Mester-módszer

- „Recept" a $T(n) = aT\left(\frac{n}{b}\right) + f(n)$ alakú rekurzív egyenletek megoldására $a \geq 1$, $b > 1$ ($f(n)$ AP)
- Vezd meg rekurzív algoritmusok elemzésére
- Lehet, hogy $\frac{n}{b}$ nem egész, ha $T\left(\left\lfloor\frac{n}{b}\right\rfloor\right)$, $T\left(\left\lceil\frac{n}{b}\right\rceil\right)$-t használunk, az nem változtat az aszimptotikus viselkedésen
- A módszer alapja: $\boxed{\text{Mester-tétel}}$

Mester-tétel: legyen $a \geq 1$, $b > 1$ állandók, $f(n)$ AP

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n)$$

- ha $f(n) \in O\left(n^{\log_b a - \varepsilon}\right)$, ahol $\varepsilon > 0$, akkor $T(n) = \Theta\left(n^{\log_b a}\right)$

$$O\left(\frac{n^{\log_b a}}{n^{\varepsilon}}\right)$$

- ha $f(n) = \Theta\left(n^{\log_b a}\right)$, akkor $T(n) = \Theta\left(n^{\log_b a} \log n\right)$
- ha $f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right)$, $\varepsilon > 0$ és $\exists N \in \mathbb{N}, \forall n \geq N$ $a\,f\left(\frac{n}{b}\right) \leq c\,f(n)$, ahol $c < 1$ „elég nagy n esetén" akkor $T(n) = \Theta(f(n))$

Példa:

$$T(n) = \underbrace{9}_{a}\, T\left(\frac{n}{\underbrace{3}_{b}}\right) + \underbrace{n}_{f(n)}$$

$$n^{\log_b a} = n^{\log_3 9} = n^2$$

$$f(n) = O\left(n^{2-\varepsilon}\right) \quad \varepsilon = 1$$

„első eset” $T(n) = \Theta(n^2)$

$$T(n) = T\left(\frac{2n}{3}\right) + 1$$

$a = 1$, $b = \frac{3}{2}$, $f(n) = 1$

$$n^{\log_b a} = n^{\log_{3/2} 1} = n^0 = 1$$

$$f(n) = \Theta(1)$$

$$T(n) = \Theta(\underbrace{1 \cdot \log(n)}_{\Theta(\log(n))})$$

$$T(n) = 3T\left(\frac{n}{4}\right) + n\log n$$

$a = 3$, $b = 4$, $f(n) = n\log n$

$$n^{\log_b a} = n^{\log_4 3} \approx n^{0.793}$$

$$f(n) = \Omega\left(n^{0.793+\varepsilon}\right) \qquad \varepsilon = 0.2$$

$$a f\left(\frac{n}{b}\right) \leq c f(n) \qquad // c < 1$$

$$3 f\left(\frac{n}{4}\right) = 3 \frac{n}{4} \log \frac{n}{4} \leq \frac{3}{4} n \log n = \frac{3}{4} f(n)$$

$$c = \frac{3}{4}$$

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

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

$a = 2$, $b = 2$, $f(n) = n \log n$

$$n^{\log_b a} = n^{\log_2 2} = n^1$$

$$\uparrow$$

$$1$$

$f(n) = \Omega(n)$, de $f(n) \neq \Omega(n^{1+\varepsilon})$ $\varepsilon > 0$

$n^{1+\varepsilon} > n^1 \log n$ $\quad n^c \log n < n^{c+d}$

$\Rightarrow$ a módszer nem alkalmazható

Többségi elem

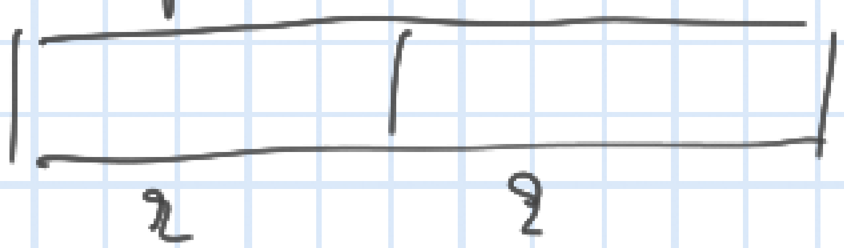
$A[1..n]$ tömbben x többségi elem, ha $\frac{n}{2}$-nél többször fordul elő. Döntsük el, hogy van-e többségi elem!

<u>Megszorítás</u>: csak $A[i] = A[j]$ vizsgálat végezhető!

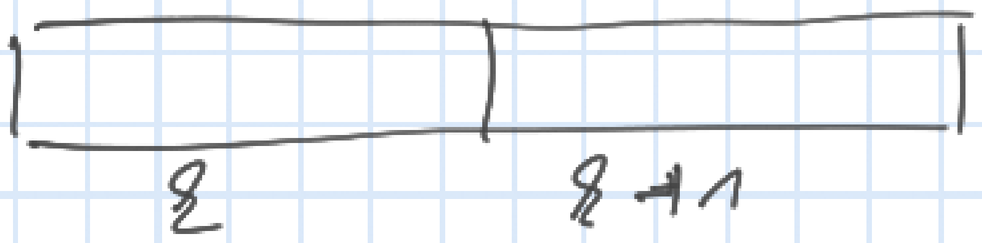
Oszd meg és uralkodj m.o.

- Észrevétel:
  - ha x többségi elem $A[1..n]$-ben, akkor $A[1..q]$ vagy $A[q+1..n]$ $(q = \lfloor \frac{n}{2} \rfloor)$ legalább egyikében többségi elem

$n = 2k$



$n = 2k+1$



x legalább $k+1$-szer fordul elő

- Algoritmus

1.) Két „egyenlő" részre bontjuk $A[1..n]$-et $A[1..q]$ és $A[q+1..n]$, ahol $q = \left\lfloor \frac{n}{2} \right\rfloor$

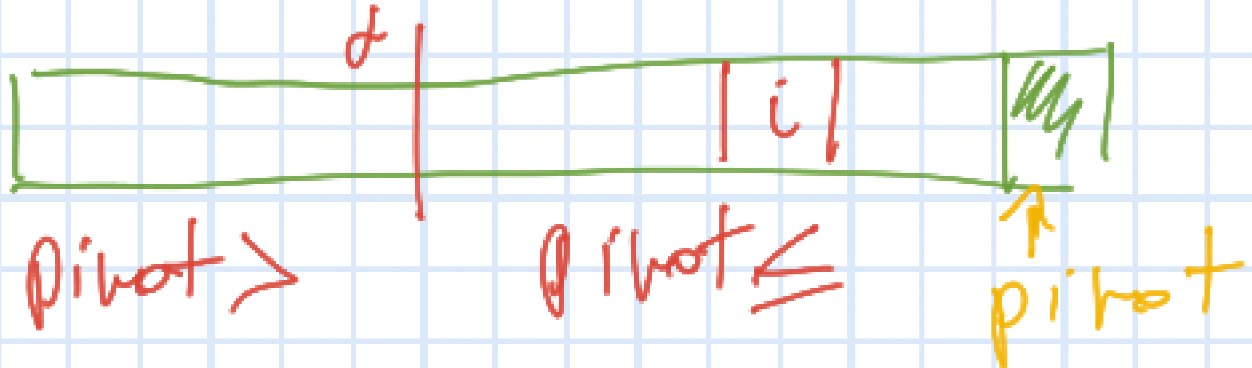
2.) Rekurzívan keressük a többségi elemet $A[1..q]$-ban és $A[q+1..n]$-ben.
   - Ha a résztömb 1 elemű, akkor azt adjuk vissza, mint többségi elem

3.) A visszaadott elemekről ellenőrizzük valóban többségi elem-e valamelyik $\rightarrow$ 2 megszámlálás

- Költség: $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$ $\quad T(n) = O(n \log n)$

Javítás:

– Ötlet: quicksort particionáló módszere



• ha $A[i] \geq$ pivot, akkor helyben marad, ha $A[i] < A[j] \Rightarrow$ csere$(A[i], A[j])$

– Észrevétel:

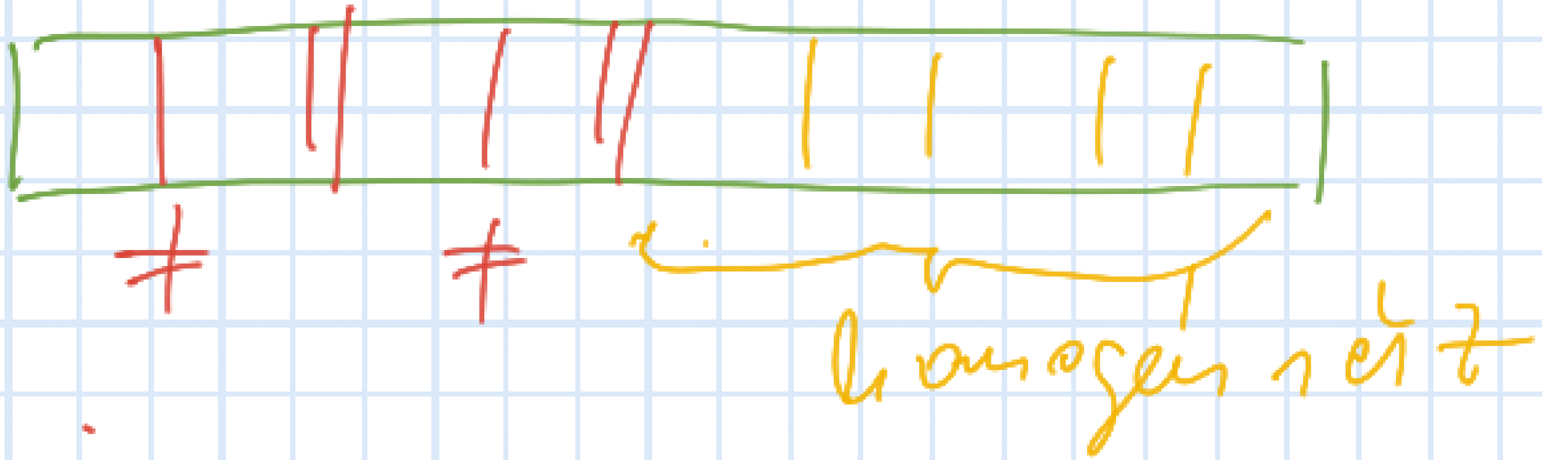
• Ha $A[1..n]$-ben x többségi $\Rightarrow$ $A[3..n]$-ben is

– ha $x \neq A[1], A[2]$, akkor nyilván

– ha $x = A[1]$ vagy $x = A[2]$ (mindkettő lehet), akkor x több, mint $\left\lceil \frac{n-2}{2} \right\rceil$-ször elő fordul $A[3..n]$-ben

— Algoritmus   Költség: $O(n)$

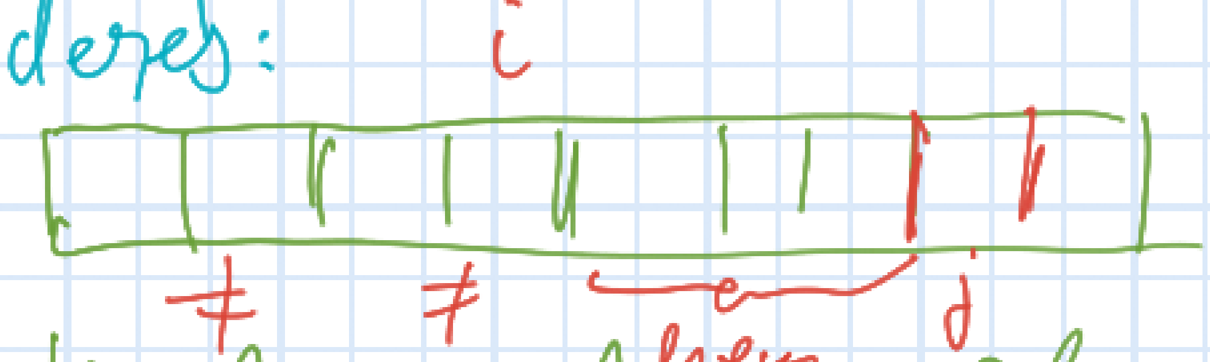
- átrendezzük a tömböt: egy homogén és egy nem homogén részre



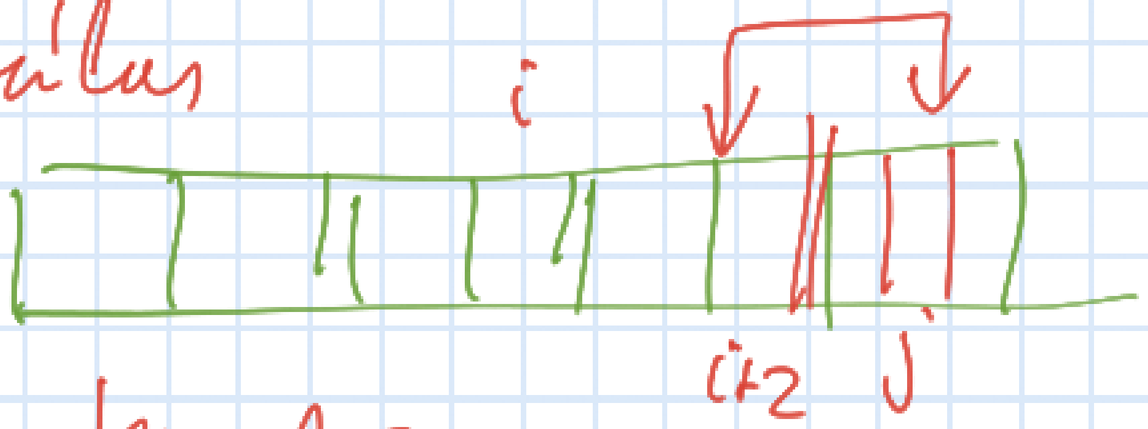
- ha van többségi elem ⇒ akkor az csak a homogén részben lehet

→ ellenőrizni kell: megszámlálás

Átrendezés:



ha $A[i+1] = A[j]$, akkor OK



ha $A[i+1] \neq A[j]$

Maximális "összefüggő" összeg

$A[1..n]$ valós tömbben két olyan $1 \leq i \leq j \leq n$ indexet keresünk, melyekre $A[i] + A[i+1] + \ldots + A[j]$ összeg maximális!

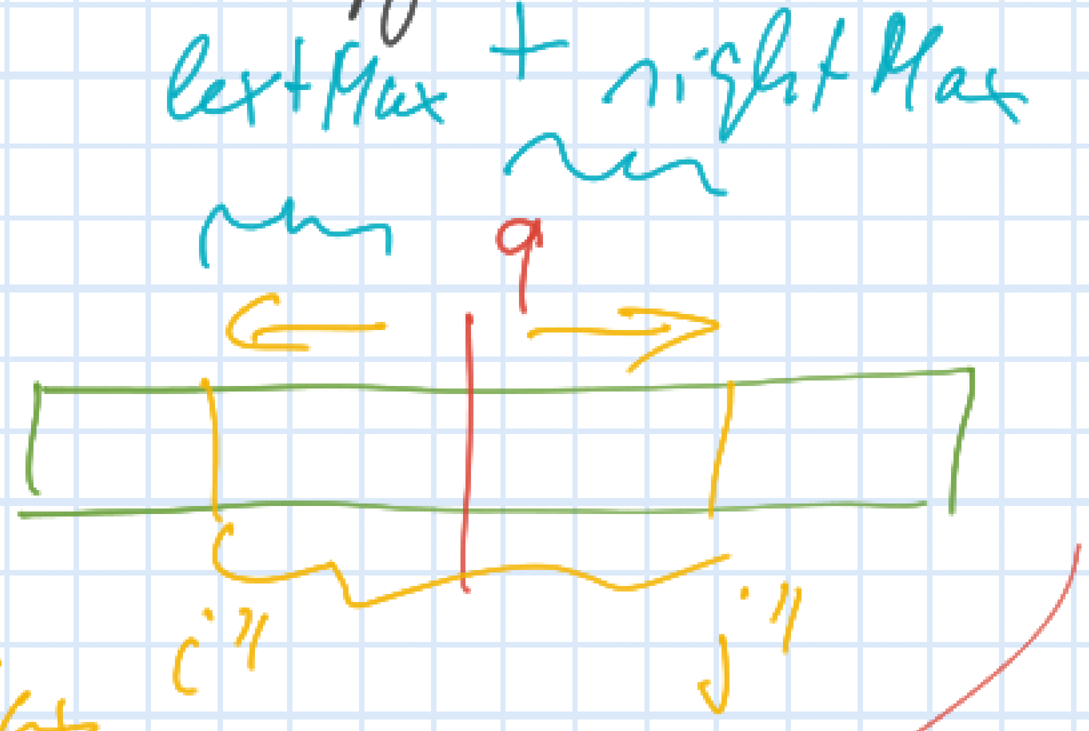
Pl.: $-2, -3, \underbrace{\boxed{4, -1, -2, 1, 5,}}_{\Sigma\, 7} -3$

– Naiv m.o.: egymásba ágyazott ciklusok $O(n^2)$ költség

Oszd meg és uralkodj m.o.

Költség: $T(n) = 2T\left(\frac{n}{2}\right) + O(n) \Rightarrow O(n \log n)$

- Osszuk fel a tömböt két részre $A[1..q]$ és $A[q+1..n]$, $q = \lfloor \frac{n}{2} \rfloor$
- Rekurzívan keressük a megoldást a két résztömbben
  - ha a résztömb 1 elemű $\Rightarrow$ azt az elemet adjuk vissza
- A teljes feladat megoldása az alábbiak egyike
  - (i, j) $A[1..q]$-ból
  - (i', j') $A[q+1..n]$-ből
  - (i'', j'') $1 \leq i'' \leq q < j'' \leq n$
    - elindulunk q-tól visszafelé $\Rightarrow$ leftMax
    - elindulunk $q+1$-től $\Rightarrow$ rightMax



a 3 eset közül a max

Javítás: Kadane algoritmus

3, 5, -9, 1, 3, -2, 3, 4, 7, 2, -9, 6, 3, 1

$|3+5| < |-9|$

Költség: $O(n)$