

Merge-Sort

- divide arrays into two halves
- recursively sort each half
- merge two halves to make a whole

H E L L O W O R L D

H E L L O

W O R L D

divide $O(n)$

E H L L O

D L O R W

sort $2T(n/2)$

D H L L O O R W

merge $O(n)$

$T(n) = O(n \log n)$

Ways to prove:

- recursion tree
- Telescoping

If $T(n)$ satisfies, $T(n) = n \log_2 n$

$$T(n) = \underbrace{2T\left(\frac{n}{2}\right)}_{\text{sorting}} + \underbrace{n}_{\text{merging}}$$

For $n > 1$

$$\frac{T(n)}{n} = \frac{2T(n/2)}{n} + 1$$

$$= \frac{T(n/2)}{n/2} + 1$$

$$= \frac{T(n/4)}{n/4} + 1 + 1$$

$$= \frac{T(n/n)}{n/n} + 1 + \dots + 1 \quad (\log_2 n)$$

$$= \log_2 n$$