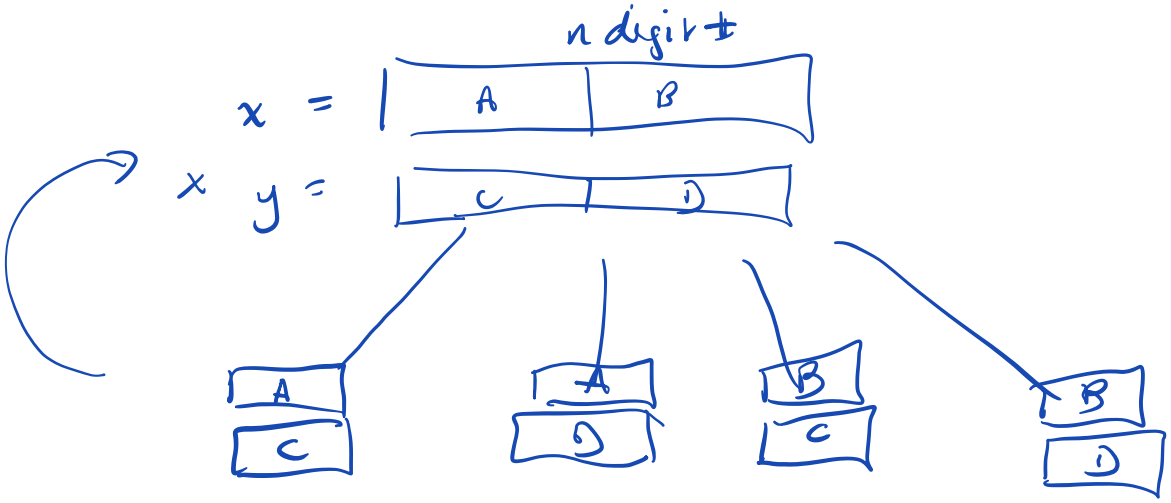
Divide and conquer: Sorting + median finding

- Hw 1 due today
- Hw 2 out today or tomorrow
- First quiz is Feb 3
- Weekly checkins "quizzes" Sat due Mon
- Recitation Fri 3-5 Karthik if needed

Last time multiplication



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

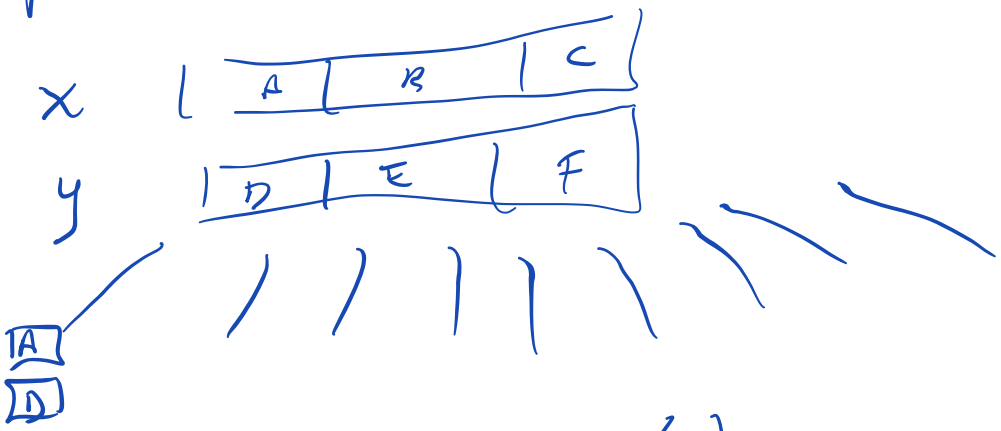
$$\Rightarrow T(n) = O\left(n^{\log_2 4}\right) = O(n^2)$$

Karatsuba actually...

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

$$\Rightarrow T(n) = O\left(n^{\log_2 3}\right) = O(n^{1.59})$$

Other partitions?



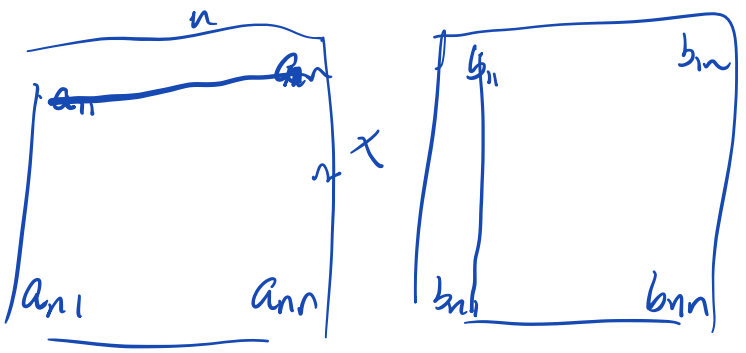
$$T(n) = 9T\left(\frac{n}{3}\right) + O(n)$$

$$\Rightarrow T(n) = O(n^{\log_3 9}) = O(n^2)$$

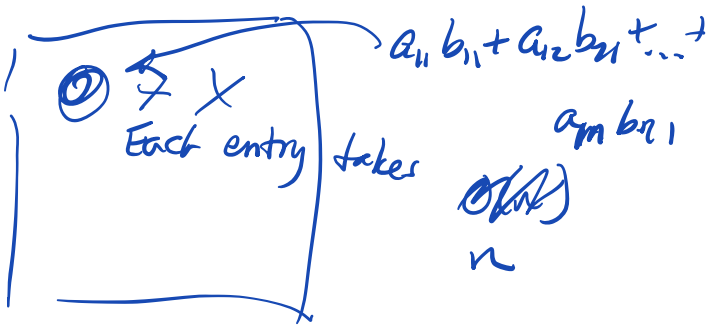
If 8 subprob $O(n^{\log_3 8}) = O(n^{1.\sim})$

Matrix Multiplication

How many integer multiplications?



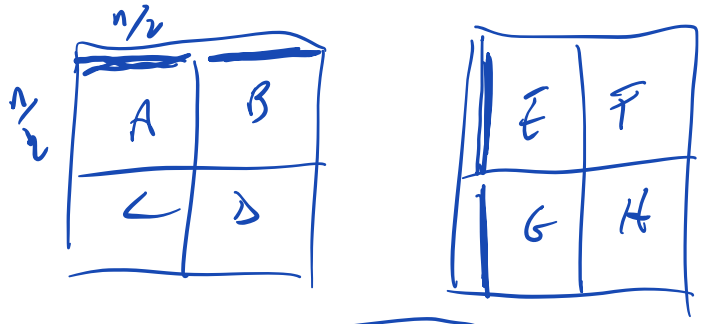
11th grade alg



$a_{11}b_{11} + a_{12}b_{21} + \dots + a_{1n}b_{n1}$

$n \times \#\text{ of entries} = n^3$

Divide + conquer



$AE+BG$ | $AF+BH$

$CE+DG$ | $CF+DH$

time to $\times$ 2 $n \times n$ matrices

This works!

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

D&C approach

$$T(n) = O\left(n^{\log_2 8}\right) = O(n^3)$$

If 7 subproblems:

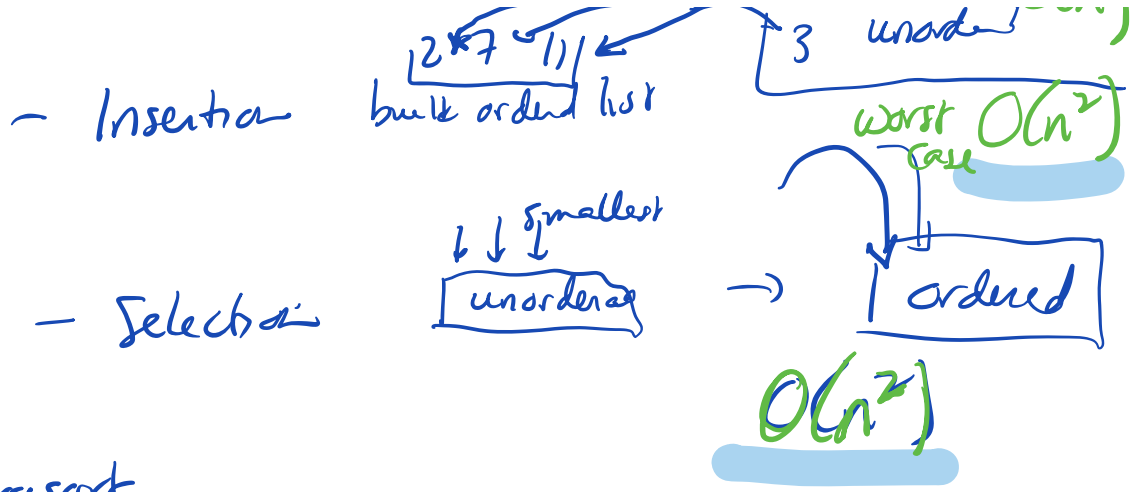
$$T(n) = O\left(n^{\log_2 7}\right) = O(n^{2.\ldots})$$

Sorting

- Mergesort $O(n \log n)$

- Quicksort

best $O(n)$



Mergesort

Input: an array $a[1,\ldots n]$ of #s

Output: a sorted array with those #s

$n=2^k$

```
If n>1
    return merge(mergesort(a[1,..,n/2]),
                 mergesort(a[n/2+1,...,n]))
Else
  return a
```

merge ($X[1,\ldots k]$, $Y[1,\ldots l]$)

```
if k=0    return Y[1,.. l]
if l=0    return X[1,..,k]
if X[1] ≤ Y[1]
        return X[1] ∘ merge(X[2,..k] Y[1,..,l])
else return Y[1] ∘ merge(X[1,.. k] Y[2,..l])
```

Analysis

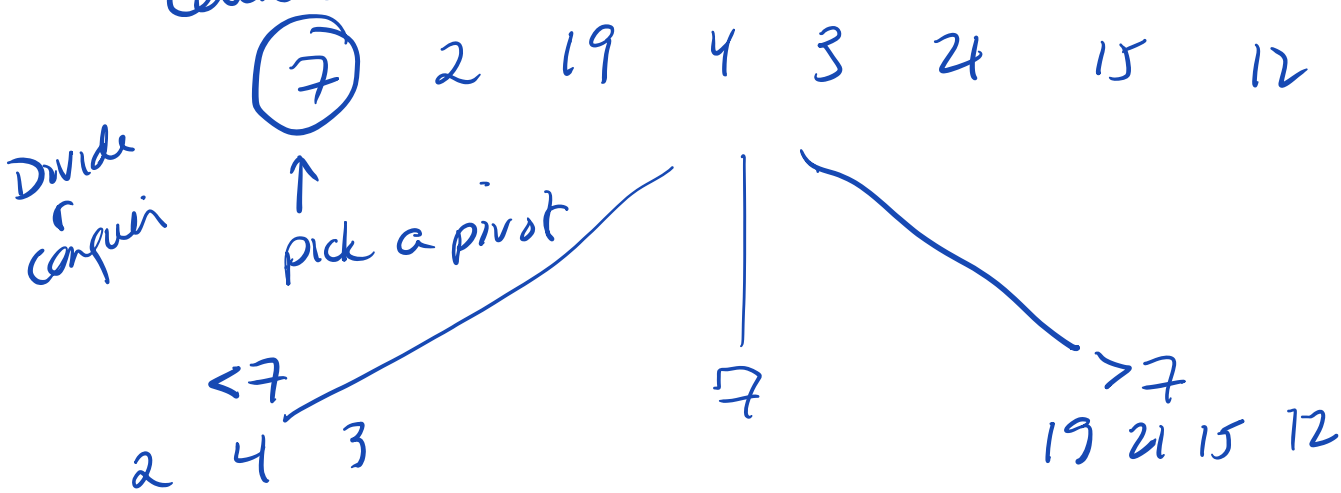
$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

$\Rightarrow O(n \log n)$ avg

Best case : $O(n \log n)$

Worst case : $O(n \log n)$

Quicksort



If we're lucky

If we always get pretty good pivots

$T(n) = T(\text{First piece}) + T(\text{second piece}) + O(n)$

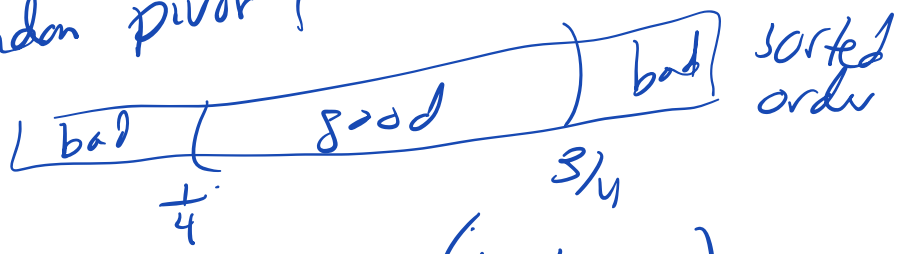
$T(n) = O(n \log n)$

If pieces are always about half

If pivots are not good
this can take $O(n^2)$

Pivot? 1st element might be bad

Random pivot?



Expected time $O(n \log n)$

Chance of 10 bad pivots in a row

$$< \frac{1}{2^{10}} \sim \frac{1}{1000}$$

What if we could always get a pivot?
What if we could always get the best pivot?

in $O(n)$ time

This is the Median!

Why can't we do better than $O(n \log n)$?

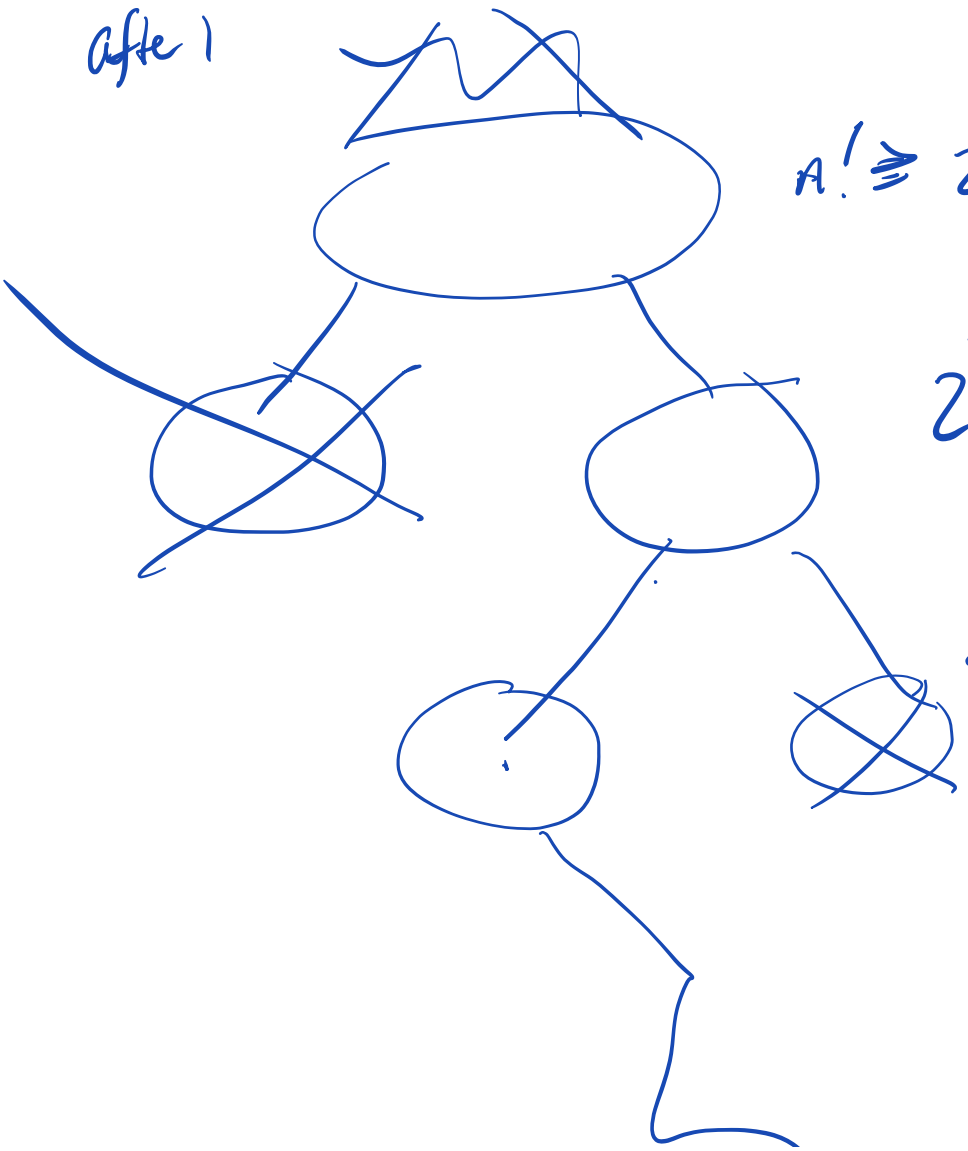
permutations: $n!$

$$n! = 1 \cdot 2 \cdot 3 \cdots \frac{n}{2}+1 \cdots n$$

$$> 1 \cdot 1 \cdot 1 \cdots 1 \cdot \underbrace{\frac{n}{2} \cdot \frac{n}{2} \cdot \frac{n}{2} \cdots \frac{n}{2}}_{\frac{n}{2}} > \left(\frac{n}{2}\right)^{\frac{n}{2}}$$

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

after 1



$$n! \geq 2^{\frac{n}{2}\log\frac{n}{2}}$$

$$2^{\frac{n}{2}\log\frac{n}{2}-1}$$

$$\frac{n}{2}\log\frac{n}{2}-2$$

$$2$$

$$1$$

Of take $n \log n$ comparisons! $1 = 2^0$