SLOŽENOST ALGORITAMA I ODABRANE METODE OPTIMIZACIJE

ZA STUDENTE ELEKTROTEHNIKE

Branko Malešević i Ivana Jovović

Glava 1

1. Složenost algoritama

1.1 Složenost algoritama

1.1.1 Veliko *O* notacija

Veliko O notacija koristi se za procenu efikasnosti algoritama. Neka su date dve funkcije $f : \mathbb{N}_0 \to \mathbb{R}$ i $g : \mathbb{N}_0 \to \mathbb{R}$. Kažemo da je $g(n) = O(f(n))$ ako postoji konstanta $C \in \mathbb{N}$ takva da za skoro svako $n \in \mathbb{N}_0$ važi da je $|g(n)| \leq C\,|f(n)|$. Počevši od nekog prirodnog broja vrednost funkcije g u nekoj tački nije veća od vrednosti funkcije f u toj tački pomnožene sa konstantom C. Oznaka $O(f(n))$ se odnosi na klasu funkcija, tako da je $g(n) = O(f(n))$ drugi zapis za $g(n) \in O(f(n))$. Oznaka $O(1)$ odnosi se na klasu ograničenih funkcija. Lako se pokazuje da važi da je $O(f(n)) + O(g(n)) = O(f(n) + g(n))$ i da je $O(f(n)) \cdot O(g(n)) = O(f(n) \cdot g(n))$. Uobičajeno je da osnovne aritmetičke operacije smatramo elementarnim koracima. Broj elementarnih koraka predstavlja složenost algoritma i zgodono je da se izrazi notacijom O. Najbolje je da nam složenost nekog algoritma bude konstanta, ali kod realnih problema to se retko dešava. Klase algoritama sublinearne složenosti npr. $O(\ln n)$ ili $O(\sqrt{n})$ smatraju se vrlo efikasnim. Tu spadaju algoritmi čija je složenost manja od $O(n)$. Jedan takav primer je binarna pretraga. Zatim imamo klasu algoritama linearne složenosti $O(n)$. Iza toga dolaze algoritmi superlinearne složenosti npr. $O(n \ln n)$ ili polinomske složenost $O(n^k)$. Ovi algoritmi se smatraju dovoljno dobrim za implementaciju na računaru. Eksponencijalne složenosti $O(e^n)$ i $O(n^n)$ ili faktorijelna složenost $O(n!)$ su nepraktične za implemetaciju.

■ **Primer 1.1** Neka su date funkcije $f : \mathbb{N}_0 \to \mathbb{R}$ i $g : \mathbb{N}_0 \to \mathbb{R}$:

(a) $f(n) = \dfrac{n^2 - n}{2}$ i $g(n) = 2n$;

(b) $f(n) = 2n + \sqrt{n}$ i $g(n) = n^2$;

(c) $f(n) = n + \log_2 n$ i $g(n) = n\sqrt{n}$.

Odrediti koja od relacija $g(n) = O(f(n))$ ili $f(n) = O(g(n))$ važi.

■ **Primer 1.2** Poređati funkcije:

(a) $f_1(n) = \sqrt{n}$, $f_2(n) = n$, $f_3(n) = 2^n$, $f_4(n) = n \log_2 n$, $f_5(n) = 2n^2$, $f_6(n) = n^2 + \log_2 n$;

(b) $f_1(n) = n^{\frac{1}{3}}$, $f_2(n) = (\log_3 n)^3$, $f_3(n) = 3^n$, $f_4(n) = n!$, $f_5(n) = \left(\frac{1}{3}\right)^n$, $f_6(n) = 3$

u niz po efikasnosti izvršavanja, tj. tako da je svaka funkcija u nizu u klasi veliko O od sledeće funkcije.

Zadatak 1.1 Odrediti složenost algoritma za izračunavanje sume i proizvoda prvih n brojeva.

Rešenje. Ako za elementarni korak uzimamo sabiranje, složenost algoritma za računanje sume prvih n brojeva $S_n = \sum_{k=1}^{n} k = 1 + 2 + \ldots + n$ jednaka je $n - 1 = O(n)$. Ukoliko podrazumevamo da su množenje i deljenje takođe elementarni koraci za računanje sume prvih n brojeva možemo iskoristiti formulu $S_n = \sum_{k=1}^{n} k = \frac{n(n+1)}{2}$, i dobijamo da je složenost jednaka $2 = O(1)$ (koristimo jedno množenje i jedno deljenje). Ovo razmatranje se može primeniti na računanje sume bilo koje aritmetičke progresije. Kod računanja proizvoda prvih n brojeva, $n!$, imamo $n - 1$ množenje i složenost je $n - 1 = O(n)$.

□

Zadatak 1.2 Odrediti složenost algoritma za izračunavanje sume prvih n kvadrata.

Rešenje. Složenost algoritma za računanje sume prvih n kvadrata $K_n = \sum_{k=1}^{n} k^2 = 1^2 + 2^2 + \ldots + n^2$ jednaka je $n - 1 + n = 2n - 1 = O(n)$, $n - 1$ sabiranje i n množenja. Za računanje sume prvih n kvadrata možemo iskoristiti formulu $K_n = \sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$, i dobijamo da je složenost jednaka $2 + 1 + 2 + 1 = 6 = O(1)$ (koristimo tri množenje, dva sabiranja i jedno deljenje).

□

Zadatak 1.3 Odrediti složenost algoritma za izračunavanje sume geometrijske progresije zadate svojim prvim članom i količnikom.

Rešenje. Neka je a_1 prvi član, a q količnik geometrijske progresije, k-ti član progresije je oblika $a_k = a_1 q^{k-1}$. Složenost njegovog izračunavanja je $k - 1$, ako smatramo da je elementaran korak množenje. Da bismo sračunali sumu prvih n članova geometrijske progresije $Q_n = \sum_{k=1}^{n} a_k = \sum_{k=1}^{n} a_1 q^{k-1}$ potrebno nam je $n - 1$ sabiranje i $\sum_{k=0}^{n-1} k = \frac{n(n-1)}{2}$ množenja. Složenost algoritma je $n - 1 + \frac{n(n-1)}{2} = \frac{(n+2)(n-1)}{2} = O(n^2)$. Ukoliko iskoristimo formulu za računanje sume geometrijske progresije $Q_n = \sum_{k=1}^{n} a_k = \sum_{k=1}^{n} a_1 q^{k-1} = a_1 \frac{q^n - 1}{q - 1}$ dobijamo da je složenost jednaka $1 + (n - 1) + 2 + 1 = n + 3 = O(n)$ (imamo n množenja, dva oduzimanja i jedno deljenje).

□

Zadatak 1.4 Neka je dat niz $a_1, a_2, \ldots, a_n$. Odrediti složenost algoritma za izračunavanje sume svih podnizova $a_1, a_2, \ldots, a_k$, $k \leq n$, datog niza.

Rešenje. Suma S svih podnizova $a_1, a_2, \ldots, a_k$, $k \leq n$, datog niza jednaka je

$$S = a_1 + (a_1 + a_2) + \ldots + (a_1 + a_2 + \ldots + a_n).$$

Složenost izračunavanja sume proizvoljnog podniza $a_1, a_2, \ldots, a_k$, $k \leq n$ jednaka je $k-1$. Broj podnizova je jednak n, pa je složenost za izračunavanje sume S jednaka $n-1+\sum_{k=1}^{n}(k-1) = n-1+\frac{n(n-1)}{2} = \frac{(n-1)(n+2)}{2} = O(n^2)$. Sa druge strane, u sumi S možemo pregrupisati članove na sledeći način $S = n\,a_1 + (n-1)\,a_2 + \ldots + a_n$. U ovom slučaju imamo $n-1$ sabiranje i $n-1$ množenje, što nam daje složenost $2(n-1) = O(n)$.

□

Zadatak 1.5 Odrediti složenost algoritma za izračunavanje proizvoda dve kvadratne matrice reda n.

Rešenje. Neka su date matrice $A = [a_{ij}]_{n\times n}$ i $B = [b_{ij}]_{n\times n}$. Matrica $C = [c_{ij}]_{m\times k}$ jednaka je proizvodu matrica A i B ako i samo ako važi da je $m = k = n$ i za svako i i j, $1 \leq i, j \leq n$, važi da je $c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$. Za računanje proizvoljnog elementa c_{ij} matrice C, potrebno je n množenja pošto je svaki sabirak u prethodnoj sumi proizvod dva broja, i $n-1$ sabiranje pošto navedena suma ima n sabiraka. Broj elementa u matrici C je jednak n^2. Prema tome, ukupan broj elementarnih koraka, pod kojim u ovom zadatku podrazumevamo sabiranje i množenje, je $n^2(n+n-1) = 2n^3 - n^2 = O(n^3)$. Složenost algoritma za množenje matrica je $O(n^3)$.

□

Zadatak 1.6 Odrediti složenost Strassen-ovog algoritma za izračunavanje proizvoda dve kvadratne matrice reda n.

Rešenje. Neka su A i B dve kvadratne matrice reda $n = 2^k$, za neko $k \in \mathbb{N}$. Ukoliko red matrica A i B nije stepen dvojke, dopunimo date matrice vrstama i kolonoma čiji su svi elementi jednaki 0 tako da dobijemo matrice čiji je red 2^k, za $k \in \mathbb{N}$ za koje važi $2^{k-1} < n \leq 2^k$. Podelimo matrice A, B i $C = A \cdot B$ na blokove jednakih dimenzija

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}, \quad C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},$$

gde su A_{ij}, B_{ij} i C_{ij}, za $1 \leq i, j \leq 2$, matrice reda 2^{k-1}. Imamo da je

$$\begin{aligned} C_{11} &= A_{11} \cdot B_{11} + A_{12} \cdot B_{21} \\ C_{12} &= A_{11} \cdot B_{12} + A_{12} \cdot B_{22} \\ C_{21} &= A_{21} \cdot B_{11} + A_{22} \cdot B_{21} \\ C_{22} &= A_{21} \cdot B_{12} + A_{22} \cdot B_{22}. \end{aligned}$$

Kod računanja matrice C algoritmom podeli pa vladaj broj množenja i sabiranja elementata matrica A i B ostao je nepromenjen u odnosu na standardno množenje. Zaista, primetimo da u ovom slučaju imamo 8 množenja i 4 sabiranja matrica reda 2^{k-1}. Ako označimo sa $T\left(2^k\right)$ složenost množenja dve matrice reda 2^k dobijamo da je $T\left(2^k\right) = 8T\left(2^{k-1}\right) + 4\left(2^{k-1}\right)^2 = 8T\left(2^{k-1}\right) + 4^k$. Nastavljajući ovaj postupak imamo da je $T\left(2^k\right) = 8T\left(2^{k-1}\right) + 4^k = 8\left(8T\left(2^{k-2}\right) + 4^{k-1}\right) + 4^k = 8^2 T\left(2^{k-2}\right) + 4^k(1+2)$, odnosno $T\left(2^k\right) = 8^i T\left(2^{k-i}\right) + 4^k \sum_{j=0}^{i-1} 2^j$, za $1 \leq i \leq k$. Odakle zaključujemo

da je $T\left(2^k\right) = 8^k T(1) + 4^k \sum_{j=0}^{k-1} 2^j = 8^k + 4^k\left(2^k - 1\right) = 2^{3k+1} - 2^{2k} = 2n^3 - n^2 = O(n^3)$. Kako je složenost sabiranja dve kvadratne matrice reda n jednaka $O(n^2)$, a množenja $O(n^3)$, ako bismo smanjili broj množenja blokova reda 2^{k-1} nauštrb sabiranja, smanjili bismo složenost izračunavanja matrice C. Definišemo nove matrice

$$\begin{aligned}
M_1 &= (A_{11} + A_{22}) \cdot (B_{11} + B_{22}) \\
M_2 &= (A_{21} + A_{22}) \cdot B_{11} \\
M_3 &= A_{11} \cdot (B_{12} - B_{22}) \\
M_4 &= A_{22} \cdot (B_{21} - B_{11}) \\
M_5 &= (A_{11} + A_{12}) \cdot B_{22} \\
M_6 &= (A_{21} - A_{11}) \cdot (B_{11} + B_{12}) \\
M_7 &= (A_{12} - A_{22}) \cdot (B_{21} + B_{22}).
\end{aligned}$$

Izrazimo blokove C_{ij}, $1 \leq i, j \leq 2$, koristeći matrice M_k, $1 \leq k \leq 7$. Imamo

$$\begin{aligned}
C_{11} &= M_1 + M_4 - M_5 + M_7 \\
C_{12} &= M_3 + M_5 \\
C_{21} &= M_2 + M_4 \\
C_{22} &= M_1 - M_2 + M_3 + M_6.
\end{aligned}$$

Strassen-ov algoritma je smanjio broj množenja na 7 i doduše poveća broj sabiranja na 18. Ako označimo sa $S\left(2^k\right)$ složenost množenja dve matrice reda 2^k dobijamo da je

$$S\left(2^k\right) = 7S\left(2^{k-1}\right) + 18\left(2^{k-1}\right)^2 = 7S\left(2^{k-1}\right) + 18 \cdot 4^{k-1}.$$

Nastavljajući ovaj postupak imamo da je

$$\begin{aligned}
S\left(2^k\right) &= 7S\left(2^{k-1}\right) + 18 \cdot 4^{k-1} \\
&= 7\left(7S\left(2^{k-2}\right) + 18 \cdot 4^{k-2}\right) + 18 \cdot 4^{k-1} \\
&= 7^2 S\left(2^{k-2}\right) + 18 \cdot 4^{k-1}\left(1 + \frac{7}{4}\right) \\
&\vdots \\
&= 7^i S\left(2^{k-i}\right) + 18 \cdot 4^{k-1} \sum_{j=0}^{i-1} \left(\frac{7}{4}\right)^j \\
&\vdots \\
&= 7^k S(1) + 18 \cdot 4^{k-1} \sum_{j=0}^{k-1} \left(\frac{7}{4}\right)^j \\
&= 7^k + 18 \cdot 4^{k-1} \frac{\left(\frac{7}{4}\right)^k - 1}{\frac{7}{4} - 1} \\
&= 7^{k+1} - 6 \cdot 4^k = 7 \cdot 7^{\log_2 n} - 6 \cdot n^2 = O(n^{\log_2 7}).
\end{aligned}$$

□

Zadatak 1.7 Odrediti složenost algoritma za izračunavanje n-tog Fibonacci-jevog broja F_n, korišćenjem matrične jednakosti

$$\begin{bmatrix} F_n \\ F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} F_{n-1} \\ F_{n-2} \end{bmatrix}, \quad \begin{bmatrix} F_1 \\ F_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Rešenje. Za izračunavanje n-tog Fibonacci-jevog broja F_n, $n \geq 2$, potrebno je samo jedno sabiranje brojeva F_{n-1} i F_{n-2}. Ako označimo sa $F(n)$ složenost izračunavanja n-tog Fibonacci-jevog broja F_n, imamo da je $F(n) = F(n-1) + F(n-2) + 1$, uvođenjem smene $T(n) = F(n) + 1$ dobijamo $T(n) = T(n-1) + T(n-2)$. Rešenje date homogene linearne diferencne jednačine drugog reda dobijamo rešavanjem odgovarajuće karakteristične jednačine $t^2 - t - 1 = 0$ i iz početnih uslova $T(0) = F(0) + 1 = 1$ i $T(1) = F(1) + 1 = 1$. Koreni karakteristične jednačine su $t_{1,2} = \frac{1 \pm \sqrt{5}}{2}$, pa je opšte rešenje $T(n) = C_1 t_1^n + C_2 t_2^n = C_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + C_2 \left(\frac{1-\sqrt{5}}{2}\right)^n$. Zamenom početnih uslova $T(0) = C_1 + C_2 = 1$ i $T(1) = \frac{1}{2}\left(C_1 + C_2 + (C_1 - C_2)\sqrt{5}\right) = 1$ dobijamo sistem $C_1 + C_2 = 1$ i $C_1 - C_2 = \frac{\sqrt{5}}{5}$, tj. imamo da je $C_1 = \frac{\sqrt{5}+1}{2\sqrt{5}}$ i $C_2 = \frac{\sqrt{5}-1}{2\sqrt{5}}$. Prema tome, $T(n) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^{n+1}$ i složenost izračunanja n-tog Fibonacci-jevog broja F_n je $O\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$.

Ako bismo n-ti Fibonacci-jev broj F_n tražili rešavanjem homogene linearne diferencne jednačine drugog reda $F_n = F_{n-1} + F_{n-2}$ sa početnim uslovima $F_0 = F_1 = 1$, dobili bismo da je $F_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^{n+1}$. Prema tome, potrebno nam je dva puta po $n+1$ množenje i jedno oduzimanje. Složenost računanja Fibonacci-jevih brojeva eksplicitnom formulom je linearna $O(n)$.

□

Zadatak 1.8 Odrediti složenost algoritma za izračunavanje k-tog stepena kvadratne matrice reda n.

Rešenje. Složenost algoritma za množenje dve kvadratne matrice reda n jednaka je $2n^3 - n^2 = O(n^3)$. U postupku određivanja k-tog stepena matrice A izvršavamo $k-1$ množenja. Složenost algoritma za određivanje k-tog stepena matrice reda n je $(k-1)(2n^3 - n^2) = O(kn^3)$. Ako bismo određivali k-ti stepen matrice A reda n kvadriranjem, tj. korisćenjem činjenice da je $A^k = \begin{cases} (A^2)^m, & k = 2m, \\ A(A^2)^m, & k = 2m+1, \end{cases}$ složenost bi bila $(1 + (m-1))(2n^3 - n^2) = O(kn^3)$, odnosno $(1 + 1 + (m-1))(2n^3 - n^2) = O(kn^3)$.

□

Zadatak 1.9 Odrediti složenost algoritma za izračunavanje vrednosti determinante kvadratne matrice reda n.

Rešenje. Determinanta kvadratne matrice $A = [a_{ij}]_{n \times n}$ je broj

$$D_n = \det A = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{vmatrix} = \sum_{(i_1, i_2, \dots, i_n) \in S_n} (-1)^I a_{1i_1} a_{2i_2} \dots a_{ni_n},$$

gde je S_n skup svih permutacija skupa $\{1, 2, \dots, n\}$, a I broj inverzija u permutaciji $(i_1, i_2, \dots, i_n)$. Za elemente i_j i i_k u permutaciji $(i_1, i_2, \dots, i_n)$ kažemo da obrazuju inverziju ako važi $i_j > i_k$ za $1 \leq j < k \leq n$. U daljem razmatranju smatramo da je znak permutacije $(-1)^I$ poznat. Skup S_n ima $n!$ elemenata. U sumi D_n imamo $n!$ sabiraka $(-1)^I a_{1i_1} a_{2i_2} \dots a_{ni_n}$, samim tim $n! - 1$ sabiranje. U

svakom sabirku imamo n činioca, pa je broj množenja po sabirku jednak $n-1$. Kako imamo $n!$ sabiraka broj množenja je $n!\,(n-1)$. Složenost algoritma za izračunavanje vrednosti determinante po definiciji je $n!\,(n-1)+n!-1=nn!-1=O(nn!)$. Možemo zaključiti da zbog velike složenosti nije praktično računati vrednost determinante po definiciji. Vrednost determinate možemo takođe odrediti koristeći Laplace-ov razvoj ili Gauss-ov algoritam. Prema Laplace-ovoj teoremi o razvoju determinante, determinanta D_n može se predstaviti kao $D_n=\sum_{i=1}^{n} a_{ij}A_{ij}$ (razvoj po j-toj koloni) ili $D_n=\sum_{j=1}^{n} a_{ij}A_{ij}$ (razvoj po i-toj vrsti), gde je $A_{ij}=(-1)^{i+j}D_{ij}$ algebarski komplement ili kofaktor elementa a_{ij}, a D_{ij} determinanta koja se dobija iz D_n izostavljanjem i-te vrste i j-te kolone (minor). Laplace-ovim razvojem determinatu D_n računamo kao zbir n determinanata reda $n-1$ pomnoženih odgovarajućim elementima. Prema tome, složenost računanja vrednosti determinate D_n jednaka je zbiru složenosti računanja vrednosti determinanata D_{ij} i $2n-1$ (jer imamo n množenja odgovarajućeg elementa i determinante reda $n-1$ i $n-1$ sabiranje ovakvih proizvoda). Kako imamo n determinanata reda $n-1$ složenost za računaje determinate D_n će biti jednaka proizvodu n i složenosti za računanje vrednosti determinante reda $n-1$. Odakle zaključujemo da smo na ovaj način uspeli da smanjim složenost za izračunavanje vrednosti determinante na $O(n!)$. Ako primenimo Gauss-ov algoritam, pod pretpostavkom da je $a_{11}\neq 0$ imamo

$$D_n=\det A=\begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n}\\ a_{21} & a_{22} & \dots & a_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ a_{n1} & a_{n2} & \dots & a_{nn}\end{vmatrix}=\begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n}\\ 0 & a_{22}-\frac{a_{21}}{a_{11}}a_{12} & \dots & a_{2n}-\frac{a_{21}}{a_{11}}a_{1n}\\ \vdots & \vdots & \ddots & \vdots\\ 0 & a_{n2}-\frac{a_{n1}}{a_{11}}a_{12} & \dots & a_{nn}-\frac{a_{n1}}{a_{11}}a_{1n}\end{vmatrix}.$$

U navedenoj realizaciji prve iteracije Gauss-ovog algoritma sproveli smo sledeće operacije nad elementima; $n-1$ deljenje (prvo smo elemente a_{i1} podelili sa elementom a_{11}, $2\leq i\leq n$), $(n-1)^2$ množenje (zatim smo element $\frac{a_{i1}}{a_{11}}$ pomnožili elementima a_{1j}, $2\leq i,j\leq n$), $(n-1)^2$ oduzimanje (i na kraju smo element $\frac{a_{i1}}{a_{11}}a_{1j}$ oduzimali od elemenata a_{ij}, $2\leq i,j\leq n$). Ukupan broj elementarnih koraka je $n-1+(n-1)^2+(n-1)^2=(n-1)(2(n-1)+1)$. U drugoj iteraciji anuliramo elemente u drugoj koloni, tj. ponavljamo predloženi postupak na minoru koji se dobija brisanjem prve vrste i prve kolone iz dobijene determinante. Broj elementarnih koraka je $n-2+(n-2)^2+(n-2)^2=(n-2)(2(n-2)+1)$. U k-toj iteraciji, $1\leq k\leq n-1$, prateći ovo razmatranje, dobijamo da je broj elementarnih koraka $n-k+(n-k)^2+(n-k)^2=(n-k)(2(n-k)+1)$. Ukupno imamo $n-1$ koraka. Prema tome, složenost algoritma za izračunavanje vrednosti determinante matrice reda n svođenjem na gornje trougaoni oblik je $\sum_{k=1}^{n-1}(n-k)(2(n-k)+1)+n-1=\sum_{k=1}^{n-1}k(2k+1)+n-1=$
$2\frac{(n-1)n(2n-1)}{6}+\frac{n(n-1)}{2}+(n-1)=(n-1)\frac{4n^2+n+6}{6}=O(n^3)$.

□

Zadatak 1.10 Odrediti složenost algoritma za izračunavanje vrednosti tridijagonalne determinante

$$D_n=\begin{vmatrix} a_1 & b_1 & 0 & \dots & 0 & 0\\ c_2 & a_2 & b_2 & \dots & 0 & 0\\ 0 & c_3 & a_3 & \dots & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 0 & 0 & \dots & a_{n-1} & b_{n-1}\\ 0 & 0 & 0 & \dots & c_n & a_n\end{vmatrix}.$$

Rešenje. Ovaj zadatak možemo uraditi na dva načina. Korišćenjem Gauss-ovog algoritama za određivanje vrednosti determinante dobijamo

$$D_n = \begin{vmatrix} a_1 & b_1 & 0 & \dots & 0 & 0 \\ c_2 & a_2 & b_2 & \dots & 0 & 0 \\ 0 & c_3 & a_3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_{n-1} & b_{n-1} \\ 0 & 0 & 0 & \dots & c_n & a_n \end{vmatrix} = \begin{vmatrix} a_1 & b_1 & 0 & \dots & 0 & 0 \\ 0 & a_2 - \frac{c_2 b_1}{a_1} & b_2 & \dots & 0 & 0 \\ 0 & 0 & a_3 - \frac{c_3 b_2}{a_2'} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_{n-1} - \frac{c_{n-1} b_{n-2}}{a_{n-2}'} & b_{n-1} \\ 0 & 0 & 0 & \dots & 0 & a_n - \frac{c_n b_{n-1}}{a_{n-1}'} \end{vmatrix},$$

gde je $a_k' = a_k - \dfrac{c_k b_{k-1}}{a_{k-1}'}$ i $a_1 = a_1'$, $2 \le k \le n$. U svakoj iteraciji Gauss-ovog algoritma imamo jedno množenje, jedno deljenje i jedno oduzimanje. Kako imamo $n-1$ iteraciju složenost ovog algoritma je $3(n-1)$. Za računanje vrednosti determinante potrebno je još da izvršimo $n-1$ množenje elemenata na glavnoj dijagonali, dobijamo složenost $4(n-1) = O(n)$. Ako zadatak rešavamo Laplace-ovim razvojem imamo da je

$$\begin{aligned} D_n = \begin{vmatrix} a_1 & b_1 & 0 & \dots & 0 & 0 \\ c_2 & a_2 & b_2 & \dots & 0 & 0 \\ 0 & c_3 & a_3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_{n-1} & b_{n-1} \\ 0 & 0 & 0 & \dots & c_n & a_n \end{vmatrix} &= a_1 D_{n-1} - b_1 \begin{vmatrix} c_2 & b_2 & \dots & 0 & 0 \\ 0 & a_3 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & a_{n-1} & b_{n-1} \\ 0 & 0 & \dots & c_n & a_n \end{vmatrix} \\ &= a_1 D_{n-1} - b_1 c_2 D_{n-2}. \end{aligned}$$

Označimo sa $T(n)$ broj potrebnih elementarnih koraka za izračunavanje vrednosti determinante D_n. Tada je $T(n) = 3+1+T(n-1)+T(n-2)$, odnosno $T(n)+4 = (T(n-1)+4)+(T(n-2)+4)$. Uvođenjem smene $F(n) = T(n)+4$, dobijamo rekurentnu relaciju za Fibonnaci-jev niz $F(n) = F(n-1)+F(n-2)$. Rešenje date homogene linearne diferencne jednačine drugog reda dobijamo rešavanjem odgovarajuće karakteristične jednačine $t^2 - t - 1 = 0$ i iz početnih uslova $F(1) = T(1)+4 = 0+4 = 4$ i $F(2) = T(2)+4 = 3+4 = 7$. Koreni karakteristične jednačine su $t_{1,2} = \frac{1\pm\sqrt{5}}{2}$, pa je opšte rešenje $F(n) = C_1 t_1^n + C_2 t_2^n = C_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + C_2 \left(\frac{1-\sqrt{5}}{2}\right)^n$. Zamenom početnih uslova $F(1) = \frac{1}{2}\left(C_1 + C_2 + (C_1 - C_2)\sqrt{5}\right) = 4$ i $F(2) = \frac{1}{2}\left(3(C_1+C_2) + (C_1-C_2)\sqrt{5}\right) = 7$ dobijamo sistem $C_1 + C_2 = 3$ i $C_1 - C_2 = \sqrt{5}$, tj. imamo da je $C_1 = \frac{3+\sqrt{5}}{2} = \left(\frac{1+\sqrt{5}}{2}\right)^2$ i $C_2 = \frac{3-\sqrt{5}}{2} = \left(\frac{1-\sqrt{5}}{2}\right)^2$. Prema tome, $F(n) = \left(\frac{1+\sqrt{5}}{2}\right)^{n+2} + \left(\frac{1-\sqrt{5}}{2}\right)^{n+2}$ i složenost izračunanja vrednosti determinante korišćenjem Laplace-ovog razvoja je $O\left(\left(\frac{1+\sqrt{5}}{2}\right)^n\right)$.

□

Zadatak 1.11 Odrediti složenost algoritma za izračunavanje inverzne matrice regularne matrice reda n.

Rešenje. Neka je $A = [a_{ij}]_{n\times n}$ regularna matrica reda n. Adjungovana matrica matrice A je matrica $\operatorname{adj} A = [A_{ji}]_{n\times n}$, gde je A_{ij} algebarski komplement (kofaktor) elementa a_{ij} . Za inverznu matricu matrice A važi $A^{-1} = \dfrac{1}{\det A}\operatorname{adj} A$. Primenom Gauss-ovog algoritma dobili smo da je složenost izračunavanja vrednosti determinante kvadratne matrice reda n jednaka $O(n^3)$. Za računanje adjungovane matrice matrice A potrebno je izračunati n^2 algebarskih komplemenata. Kako je

algebarski komplement do na znak jednak minoru reda $n-1$, imamo da je složenost računanja adjungovane matrice matrice A jednaka $n^2\,O\left((n-1)^3\right) = O(n^5)$. Za računanje inverzne matrice matrice A broj elementarnih koraka je $O(n^3)+O(n^5)+n^2 = O(n^5)$ (potrebno je da izračunamo determinantu i adjungovanu matricu matrice A i da svaki element matrice $\operatorname{adj} A$ podelimo sa $\det A$).

Inverznu matricu matrice A možemo odrediti i Gauss–Jordan-ovom metodom. Gauss–Jordan-ova metoda je u direktnoj vezi sa rešavanjem sistema linearnih algebarskih jednačina. Matrična jednačina $A \cdot X = I$, gde je A data regularna matrica reda n, I jedinična matrica reda n i X nepoznata matrica reda n, ima jedinstveno rešenje $X = A^{-1}$. Datu jednačinu možemo zapisati kao $A \cdot [X_{\downarrow 1}\ X_{\downarrow 2}\ \ldots\ X_{\downarrow n}] = [e_1\ e_2\ \ldots\ e_n]$, gde je $X_{\downarrow i}$ i-ta kolona matrice X, a e_i kolona koja ima sve elemente jednake 0 izuzev elementa u i-toj vrsti koji je jednak 1, $1 \le i \le n$. Dalje imamo $A \cdot [X_{\downarrow 1}\ X_{\downarrow 2}\ \ldots\ X_{\downarrow n}] = [A \cdot X_{\downarrow 1}\ A \cdot X_{\downarrow 2}\ \ldots\ A \cdot X_{\downarrow n}]$. Prema tome, datu jednačinu možemo zapisati kao n sistema linearnih algebarskih jednačina u matričnom obliku

$$\begin{aligned} A \cdot X_{\downarrow 1} &= e_1 \\ A \cdot X_{\downarrow 2} &= e_2 \\ &\vdots \\ A \cdot X_{\downarrow n} &= e_n. \end{aligned}$$

Odakle dobijamo $[X_{\downarrow 1}\ X_{\downarrow 2}\ \ldots\ X_{\downarrow n}] = [A^{-1} \cdot e_1\ A^{-1} \cdot e_2\ \ldots\ A^{-1} \cdot e_n] = A^{-1}$. Datih n sistema možemo rešavati istovremeno korišćenjem elementarnih transformacija vrsta, tj. $[A \mid e_1\ e_2\ \ldots\ e_n] = [A \mid I] \cong \ldots \cong [I \mid A^{-1}]$. Broj elementarnih transformacija vrsta koje je potrebno izvesti da se dobije gornje trougaona matrica G od matrice A korišćenjem Gauss-ove metode jeste $O(n^3)$. Iste transformacije se sprovode i na jediničnoj matrici. Dobijenu matricu od matrice I ovim transformacijama označimo sa B. Imamo da je $[A \mid I] \cong [G \mid B]$. Dalje, da bismo dobili od gornje trougaone matrice G dijagonalnu matricu D potrebno nam je $\displaystyle\sum_{k=0}^{n-1} k = \frac{n(n-1)}{2} = O(n^2)$ deljenja i oduzimanja i $O(n^3)$ transformacija na matrici B. Označimo sa C matricu dobijenu od matrice B ovim transformacijama. Važi da je $[G \mid B] \cong [D \mid C]$. Uz još n deljenja dijagonalnih elemenata matrice D i n^2 deljenja elemenata matrice C dobijamo jediničnu matricu i inverznu matricu matrice A, $[D \mid C] \cong [I \mid A^{-1}]$. Broj elementarnih koraka je $2\,O(n^3)+2\,O(n^2)+O(n^3)+n+n^2 = O(n^3)$.

□

Zadatak 1.12 Odrediti složenost algoritma za rešavanje određenog sistema linearnih algebarskih jednačina koji ima n nepoznatih:

(a) Gauss-ovom metodom;

(b) Cramer-ovim formulama;

(c) nalaženjem inverzne matrice (inverznu matricu matrice sistema A računamo kao $A^{-1} = \dfrac{1}{\det A}\operatorname{adj} A$);

(d) nalaženjem inverzne matrice (inverznu matricu matrice sistema računamo Gauss–Jordan-ovom metodom);

(e) LU dekompozicijom matrice sistema.

Rešenje. □

Zadatak 1.13 Odrediti složenost algoritma za izračunavanje vrednosti polinoma $P_n(x) = \displaystyle\sum_{k=0}^{n} a_k x^k$ u tački $\alpha \in \mathbb{R}$.

Rešenje. Vrednost polinoma $P_n(x) = \sum_{k=0}^{n} a_k x^k$ u tački α jednaka je

$$P_n(\alpha) = \sum_{k=0}^{n} a_k \alpha^k = a_n \alpha^n + a_{n-1} \alpha^{n-1} + \ldots + a_1 \alpha + a_0.$$

U navedenoj sumi imamo $n+1$ sabirak, pa samim tim i n sabiranja. Kako su sabirci stepeni broja α pomnoženi koeficijentima polinoma, broj množenja jednak je $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$. Složenost izračunavanja vrednosti polinoma u tački α jednaka je $n + \frac{n(n+1)}{2} = \frac{n(n+3)}{2} = O(n^2)$.

Drugi način da izračunamo vrednost polinoma $P_n(x) = \sum_{k=0}^{n} a_k x^k$ u tački α jeste primenom Hornerove šeme.

α	a_n	a_{n-1}	$\ldots$	a_1	a_0
	$a_n = b_n$	$\alpha \cdot b_n + a_{n-1} = b_{n-1}$	$\ldots$	$\alpha \cdot b_2 + a_1 = b_1$	$\alpha \cdot b_1 + a_0 = b_0$

Vrednost polinoma $P_n(x)$ u tački α jednaka je b_0. U svakom koraku ovog postupka imamo jedno množenje i jedno sabiranje. Ukupno imamo n koraka, pa je složenost jednaka $2n = O(n)$. □

Zadatak 1.14 Neka su $P_n(x)$ i $Q_m(x)$ dva polinoma stepena n i m, redom, nad poljem realnih brojeva $\mathbb{R}$, $n, m \in \mathbb{N}$, $n \geq m$. Odrediti složenost algoritma za izračunavanje:

(a) zbira i proizvoda polinoma $P_n(x)$i $Q_m(x)$;

(b) količnika i ostatka pri deljenju polinoma $P_n(x)$ polinomom $Q_m(x)$;

(c) Euklidovog algoritma za nalaženje najvećeg zajedničkog delioca polinoma $P_n(x)$i $Q_m(x)$.

(d) koeficijenata polinoma $P_n(x)$ korišćenjem Vijetovih formula, ako su poznati njegovi koreni;

(e) vrednosti prvog izvoda $P_n'(x)$ polinoma $P_n(x)$ u tački $\alpha \in \mathbb{R}$;

(f) vrednosti racionalne funkcije $f(x) = \frac{P_n(x)}{Q_n(x)}$ i njenog prvog izvoda $f'(x)$ u tački $\alpha \in \mathbb{R}$;

Rešenje. □

Zadatak 1.15 Odrediti složenost algoritma za izračunavanje vrednosti polinoma

$$P_n(x,y) = \sum_{0 \leq i+j \leq n} a_{ij} x^i y^j = \sum_{i=0}^{n} \sum_{j=0}^{i} a_{i-j,j} x^{i-j} y^j$$

u tački $(\alpha, \beta) \in \mathbb{R}^2$.

Rešenje. Vrednost polinoma $P_n(x,y) = \sum_{0 \leq i+j \leq n} a_{ij} x^i y^j = \sum_{i=0}^{n} \sum_{j=0}^{i} a_{i-j,j} x^{i-j} y^j$ u tački $(\alpha, \beta) \in \mathbb{R}^2$ jednaka je

$$\begin{aligned} P_n(x,y) = \sum_{i=0}^{n} \sum_{j=0}^{i} a_{i-j,j} x^{i-j} y^j &= \left(a_{n,0} \alpha^n + a_{n-1,1} \alpha^{n-1} \beta + \ldots + a_{1,n-1} \alpha \beta^{n-1} + a_{0,n} \beta^n\right) + \\ \ldots &+ \left(a_{2,0} \alpha^2 + a_{1,1} \alpha, \beta + a_{0,2} \beta^2\right) + \left(a_{1,0} \alpha + a_{0,1} \beta\right) + a_{0,0}. \end{aligned}$$

U navedenoj sumi imamo $\sum_{k=1}^{n+1} k = \frac{(n+1)(n+2)}{2}$ sabirak, pa samim tim i $\frac{(n+1)(n+2)}{2} - 1 = \frac{n(n+3)}{2}$ sabiranja. Kako su sabirci stepeni brojeva α i β pomnoženi koeficijentima polinoma, broj množenja jednak je $\sum_{k=0}^{n} k(k+1) = \sum_{k=0}^{n} k^2 + \sum_{k=0}^{n} k = \frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2} = \frac{n(n+1)(n+2)}{3}$. Složenost izračunavanja vrednosti polinoma u tački α jednaka je $\frac{n(n+3)}{2} + \frac{n(n+1)(n+2)}{3} = \frac{n(2n^2+9n+13)}{6} = O(n^3)$.

□

Zadatak 1.16 Odrediti složenost algoritma za izračunavanje vrednosti polinoma $P_n(x,y,z) = \sum_{0 \le i+j+k \le n} a_{ijk} x^i y^j z^k$ u tački $(\alpha,\beta,\gamma) \in \mathbb{R}^2$.

Rešenje. Broj sabiraka u sumi $\sum_{0 \le i+j+k \le n} a_{ijk} x^i y^j z^k$ jednak je sumi broja kompozicija broja t, $0 \le t \le n$, na tri sabirka koji mogu biti jednaki i 0, tj. $\sum_{t=0}^{n} \binom{t+2}{2}$. Prema tome, imamo $\sum_{t=0}^{n} \binom{t+2}{2} - 1$ sabiranje. Za fiksirano t, $0 \le t \le n$, sabirci su oblika $a_{i,j,k} x^i y^j z^k$, $i+j+k = t$. Broj množenja u jednom takvom sabirku je t. Dakle, ukupan broj množenja je $\sum_{t=0}^{n} \binom{t+2}{2} t$. Aritmetičkih operacija za izračunavanje vrednosti polinoma $P_n(x,y,z)$ u tački (α,β,γ) imamo

$$\sum_{t=0}^{n} \binom{t+2}{2}(t+1) - 1 = \sum_{t=1}^{n+1} \binom{t+1}{2} t - 1 = \sum_{t=1}^{n+1} \frac{(t+1)t^2}{2} - 1 = \sum_{t=1}^{n+1} \frac{t^3}{2} + \sum_{t=1}^{n+1} t^2 + \sum_{t=1}^{n+1} \frac{t}{2} - 1.$$

Za sumu trećih stepena prvih n brojeva važi da je jednaka $\left(\frac{n(n+1)}{2}\right)^2$. Imamo da je

$$\begin{aligned} \sum_{t=1}^{n+1} \frac{t^3}{2} + \sum_{t=1}^{n+1} t^2 + \sum_{t=1}^{n+1} \frac{t}{2} - 1 &= \frac{(n+1)^2(n+2)^2}{8} + \frac{(n+1)(n+2)(2n+3)}{6} + \frac{(n+1)(n+2)}{4} - 1 \\ &= \frac{(n+1)(n+2)(3n^2+17n+24)}{24} - 1. \end{aligned}$$

Složenost izračunavanja vrednosti polinoma $P_n(x,y,z)$ u tački (α,β,γ) jednaka je $O(n^4)$.
Ovo razmatranja možemo uopštiti i na polinome sa proizvoljnim brojem promenljivih. Ako je broj promenljivih jednak $s \in \mathbb{N}$, onda je složenost izračunavanja polinoma u datoj tački jednak $\sum_{t=0}^{n} \binom{t+s-1}{s-1}(t+1) - 1 = O(n^{s+1})$.

□

Zadatak 1.17 Odrediti složenost algoritma za izračunavanje karakterističnog polinoma $\varphi_A(x)$ matrice A reda n, ako se zna da je koeficijent uz x^k u polinomu $\varphi_A(x)$ jednak proizvodu $(-1)^k$ i sume svih glavnih minora reda k matrice A.

Rešenje.

□