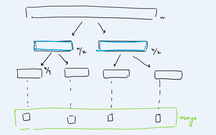
Merge Sort

→ Algorithm
→ Big Innovation
→ Time Complexity



Recurrence Relation

$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$ — ①

→ Substitution method

$n \to \frac{n}{2}$

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

$T(n) = 2\left[2T\left(\frac{n}{2^2}\right) + \frac{n}{2}\right] + n$

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

$T\left(\frac{n}{2^2}\right) = 2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}$ $\quad n \to \frac{n}{2^2}$

$T(n) = 2^2\left[2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}\right] + 2n$

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

$T(n) = 2^k T\left(\frac{n}{2^k}\right) + kn$

$T\left(\frac{n}{2^k}\right) = T(1)$

$\therefore \frac{n}{2^k} = 1 \Rightarrow n = 2^k$

④ $\log_2 n$

$T(n) = 2^k T(1) + kn$

$T(n) = 2^{\log n} T(1) + n\log_2 n$

$T(n) = n + n\log_2 n$

$\boxed{T(n) = O(n\log_2 n)}$

$k = \log_2 n$

Merge Sort (TC)

- Worst case → $O(n\log_2 n)$
- Best case → $O(n\log_2 n)$
- Avg case → $O(n\log_2 n)$

SC → $O(n)$

Inversion Count

Inversion count for an array tells about how far the array is from sorted.

For ending, two elements $a[i], a[j]$ form an inversion if $a[i] > a[j]$ and $i < j$.

Eg arr[] = {2, 4, 1, 3}

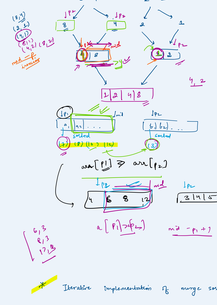
output → 3

{2,1}, {4,1}, {4,3}, {2,3}, {1,3}

Eg arr[] = {3, 1, 2}

output → 2

Count Inversions (Z) + Count Inversions (Y) + ____


8 4 2 1
8 4
2 1
8 4 2 1
2 1 8 4
1 2 4 8
{2,1}
{8,4}
{4,1}
{8,2}
{8,1}
{4,2}
{4,1}
{8,2}
{8,1}
{4,2}
{4,1}
{8,2}
{8,1}
{4,2