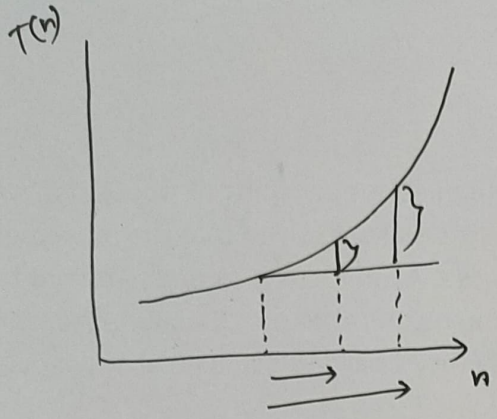
CSE 373 / L-3 / 13.02.2024

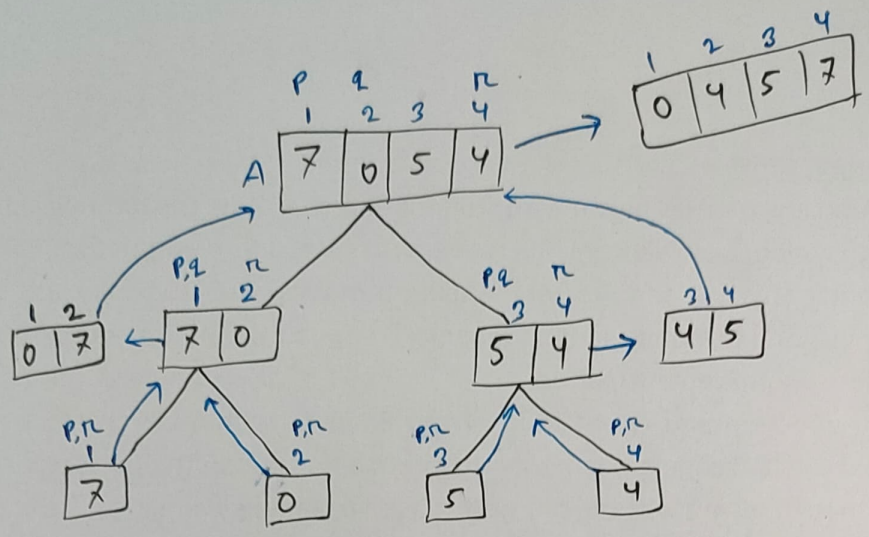


$\theta(n^2)$

Divide and Conquer (Merge Sort):

- Use recursive function
- To solve a given problem, they call themselves recursively one or more times to deal with closely related sub-problems.
- Divide: the problem into one or more subproblems that are ~~similar~~ smaller instances of the same problem.
- Conquer: the subproblems by solving them recursively
- Combine: the subproblems solutions to form a solution to the original problem.

Merge Sort Operation:



⇒ Faster, Linear function

⇒ Takes a lot of memory

Algorithm:

MERGE-SORT (A, p, r)

C.T { if p ≥ r
return
q = ⌊(p+r)/2⌋

2T(n/2) { MERGE-SORT (A, p, q)
MERGE-SORT (A, q+1, r)

θ(n) { MERGE (A, p, q, r)

MERGE (A, p, q, r)

C.T.
- $n_L = q - p + 1$ C.T.
- $n_R = r - q$ C.T.
- let $L[0 : n_L - 1]$ and $R[0 : n_R - 1]$ be new arrays C.T.

$\Theta(n_L + n_R) = \Theta(n)$

```
for i = 0 to n_L - 1
    L[i] = A[p + i]
for j = 0 to n_R - 1
    R[j] = A[q + j + 1]
```

C.T.
- $i = 0$
- $j = 0$
- $k = p$

$\Theta(n)$

```
while i < n_L and j < n_R
    if L[i] ≤ R[j]
        A[k] = L[i]
        i = i + 1
    else
        A[k] = R[j]
        j = j + 1
    k = k + 1
while i < n_L
    A[k] = L[i]
    i = i + 1
    k = k + 1
while j < n_R
    A[k] = R[j]
    j = j + 1
    k = k + 1
```

Time $\Rightarrow \Theta(n) + \Theta(n) + \text{C.T.} \neq \text{C.T}$

$= \boxed{\Theta(n)}$

Analysis :

Divide : The divide step just compute the middle of the subarray, which takes constant time. Thus,

$$D(n) = \theta(1)$$

Conquer : We recursively solve two subproblem, each of size $(n/2)$, which contributes $2T(n/2)$ to the running time.

Combine : We have already noted that the MERGE procedure on an n element subarray takes time $\theta(n)$,

$$C(n) = \theta(n)$$

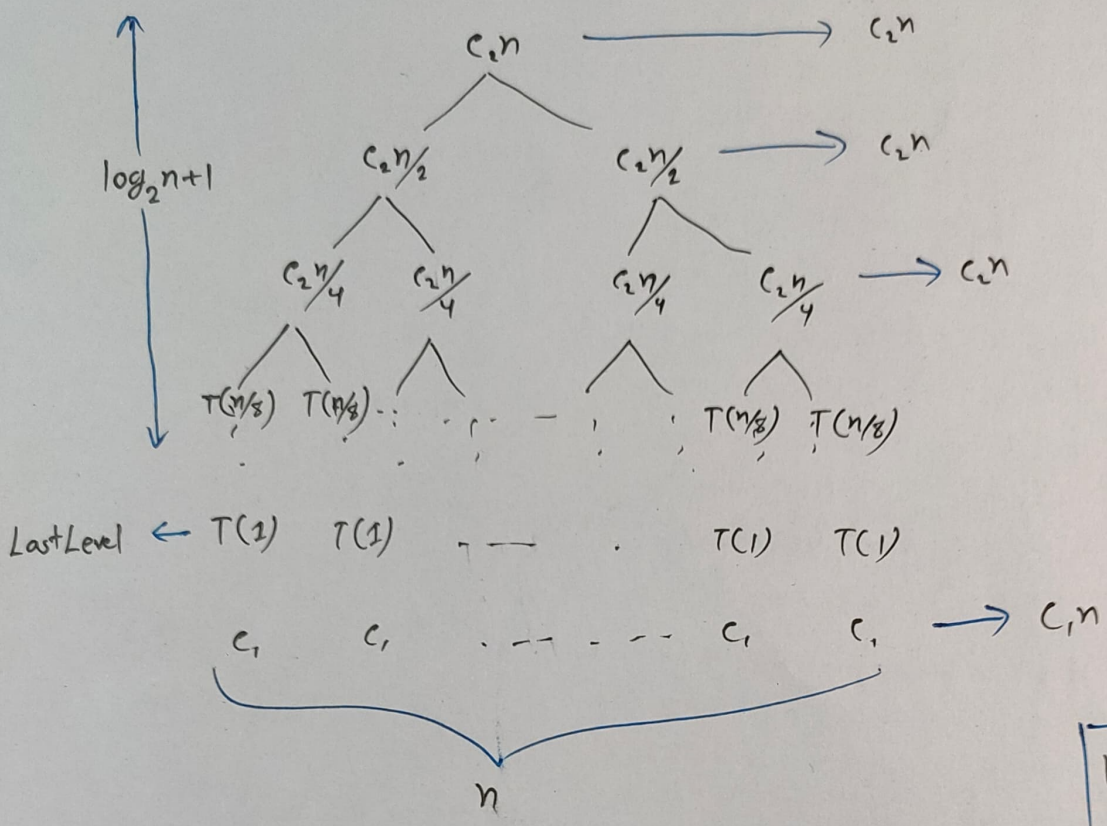
⇒ Worst-Case running time $T(n)$ of merge sort

$$T(n) = \begin{cases} \theta(1) & ; \text{if } n < n_0 \\ D(n) + aT(n/b) + C(n) & ; \text{otherwise} \end{cases}$$

$$T(n) = \begin{cases} \theta(1) & ; n = 1 \\ 2T(n/2) + \theta(n) & ; n > 1 \end{cases}$$

$$T(n) = \begin{cases} c_1 & ; n = 1 \\ 2T(n/2) + c_2 n & ; n > 1 \end{cases}$$

Merge-Sort Analysis using recursive tree method:



$\lg n = \log_2 n$

$\Rightarrow$ - recursive tree has $\lg n + 1$ level

- The level above the leaves each cost $c_2 n$
- The level cost $c_1 n$

$\Rightarrow$ Total cost $\Rightarrow$ $c_2 n \log_2 n + c_1 n$

$= \theta(n \log_2 n)$

L-4 / 18.02.2024

H.W.

2.1 $\Rightarrow$ 1, 3, 4

2.2 $\Rightarrow$ 2, 3, 4

2.3 $\Rightarrow$ 1, 2, 8