Nov 9

Collaborative filtering (Netflix)

Each user ≡ ranking (on movies)

User 1 is "close" to User 2 if user 1's rankings is "close" to user 2's rankings.

Assumption: Each user ranks all n movies (not true in real life)

Input: a ranking $a = a_1, \ldots, a_n$ (permutation of $[n] = \{1, \ldots, n\}$)

"base ranking"

Output: # inversion in a

(i,j) is an inversion if

(I) $i < j$

(II) $a_i > a_j$

$\#inv(1, 2, \ldots, n) = 0$ ← min ← none of the pairs are inversions

$\#inv(n, n-1, \ldots, 2, 1) = \binom{n}{2}$ ← all pairs are inversions (***) ← max

---

Brute-force algo: check all $\binom{n}{2}$ pairs (i,j) $i<j$ & count how many have $a_i > a_j$

$\Rightarrow O(n^2)$

Obs. If you had to output ALL inversions then $O(n^2)$ is best possible as output size for (***) is $\Omega(n^2)$.

Goal: count # inversions faster than $O(n^2)$

$\Rightarrow$ cannot compute ALL inversions.

CountInv(a, n)

If n == 1 return 0

If n == 2 return $a_1 > a_0$

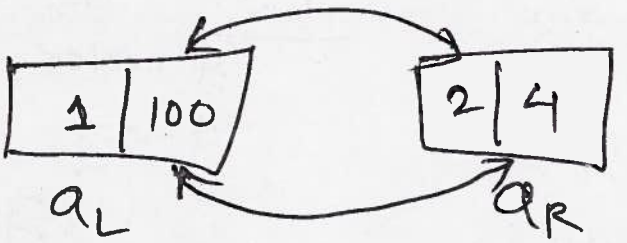
$a_L = a_1, \ldots, a_{\lceil \frac{n}{2} \rceil}$

$a_R = a_{\lceil \frac{n}{2} \rceil + 1}, \ldots, a_n$

return CountInv(a_L, $\lceil \frac{n}{2} \rceil$) + CountInv($a_R$, $n - \lceil \frac{n}{2} \rceil$)

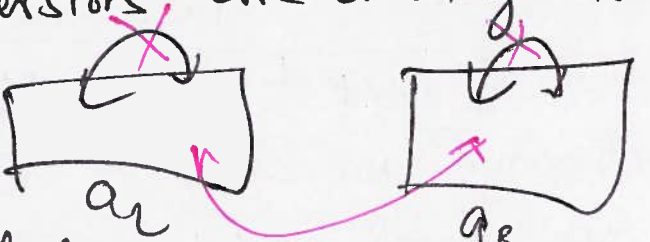
Problem: Doesn't consider "crossing" inversions.

↳ Infact all inversions could be crossing inversions.



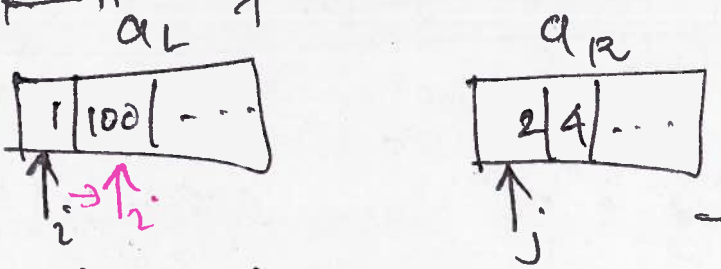
Consider case: All inversions are crossing inversions

⇒ no inversion



⇒ a_L & a_R are sorted.

Claim: Above case can be solved in O(n) time.



Solution: Modify MERGE algo

Case 1: $a_i = 1$, $a_j = 2$

$1 < 2$

⇒ (i, j) is not an inversion + (i, j') is not an inversion $\forall j' \geq j$

⇒ can advance i (i++)

Case 2: $a_i = 100 > a_j = 2$ ⇒ (i', j) is an inversion $\forall i' \geq i$

⇒ n' - i + 1 inversions + advance j!

In general:

a_L a_R

$l_1 \leq l_2 \leq \dots \leq l_{n'}$ $r_1 \leq r_2 \dots \leq r_m$

~~output all~~ count all pairs (i,j) s.t $l_i > r_j$

MergeCount (a_L, a_R)

0. $c = 0$
1. $i, j = 1$
2. while $i \leq n'$ AND $j \leq m$
   If $l_i \leq r_j$
   ~~c++~~ Append l_i to output; i++
   else
   $c = n' - i + 1$
   ~~j++~~.
   Append r_j to output; j++
3. Append whatever remains to the output

Q: Modify this to also merge a_L & a_R into one sorted list.