



Algoritmusok és adatszerkezetek I.

4. Előadás

Gyorsrendezés

Gyorsrendezés lépései

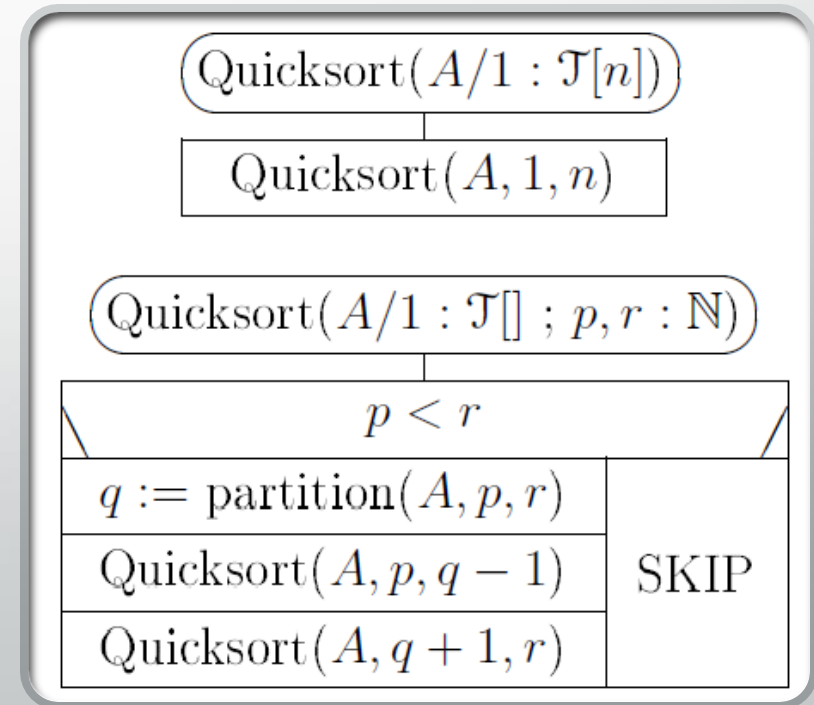
➤ „oszd meg és uralkodj” elv

➤ Lépései:

- **Válaszd ki** a rendezendő (rész)tömb egy tetszőleges elemét! Ez lesz a tengely (angolul **pivot**).
- **Részekre bontás (partitioning)**: Rendezd át úgy a tömböt, hogy minden, a tengelynél kisebb elem a tengely előtt, a nagyobbak pedig utána jöjjenek! (A tengellyel egyenlők bármelyik részbe kerülhetnek.) Ezzel az ún. particionálással (partition) a tengely már a végleges helyére került.
- **Alkalmazd rekurzívan** a fenti lépéseket, külön a **tengelynél kisebb** elemek résztömbjére, és külön a **tengelynél nagyobb elemek résztömbjére**!
- **Az üres és az egyelemű résztömbök a rekurzió alapesetei**. Ezek ui. már eleve készen vannak, így nem is kell őket rendezni.

Gyorsrendezés algoritmus

- A tengely kiválasztása és a részekre bontás lépései
 - Többféleképpen
 - A módszerek konkrét megválasztása erősen befolyásolja a rendezés hatékonyságát.
 - együtt lineáris időben befejeződjenek!



Partition függvény:

➤ Jelölések:

➤ $A[k..m] \leq x \Leftrightarrow \text{ha } \forall l (k \leq l \leq m) : A[l] \leq x$

➤ $A[k..m] \geq x \Leftrightarrow \text{ha } \forall l (k \leq l \leq m) : A[l] \geq x$

➤ Bemenet: $A[p..r]$ résztömb, pivot: $x=5$

	p							r
A:	5	3	8	5	6	4	7	1

	p							r	
A:	5	3	8	1	6	4	7		$x=5$

➤ 1. ciklus: megkeresi az 1. tengelynél > elemet (ha van)

	i=p	i	i					r	
A:	5	3	8	1	6	4	7		$x=5$

➤ A j a következő elemre áll:

	p		i	j				r	
A:	5	3	8	1	6	4	7		$x=5$

➤ $p \leq i < j \leq r$

➤ Az $A[p..r]$ résztömb szakaszai:

➤ $A[p..(i-1)] \leq x$

➤ $A[i..(j-1)] \geq x$

➤ $A[j..(r-1)]$ ismeretlen

➤ $A[r]$ definiálatlan (ez a tengely üres helye)

$\text{partition}(A[1 : \mathcal{T}[] ; p, r : \mathbb{N}) : \mathbb{N}$

$i := \text{random}(p, r)$

$x := A[i] ; A[i] := A[r]$

$i := p$

$i < r \wedge A[i] \leq x$

$i := i + 1$

$i < r$

$j := i + 1$

$j < r$

$A[j] < x$

$\text{swap}(A[i], A[j])$

$i := i + 1$

$j := j + 1$

$A[r] := A[i] ; A[i] := x$

return i

$A[r] := x$

SKIP

Partition függvény

➤ A partition fv helyességének ellenőrzéséhez vezessük még be a következő jelöléseket:

- A_0 : A tömb kezdeti állapota a partition függvény meghívásakor.
- $A[u..v] + x$ tömb objektum az x elemnek az $A[u..v]$ résztömb végéhez kapcsolásával adódik.

➤ A partition fv előfeltétele:

- $1 \leq p < r \leq A.length$ (p, r a fv-en belül konstansok.)

➤ A partition fv második ciklusának invariánsa:

- $A[p..(r-1)] + x$ egy permutációja az $A_0[p..r]$ résztömbnek \wedge
- $p \leq i < j \leq r \wedge A[p..(i-1)] \leq x \wedge$
- $A[i..(j-1)] \geq x$

➤ A partition fv utófeltétele:

- $A[p..r]$ az $A_0[p..r]$ permutációja \wedge
- $p \leq i \leq r \wedge A[p..(i-1)] \leq A[i] \wedge$
- $A[(i+1)..r] \geq A[i]$, ahol i a visszatérési érték.

➤ A 2. ciklus cseréi:

	p		i	j				r	
A :	5	3	8	1	6	4	7		$x = 5$

	p			i	j	j		r	
A :	5	3	1	8	6	4	7		$x = 5$

	p				i		j	j=r	
A :	5	3	1	4	6	8	7		$x = 5$

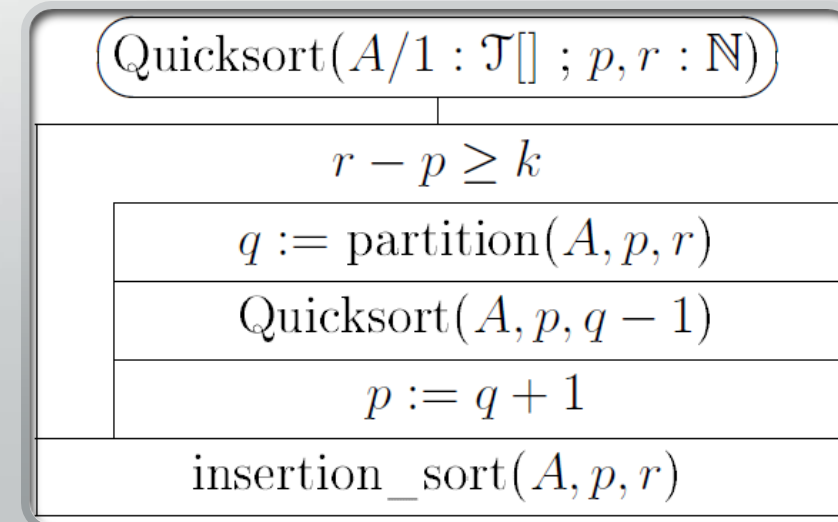
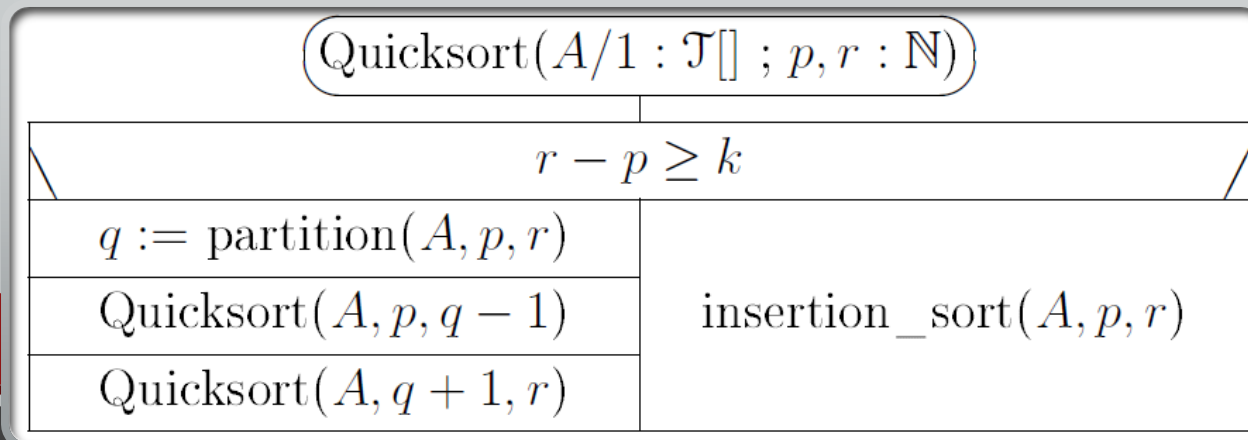
	p				i			j=r	
A :	5	3	1	4	5	8	7	6	


A gyorsrendezés (quicksort) műveletigénye

- A fenti szétvágás (partition) műveletigénye lineáris
 - a két ciklus együtt $r - p - 1$ vagy $r - p$ iterációt végez
- Műveletigény
 - $mT(n), AT(n) \in \Theta(n \log n)$
 - $MT(n) \in \Theta(n^2)$
 - A várható vagy átlagos műveletigény aszimptotikusan a legjobb esethez esik közel
 - a legrosszabb eset valószínűsége nagyon kicsi.

Vegyes gyorsrendezés

- Legfeljebb néhányszor tíz rendezendő elem esetén a beszűrő rendezés hatékonyabb, mint a gyors rendezések (merge sort, heap sort, quicksort)
- Quicksort(A, p, r) eljárás jelentős gyorsítása:
 - kis méretű résztömböknél áttérünk beszűrő rendezésre
 - $k \in \mathbb{N}$ (20 és 40 között)
- Legrosszabb esetének javítása ($n^2 \rightarrow \Theta(n \log n)$)
 - rekurzív eljárásában figyeljük a rekurziós mélységet is
 - pl. $2 \log n$ mélység meghaladása esetén áttérünk valamelyik gyors rendezésre (pl. heap, merge).





Köszönöm a figyelmet!

Pusztai Kinga