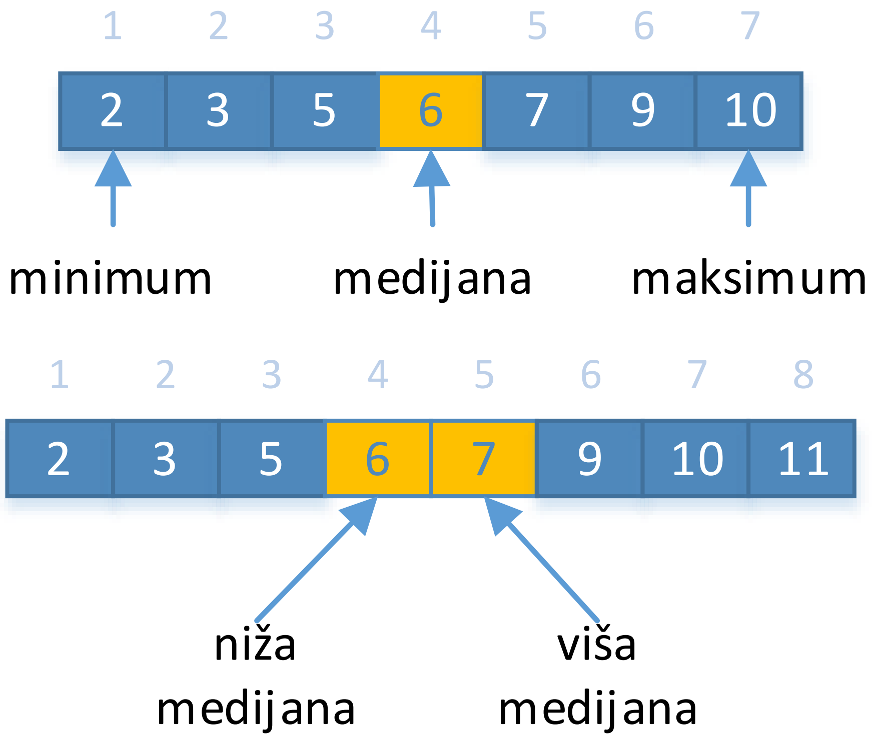
Algoritmi

REDOSLEDNA STATISTIKA

Redosledna statistika (*Order Statistics*)

- Posmatra je: *i*-ti redosledni element iz skupa od *n* elemenata je *i*-ti najmanji elemenat

- Minimum = redosledno 1. elemenat $i = 1$
- Maksimum = redosledno *n*-ti elemenat $i = n$
- Medijana = redosledno na sredini minimuma i maksimuma
  - Za neparan broj elemenata $i = \lceil n/2 \rceil$
  - Za paran broj elemenata
    - Viša medijana: $i = \lfloor (n+1)/2 \rfloor$
    - Niža medijana: $i = \lceil (n+1)/2 \rceil$



Definicija problema

- Problem: Odrediti *i*-ti redosledni elemenat na skupu on *n* različitih brojeva
  - Ulaz: skup A sa *n* različitih elemenata, celobrojna *i* ($\mathrm{i} \leq i \leq n$)
  - Izlaz: elemenat koji je veći od $i - 1$ drugih elemenata

- Ovo je **problem selekcije** koji se može rešiti u dva koraka:
  1. Sortiranjem niza A, i
  2. Izborom *i*-tog elementa
- Prethodno rešenje ima složenost $O(n\log_2 n)$ i asimptotski je neefikasno
- traže se brža rešenja, složenosti $O(n)$

Minimum i maksimum

- Koliko je najmanje operacija poređenja neophodno za pronalaženje mimimuma?
  Odgovor: $n - 1$
  (jer se najmanji broj mora uporediti sa svakim drugim brojem)
  Primetiti da se ovde ne govori o asimtotoskoj složenosti, nego o tačnom broju poređenja.

- Algoritmi:

```
MINIMUM(A)
m = A(1)
for i=2 to A.length
    if A[i] < m
        m = A[i]
return m
```

```
MAKSIMUM(A)
m = A(1)
for i=2 to A.length
    if A[i] > m
        m = A[i]
return m
```

Jednovremeno traženje minimuma i maksimuma

- Da li je potrebno $2(n-1)$ poređenja?
  Odgovor: Nije, treba najviše $3\lfloor n/2 \rfloor$ poređenja.

- Algoritam:
  Umesto poređenja (jednog) *i*-tog elementa sa tekućom najmanjom i najvećom vednosti, treba posmatrati po 2 elementa (parove elemenata) iz niza A:
  1. međusobno se uporede elementi u paru
  2. manji se poredi sa tekućim minimumom, a
  3. veći se poredi sa tekućim maksimumom.

Ukupno 3 poređenja
za svaka 2 elementa

Selekcija

- Problem selekcije *i*-tog elementa je teži od traženja minimuma ili maksimuma.
- Posmatramo dva algoritma selekcije:
  - SELEKTUJ-SA-ZAMENOM je modifikovana verzija QUICKSORT algoritma jer se rekurzivno particionisanje odnosi samo na jednu stranu (polovinu) niza A gde se nalazi traženi elemenat
    - To čini da je očekivana složenost algoritma $\Theta(n)$ (pod uslovom da su svi elementi različiti).
  - SELECT algoritam vrši selekciju u linearnom vremenu za najgori slučaj.

Selektuj-sa-Zamenom algoritam

```
Selektuj-sa-Zamenom(A,p,r,i)
if p==r
    return A[p]
q = Podeli-sa-Zamenom(A,p,r)
k = q – p + 1
if i==k
    return A[q]
elseif i < k
    return Selektuj-sa-Zamenom(A,p,q-1,i)
else return Selektuj-sa-Zamenom(A,q+1,r,i-k)
```

```
Podeli-sa-Zamenom(A,p,r)
i = Random(p,r)
A[r] ↔ A[i]
return Podeli(A,p,r) // vidi Quicksort
```

Vreme izvršavanja SELEKTUJ-SA-ZAMENOM

- Najgori slučaj ima složenost $\Theta(n^2)$, ali samo kada se podele dešavaju oko najvećeg (ili najmanjeg) elementa.
  - Ovo je analizirano kod *Quick sort* algoritma
- Slučajan izbor pivota (koja je posledica poziva RANDOM funkcije) pomaže jer se izbegavaju nepovoljni ulazi (kada je ulaz sortiran).
- Može se pokazati da je očekivano vreme izvršavanja ovog algoritma $\mathrm{E}\{T(n)\} = O(n)$ (Ovde je $E\{.\}$ matematičko očekivanje, tj. sredja vrednost višestrukog ponavljanja algoritma.)

- Očekivano vreme izvršavanja statistike bilo kog reda je linearno, podrazumevajući da su svi elementi različiti.

SELEKTUJ algoritam

SELEKTUJ algoritam (SELECT):

1. Podeli se A na $\lceil n/5 \rceil$ grupa, gde svaka grupa ima 5 elemenata (poslednja može imati manje elemenata)
2. Nađe se medijana u svakoj od grupa upotrebom *insertion sort*-a
3. Upotrebi se SELEKTUJ rekurzivno na novi niz sastavljen od medijana svih grupa iz prethodnog koraka i nađe se x
4. Primeni se modifikovan PODELI oko medijane-medijana (x iz koraka 3) koja je k-ti najmanji element
   – Levu grupu čine elementi $1..k-1$
   – Desnu grupu čine elementi $k+1..n$
5. Ako je $i == k$ tada je rešenje pronađeno, inače nastavi se rekurzivno sa levom grupom za $i < k$, ili se traži $(i-k)$-ti element u desnoj grupi (gde je $i > k$).

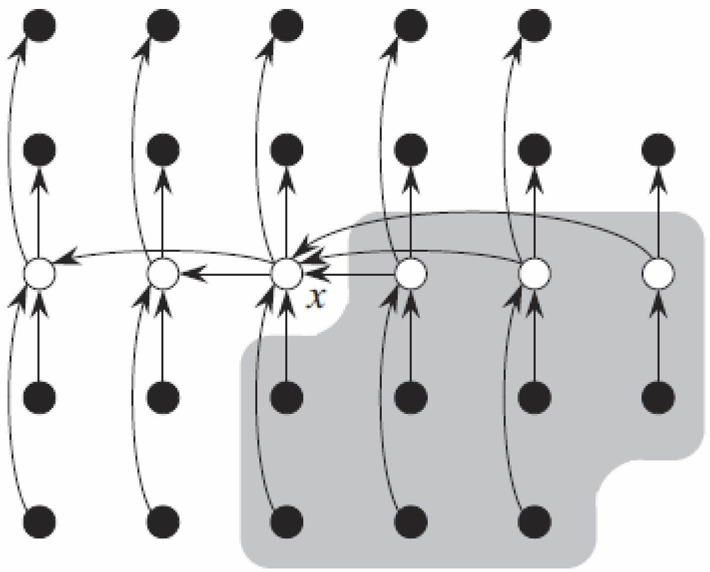
Vreme izvršavanja Selektuj

- Koliko je elemenata veće od x?
Pola grupa ima medijanu veću od x, a to znači da 3 elementa u tim grupama ima vrednost veću od x (sem 2 grupe: poslednje i gde je x), tj. $3\left(\left\lceil\frac{1}{2}\left\lceil\frac{n}{5}\right\rceil\right\rceil-2\right)\geq\frac{3\mathrm{n}}{10}-6.$
- Elemenata manjih od x ima: $n-\left(\frac{3\mathrm{n}}{10}-6\right)=\frac{7n}{10}+6.$
- Vreme izvršavanja koraka 1, 2 i 4 je $O(n)=an$, gde je a konstanta (ovde spada i vreme insertion sort-a koje smatramo konstantnim za sortiranje 5 elemenata); korak 3 rekurzivni poziv $T(\lceil n/5\rceil)$, i korak 5 je takođe rekurzivni poziv $T(7n/10+6)$. Sve zajedno daje:

$$T(n)=\begin{cases} O(1), n<n_0 \\ T\left(\left\lceil\frac{n}{5}\right\rceil\right)+T\left(\frac{7n}{10}+6\right)+O(n), n\geq n_0\end{cases}$$

$$T(n)\leq c\left\lceil\frac{n}{5}\right\rceil+c\left(\frac{7n}{10}+6\right)+an\leq\frac{cn}{5}+c+\frac{7cn}{10}+6c+an$$
$$=\frac{9cn}{10}+7c+an=cn+\left(-\frac{cn}{10}+7c+an\right)\leq cn$$

- $-\frac{cn}{10}+4c+an\leq 0$ daje $c=10a\,\frac{n}{n-70}\leq 20a$ za $n\geq 140$
- Znači: za $n\geq n_0=140$, $T(n)\leq cn$ (izbor za n_0 nije suštinski bitan jer se „samo“ dobija drugačija konstanta c)



Kružići su elementi
Beli kružići su medijane grupa
x je medijana-medijana
Strelice pokazuju na veći elemenat

Zaključak: Dokazana je složenost $O(n)$, što je efikasnije od „najboljeg sortiranja“, čija je složenost $\Omega(n\log_2 n)$, pa uzimanja i-tog elementa po redu.