ese 201 1-28-21

---

Continue iteration method.

Ex. $T(n) = \begin{cases} 1 & 1 \le n < 3 \\ 2T(\lfloor \frac{n}{3} \rfloor) + n & n \ge 3 \end{cases}$

We showed $T(n) = O(n)$

also $T(n) = 2T(\lfloor \frac{n}{3} \rfloor) + n \ge n = \Omega(n)$

$\therefore T(n) = \Omega(n)$

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

check using master-thm:

simplify $T(n) = 2T(\frac{n}{3}) + n$

compare: $n$ to $n^{\log_3 2}$

let $\varepsilon = 1 - \log_3 2$. Then $\varepsilon > 0$

and $1 = \log_3 2 + \varepsilon$, and

$$n = \Omega(n) = \Omega\left(n^{\log_3 2 + \varepsilon}\right)$$

✓ regularity cond. $af\left(\frac{n}{b}\right) \le cf(n)$ for some $0 < c < 1$ and all sufft. large $n$.

need $2\left(\frac{n}{3}\right) \le cn$ ✓

$\frac{2}{3} n \le cn$ ✓

$\frac{2}{3} \le c$ ✓ (note: $\frac{2}{3} < 1$)

Pick any $c$ in range $\frac{2}{3} \le c < 1$.

By case 3: $\boxed{T(n) = \Theta(n).}$

Ex.

$$T(n) = \begin{cases} 5 & n = 0, 1 \\ T(n-2) + n & n \geq 2 \end{cases}$$

$$\begin{aligned} T(n) &= n + T(n-2) \\ &= n + (n-2) + T(n-2\cdot 2) \\ &= n + (n-2) + (n-2\cdot 2) + T(n-3\cdot 2) \\ &\vdots \\ &= \sum_{i=0}^{k-1} (n-2i) + T(n-2k) \end{aligned}$$

stop when $0 \leq n-2k < 2$

$\therefore 2k \leq n < 2k+2$

$\therefore k \leq \frac{n}{2} < k+1$

$\therefore k = \lfloor \frac{n}{2} \rfloor$

with this $k$, $T(n-2k) = 5$

so

$$T(n) = \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor - 1} (n - 2i) + 5$$

$$= \sum_{i=0}^{K-1} n - 2 \sum_{i=\cancel{0}\,1}^{K-1} i + 5$$

$$= Kn - \cancel{2} \cdot \frac{K(K-1)}{\cancel{2}} + 5$$

$$\therefore \boxed{T(n) = \lfloor \tfrac{n}{2} \rfloor n - \lfloor \tfrac{n}{2} \rfloor \left( \lfloor \tfrac{n}{2} \rfloor - 1 \right) + 5}$$

Simplify:

$$T(n) = \Theta(n^2).$$

or

$$T(n) \sim \tfrac{1}{4} n^2$$

Ex. $T(n) = \begin{cases} 0 & n = 1 \\ T(\lfloor \frac{n}{2} \rfloor) + 1 & n \geq 2 \end{cases}$

$$T(n) = 1 + T(\lfloor \tfrac{n}{2} \rfloor)$$

$$= 1 + 1 + T(\lfloor \tfrac{\lfloor n/2 \rfloor}{2} \rfloor)$$

$$= 2 + T(\lfloor \tfrac{n}{2^2} \rfloor)$$

$$= 2 + 1 + T(\lfloor \tfrac{\lfloor n/2^2 \rfloor}{2} \rfloor)$$

$$= 3 + T(\lfloor \tfrac{n}{2^3} \rfloor)$$

$$\vdots$$

$$= k + T(\lfloor \tfrac{n}{2^k} \rfloor)$$

stop when $1 \leq \lfloor \frac{n}{2^k} \rfloor < 2$

$\therefore \quad 1 \leq \frac{n}{2^k} < 2$

$\therefore \quad 2^k \leq n < 2^{k+1}$

$\therefore \quad k \leq \lg(n) < k+1$

$\therefore \quad k = \lfloor \lg n \rfloor$

$\therefore T(n) = \lfloor \lg n \rfloor$

hence $T(n) = \Theta(\log n)$

Exercise : $S(n) = \begin{cases} 0 & n = 1 \\ S(\lceil \frac{n}{2} \rceil) + 1 & n \geq 2 \end{cases}$

show that $S(n) = \lceil \lg n \rceil$

and hence $S(n) = \Theta(\log n)$

Ex.

$$T(n) = \begin{cases} c & 1 \leq n < n_0 \\ T(\lfloor \frac{n}{2} \rfloor) + d & n \geq n_0 \end{cases}$$

$$T(n) = d + T(\lfloor \tfrac{n}{2} \rfloor)$$
$$= d + d + T(\lfloor \tfrac{n}{2^2} \rfloor)$$
$$= 2d + T(\lfloor \tfrac{n}{2^2} \rfloor)$$

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

halt at smallest $k$ for which

$$1 \leq \left\lfloor \frac{n}{2^k} \right\rfloor < n_0$$

$$\therefore \quad 1 \leq \frac{n}{2^k} < n_0$$

$$\therefore \quad 2^k \leq n < n_0 \cdot 2^k$$

$$\therefore \quad k \leq \lg(n) < k + \lg n_0$$

we seek smallest $k$ such that

$$\lg(n) < k + \lg n_0$$

$$\therefore \quad \lg(n) - \lg n_0 < k$$

$$\therefore \quad k - 1 \leq \lg n - \lg n_0 < k$$

$$\therefore \quad k - 1 = \lfloor \lg n - \lg n_0 \rfloor$$

$$\therefore \quad k = \lfloor \lg n - \lg n_0 \rfloor + 1$$

for this $k$,

$$\therefore \quad T(n) = d\left(\lfloor \lg n - \lg n_0 \rfloor + 1\right) + c$$

Hence $\boxed{T(n) = \Theta(\log n).}$

# Divide and Conquer (chap 4 in CLRS)

Ex. Merge Sort

Let A be an array of integers.

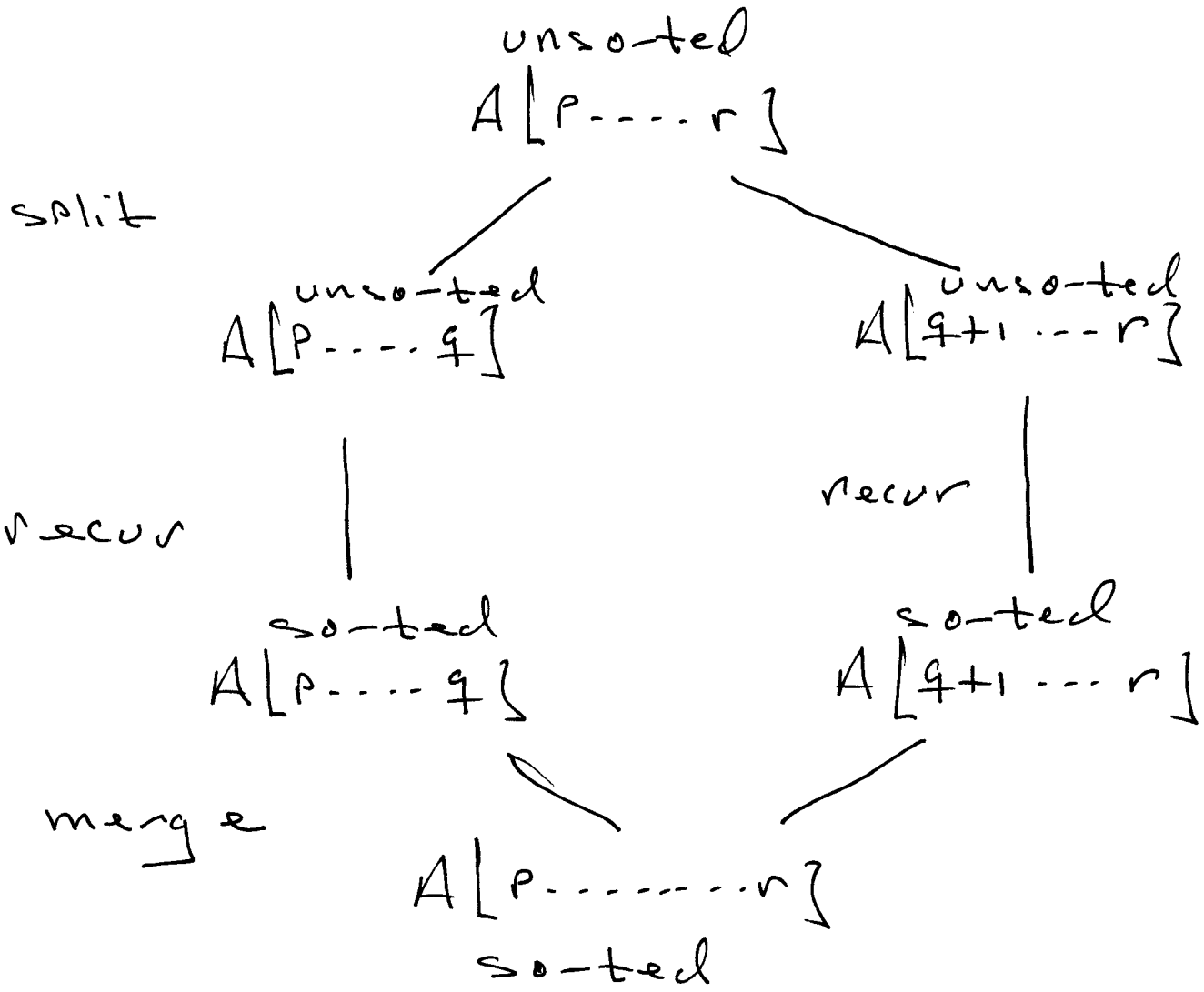
notation:

$A[p \ldots r]$ sub-array index p to r (inclusive):

$A[p], A[p+1], \ldots, A[r]$

$A[1 \ldots n]$ full array of length n.

$$\text{length}(A[p \ldots r]) = r - p + 1$$



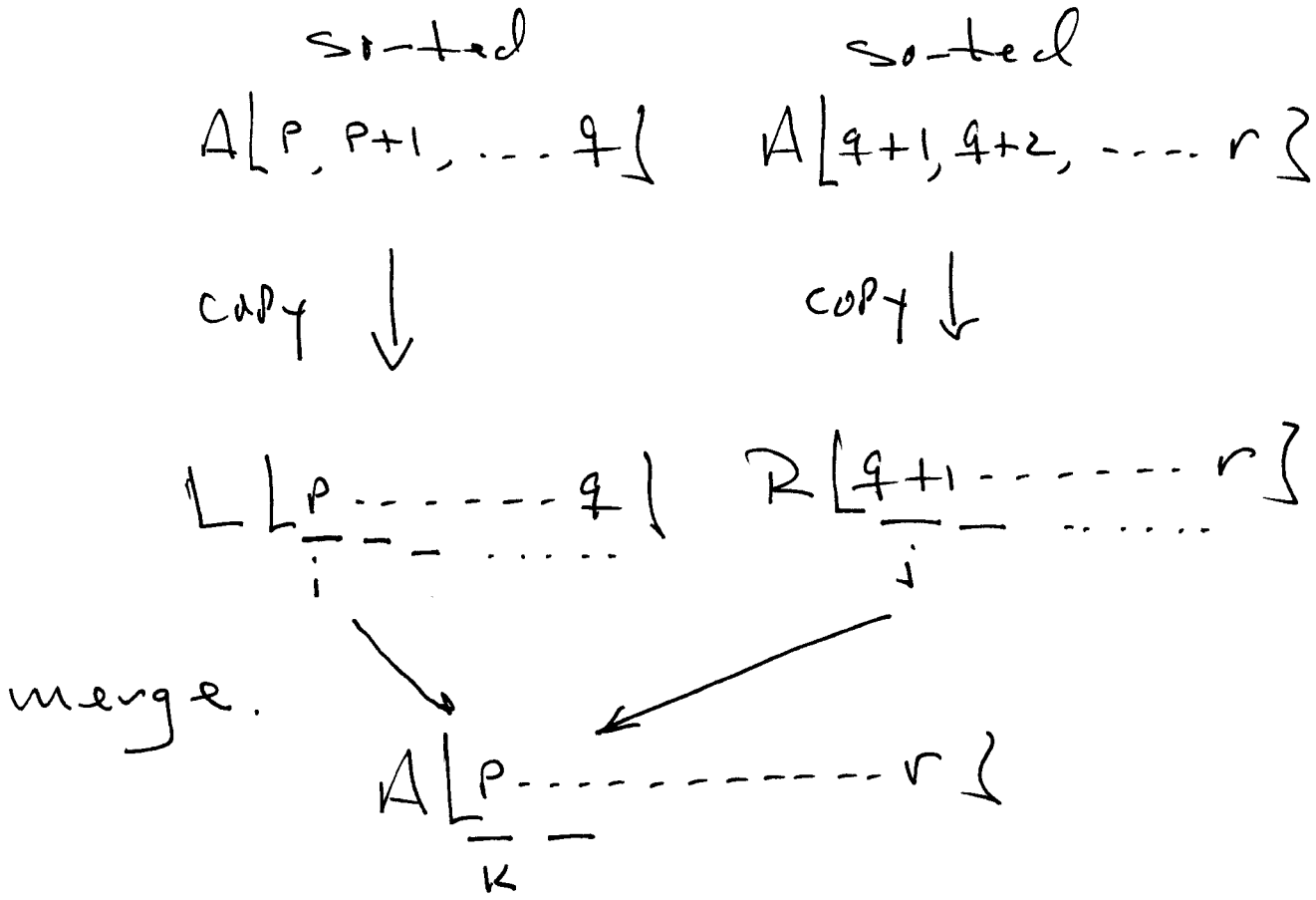

## MergeSort(A, p, r)

1. If $p < r$
2. $q = \lfloor \frac{p+r}{2} \rfloor$
3. MergeSort(A, p, q)
4. MergeSort(A, q+1, r)
5. Merge(A, p, q, r)

Exercise

write pseudo-code for Merge(A, p, q, r).

or look it up




- To do:
  - prove correctness: induction
  - write recurrence for runtime and solve it.