

Mergesort

↳ Merge Procedure

↳ combining task

[50, 70, 45, 5, 79, 47, 29, 52] $n = 8$

0 1 2 3 4 5 6 7

Small Problem

↳ single element

↳ sorted array

Big Problem

$n > 1$

Sorted array
↑

Recursive Tree

$mid = 3$

5, 45, 50, 70

C2

arr, 0, 3

$$\frac{0+3}{2} = 1$$

50, 70

C3

arr, 0, 1

50

C4

arr, 0, 0

70

C5

arr, 1, 1

5, 45

C6

arr, 2, 3

C7

arr, 2, 2

$$\frac{4+7}{2} = 5$$

47, 79

C10

arr, 4, 5

29, 52

C13

arr, 6, 7

47

C12

arr, 5, 6

5

C8

arr, 3, 3

79

C11

arr, 4, 4

29

C14

arr, 6, 6

52

C15

arr, 7, 7

↳ Small Problem

$i == j \Rightarrow$ array contains only single element

Stack data structure = $O(\log_2 n)$



Top to Down \rightarrow Divide & conquer

↳ Small problem

Down to Top \rightarrow combine

Merge Procedure

↓

fully sorted

subarray

Worst case scenario

Time complexity of merge procedure

$m=4$
original array →



Number of moves

Number of comparisons

Right side sorted subarray $m=4$

Left side sorted subarray

10	20	30	40	11	21	31	41
----	----	----	----	----	----	----	----

0 1 2 3 4 5 6 7
 ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑

↓ Merge Procedure

$$\textcircled{1} \quad 10 < 11$$

$$\textcircled{2} \quad 20 > 11$$

$$\textcircled{3} \quad 20 < 21$$

$$\textcircled{4} \quad 30 > 21$$

$$\textcircled{5} \quad 30 < 31$$

$$\textcircled{6} \quad 40 > 31$$

$$\textcircled{7} \quad 40 < 41$$

New array →

10	11	20	21	30	31	40	41
----	----	----	----	----	----	----	----

Worst case scenario

$$\hookrightarrow m + m - 1$$

Best case scenario

original array

$m=4$

$m=2$

30	40	50	60	5	10
----	----	----	----	---	----

Left sorted subarray Right sorted subarray

$$30 > 5 \textcircled{1}$$

$$30 > 10 \textcircled{2}$$

extra space

New array ←

5	10	30	40	50	60
---	----	----	----	----	----

Best case scenario

$$\# \text{ comparisons} = \min(m, n)$$

$$= \min(2, 4) = 2$$

$$\# \text{ moves} = m+n$$

$$m+n = N$$

Time complexity $\rightarrow O(m+n)$
(Merge Procedure) $\rightarrow O(N)$



moves + # comparisons

$$(m+n) + (m+m-1) \\ \Rightarrow O(m+n)$$

Space complexity $\rightarrow O(N)$

↓
extra space (new array)

↓
Outplace sorting algorithm

$T(n) \Rightarrow \underline{\text{mergeSort}}(\text{arr}, i, j):$
Pseudocode

```

if i == j:
    return arr
else:
    c - Divide → {
        mid = i + (j - i) / 2
        T(N/2)
        mergeSort(arr, i, mid)
        mergeSort(arr, mid + 1, j)
    }
    c - conquer → {
        mergeProcedure(arr, i, mid,
                      mid + 1, j)
    }
    c - combine → {
        return arr
    }
}

```

$\uparrow c$
small Problem

Recurrence Relation

$$T(n) = 2T\left(\frac{n}{2}\right) + N \quad \text{Master's Theorem}$$

$$a = 2 \quad k = 1$$

$$b = 2 \quad p = 0$$

Masters

Theorem

$$\log_b^a = \log_2^2 = 1$$

$$\log_b^a = k$$

$$\Rightarrow \Theta(N \log N)$$

$$\begin{aligned}
 T(n) &= 2T\left(\frac{n}{2}\right) + n \\
 &= 2\left(2T\left(\frac{n}{2^2}\right) + \frac{n}{2}\right) + n \\
 \underline{\text{Substitution}} \quad \underline{\text{Method}} \quad &\Rightarrow 2^2 T\left(\frac{n}{2^2}\right) + 2n \\
 &\Rightarrow 2^2 \left(2T\left(\frac{n}{2^3}\right) + \frac{n}{2^2}\right) + 2n \\
 &\Rightarrow 2^3 T\left(\frac{n}{2^3}\right) + 3n \\
 &\qquad\qquad\qquad \downarrow k \\
 \frac{n}{2^k} = 1 &\qquad\qquad\qquad \downarrow k \\
 k = \log_2 n &
 \end{aligned}$$

$$\begin{aligned}
 2^k T\left(\frac{n}{2^k}\right) + k \cdot n & \\
 2^{\log_2 n} T\left(\frac{n}{2^{\log_2 n}}\right) + \log_2 n \cdot n & \\
 n^{\log_2 2} T\left(\frac{n}{n^{\log_2 2}}\right) + n \log_2 n & \\
 n + n \log_2 n &
 \end{aligned}$$

$\Rightarrow \underline{\circ(\log n)}$