Predavanje 4

Donja međa za sortiranje

Može li se sortirati brže od $O(n \lg n)$?

__Def.__: Algoritmi za sortiranje koji koriste operacije $<, <=, >, >=, =$ (Operatore usporedbe) da bi uspostavili relativan odnos među elementima nazivamo 'comparison sort' algoritmima

Napomena: Svi algoritmi do sada su bili 'comparison sort'

__TEOREM__: Svaki 'comparison based' algoritam zahtjeva $\Omega(n \lg n)$ u najgorem slučaju

Posljedica ove tvrdnje:

Merge Sort je optimalan alg. (najbolji mogući po dizajnu) za sortiranje

__Def.__: (Model usporedbe / Comparison model / decision tree model) -

- binarno stablo gdje su unutrašnji vrhovi labelirani kao $a_i \leq a_j$, (a_i, a_j i-ti i j-ti ulazni elementi)

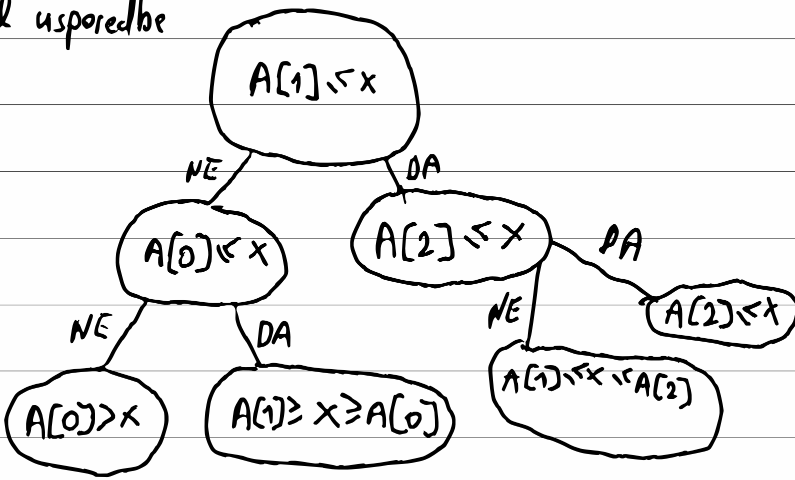
Izvršavanje algoritma odgovara jednom korijen - list putu

Listovi sadrže rezultat računanja alg.

Primjer: binarno pretraživanje $n = 3,\ x$

$A = $ [0 | 1 | 2]

Model usporedbe



- $\underbrace{\text{visina stabla}}_{\text{duljina najdužeg korijen-list puta}}$ (height) odluke odgovara $\underbrace{\text{vremenu izvršavanja alg.}}_{\text{WORST\_CASE}}$

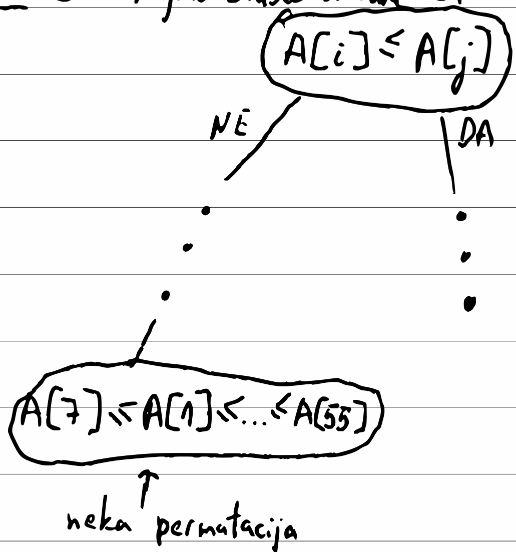
LEMA: BS je $\Omega(\lg n)$

DOKAZ:

br. listova stabla je $\approx n \Rightarrow$ visina stabla najmanje $\lg n$

Dokaz teorema:

<u>IDEJA</u>: Skicirajmo stablo odluke za 'comparison sort' alg.



$A[1 \ldots n]$

Zanima nas visina ovog stabla?

$\#$ listova iznosi $n!$ (broj svih permutacija n elemenata polja A)

Visina stabla (barem) $\geq \lg(n!) = \lg(n(n-1)(n-2)\cdot\ldots\cdot 2\cdot 1)$

$$= \sum_{i=1}^{n} \lg i \geq \sum_{i=\frac{n}{2}}^{n} \lg i \geq \sum_{i=\frac{n}{2}}^{n} \lg \frac{n}{2}$$

$$= \sum_{i=\frac{n}{2}}^{n} (\lg n - 1) = \frac{n}{2}\lg n - \frac{n}{2} \Rightarrow \text{VISINA STABLA} \in \Omega(n \lg n)$$

Sortiranje u linearnom vremenu

Pretpostavka: elementi $a_1, \dots, a_n$ koje sortiramo su integeri

COUNTING SORT:

- $a_1, \dots, a_n$ dolaze iz skupa $\{0, 1, 2, \dots, k\}$

TVRDNJA: Za "nepreveliki" k cijele brojeve možemo sortirati u $\Theta(n)$ vremenu

IDEJA: Alocirati pomoćno polje L veličine $(k+1)$ u kojem ćemo brojati broj pojavljivanja pojedinog integera

Primjer: 1 4 3 3 2 3 4 1 $\quad \{1, 2, 3, 4\}$

L duljine 5

	0	1	2	3	4
L:	0	2	1	3	2

Pseudokod

VSA

```
L = polje od k praznih lista                  → Θ(k)
for i in range(n):                            } → Θ(n)
|   L[A[j]].append(A[j])                      }
output = []                                   →
for i in range(k):                            } → Θ(k+n)
        output.extend(L[i])                   }
```

VSA

$\Rightarrow \Theta(n+k)$

tj. ovaj alg. je $\Theta(n)$ za $k = O(n)$

RADIX SORT alg.:

- za $k = n^{O(1)}$ Radix Sort sortira u $\Theta(n)$ vremenu

Pretp. da su brojevi $a_1, \dots, a_n$ zapisani u bazi b

$\Rightarrow$ broj znamenaka $d \ll \log_b k$

Ideja algoritma je sljedeća:

- sortiraj brojeve po najmanje značajnoj znamenci (najdesnija)
- ⋮

INTEGERI

- $\underbrace{\text{sortiraj brojeve}}_{\text{COUNTING SORT}}$ po $\underbrace{\text{najviše značajnoj znamenci}}_{\text{najljevija}}$

Primjer:

	CS → SČ		CS → SČ		COUNT SORT → SAČUVAJ UREĐAJ	
123		472		212		123
472		322		216		132
322		132		322		212
376		212		123		216
216		123		132		322
132		376		472		376
212		216		376		472

Vremenska složenost: $\underbrace{\log_b k}_{\#\text{znamenaka}} \cdot \underbrace{O(n+b)}_{\text{COUNTING SORT}}$

$O((n+b)\log_b k) \to$ koju bazu odabrati t.d. ovaj izraz bude minimalan

$k = n^c$, za neki konst. C

$\Rightarrow O(n \log_n n^c) = O(n)$

Amortizirana analiza

- želimo spremiti n različitih elemenata u polje A, a n je nepoznanica

DINAMIČKO POLJE:

- želimo da polje $A \in \Theta(n)$

Klasa 'vektor' C++ STL

Primjer:

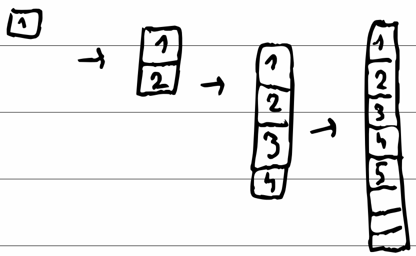
INSERT(1)

INSERT(2)

INSERT(3)

INSERT(4)

INSERT(5)



Zanima nas VSA nakon n ISERT operacija:

Neka c_i = broj operacija pri ubacivanju i-tog inserta (cijena i-tog inserta)

$$c_i = \begin{cases} i, & \text{akoliko } (i-1) \text{ potencija broja } 2 \\ 1, & \text{inače} \end{cases}$$

i-ti INSERT	1	2	3	4	5	6	7	8	9		
veličina	1	2	4	4	8	8	8	8	16	...	n
c_i	1	1+1	2+1	1	4+1	1	1	1	8+1		

move insert

NAS ZANIMA

$$\sum_{i=1}^{n} c_i = n + \underbrace{\sum_{i=0}^{\lfloor \log(n-1) \rfloor} 2^i}_{\text{SPA1 ☺}} \leq n + 2n = 3 \cdot n$$

GEOMETRIJSKI RED

VSA od n INSERT operacija je $\Theta(n)$

$\frac{3n}{n} \approx \boxed{3}$ INSERT u AMORTIZIRANOM SMISLU