Oblikovanje i analiza algoritama

Matej Mihelčić

Prirodoslovno-matematički fakultet, Sveučilište u Zagrebu

matmih@math.hr

20. listopada, 2023.

Primjeri rekurzivnih algoritama

- Rekurzivno traženje najvećeg (najmanjeg) elementa u polju.
- binarnom stablu traženja
- Binarno pretraživanje
- Rekurzivno množenje matrica:
  - Strassenov algoritam
  - Winogradov algoritam
- Množenje velikih brojeva (podijeli pa vladaj)
- FFT (Fast Fourier Transform)
- Algoritmi sortiranja
- Dinamičko programiranje
  - ulančano množenje matrica
  - neprekidni podvektor maksimalne sume

Matrično množenje - rekurzivno

- Pretpostavka: $n = 2^k$
- Rekurzivni algoritam:
  - $$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},\ B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},\ C = A \cdot B,$$
  - $T(n) = \begin{cases} c_1, & n = 2 \\ 8T(\frac{n}{2}) + c_2 \cdot n^2, & n > 2 \end{cases}$
  - $T(n) \in \Theta(n^{log_2 8})$
- Strassenov algoritam (Volker Strassen, 1969.)
  - $M_1 = (A_{11} + A_{22})(B_{11} + B_{22}),\ M_2 = (A_{21} + A_{22})B_{11},$
  - $M_3 = A_{11}(B_{12} - B_{22}),\ M_4 = A_{22}(B_{21} - B_{11}),\ M_5 = (A_{11} + A_{12})B_{22},$
  - $M_6 = A_{21} - A_{11}B_{11} + B_{12},\ M_7 = A_{12} - A_{22}B_{21} + B_{22}$
  - $C_{11} = M_1 + M_4 - M_5 + M_7,\ C_{12} = M_3 + M_5,$
  - $C_{21} = M_2 + M_4,\ C_{22} = M_1 + M_3 - M_2 + M_6$
  - $T(n) = \begin{cases} c_1, & n = 2 \\ 7T(\frac{n}{2}) + c_2 \cdot n^2, & n > 2 \end{cases}$
  - $T(n) \in \Theta(n^{log_2 7}) = \Theta(n^{2.80735})$

Matrično množenje - rekurzivno

- Shmuel Winograd
  - $M_1 = (A_{21} + A_{22} - A_{11})(B_{22} - B_{12} + B_{11}),\ M_2 = A_{11}B_{11},$
  - $M_3 = A_{12}B_{21},\ M_4 = (A_{11} - A_{21})(B_{22} - B_{12}),$
  - $M_5 = (A_{21} + A_{22})(B_{12} - B_{11}),\ M_6 = (A_{12} - A_{21} + A_{11} - A_{22}) \cdot B_{22}$
  - $M_7 = A_{22} \cdot (B_{11} + B_{22} - B_{12} - B_{21}),$
  - $C_{11} = M_2 + M_3,\ C_{12} = M_1 + M_2 + M_5 + M_6,$
  - $C_{21} = M_1 + M_2 + M_4 - M_7,\ C_{22} = M_1 + M_2 + M_4 + M_5$
  - $T(n) = \begin{cases} c_1, & n = 2 \\ 7T(\frac{n}{2}) + c_2 \cdot n^2, & n > 2 \end{cases}$
  - $T(n) \in \Theta(n^{log_2 7}) = \Theta(n^{2.80735})$
  - Računanjem uz spremanje međurezultata u memoriju se broj zbrajanja matrica može spustiti na 15 (taj broj je jednak 18 kod Strassenovog algoritma).

Matrično množenje - rekurzivno

- Coppersmith - Winograd (1986.), $\mathcal{O}(n^{2.376})$ (kombinatorna konstrukcija, metoda lasera, rekurzija).
- Williams, Xu, Xu i Zhou (2023.), $\mathcal{O}(n^{2.371552})$ (poboljšana metoda lasera + rekurzija i kombinatorna konstrukcija).

Množenje brojeva - rekurzivan algoritam

- A. Karatsuba (1960.):
  - $X,\ Y$ su n znamenkasti, veliki cijeli brojevi (zapisani u bazi b).
  - $X = \sum_{i=0}^{n-1} x_i b^i = x_1 \cdot b^k + x_0$, ($x_1$ i x_0 imaju po $n/2 = k$ znamenki).
  - $x_1 = \sum_{i=0}^{k-1} x_{i+k} b^i,\ x_0 = \sum_{i=0}^{k-1} x_i b^i$
  - Analogno za Y:
  - $Y = \sum_{i=0}^{n-1} y_i b^i = y_1 \cdot b^k + y_0,\ y_1 = \sum_{i=0}^{k-1} y_{i+k} b^i,\ y_0 = \sum_{i=0}^{k-1} y_i b^i.$
  - $Z = X \cdot Y$ u toj bazi ima oblik:
  - $Z = (x_1 \cdot y_1) \cdot b^{2k} + (x_1 \cdot y_0 + x_0 \cdot y_1) \cdot b^k + y_0 \cdot x_0$
  - $x_1 \cdot y_0 + x_0 \cdot y_1 = (x_0 + x_1) \cdot (y_0 + y_1) - x_1 \cdot y_1 - x_0 \cdot y_0$
  - Trebamo izračunati 3 umnoška $\approx \frac{n}{2}$ znamenkastih brojeva.
  - $p = x_1 \cdot y_1,\ q = x_0 \cdot y_0,\ r = (x_0 + x_1) \cdot (y_0 + y_1).$
  - Tada: $Z = p \cdot b^{2k} + (r - p - q) \cdot b^k + q$
  - $T(n) = 3 \cdot T(\frac{n}{2}) + c_0 n,\ n > 1.$
  - $T(n) \in \Theta(n^{log_2 3})$

Zadatak

Treba naći lokalni maksimum (tzv. **vrh**) u polju s n elemenata, odnosno u matrici s $n \times n$ elemenata, gdje je n zadani prirodni broj. Elementi su (primjerice) cijeli brojevi. Dodatno, pretpostavljamo da su svi elementi u polju/matrici međusobno **različiti**. Složenost algoritma mjerimo brojem usporedbi elemenata u polju/matrici. U oba slučaja treba pokazati da algoritam radi korektno i da zadovoljava traženu složenost.

a) U polju A s n elemenata. element $A[i]$ je **vrh** ako je strogo veći od oba susjedna elementa $A[i-1]$ i $A[i+1]$. Ako je $A[i]$ na rubu polja, dovoljno je da je strogo veći od onog susjeda koji se nalazi unutar polja, uz $n \geq 2$.

   Sastavite algoritam koji nalazi neki vrh u polju A i vraća njegov indeks i. Složenost algoritma **mora** biti u $\mathcal{O}(log_2 n)$.

b) U matrici A s $n \times n$ elemenata, element $A[i][j]$ je **vrh** ako je strogo veći od sva četiri susjedna elementa $A[i-1][j],\ A[i+1][j],\ A[i][j-1],\ A[i][j+1]$ (dijagonalne susjede ne gledamo).

Zadatak - nastavak

b) Ako je $A[i][j]$ na rubu ili u kutu matrice, dovoljno je da je strogo veći od svih susjeda koji se nalaze unutar matrice (dva ili tri susjeda uz $n \geq 2$.

Sastavite algoritam koji nalazi neki vrh u matrici A i vraća njegove indekse i, j. Složenost algoritma **mora** biti u $\mathcal{O}(nlog_2 n)$.

Može li se pretpostavka o različitosti **svih** elemenata oslabiti a da navedeni algoritmi još uvijek rade korektno?

Zadatak

Zločesti kralj ima podrum s n bačvi vina. Špijun je uspio ubaciti otrov u jednu bačvu, ali kralj ne zna koju. Otrov je vrlo djelotvoran, ali spor - bez obzira na to koliko je otrov razrijeđen. **Bilo koja** količina otrova je dovoljna da ubije neku osobu i osoba umire nakon **točno** 30 dana od konzumacije otrova. Kralj je spreman žrtvovati neki broj svojih robova kao "kušače vina", ali želi što prije saznati koja je bačva otrovana. Na kraljevu žalost, broj robova je bitno manji od broja bačvi u podrumu.

a) Nađite postupak kojim kralj, nakon točno 30 dana (tj. u najkraće moguće vrijeme), može odrediti koja od n bačvi je otrovana, s time da žrtvuje najviše $\mathcal{O}(log_2 n)$ robova.

b) Može li se problem riješiti žrtvujući **manje** robova, ali uz **dulji** rok? Primjerice, je li moguće iskoristiti samo jednog roba? Ako je, koliki je rok u kojem kralj nalazi otrovanu bačvu.

Ako kralj ima 1000 bačvi, a slavlje je planirano za 5 tjedana, je li 8 robova dovoljno da odredi otrovanu bačvu prije tog roka? Ako da, kako?

Sortiranje - uvod

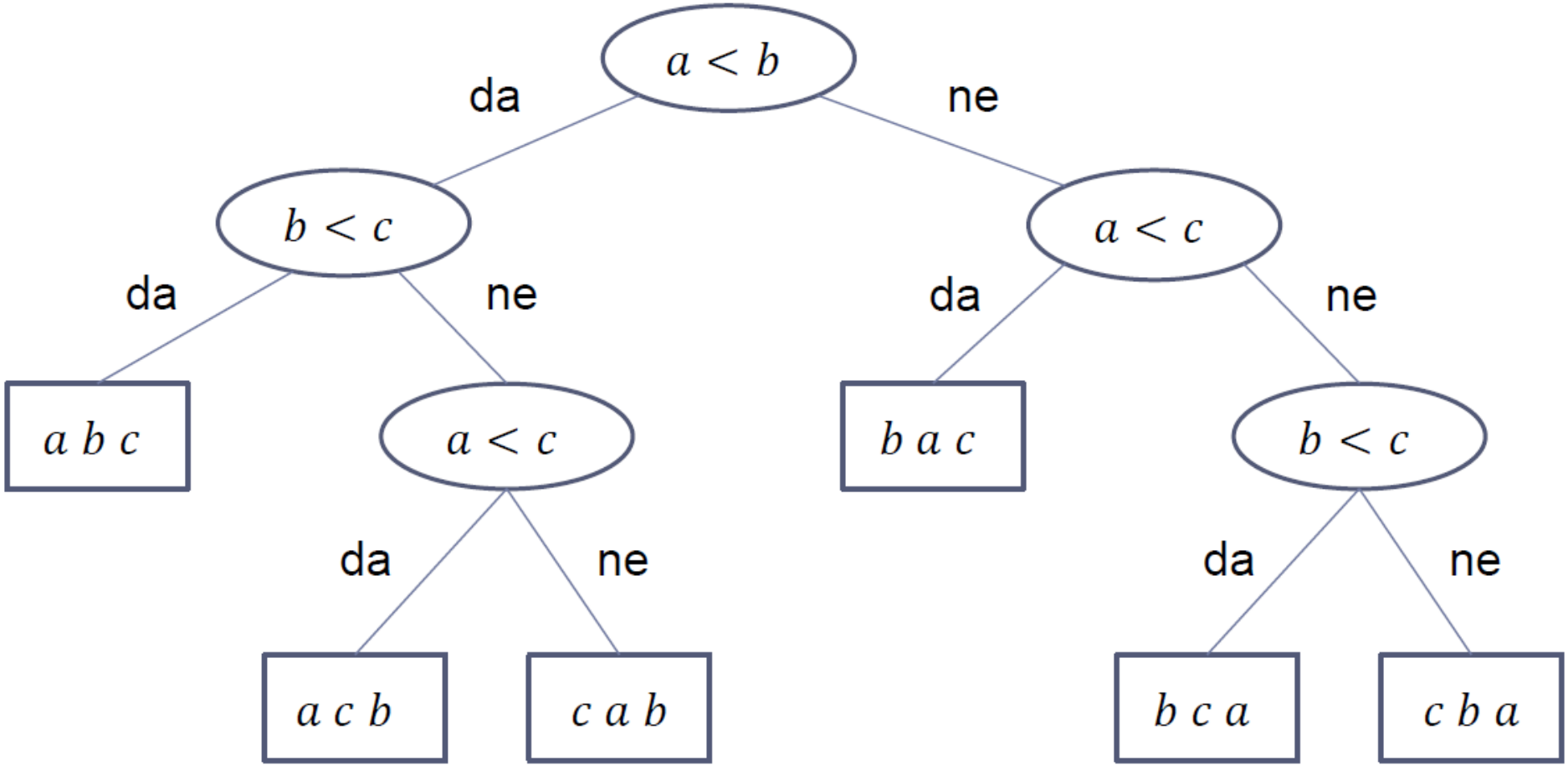
- Primjene:
  - brže pretraživanje: traženje u sortiranom nizu, u najgorem slučaju, ovisi logaritamski o duljini niza.
  - lakše rješavanje nekih problema:
    - minimalno razapinjuće stablo (sortiramo bridove prema težinama uzlazno)
    - median
    - konveksna ljuska (sortiramo točke ulazno prema x koordinatama)
    - najbliži par točaka (sortiramo točke prema x koordinatama)
- Promatrani model sortiranja:
  - niz objekata s pripadnim ključevima
  - definirana je relacija linearnog uređaja na skupu ključeva , u oznaci $\leq$.
  - zadani niz želimo sortirati *uzlazno*.

Sortiranje usporedbom

- Dodatne pretpostavke:
  - pristup do bilo kojeg objekta u nizu je elementarna operacija (unutarnje, interno sortiranje), tj. ne ovisi o položaju objekta u nizu
  - u postupk u sortiranja koristimo najviše konstantnu veličinu dodatne memorije (sortiranje *u mjestu* zamjenama objekata).
- Elementarne operacije
  - usporedba ključeva
  - kopiranje ili pomak objekta
- Donja ograda za složenost sortiranja uspoređivanjem
- Primjer. Neka su a, b i c tri različita ključa:

Sortiranje usporedbom

Stablo odluke (*decision, comparison tree*):



Sortiranje usporedbom

Teorem. Svaki deterministički algoritam sortiranja uspoređivanjem zahtijeva $\Omega(n log_2 n)$ usporedbi u najgorem slučaju.

Skica dokaza:

- h - visina stabla odluke
- l - broj listova

$$n! \leq l \leq 2^h \Rightarrow h \geq log_2 n! \geq n log_2 n - \frac{n}{ln2}$$

Koristimo **Stirlingovu fomulu:** $n! \sim \sqrt{2\pi n}(\frac{n}{e})^n$.