Problem Have lists A, B, C of numbers decide $\exists a \in A, b \in B, c \in C$ s.t. $a+b=c$

1) Put C in hash. Loop over $a+b \in C$? $\rightarrow O(n^2)$

2) FFT to convolve P_A, P_B and get possible sums

$\sum_{i \in A} x^i$

$P_A P_B$ tells us $\{A+B\}$

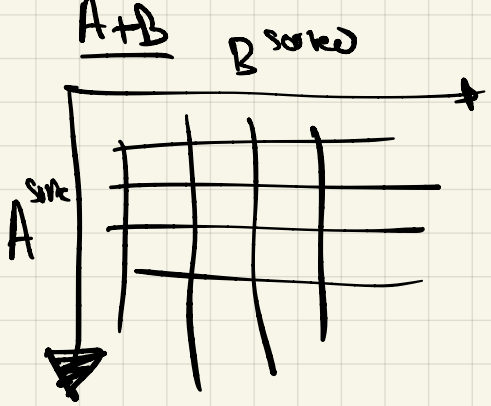
3) Do but BST on C $\rightarrow n^2 \log n$

3SUM

Thm Can solve this prob with $n^{1.5} \log n$ comparisons! $A, B, C \subseteq \mathbb{R}$

Thm 3 sum is $\frac{n^2 (\lg \lg n)^2}{\lg n}$

Let us do n^2 soln for prob

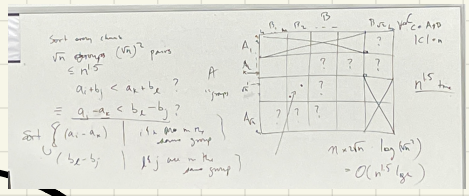


$\forall c \in C$ search for $c \in A+B$

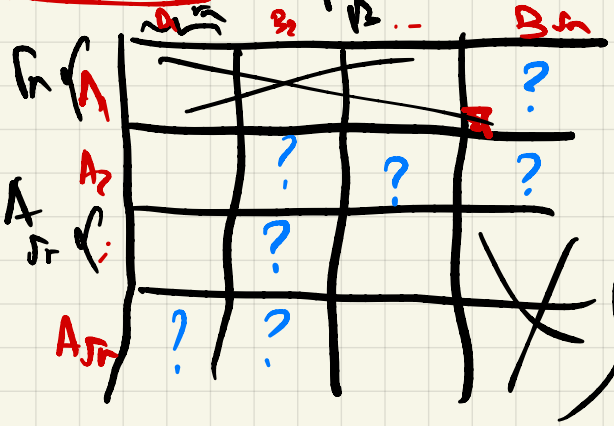
Start by probing top right cell

if too big elim column, else row!

$2n$ probes



Problem Chop $A+B$ mat into "grps" of $\sqrt{n}$



$\forall c \in C$ search for $c = a+b$

search with bottom left elt of grp

Similar ruling out of grps, can but have question above me!

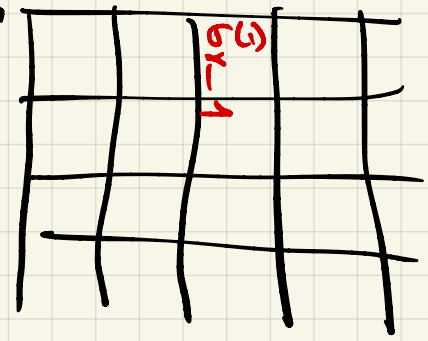
It takes $\sqrt{n}$ time for each c.

Overall $n^{1.5}$ time!

Sort every chunk $\rightarrow$ can do ... $n^{1.5}$ comp

Now BST on the chunks $\rightarrow n^{1.5} \log n$ time!

Now do this !



$\frac{n}{\sqrt{\log n}}$ / $\log\log n$

all possible perm of list of size L

$(L^2)! = (L^2)^{L^2} = e^{\frac{\log n}{\log\log n}} = n^{O(1)}$

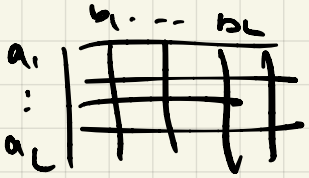
For every sorting permutation $\sigma : [L^2] \to [L] \times [L]$ decide which chunks are sorted by σ

↳ we need $\sigma(1) = (1,1)$ (smallest elt) (has to be in top left)

$\sigma(2) = (1,2)$ or $(2,1)$

$\sigma(i) = (\sigma_A(i), \sigma_B(i))$

Let some b_{ox}. when is σ right sort



want $a_{\sigma_A(1)} + b_{\sigma_B(1)} < a_{\sigma_A(2)} + b_{\sigma_B(2)}$

$\vdots$

$a_{\sigma_A(i)} + b_{\sigma_B(i)} < a_{\sigma_A(i+1)} + b_{\sigma_B(i+1)}$

$a_{\sigma_A(i)} - a_{\sigma_A(i+1)} < b_{\sigma_B(i+1)} - b_{\sigma_B(i)}$

blue pt

three values $\in \mathbb{R}^{L^2}$

red pt

$\in \mathbb{R}^{L^2}$

L^2 inequalities

σ is correct $\iff$ red dominates blue as elts of $\mathbb{R}^{L^2}$

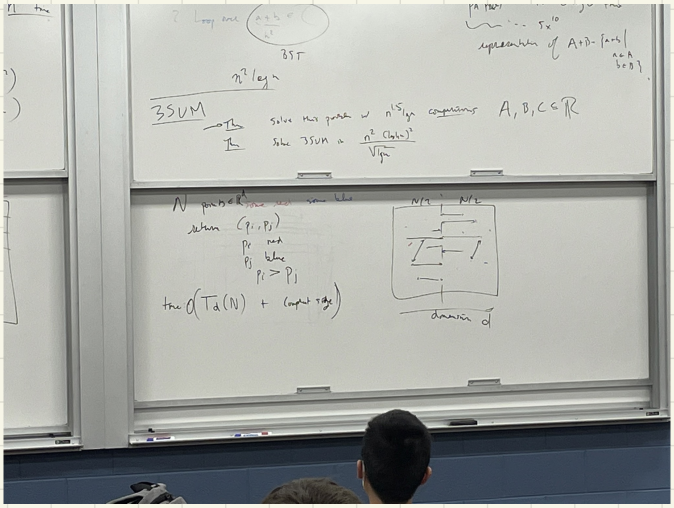
New problem

have N points in $\mathbb{R}^d$ some red some blue

return (p_i, p_j)

p_i red p_j blue so $p_i > p_j$

divide and conquer on last dimension !



Sort & iterate

$T_1(N) = N \log N$

Recurrence $T_d(N) = 2T_d(n/2) + T_{d-1}(N) + N$

~~hyp~~ $T_d(N) = N(\log N)^d \quad \forall N < n$

in our case $d = L^2$

$\leq n(\log n)^{L_2} \quad n(\log n)^{\frac{\log n}{(\log \log n)^2}} = n e^{\frac{\log n}{\log \log n}} \propto n^{1+o(1)}$ (sub poly)

$\forall \sigma \quad (\; (L^2)! = n^{o(1)}$ of these)

gen points $(p_1, \ldots p_{n/L}, q_1, \ldots q_{n/L})$

find all red blue dom pairs $(n^{1+o(1)}$ + output size$)$

now, we can sort each chunk

→ $O(1)$ access to each rank

→ each chunk sorted by only

⇒ total output size $= \left(\frac{n}{L}\right)^2 \simeq n^2 \log n$

$\forall c \in C$ look for c in $A+B$

→ same test to set [illegible] questionable

$\frac{2n}{L}$ candidate boxes

Then do a BST for C → $\log(L^2)$ time $\approx \log \log n$ time

for last step → $n \cdot \frac{2n}{L} \log L^2 = O\left(\frac{n^2 (\log \log n)^2}{\sqrt{\log n}}\right)$

(annotations over the terms: # C, # P, BST)

→ walkthru