Counting inversions
Divide and conquer

Compare your rankings of movies to your friends'

ranking of movies

(eg) 5 movies : A B C D E

your ranking : D B C A E
your friend's ranking : B A C D E

} how to compare

both of you: $B > A$, $C > E$

you: $D > B$ your friend: $B > D$

$\boxed{\text{1 inversion}}$

Problem: count # of inversions

* If no inversions: identical rankings

* max # of inversions possible : * $\binom{n}{2}$ pairs of movies

* $\leq \binom{n}{2}$.

# of inversions ⟷ "how dissimilar rankings are"

Assume your ranking : 1 2 3 . . . n
your friend's ranking : : $\sigma(1)$ $\sigma(2)$ - - - $\sigma(n)$

Inversion : ($i < j$) pair with j appearing before i in σ

. . . - j . . - - i . . . .

$\Updownarrow$

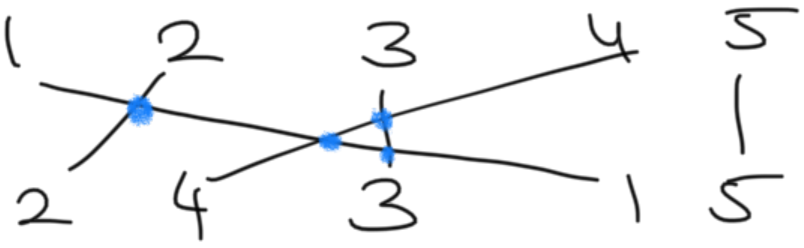
pair ($s < t$) with $\sigma(s) > \sigma(t)$

(eg) D B C A E your ranking
1 2 3 4 5

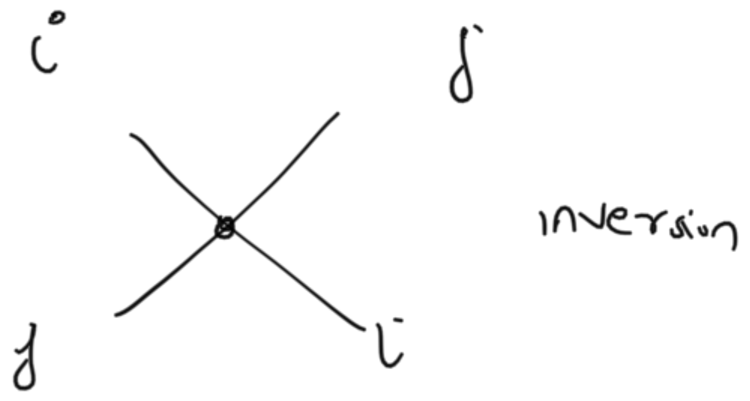
σ: B A C D E friend's ranking

2 4 3 1 5

Inversions of σ: $\{(2,1), (4,1), (3,1), (4,5)\}$



crossing of 2 lines ....



Brute force $O(n^2)$

$l = [2, 4, 3, 1, 5]$

* Check if $l[s] < l[t]$ $\forall$ $s < t$

Divide and conquer

$F = [\, i_1, i_2 \, ... \, i_N]$ ⇐ your friend's list

$L = [i_1 \ldots i_{N/2}]$ $R = [\, i_{N/2+1}, \ldots i_N]$ Divide

Find inversions(L), inversions(R) } Recursively solve smaller problems

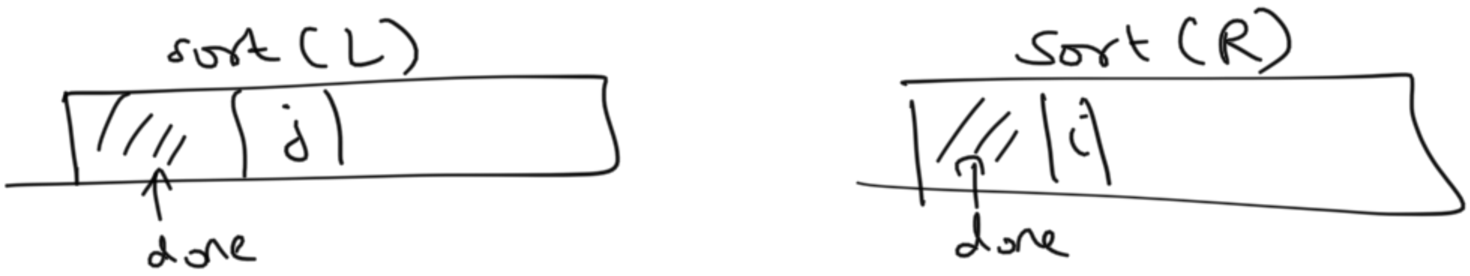
Combine | inversions (L) + inversions (R)

$+ \# \{ (l, r) \mid l \in L, r \in R, r < l \}$ = inversions(F)

Stronger problem solved during recursion

- Assume sort(L) and count # of inversions in (L)
- Assume sort(R) and count # of inversions in (R)
- when merging sort(L) and sort(R),

  count how many elements of r < how many elements of l



↓

when merging, if pulling elt from R, add current size(L) ⟸ # of inversions

$i < j$

⇒ $i <$ all elts in L currently

⇒ # of inv ++ by current size(L)

$O(n \log n)$