Projektovanje algoritama

L06. Quicksort. Linearno sortiranje

Algoritmi sortiranja

Naziv algoritma	Očekivano vreme trajanja	Prostorni zahtev
Insertion Sort	$O(n^2)$	unutar niza
Merge Sort	$O(n \lg n)$	dodatni prostor
Heapsort	$O(n \lg n)$	unutar niza
	$O(n)$	

Quicksort

```
QUICKSORT(A,p,r)
  if p < r
    q = PARTITION(A,p,r)
    QUICKSORT(A,p,q-1)
    QUICKSORT(A,q+1,r)
```

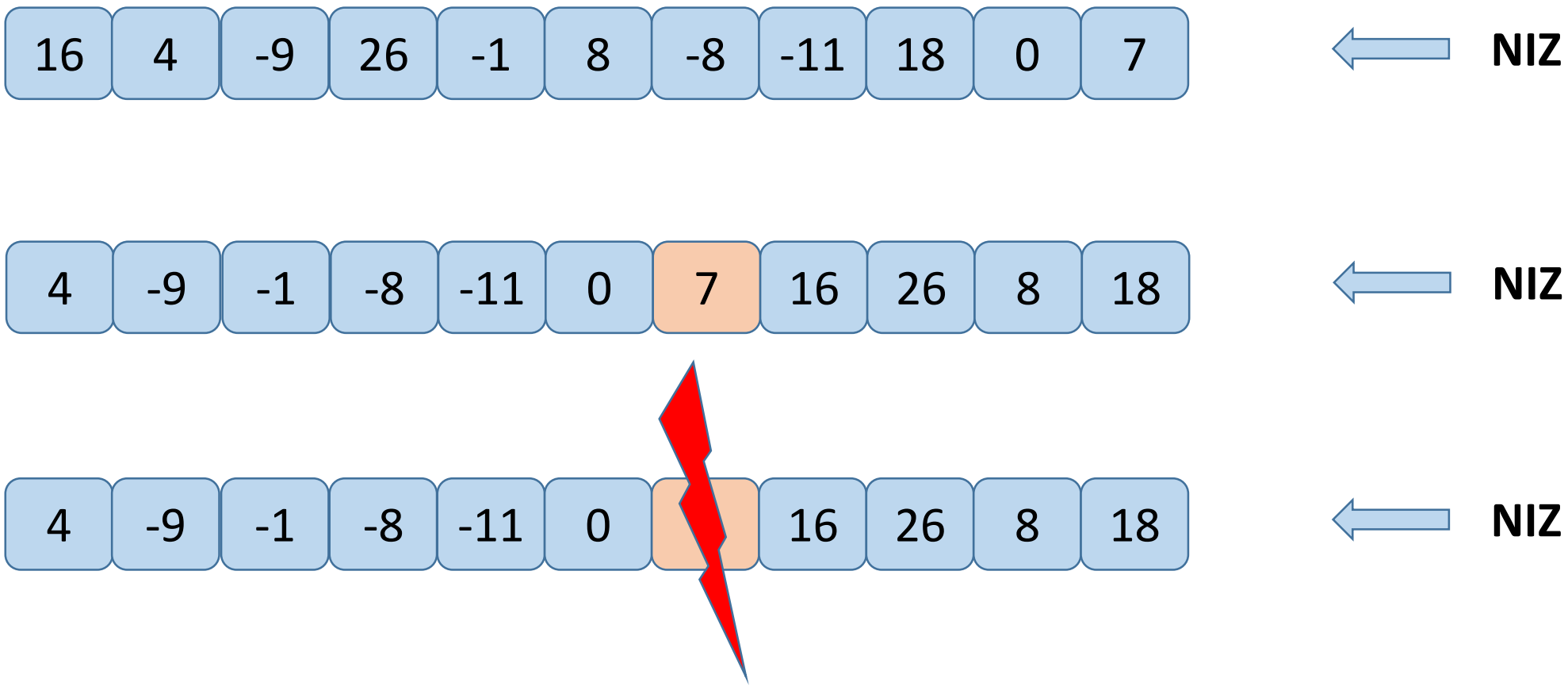
A[q] – PIVOT ELEMENT

Zavadi (*divide*): Podelićemo niz A[p..r] u **tri podniza** A[p..q-1], A[q], A[q+1..r]
Uslov: svi elementi podniza su veći od svih elemenata prethodnog podniza

Vladaj *(conquer):* Sortiraćemo svaki podniz rekurzivno (drugi ima samo jedan element)

***Combine*:** Nema potrebe za kombinovanjem; niz je već sortiran.

Quicksort


16 4 -9 26 -1 8 -8 -11 18 0 7
NIZ
4 -9 -1 -8 -11 0 7 16 26 8 18
NIZ
4 -9 -1 -8 -11 0 16 26 8 18
NIZ

Quicksort - particionisanje

```
PARTITION(A,p,r)
  x = A[r]
  i = p - 1
  for j = p to r - 1
    if A[j] <= x
      i = i + 1
      exchange A[i] with A[j]
  exchange A[i+1] with A[r]
  return i + 1
```

https://visualgo.net/sorting

Quicksort – particionisanje (dokaz tačnosti)

Invarijanta petlje:

- levi podniz: $p \leq k \leq i \;\; \rightarrow \;\; A[k] \leq x$
- desni podniz: $i + 1 \leq k \leq j - 1 \;\; \rightarrow \;\; A[k] > x$
- pivot: $k = r \;\; \rightarrow \;\; A[k] = x$

Inicijalizacija: $i = p - 1, \quad j = p$

Održavanje: proveriti oba slučaja **if** uslova

Terminacija: $j = r$

Quicksort – analiza složenosti

Worst-case

Dešava se kada particionisanje napravi potproblem veličine N-1 i potproblem veličine 0.

- ukoliko pivot bude na kraju!

$$T(n) = T(n-1) + T(0) + \theta(n)$$

$\theta(n)$ - particionisanje

$$\boldsymbol{T(n) = \theta(n^2)}$$

Quicksort – analiza složenosti

Best-case

Dešava se kada particionisanje napravi potprobleme veličine N/2.

- ukoliko pivot bude na sredini u svakoj iteraciji!

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

$$\boldsymbol{T(n) = \theta(n \lg n)}$$

Quicksort – analiza složenosti

Average-case

Koje je onda očekivano vreme trajanja Quicksort algoritma?

Posmatrajmo slučaj particionisanja u odnosu 9:1, odn. da se pivot nalazi blizu kraja.

$$T(n) = T\left(\frac{9n}{10}\right) + T\left(\frac{n}{10}\right) + \theta(n)$$

$$\boldsymbol{T(n) = \theta(n \lg n)}$$

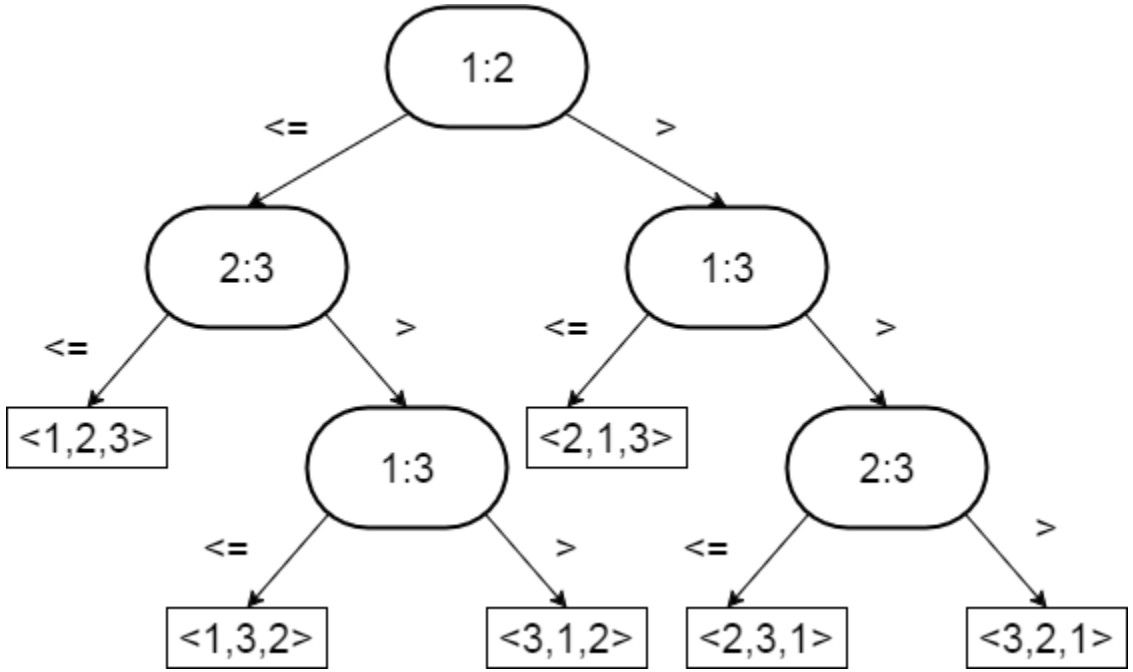
Ista složenost će biti kod bilo kog konstantnog odnosa u proporciji!

Algoritmi sortiranja

Naziv algoritma	Očekivano vreme trajanja	Prostorni zahtev
Insertion Sort	$O(n^2)$	unutar niza
Merge Sort	$O(n \lg n)$	dodatni prostor
Heapsort	$O(n \lg n)$	unutar niza
Quicksort	$O(n \lg n)$	unutar niza
	$O(n)$	

Donja granica za sortiranje?

Stablo odlučivanja



Sortiranja zasnovana na poređenju

$\boldsymbol{\Omega(n \lg n)}$

Linearno sortiranje - Counting Sort

```
COUNTING-SORT(A,B,k)
let C[0..k] be a new array
for i = 0 to k
  C[i] = 0
for j = 1 to A.length
  C[A[j]] = C[A[j]] + 1
for i = 1 to k
  C[i] = C[i] + C[i-1]
for j = A.length downto 1
  B[C[A[j]]] = A[j]
  C[A[j]] = C[A[j]] - 1
```

$$T(n) = \theta(n)$$

Algoritmi sortiranja

Naziv algoritma	Očekivano vreme trajanja	Prostorni zahtev
Insertion Sort	$O(n^2)$	unutar niza
Merge Sort	$O(n \lg n)$	dodatni prostor
Heapsort	$O(n \lg n)$	unutar niza
Quicksort	$O(n \lg n)$	unutar niza
Counting Sort Radix Sort Bucket Sort	$O(n)$	dodatni prostor

thank you!