Algoritmi in podatkovne strukture 1

Visokošolski strokovni študij Računalništvo in informatika

Deli in vladaj

Jurij Mihelič, UniLj, FRI

Deli in vladaj

- Divide et impera (*divide & conquer*)
  - Deli:
    - problem **delimo** na
    - **manjše** probleme
    - dokler ne dobimo
    - ***obvladljivega*** problema
  - Vladaj
    - majhne probleme
    - enostavno oz. trivialno rešimo

Gaj Julij Cezar
100 pr. n. št. – 44 pr. n. št.

Deli in vladaj

- Metoda snovanja algoritmov
  - rekurzivni algoritem
- Koraki
  - delitev naloge:
    - na eno ali več **manjših** nalog
  - *reši manjše naloge:*
    - uporaba rekurzije
  - združevanje rešitev:
    - iz rešitev manjših nalog sestavimo rešitev osnovne naloge

Dvojiško iskanje

- Ideja algoritma
  - tabelo delimo na **dve** polovici
  - rekurzija gre le v **eno** polovico
  - vladaj: tabela velikosti 1
  - zahtevnost delitve $O(1)$ in sestavljanja $O(1)$

Dvojiško iskanje (rekurzivno)

```
fun binarySearch(a, left, right, key) is
    if right > left then return -1
    mid = left + (right - left) / 2
    if (key < a[mid]) then
        return binarySearch(a, left, mid - 1)
    if (k > a[mid]) then
        return binarySearch(a, mid + 1, right)
    return mid
```

Dvojiško iskanje

- Časovna zahtevnost
  - Kako pridemo do $\log n$?
- Rekurzivna enačba
  - asimptotična zahtevnost

$$T(1) = \Theta(1)$$

$$T(n) = 1 \cdot T\left(\left\lceil \frac{n}{2} \right\rceil\right) + \Theta(1)$$

  - zapis s konstantami

$$T(1) = a$$

$$T(n) = T\left(\left\lceil \frac{n}{2} \right\rceil\right) + c$$

Dvojiško iskanje

- Reševanje rekurzivne enačbe $T(n) = T(\lceil \frac{n}{2} \rceil) + c$
  - metoda vstavljanja

T(1) = a

T(2) = T(1) + c = a + c

T(3) = T(2) + c = a + 2c

T(4) = T(2) + c = a + 2c

T(5) = T(3) + c = a + 3c

T(6) = T(3) + c = a + 3c

T(7) = T(4) + c = a + 3c

T(8) = T(4) + c = a + 3c

T(16) = T(8) + c = a + 4c

…

Substitucija

- $n = 2^N$
- torej $N = \lg n$

Sklepamo

- $T(2^N) = a + N c$
- $T(n) = a + c \lg n$

Urejanje z zlivanjem

- Ideja algoritma
  - tabelo delimo na **dve** polovici
  - rekurzija gre v **obe** polovici
  - vladaj: tabela velikosti 1
  - zahtevnost delitve $O(1)$ in sestavljanja $O(n)$

Urejanje z zlivanjem

- Časovna zahtevnost
  - Kako pridemo do $n \log n$?
- Rekurzivna enačba
  - asimptotična zahtevnost

$$T(1) = \Theta(1)$$
$$T(n) = 2 \cdot T(\lceil \frac{n}{2} \rceil) + \Theta(n)$$

  - zapis s konstantami

$$T(1) = a$$
$$T(n) = 2 \cdot T(\lceil \frac{n}{2} \rceil) + cn$$

Urejanje z zlivanjem

- Reševanje rekurzivne enačbe $T(n)=2\cdot T(\lceil \frac{n}{2} \rceil)+cn$
  - metoda vstavljanja

$$
\begin{aligned}
T(1) &= a \\
T(2) &= 2a + 2c \\
T(4) &= 4a + 8c \\
T(8) &= 8a + 24c \\
T(16) &= 16a + 64c \\
&\dots
\end{aligned}
$$

$$T(n) = n\,a + n \lg n\, c = \quad (n \lg n)$$

Reševanje rekurenčnih enačb

- Metoda substitucije
  - guess the form of the solution
  - verify by induction
  - solve for constants

TODO

- TODO:
  - morda še ena, ki je z vstavljanjem težko rešljiva
  - T(n) = 3T(n/4) + cn^2
  - lahko pa narediš drevo rekurzije
  - v vozliščih zahtevnost podproblema
  - nato seštevaš vse skupaj, geometrijska vrsta
  - in pride O(n^2)
  - oz. ker je cn^2, torej Omega(n^2)

Mojstrov izrek (master theorem)

- Rekurzivna enačba

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c\,n^d$$

a – *število podnalog (podproblemov)*

b – *faktor delitve naloge*

c – *konstanta iz asimptotične notacije*

d – *red velikost zahtevnosti delitve in združevanja.*

Mojstrov izrek

- Rekurzivna enačba in rešitev

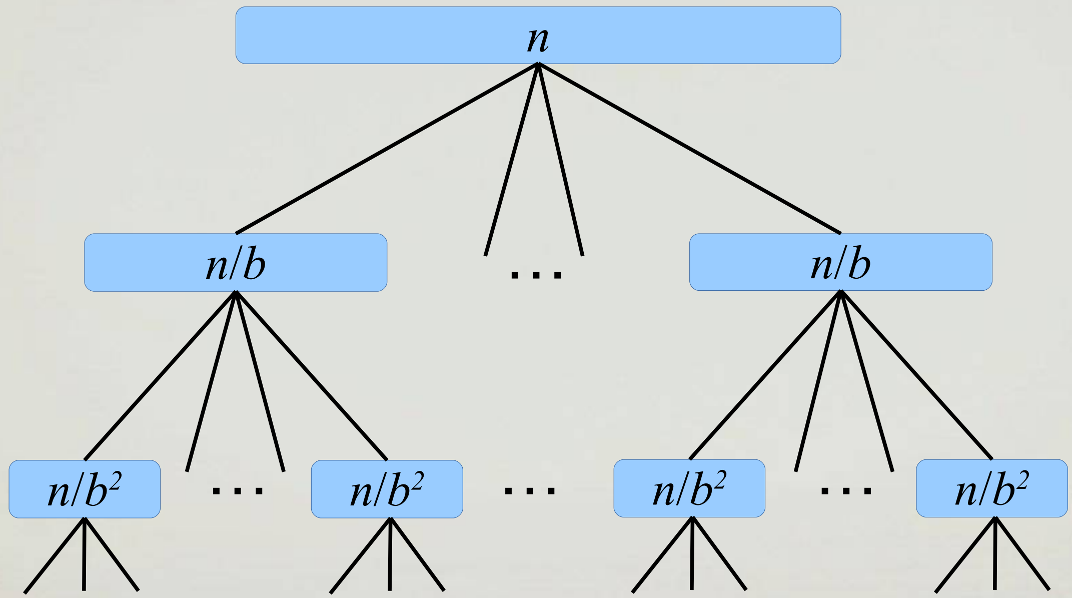
$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c\,n^d$$

$$T(n) = \begin{cases} \Theta\left(n^d\right) & a < b^d \\ \Theta\left(n^d \log n\right) & a = b^d \\ \Theta\left(n^{\log_b a}\right) & a > b^d \end{cases}$$

Mojstrov izrek

- Intuicija

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + c\,n^d$$



Mojstrov izrek

- Sile zla a vs sile dobrega b^d

$$\left(\frac{a}{b^d}\right)^i n^d$$

  - $a < b^d \Rightarrow \Theta(n^d)$
    - večja globina ► manj dela ► največ dela v korenu
  - $a = b^d \Rightarrow \Theta(n^d \log n)$
    - na vsaki globini je enako dela
  - $a > b^d \Rightarrow \Theta(n^{\log_b a})$
    - večja globina ► več dela ► največ dela v listih

Mojstrov izrek

$$T(n) = \begin{cases} \Theta\left(n^d\right) & a < b^d \\ \Theta\left(n^d \lg n\right) & a = b^d \\ \Theta\left(n^{\log_b a}\right) & a > b^d \end{cases}$$

- Primeri
  - dvojiško iskanje: $(1,2,0)$
  - kopica – dvigovanje: $(1,2,0)$
  - kopica – ugrezanje: $(1,3/2,0)$
  - urejanje z zlivanjem: $(2,2,1)$
  - štetje inverzij: $(2,2,1)$
  - hitro urejanje (najboljši primer): $(2,2,1)$
  - množenje celih števil – D&V: $(4,2,1)$
  - množenje celih števil – Karatsuba: $(3,2,1)$
  - množenje matrik – D&V: $(8,2,2)$
  - množenje matrik – Strassen: $(7,2,2)$

Množenje velikih celih števil

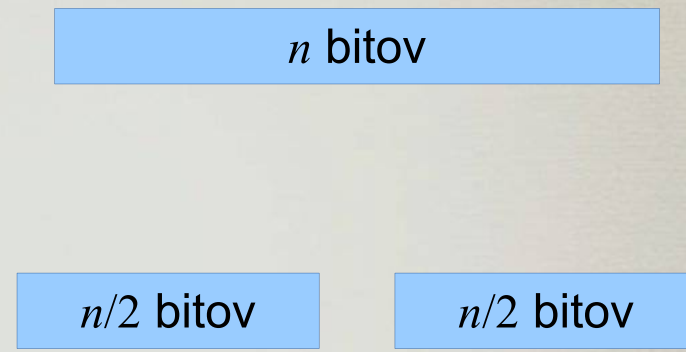
- Velika cela števila
  - Kdaj je število veliko?
  - Kje potrebujemo tako velika števila?
    - kriptografija
    - iskanje praštevil in drugi matematični izzivi

Množenje velikih celih števil

- Algoritmi
  - klasično šolsko množenje
  - množenje ruskih/francoskih trgovcev
  - deli & vladaj množenje
  - Karacubov algoritem (angl. Karatsbua)

Množenje velikih celih števil

- Delitev števil
  - n-bitna števila razdelimo na pol
  - na dva $n/2$ bitna dela
    - prva polovica
    - druga polovica



$$a = a_1 \cdot 2^{n/2} + a_0$$

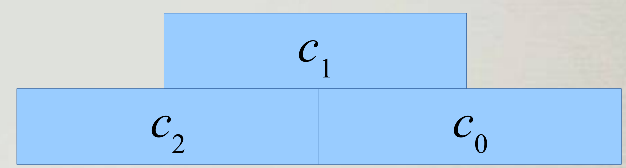
$$b = b_1 \cdot 2^{n/2} + b_0$$

Množenje velikih celih števil

- Direktno D&V množenje
  - produkt

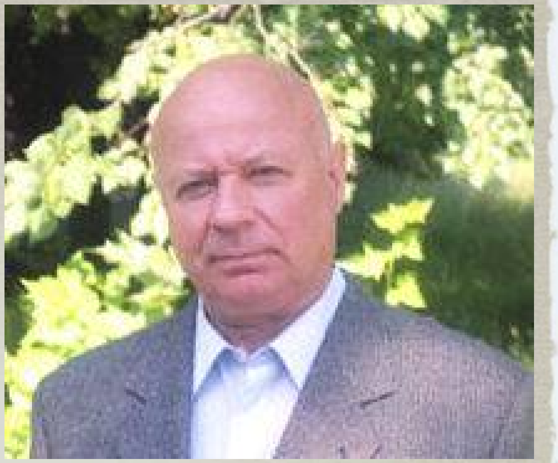
$$\begin{aligned} a\cdot b &= (a_1\cdot 2^{n/2} + a_0)\cdot(b_1\cdot 2^{n/2} + b_0) \\ &= a_1 b_1\cdot 2^n + (a_1 b_0 + a_0 b_1)\cdot 2^{n/2} + a_0 b_0 \\ &= c_2\cdot 2^n + c_1\cdot 2^{n/2} + c_0 \end{aligned}$$

$$\begin{aligned} c_2 &= a_1\cdot b_1 \\ c_1 &= a_1\cdot b_0 + a_0\cdot b_1 \\ c_0 &= a_0\cdot b_0 \end{aligned}$$



  - zahtevnost
    - $O(n^2)$

Množenje velikih celih števil

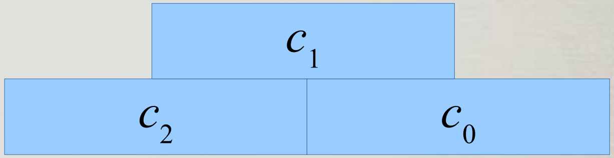


A. A. Karacuba, 1937 – 2008

- Karacubov algoritem
  - produkt

$$
\begin{aligned}
a \cdot b &= (a_1 \cdot 2^{n/2} + a_0) \cdot (b_1 \cdot 2^{n/2} + b_0) \\
&= a_1 b_1 \cdot 2^n + (a_1 b_0 + a_0 b_1) \cdot 2^{n/2} + a_0 b_0 \\
&= c_2 \cdot 2^n + c_1 \cdot 2^{n/2} + c_0
\end{aligned}
$$

  - Gaussov / Karacubov trik

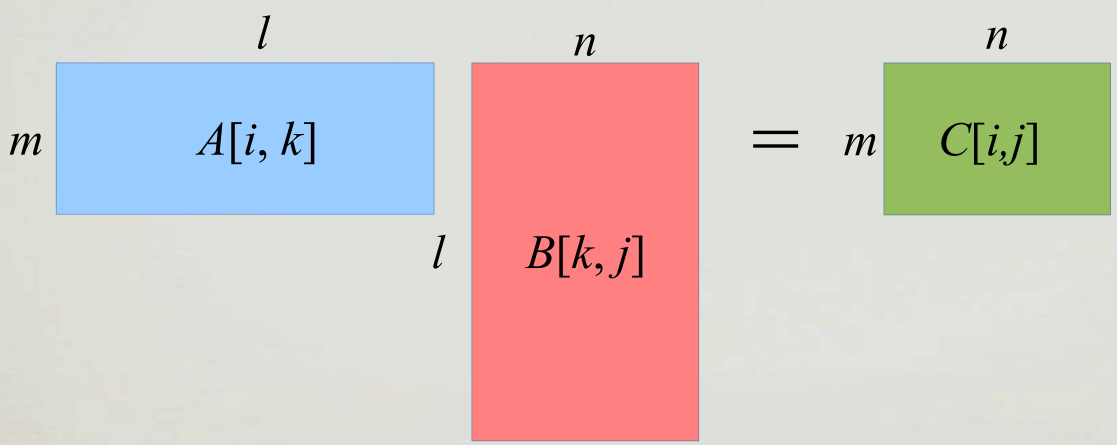


$$
\begin{aligned}
c_2 &= a_1 \cdot b_1 \\
\mathrm{c}_0 &= a_0 \cdot b_0 \\
c_1 &= (a_1 + a_0) \cdot (b_1 + b_0) - c_2 - c_0
\end{aligned}
$$

  - zahtevnost
    - $T(n) = O(n^{\lg 3}) = O(n^{1.585})$

Množenje matrik

- Problem
  - dani sta dve matriki A in B
  - iščemo njun produkt $C = AB$



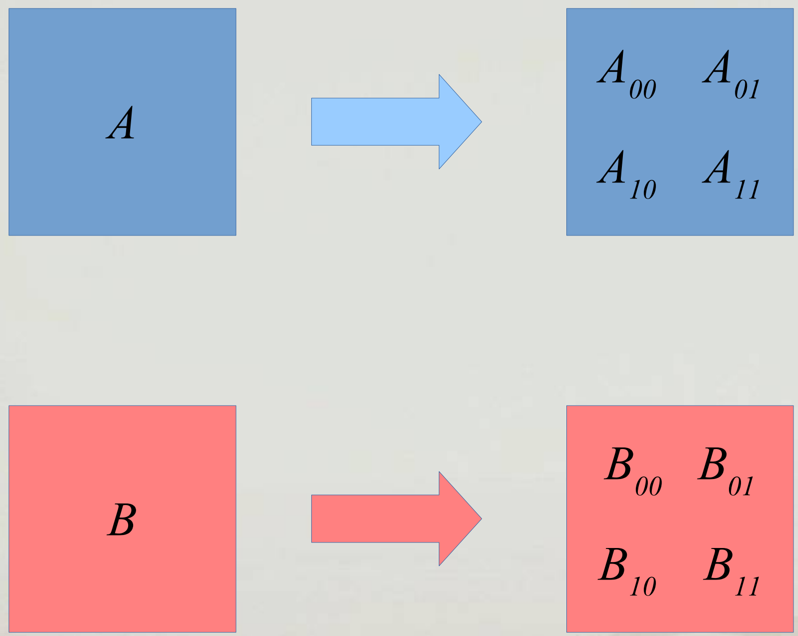
Množenje matrik

- Klasični algoritem
  - tri zanke for
  - zahtevnost $O(n^3)$

```
for i = 0 to m-1 do
   for j = 0 to n-1 do
      s = 0
      for k = 0 to l do
         s += A[i, k] * B[k, j]
      C[i, j] = s
   endfor
endfor
```

Množenje matrik

- Bločne operacije
  - delitev matrik na (disjunktne) podmatrike – bloke



Množenje matrik

- Bločno seštevanje
  - $C_{00} = A_{00} + B_{00}$
  - $C_{01} = A_{01} + B_{01}$
  - $C_{10} = A_{10} + B_{10}$
  - $C_{11} = A_{11} + B_{11}$

$$\begin{bmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} + \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix}$$

Množenje matrik

- Bločno množenje
  - $C_{00} = A_{00}B_{00} + A_{01}B_{10}$
  - $C_{01} = A_{00}B_{01} + A_{01}B_{11}$
  - $C_{10} = A_{10}B_{00} + A_{11}B_{10}$
  - $C_{11} = A_{10}B_{01} + A_{11}B_{11}$

$$\begin{bmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} \cdot \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix}$$

Množenje matrik

- Direktno D&V množenje
  - bločno delitev izvajamo rekurzivno

$$\begin{bmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} \cdot \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix}$$

  - produkti
    - $C_{00} = A_{00}B_{00} + A_{01}B_{10}$
    - $C_{01} = A_{00}B_{01} + A_{01}B_{11}$
    - $C_{10} = A_{10}B_{00} + A_{11}B_{10}$
    - $C_{11} = A_{10}B_{01} + A_{11}B_{11}$
  - zahtevnost
    - $O(n^3)$

Množenje matrik

V. Strassen, 1936

- Strassenov algoritem

$$\begin{bmatrix} C_{00} & C_{01} \\ C_{10} & C_{11} \end{bmatrix} = \begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix}$$

  - produkti
    - $C_{00} = M_1 + M_4 - M_5 + M_7$
    - $C_{01} = M_3 + M_5$
    - $C_{10} = M_2 + M_4$
    - $C_{11} = M_1 + M_3 - M_2 + M_6$

$$\begin{aligned}
M_1 &= (A_{00} + A_{11})\,(B_{00} + B_{11}) \\
M_2 &= (A_{10} + A_{11})\,B_{00} \\
M_3 &= A_{00}\,(B_{01} - B_{11}) \\
M_4 &= A_{11}(B_{10} - B_{00}) \\
M_5 &= (A_{00} + A_{01})\,B_{11} \\
M_6 &= (A_{10} - A_{00})\,(B_{00} + B_{01}) \\
M_7 &= (A_{01} - A_{11})\,(B_{10} + B_{11})
\end{aligned}$$

  - zahtevnost
    - $O(n^{2.808})$

Množenje matrik

- Strassenov algoritem v praksi
  - velika konstanta
  - prostorska potratnost
  - naivna in bločna metoda je numerično stabilnejša
  - matrike niso vedno velikosti $n = 2^k$
  - ustavljanje rekurzije pri majhnih matrikah
    - preklop na drug algoritem
  - upoštevanje predpomnilnika (transponiranje matrik)
  - glej tudi članek: Matrix Multiplication: Practical Use of a Strassen-like Algorithm, Rozman Mitja and Elеršič Miha

Množenje matrik

- Asimptotično najhitrejši algoritmi: $O(n^{\alpha})$
  - naivno množenje: $\alpha = 3$
  - Strassen, $\alpha = 2.\mathbf{808}$
  - Coopersmith, Winograd, 1990, $\alpha = 2.37\mathbf{6}$
  - Stothers, 2010, $\alpha = 2.37\mathbf{4}$ $(\alpha = 2.3736)$
  - Williams, 2011, $\alpha = 2.37\mathbf{3}$ $\mathbf{(}\alpha = 2.37286\mathbf{42)}$
  - Le Gall, 2014, $\alpha = 2.37286\mathbf{39}$