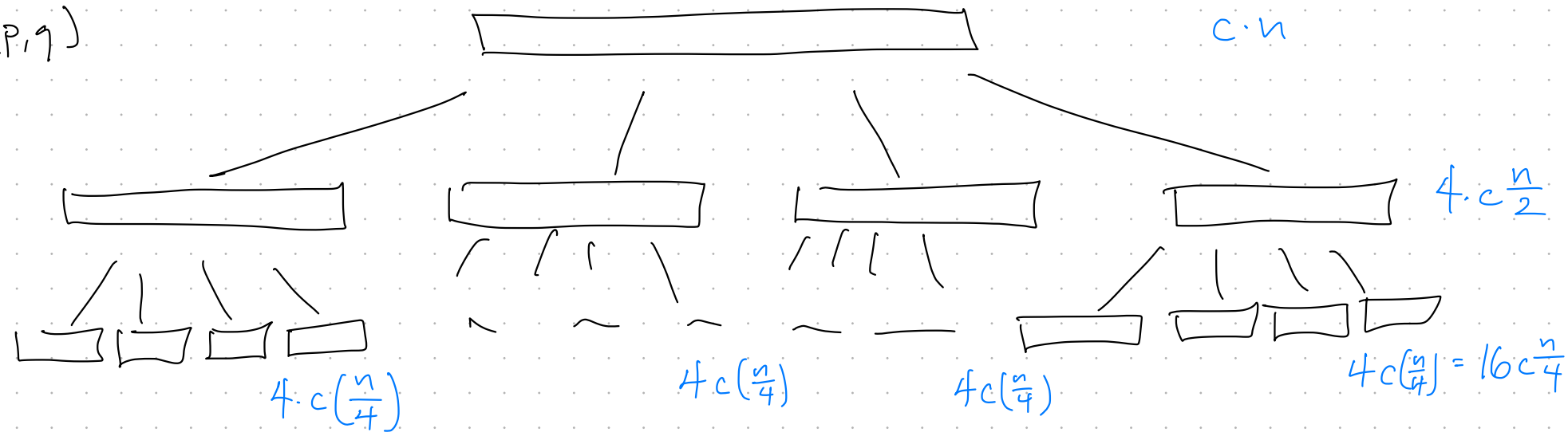
RM(p,q)



Recurrence Relations

* Tally up total work over Recursive Calls.

RM(p,q)



How many levels?

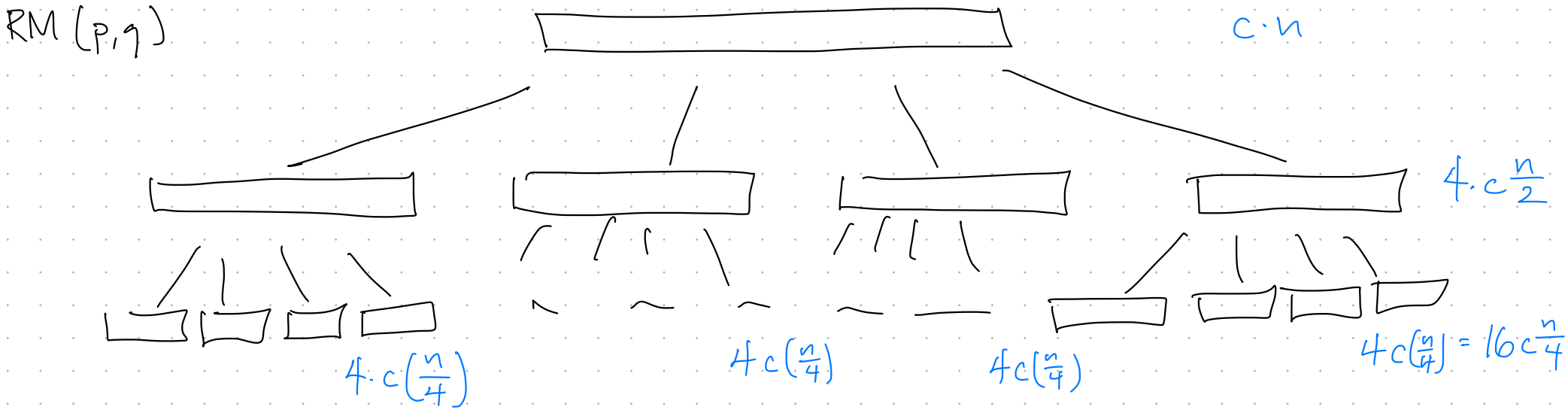
$d = \log_2 n$

How much work per level i?

$cn \cdot 4^i \cdot \frac{1}{2^i} = cn \cdot 2^i$

Recurrence Relations

* Tally up total work over Recursive Calls.



Sum over levels

$$\sum_{i=0}^{d=\log n} cn\,2^i = cn \cdot \Theta(2^d)$$

$$= \Theta(n^2)$$

Original Recurrence

$$(ax + b)(cx + d)$$

$$= acx^2 + (ad + bc)x + bd$$

Karatsuba Recurrence

Consider $w = (a+b)(c+d) = ac + bc + ad + bd$

Original Recurrence

$$(ax + b)(cx + d)$$

$$= acx^2 + (ad + bc)x + bd$$

Karatsuba Recurrence

Consider $w = (a+b)(c+d) = ac + bc + ad + bd$

$$= acx^2 + (w - ac - bd)x + bd$$

$\Rightarrow$ Only Requires 3 multiplications

Karatsuba Multiply (p, q) // Base case: single bits

* Split into $p_h p_\ell \cdot q_h q_\ell$
* $w \leftarrow KM(p_h + p_\ell, q_h + q_\ell)$
* $h \leftarrow KM(p_h, q_h)$
* $\ell \leftarrow KM(p_\ell, q_\ell)$

Return $h \cdot 2^n + (w - h - \ell) \cdot 2^{n/2} + \ell$

Karatsuba Multiply (p, q) // Base case: single bits

* Split into $p_h p_\ell \cdot q_h q_\ell$
* $w \leftarrow KM(p_h + p_\ell, q_h + q_\ell)$
* $h \leftarrow KM(p_h, q_h)$
* $\ell \leftarrow KM(p_\ell, q_\ell)$

Return $h \cdot 2^n + (w - h - \ell) \cdot 2^{n/2} + \ell$

Running Time

$$T(n) \leq 3 \cdot T(n/2) + c \cdot n$$

$$T(n) \leq 3 \cdot T(n/2) + c \cdot n$$

levels

Work @ level i

$$T(n) \leq 3 \cdot T(n/2) + c \cdot n$$

levels

$$1 = n \cdot (1/2)^d \Rightarrow d = \log_2 n$$

base case

shrinkage rate

Work @ level i

problems: 3^i

Size of problems: $n/2^i$

$\Big\} \; n \cdot (3/2)^i$

Total work.

$$T(n) \leq 3 \cdot T(n/2) + c \cdot n$$

levels

$$1 = n \cdot (1/2)^d \Rightarrow d = \log_2 n$$

base case

shrinkage rate

Work @ level i

problems: 3^i

Size of problems: $n/2^i$

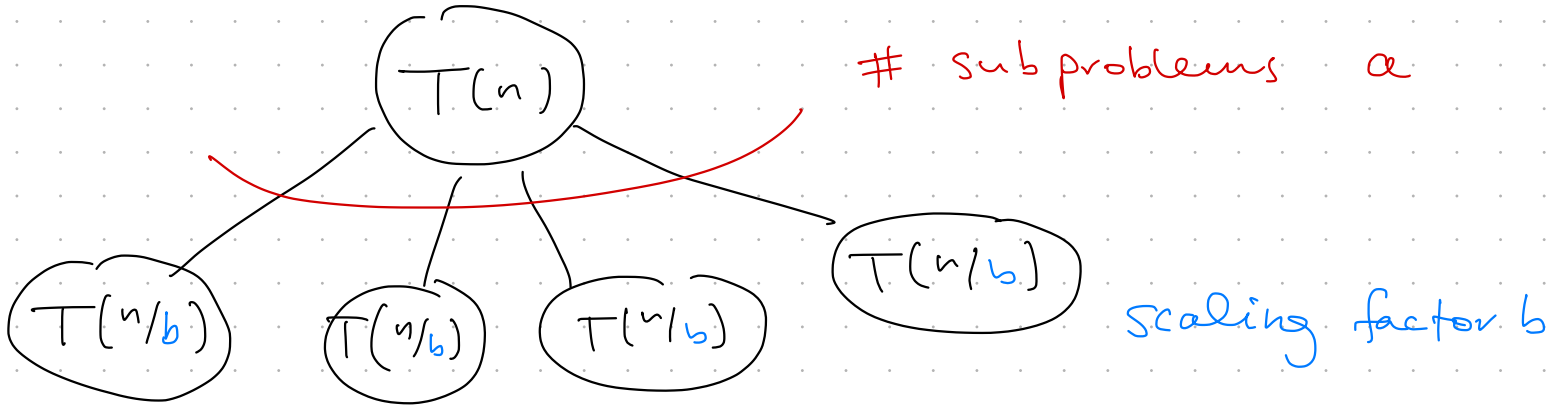
$\Big\}$ $n \cdot (3/2)^i$

Total work.

$$c \cdot n \cdot \sum_{i=0}^{\log_2 n} \left(\frac{3}{2}\right)^i \leq O(n^{\log_2 3})$$

$$\leq O(n^{1.59})$$

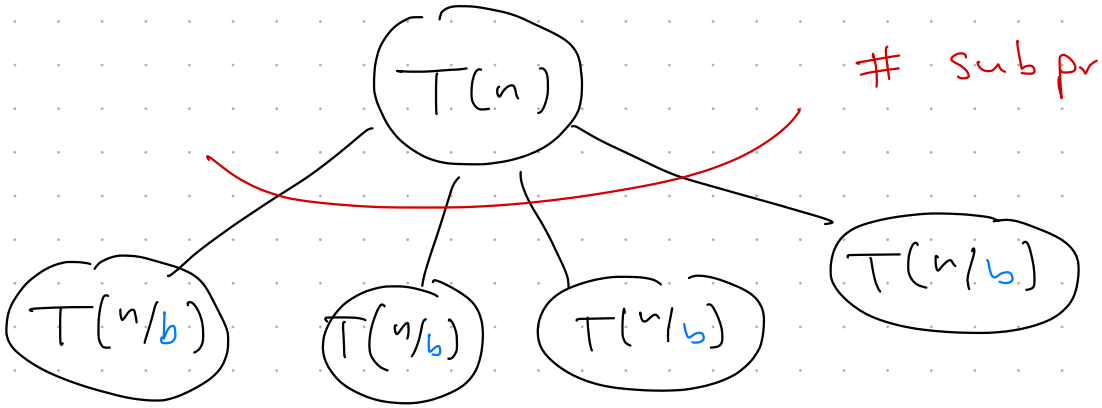
Divide & Recurse Running Times



Common Recurrence

$T(n) \leq 2 \cdot T(n/2) + O(n)$ e.g. Merge Sort

Divide & Recurse Running Times



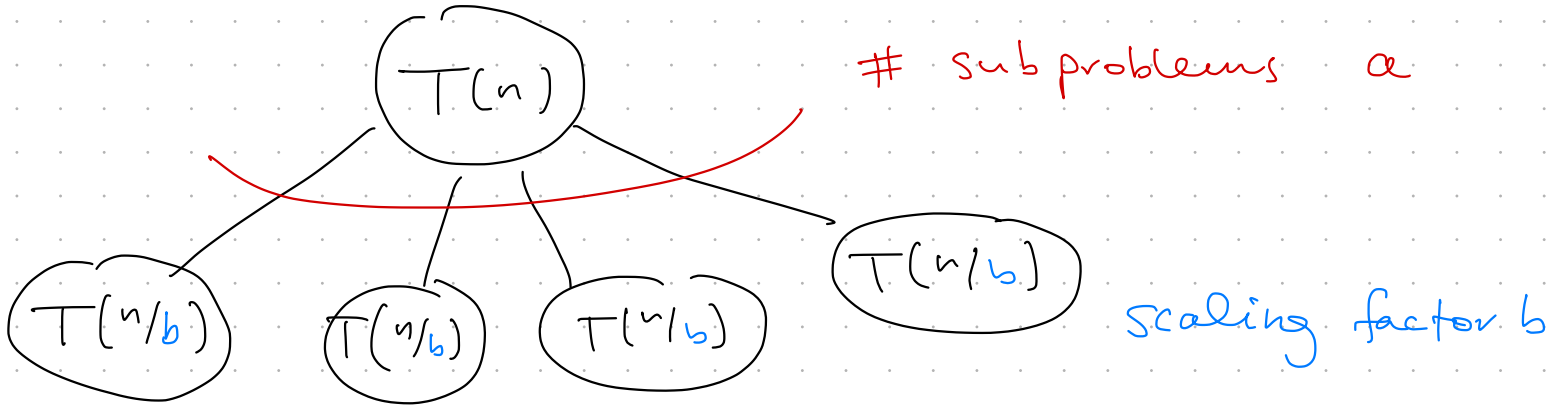
Common Recurrence

$T(n) \leq 2 \cdot T(n/2) + O(n)$ e.g. Merge Sort

* $\Theta(n)$ work per level
* $\Theta(\log n)$ levels

↳ $O(n \log n)$

Divide & Recurse Running Times

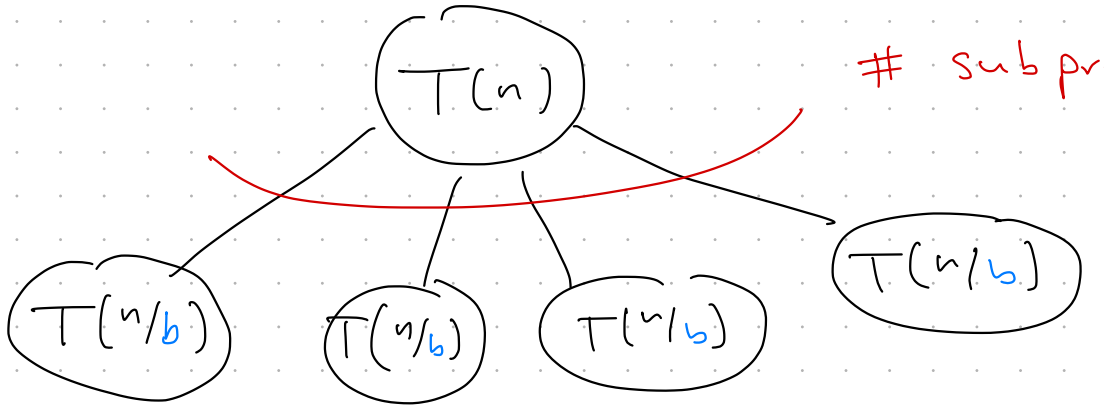


Common Recurrence

$$T(n) \leq a \cdot T(n/b) + O(n)$$

Suppose $b > a$

Divide & Recurse Running Times



Common Recurrence

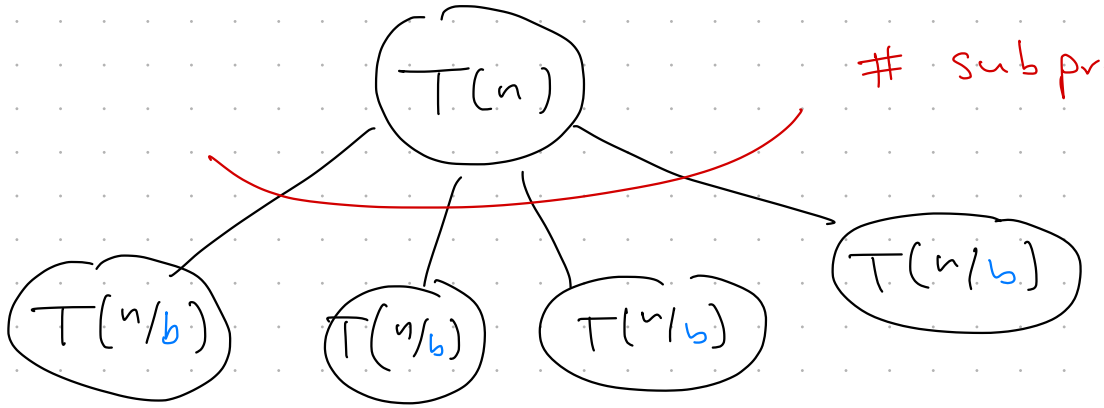
$$T(n) \leq a \cdot T(n/b) + O(n)$$

Suppose $b > a$

$\Rightarrow$ Work per level shrinks at an exponential $(\frac{a}{b})^d$ rate.

$\Rightarrow O(n)$

Divide & Recurse Running Times



Common Recurrence

$$T(n) \leq a \cdot T(n/b) + O(n)$$

Suppose $b < a$

$\Rightarrow$ Work per level grows at an exponential $(\frac{a}{b})^d$ rate.

Geometric Sum & change of basis for logs $\Rightarrow$ $O(n^{\log_b a})$