Skripta predmeta
Diskretna matematika

Zoran Ognjanović
Matematički fakultet u Beogradu

2010, 2011.

1

Uvod

Pre četrdesetak godina računari su uglavnom bili korišteni kao sredstva za ubrzavanje numeričkog rešavanja raznih inžinjerskih problema. Zbog toga, oblasti matematike koje su bile korisne u tim postupcima pre svega su bile matematička analiza, diferencijane jednačine, numerička analiza itd., koje možemo jednim imenom zvati *kontinualna (neprekidna) matematika*. Vremenom je, međutim, računarstvo postalo nauka za sebe, što je dovelo do toga da neke druge matematičke discipline dobiju veći značaj i u primenama i u obrazovanju. Danas je uobičajeno da se naziv *diskretna matematika* koristi kao zajednički imenitelj tih baziĉnih oblasti.

Diskretna matematika proučava prirodne, cele i racionalne brojeve, diskretne (konačne ili najviše prebrojive) skupove, relacije i funkcije definisane na njima, jezike koji se koriste u matematičkom rezonovanju primenljivom u računarstvu, algoritme itd. Drugim rečima, diskretna matematika se bavi (diskretnim) objektima, njihovim svojstvima i odnosima, koji se danas prihvataju kao osnova za zasnivanje računarstva kao nauke. Primeri pitanja na koje se pri tome nailazi su: za dati skup znaka i datu dužinu, koliko se različitih lozinki može konstruisati, koliko puteva između dve fiksirane tačke postoji u datom grafu, ili koje je ograničenje za dužinu izvršavanja nekog programa.

Polazeći od najčešće postavljenih ciljevi kurseva u ovoj oblasti:

- definisanje važnih diskretnih formalnih sistema i odgovarajućih tehnika zaključivanja i davanje motiva za njihovu upotrebu u konkretnim primenama u računarstvu,
- razumevanje formalnih dokaza na kojima se zasniva matematičko rezonovanje i
- uvođenje opštih algoritamskih postupaka rešavanja problema.

kao osnovne teme čije proučavanje se predlaže uglavnom se navode:

- osnovni matematički koncepti, poput skupova, relacija, funkcija, formalnih sistema i dokaza, ...,
- diskretne strukture, grafovi i drveta, modularna aritmetika,
- apstraktne koncepti za zasnivanje i analizu algoritama,
- tehnike prebrojavanja, diskretne verovatnoća, itd.

2

Skupovi, relacije, funkcije

2.1 Osnovne definicije i primeri

2.1.1 Skupovi

Pojam skupa je jedan od osnovnih u matematici. Ovde ćemo proizvoljan skup shvatati kao *kolekciju* objekata, ili *elemenata*, koji kao zajedničku karakteristiku imaju upravo pripadnost tome skupu.

Koristićemo sledeću notaciju:

- simbol $\in$ označava pripadanje elementa skupu, tj.
  - $a \in A$ znači da element a pripada skupu A, dok
  - $a \notin A$ znači da element a ne pripada skupu A, tj. $\neg(a \in A)$,
- obično se skupovi označavaju velikim (A, B, ...), a elementi malim (a, b, ...) slovima; skupovi prirodnih celih i racionalnih brojeva se označavaju redom sa $\mathbb{N}$ (pretpostavićemo na dalje da je $0 \in \mathbb{N}$), $\mathbb{Z}$ i $\mathbb{Q}$,
- reprezentacija skupa se vrši:
  - ekstenzionalno, tj. navođenjem svih elemenata skupa između vitičastih zagrada, na primer $\{a, b, c\}$, $\{1, 2, 3, \ldots, 17\}$, ili $\{2, 4, 6, \ldots\}$,
  - intenzionalno, tj. navođenjem osobine koju imaju elementi skupa i samo oni, na primer $\{x : P(x)\}$, što se čita kao skup svih x za koje važi $P(x)$,
- simbol $\emptyset$ označava prazan skup, tj. skup koji ne sadrži ni jedan element, odnosno $\forall x (x \notin \emptyset)$.

Primer 2.1.1 Skup $\{1, 2, 3\}$ sadrži tri elementa - brojeve 1, 2 i 3. Skup $\{\{1, 2\}, \{\{3\}, 2\}, \{1\}\}$ sadrži tri elementa (koji su i sami takođe skupovi)

$\{1,2\}$, $\{\{3\},2\}$ i $\{1\}$. U slučaju skupa $\{1,2,3,\ldots,17\}$ oznaka $\ldots$ ima (donekle nepreciznu) namenu da zameni eksplicitno navođenje svih prirodnih brojeva između 3 i 17, dok kod skupa $\{2,4,6,\ldots\}$ zamenjuje (neograničenu) listu svih parnih brojeva većih od 6.

Zapis $\{x : x \text{ je realan broj i } 1 \leq x \geq 2\}$ predstavlja skup svih realnih brojeva između 1 i 2. Intenzionalni $\{x : x \text{ je prirodan broj manji od 100 i kvadrat prirodnog broja }\}$ i ekstenzionalni $\{0,1,4,9,16,25,36,49,64,81\}$ zapisi predstavljaju isti skup.

■

U kontekstu materijala koj se ovde izlaže, posebno značajan vid intenzionalnog predstavljanja skupova je kada se osobina koja karakteriše elemente skupa opisuje nekim postupkom. U primeru 2.1.2 taj postupak je induktivan, tj. zadaje se početni element, kao i način za generisanje svih narednih elemenata.

Primer 2.1.2 Skup $\mathbb{N}$ prirodnih brojeva može se definisati na sledeći način:

1. $0 \in \mathbb{N}$,
2. za bilo koji x, ako je $x \in \mathbb{N}$, onda je i $x+1 \in \mathbb{N}$ i
3. $\mathbb{N}$ sadrži one i samo one x dobijene koracima 1 i 2.

Primetimo da koraci 1 i 2 generišu prirodne brojeve, dok korak 3 obezbeđuje da se recimo 0.5 i a ne mogu naći u skupu. ■

Definicija 2.1.3 Skup A je *podskup* skupa B, a skup B je *nadskup* skupa A, u oznaci $A \subset B$, odnosno $B \supset A$, ako važi da je svaki element skupa A ujedno i element skupa B, odnosno $\forall x(x \in A \rightarrow x \in B)$.

Skupovi A i B su *jednaki*, u oznaci $A = B$, ako je $A \subset B$ i $B \subset A$, odnosno $\forall x(x \in A \leftrightarrow x \in B)$.

Skup A je *pravi podskup* skupa B, ako je $A \subset B$ i nije $A = B$.

$A \not\subset B$ znači da nije $A \subset B$. ■

Za svaki skup A je $\emptyset \subset A$, odakle direktno sledi da su svaka dva prazna skupa međusobno jednaka. Takođe, primetimo da za svaki skup A važi $A \subset A$ i $A = A$. Redosled navođenja elemenata skupa nije od značaja, tako da su skupovi $\{1,2\}$ i $\{2,1\}$ jednaki. Ni višestruko navođenje istog elementa ne utiče na formiranje skupa, tako da je $\{a,b,a\} = \{a,b\} = \{b,a,a,a,a\}$. Da bi se označilo da je A pravi podskup skupa B koristi se i oznaka $A \subsetneqq B$.

Primer 2.1.4 Neka je $A = \{1,2,3\}$, $B = \{1,2,\{1,2,3\}\}$ i $C = \{1,2,3,\{1,2,3\}\}$. Najpre primetimo da su 1, 2 i $\{1,2,3\}$ elementi skupa B, da je $\{1\} \subset B$, ali da $\{1,2,3\}$ nije podskup skupa B, jer $3 \in \{1,2,3\}$, ali $3 \notin B$. Prema tome, važi da $A \in B$, ali i $A \not\subset B$. Sa druge strane je $A \in C$ i $A \subset C$. ■

Definicija 2.1.5 *Partitivni skup* skupa A, u oznaci $\mathbb{P}(A)$, je skup svih podskupova od A, odnosno $\mathbb{P}(A) = \{B : B \subset A\}$. ■

Pošto je za svaki skup A, $\emptyset \subset A$ i $A \subset A$, onda uvek važi $\emptyset \in \mathbb{P}(A)$ i $A \in \mathbb{P}(A)$.

Primer 2.1.6 Neka je $A = \{a, b\}$. Tada je $\mathbb{P}(A) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$. ■

Redosled elemenata u nekom skupu nije od značaja, tako da je $\{a, b\} = \{b, a\}$. Međutim, nekada je bitno da se istakne redosled, na primer tačke sa koordinatama $(6, 2)$ i $(2, 6)$ se razlikuju.

Definicija 2.1.7 *Uređeni par* objekata x i y, u oznaci $\langle x, y\rangle$, je skup

$$\{\{x\}, \{x, y\}\}.$$

Pri tome, x je prva, a y druga koordinata (projekcija, komponenta) uređenog para $\langle x, y\rangle$. ■

Upravo insistiranje na redosledu koordinata uređenih parova, dovodi do važnog svojstva da su dva uređena para jednaka ako i samo ako su im jednake odgovarajuće koordinate, koje je formulisano tvrđenjem 2.1.8.

Teorema 2.1.8 Uređeni parovi $\langle x, y\rangle$ i $\langle a, b\rangle$ su jednaki ako i samo ako važi $x = a$ i $y = b$.

Dokaz. ($\Rightarrow$) Lako se vidi da je bilo koji uređeni par $\langle x, y\rangle$ jednočlan skup $\{\{x\}\}$ ako i samo ako je $x = y$. Neka je dakle $\langle x, y\rangle = \langle a, b\rangle$. Ako je $x = y$, onda je i $a = b$, a zatim i $x = y = a = b$. Pretpostavimo da je $x \neq y$. Tada $\langle x, y\rangle$ sadrži dva elementa - skupove $\{x\}$ i $\{x, y\}$. Sada mora biti i $a \neq b$ jer bi u suprotnom $\langle a, b\rangle$ sadržao jedan element - skup $\{a\}$. Zbog jednakosti $\langle x, y\rangle$ i $\langle a, b\rangle$ i činjenice da jednočlan skup ne može biti jednak dvočlanom, mora važiti da je $x = a$. Takođe je i $\{x, y\} = \{a, b\}$. Sada je $y \in \{a, b\}$. Ako bi bilo $y = a$, onda bi se dobilo $x = y$, što ovde nije slučaj, tako da važi $y = b$.
($\Leftarrow$) Obrnuto, ako važi $x = a$ i $y = b$, trivijalno sledi jednakost uređenih parova. ■

Pojam uređenog para se prirodno uopštava na (uređenu) k-torku objekata, u oznaci $\langle x_1, x_2, \ldots, x_k\rangle$, u kojoj se tačno zna ko je koja od k-koordinata.

Definicija 2.1.9 *Dekartov proizvod* skupova X_1, X_2, …, X_k, u oznaci $X_1 \times X_2 \times \cdots \times X_k$, ili $\times_{i=1}^{k} X_i$, je skup svih uređenih k-torki $\langle x_1, x_2 \ldots, x_k\rangle$, za koje je $x_1 \in X_1$, $x_2 \in X_2$, …, $x_k \in X_k$, tj.

$$X_1 \times X_2 \times \cdots \times X_k = \{\langle x_1, x_2 \ldots, x_k\rangle : x_1 \in X_1, x_2 \in X_2, \ldots, x_k \in X_k\}.$$

■

Ako je $X_1 = X_2 = \ldots = X_k$, onda se $X_1 \times X_2 \times \cdots \times X_k$ obično označava sa X_1^k.

Primer 2.1.10 Neka su skupovi predmeta za izborne blokove u trećem i četvrtom semestru:

- $\mathrm{ML1} = \{\mathrm{DiskretnaMatematika}, \mathrm{FinansijskaMatematika}\}$, i
- $\mathrm{ML2} = \{\mathrm{Filozofija}, \mathrm{ProgramskiPaketiZaMatematiku}\}$.

Tada $\mathrm{ML1} \times \mathrm{ML2}$ predstavlja moguće izbore studenata:

$$\begin{aligned}
\{\langle \mathrm{DiskretnaMatematika} &\ ,\ \mathrm{Filozofija}\rangle,\\
\langle \mathrm{DiskretnaMatematika} &\ ,\ \mathrm{ProgramskiPaketiZaMatematiku}\rangle,\\
\langle \mathrm{FinansijskaMatematika} &\ ,\ \mathrm{Filozofija}\rangle,\\
\langle \mathrm{FinansijskaMatematika} &\ ,\ \mathrm{ProgramskiPaketiZaMatematiku}\rangle\}.
\end{aligned}$$

■

2.1.2 Relacije

Pojmovi relacija i funkcija koje uvodimo u nastavku teksta spadaju takođe u osnovne matematičke koncepte i uopštavaju osobine konretnih relacija i funkcija, poput $\geq$ ili $\ln x$, na konkretnim matematičkim strukturama.

Definicija 2.1.11 Neka su X_1, X_2, …, X_k skupovi. *Relacija* (dužine ili arnosti k, nad skupovima X_1, X_2, …, X_k) je bilo koji podskup Dekartovog proizvoda $X_1 \times X_2 \times \cdots \times X_k$. ■

Primetimo da skupovi $X_1, X_2, \ldots, X_k$ mogu biti i jednaki, u kom slučaju su relacije podskupovi od X_1^k. Ako su elementi $x_1 \in X_1$, $x_2 \in X_2$, …, $x_k \in X_k$ u relaciji $R \subset X_1 \times X_2 \times \cdots \times X_k$, piše se $\langle x_1, x_2 \ldots, x_k\rangle \in R$ ili u prefiksnom zapisu $R(x_1, x_2 \ldots, x_k)$. Posebno interesantne su relacije dužine 2, ili *binarne relacije*, kod kojih se koristi i infiksni zapis oblika aRb ako su a i b u relaciji. Slično kao i kod skupova ako elementi a i b nisu u relaciji R, piše se $\langle a, b\rangle \notin R$, ili $\neg R(a, b)$, odnosno $\neg(aRb)$.

Primer 2.1.12 Neka su A skup reka i B skup država. Jednu binarnu relaciju $R \subset A \times B$ možemo definisati kao: $R(a,b)$ akko reka $a \in A$ protiče kroz državu $b \in B$. Tada je:

- $R(\text{Dunav}, \text{Austrija})$, $R(\text{Dunav}, \text{Mađarska})$ i
- $\neg R(\text{Volga}, \text{Italija})$. ■

Definicija 2.1.13 *Kompozicija* relacija $R \subset A \times B$ i $S \subset B \times C$, u oznaci $S \circ R$ je binarna relacija $\{\langle a, c\rangle \subset A \times C\}$ za koju je $a(S \circ R)c$ ako i samo postoji $b \in B$ tako da $R(a,b)$ i $S(b,c)$. ■

U definiciji 2.1.13 treba obratiti pažnju na redosled navođenja relacija. Naime, iako je reč o kompoziciji relacija R i S, relacija R se navodi "iza" S, odnosno relacija S je u zapisu levo od R.

Primer 2.1.14 Neka je A skup osoba i neka su relacije R i S definisane nad A^2 sa: $R(x,y)$ ako i samo ako je x majka od y, odnosno $S(x,y)$ ako i samo ako je x otac od y. Tada je $x(S \circ R)y$ ako i samo ako postoji $z \in A$ tako da je x majka od z i z je otac od y. Drugim rečima $x(S \circ R)y$ ako i samo ako je x baba po očevoj liniji od y. ■

Interesantni primeri relacija su: univerzalna, prazna, indentična i inverzna. U slučaju relacija arnosti 2 nad skupovima A i B, za njih se koriste oznake:

- za univerzalnu relaciju $U = A \times B$,
- za praznu relaciju (koja ne sadrži ni jedan uređeni par) $\emptyset$,
- za relaciju identiteta $I = \{\langle a, b\rangle : a = b\}$ i
- za inverznu relaciju $R^{-1} \subset B \times A$ relacije $R \subset A \times B$, $R^{-1} = \{\langle b, a\rangle : \langle a, b\rangle \in R\}$.

Neke od posebnih osobina binarnih relacija formalizavane su definicijom 2.1.15.

Definicija 2.1.15 Neka je A skup i binarna relacija $R \subset A^2$. Tada je R:

- *refleksivna* ako i samo ako je $(\forall x \in A)R(x,x)$,
- *irefleksivna* ili *striktna* ako i samo $(\forall x \in A)\neg R(x,x)$,
- *simetrična* ako i samo ako $(\forall x, y \in A)(R(x,y) \to R(y,x))$,
- *antisimetrična* ako i samo ako $(\forall x, y \in A)(R(x,y) \wedge R(y,x) \to x = y)$,

- *tranzitivna* ako i samo ako $(\forall x, y, z \in A)(R(x, y) \wedge R(y, z) \rightarrow R(x, z))$.
  ■

Primer 2.1.16 Neka je $\mathbb{N}$ skup prirodnih brojeva. Tada je relacija $R \subset \mathbb{N}^2$, definisana sa $R(a, b)$ ako i samo ako je $a \leq b$:

- refleksivna, jer $(\forall x \in \mathbb{N}) x \leq x$
- antisimetrična, jer za sve $x, y \in \mathbb{N}$ iz $x \leq y$ i $y \leq x$ sledi $x = y$ i
- tranzitivna, jer za sve $x, y, z \in \mathbb{N}$ iz $x \leq y$ i $y \leq z$ sledi $x \leq z$,

ali nije simetrična, jer na primer važi $1 \leq 2$, ali nije $2 \leq 1$. Slično važi i za relaciju $R \subset \mathbb{P}(A)^2$, definisanu sa $R(x, y)$ ako i samo ako je $x \subset y$, za proizvoljni skup A.

Relacija $P \subset \mathbb{N}^2$, definisana sa $P(a, b)$ ako i samo ako je $a < b$ je striktna, antisimetrična i tranzitivna. Relacija $Q \subset \mathbb{N}^2 \times \mathbb{N}^2$, definisana sa $Q(\langle a, b\rangle, \langle c, d\rangle)$ ako i samo ako je $a+d = b+c$ nije antisimetrična jer važi $Q(\langle 1, 2\rangle, \langle 2, 3\rangle)$ i $Q(\langle 2, 3\rangle, \langle 1, 2\rangle)$, pošto je $1+3 = 2+2$, ali nije $\langle 1, 2\rangle = \langle 2, 3\rangle$. Ova relacija jeste refleksivna, simetrična i tranzitivna. ■

Na osnovu osobina relacija navedenih u definiciji 2.1.15, izdvajaju se neke posebno značajne vrste relacija:

Definicija 2.1.17 Neka je A skup i binarna relacija $R \subset A^2$. Tada je R relacija:

- *ekvivalencije* ako i samo ako je refleksivna, simetrična i tranzitivna,
- *parcijalnog uređenja (poretka)* ako i samo je refleksivna, antisimetrična i tranzitivna,
- *totalnog* ili *linearnog uređenja (poretka)* ako i samo ako je to relacija parcijanog uređenja za koju je $(\forall x, y \in A)(R(x, y) \vee R(y, x))$. ■

Ponekad se antisimetrična i tranzitivna relacija koja je refleksivna naziva relacija *slabog (parcijalnog) uređenja*, dok je antisimetrična i tranzitivna relacija koja je irefleksivna - relacija *striktnog (parcijalnog) uređenja*.

Relacije ekvivalencije

Primeri relacije ekvivalecije su: jednakost u nekom skupu, paralelnost pravih itd.

Primer 2.1.18 Neka su $\mathbb{N}$ i $\mathbb{Z}$ skupovi prirodnih i celih brojeva i $n \in \mathbb{N}$, takav da je $n > 0$. Tada je relacija (kongruencije po modulu n) $\equiv_n \subset \mathbb{Z}^2$, definisana sa

$$a \equiv_n b \text{ ako i samo ako je } a - b = k \cdot n, \text{ za neki } k \in \mathbb{Z}$$

jedna relacija ekvivalencije:

- refleksivnost važi jer je $a - a = 0 \cdot n$,
- simetričnost važi jer iz $a - b = k \cdot n$ sledi da je $b - a = -k \cdot n$ i
- tranzitivnost važi jer iz $a - b = k \cdot n$ i $b - c = l \cdot n$ sledi da je $a - c = (a - b) + (b - c) = k \cdot n + l \cdot n = (k + l) \cdot n$.

Alternativna oznaka za $a \equiv_n b$ je $a \equiv b \pmod{n}$. ■

Definicija 2.1.19 Neka je A skup, $x \in A$ i $R \subset A^2$ relacija ekvivalencije. *Klasa ekvivalencije* elementa x u odnosu na relaciju R, u oznaci $[x]_R$, je skup svih elemenata skupa A koji su u relaciji R sa x, odnosno

$$[x]_R = \{y \in A : xRy\}.$$

Količnički skup skupa A za relaciju ekvivalencije R, u oznaci $A_{/R}$ je

$$A_{/R} = \{[x]_R : x \in A\}.$$

■

Ako se relacija R podrazumeva uobičajeno je da se piše samo $[x]$. Imajući u vidu svojstva relacije ekvivalencije, lako se dokazuje tvrđenje 2.1.20.

Teorema 2.1.20 Neka je $R \subset A^2$ relacija ekvivalencije i $x, y \in A$. Tada je $[x] = [y]$ ako i samo ako $R(x, y)$.

Dokaz. Najpre primetimo da je zbog refleksivnosti za svaki $x \in A$ ispunjeno $x \in [x]$. Prema tome ako je $[x] = [y]$, onda je $y \in [x]$, što znači da je $R(x, y)$. Obrnuto, pretpostavimo da je $R(x, y)$. To znači da je $y \in [x]$. Za svaki $z \in [x]$ zbog simetričnosti i tranzitivnosti važi da je $R(y, x)$ i $R(x, z)$, pa i $R(y, z)$, odakle je $z \in [y]$ i $[x] \subset [y]$. Na sličan način se dobija $[y] \subset [x]$, pa i $[x] = [y]$. ■

Primer 2.1.21 Prema primeru 2.1.18 relacija $\equiv_3$ je relacija ekvivalencije. Odgovarajuće klase ekvivalencije su:

- $[0] = \{\ldots, -6, -3, 0, 3, 6, \ldots\}$,
- $[1] = \{\ldots, -5, -2, 1, 4, 7, \ldots\}$ i
- $[2] = \{\ldots, -4, -1, 2, 5, 8, \ldots\}$. ■

Relacije poretka

Relacije (parcijalnog i totalnog) poretka uopštavaju različite primere uređenja skupova: biti podskup na nekoj familiji skupova, biti veći ili jednak na skupu realnih brojeva, biti deljiv na skupu prirodnih brojeva $\mathbb{N}$, ili alfabetsko uređenje reči nekog jezika.

Primer 2.1.22 Neka je $\mathbb{Q}$ skup racionalnih brojeva. Relacija R definisana na $\mathbb{Q}^2 \times \mathbb{Q}^2$ tako da je

$$(x,y)R(a,b) \text{ ako i samo ako je } x < a, \text{ ili } x = a \text{ i } y \leq b.$$

je jedna relacija parcijalnog poretka. Uočimo najpre da iz $(x,y)R(a,b)$ sledi $x \leq a$. Refleksivnost relacije R očigledno važi, jer je $x = x$ i $y \leq y$, pa je $(x,y)R(x,y)$. Ako je $(x,y)R(a,b)$ i $(a,b)R(x,y)$, onda je najpre $x \leq a$ i $a \leq x$, odnosno $x = a$, zatim isto važi i za y i b, pa je $\langle x,y\rangle = \langle a,b\rangle$, tako da je R antisimetrična relacija. Konačno, neka je $(x,y)R(a,b)$ i $(a,b)R(u,v)$. Tada je $x \leq a$ i $a \leq u$, pa i $x \leq u$. Ako je $x < u$, važi da je $(x,y)R(u,v)$. Ako je $x = u$, tada je i $x = a$ i $a = u$, pa mora biti $y \leq b$ i $b \leq v$, odakle je $y \leq v$ i ponovo $(x,y)R(u,v)$, odnosno R je tranzitivna relacija. ■

Definicija 2.1.23 *Parcijalno uređen skup*, ili *poset*, je uređeni par $\langle A, R\rangle$, gde je A skup, a $R \subset A^2$ relacija parcijalnog poretka.

Element $a \in A$ je *minimalan* ako za svaki element $x \in A$, iz xRa sledi $x = a$. Element $b \in A$ je *maksimalan* ako za svaki element $x \in A$, iz bRx sledi $b = x$.

Element $a \in A$ je *minimum*, ili *najmanji element skupa A*, ako za svaki element $x \in A$ važi aRx. Element $b \in A$ je *maksimum*, ili *najveći element skupa A*, ako za svaki element $x \in A$ važi xRb. ■

Ako minimum (maksimum) u nekom posetu $\langle A, R\rangle$ postoji, lako se vidi da je i jedinstven. Na primer, neka su a i b dva najmanja elementa. Tada je aRb i bRa, pa pošto je R antisimetričma, sledi da je $a = b$. Sledeći primer ilustruje da minimum (maksimum) ne mora postojati.

Primer 2.1.24 Neka je $\mathbb{Q}$ skup racionalnih brojeva. Skup $(0,1) = \{x \in \mathbb{Q} : 0 < x < 1\}$, nema ni minimum ni maksimum. Skup $(0,1] = \{x \in \mathbb{Q} : 0 < x \leq 1\}$ nema minimum, ali je 1 maksimum, dok skup $[0,1) = \{x \in \mathbb{Q} : 0 \leq x < 1\}$ nema maksimum, a 0 je minimum. U skupu $[0,1] = \{x \in \mathbb{Q} : 0 \leq x \leq 1\}$ su 0 i 1 redom minimum i maksimum. ■

Za razliku od minimuma i maksimuma koji su, ako postoje, jedinstveni, minimalnih i maksimalnih elemenata u nekom posetu može biti više pošto međusobno ne moraju biti uporedivi.

Primer 2.1.25 Posmatrajmo relaciju biti podskup na familiji $A = \{\emptyset, \{a\}, \{b\}\}$ pravih podskupova skupa $\{a, b\}$ za koju važi $\{a\} \not\subset \{b\}$ i $\{b\} \not\subset \{a\}$. Uređeni par $\langle A, \subset\rangle$ je jedan poset. Očigledno je da je $\emptyset$ njegov minimum, pa sem praznog skupa nema drugih minimalnih elemenata. Međutim, ovde ne postoji maksimum, a maksimalni elementi su $\{a\}$ i $\{b\}$. ■

Jednu od osnovnih motivacija za relacije poretka predstavlja relaciji "biti manji do jednak" na skupu celih brojeva $\mathbb{Z}$. Poznato je da tu važi da su svaka dva elementa u relaciji, pa je ovo primer relacije totalnog uređenja.

Definicija 2.1.26 *Totalno (linearno) uređenje*, ili *lanac*, je uređeni par $\langle A, R\rangle$, gde je A skup, a $R \subset A^2$ relacija totalnog poretka. ■

Primer 2.1.27 Skup $\{1, 2, 4, 8\}$ sa relacijom deljivosti $|$ čini jedan konačan lanac. Međutim, $\langle\{1, 2, 4, 8, 12\}, |\rangle$ nije lanac jer niti $8|12$ niti $12|8$. Jedan beskonačan lanac u odnosu na relaciju $|$ je skup $\{2^k : k \in \mathbb{N}\}$.

Posmatrajmo skup A i njegov partitivni skup $\mathbb{P}(A)$. U opštem slučaju $\langle\mathbb{P}(A), \subset\rangle$ nije lanac, na primer različiti jednočlani podskupovi skupa A nisu uporedivi. Međutim, ako je $A = \emptyset$ ili je A jednočlan, $\langle\mathbb{P}(A), \subset\rangle$ jeste lanac. ■

Definicija 2.1.28 Parcijalno uređenje $\langle A, R\rangle$ je *dobro uređenje* ako svaki neprazan podskup skupa A ima minimum. ■

Lako se vidi da je svako dobro uređenje i totalno: pošto svaki skup $\{a, b\} \subset A$ ima minimum, mora biti ili aRb ili bRa.

Primer 2.1.29 Skup $\mathbb{N}$ prirodnih brojeva u odnosu na relaciju $\leq$ čini jedno dobro uređenje. Skup $\{-x : x \in \mathbb{N}\}$ u odnosu na relaciju $\leq$ nije dobro uređenje jer nema minimum. Skup $\mathbb{Q}$ racionalnih brojeva u odnosu na relaciju $\leq$ nije dobro uređenje, jer na primer skup $(0, 1)$ nema minimum. ■

2.1.3 Funkcije i operacije

Definicija 2.1.30 Relacija $f \subset A \times B$ je *funkcija (preslikavanje)* ako zadovoljava da za svaki $x \in A$ postoji najviše jedan $y \in B$ tako da je $(x, y) \in f$. Pri tome je *y vrednost funkcije f za element x*. Skup A je *domen*, u oznaci $\mathrm{Dom}(f)$, funkcije f. Skup B je *kodomen*, u oznaci $\mathrm{Kodom}(f)$, funkcije f. *Slika funkcije* f, u oznaci $\mathrm{Im}(f)$, je skup $\{y \in B : (\exists x \in A)(x, y) \in f\}$.

Skup svih funkcija iz skupa A u skup B se označava sa B^A.

Ako za svaki $x \in A$ postoji $y \in B$ tako da je $(x, y) \in f$, funkcija f je *totalna*. Ako postoji $x \in A$ takav da ni za jedno $y \in B$ nije $(x, y) \in f$, funkcija f je *parcijalna*.

Identička funkcija na skupu A, $id_A \subset A \times A$, je definisana sa $id_A(x) = x$, za svaki $x \in A$. ■

Uobičajeno je da se umesto $f \subset A \times B$ za funkcije piše $f : A \mapsto B$, kao i $f(x) = y$ umesto $(x, y) \in f$.

Primer 2.1.31 Neka su $\mathbb{Q}$, $\mathbb{Z}$ i $\mathbb{N}$ skupovi racionalnih, celih i prirodnih brojeva. Jedna (totalna) funkcija $f : \mathbb{Z} \mapsto \mathbb{N}$ je definisana sa:

$$f(x) = x^2.$$

Za nju važi da je skup $\mathrm{Im}(f) = \{0, 1, 4, 9, \ldots\}$ pravi podskup skupa $\mathrm{Kodom}(f) = \mathbb{N}$.

Neka je B neki skup i $A \subset B$. Funkcija $\chi_A : B \mapsto \{0, 1\}$ definisana sa $\chi_A(x) = 1$ ako je $x \in A$, $\chi_A(x) = 0$ ako je $x \notin A$ je *karakteristična funkcija skupa A.*

Funkcija $g : \mathbb{Z} \mapsto \mathbb{Q}$ definisana sa:

$$g(x) = \frac{1}{x+1}$$

je parcijalna, jer nije definisana za $x = -1$.

Relacija $h = \{\langle x, y\rangle \in \mathbb{Q}^2 : x = y^2\}$ nije funkcija, jer, na primer, za $x = 4$ postoje $y_1 = -2$ i $y_2 = 2$ za koje je $(4, -2)$, $(4, 2) \in h$. ■

Definicija 2.1.32 Neka su $f : A \mapsto B$ i $g : B \mapsto C$ funkcije. *Kompozicija funkcija* f i g, u oznaci $g \circ f$, je skup $(g \circ f) = \{\langle x, z\rangle :$ postoji $y \in B$ za koje je $f(x) = y$ i $g(y) = z\}$. ■

Imajući u vidu da su u definiciji 2.1.32 f i g funkcije, lako se vidi i da je kompozicija funkcija $g \circ f$ takođe funkcija oblika $g \circ f : A \mapsto C$ za koju je $(g \circ f)(x) = g(f(x))$. Naime, za svaki $x \in A$ postoji najviše jedan $y \in B$ za koji je $f(x) = y$, a takođe i za taj y postoji najviše jedan $z \in C$ takav da je $g(y) = z$, odakle je ispunjen uslov iz definicije 2.1.30 da za svaki $x \in A$ postoji najviže jedan $z \in C$ tako da je $\langle x, z\rangle \in g \circ f$. Primetimo i da funkcija $g \circ f$ za neki $x \in \mathrm{Dom}(g \circ f)$ ne mora biti definisana iz dva razloga: ako funkcija f nije defisana za x, ili ako jeste, a funkcija g nije definisana za $f(x)$. Pošto su funkcije skupovi uređenih parova, za jednakost dvaju funkcija je potrebno da važi jednakost tih skupova.

Primer 2.1.33 Neka su funkcije $f : \mathbb{Z} \mapsto \mathbb{Z}$ i $g : \mathbb{Z} \mapsto \mathbb{N}$ definisane sa:

- $f(x) = 4x - 1$, odnosno
- $g(x) = 2x^2$.

Tada je $(g \circ f)(2) = g(f(2)) = g(4 \cdot 2 - 1) = g(7) = 2 \cdot 7^2 = 98$, a $(g \circ g)(1) = g(2 \cdot 1^2) = g(2) = 2 \cdot 2^2 = 8$.

Neka je funkcija $f : \mathbb{Q} \mapsto \mathbb{Q}$ definisana tako da je $f(x)$ najveći celi broj koji je manji ili jednak od x, za šta se koristi oznaka $f(x) = \lfloor x \rfloor$. Za ovu funkciju važi $f \circ f = f$ jer je za svaki $y \in \mathbb{Z}$, $\lfloor y \rfloor = y$.

Neka su funkcije $f : \mathbb{Z} \mapsto \mathbb{Q}$ i $g : \mathbb{Q} \mapsto \mathbb{Q}$ definisane sa:

- $f(x) = \frac{1}{x+1}$, odnosno
- $g(x) = x + 1$.

Tada f, pa i $g \circ f$ nisu definisane za $x = -1$.

Neka su funkcije $f : \mathbb{Z} \mapsto \mathbb{N}$ i $g : \mathbb{N} \mapsto \mathbb{Q}$ definisane sa:

- $f(x) = x^2$, odnosno
- $g(x) = \frac{1}{x-1}$.

Tada $g \circ f$ nije definisano za $x \in \{-1, 1\}$, jer g nije definisano za 1. ■

Teorema 2.1.34 Neka su $f : A \mapsto B$, $g : B \mapsto C$ i $h : C \mapsto D$ funkcije. Tada je

$$h \circ (g \circ f) = (h \circ g) \circ f.$$

Dokaz. Uočimo da je za proizvoljno $x \in A$

$$(h \circ (g \circ f))(x) = h((g \circ f)(x)) = h(g(f(x))) = (h \circ g)(f(x)) = ((h \circ g) \circ f)(x),$$

odakle sledi tvrđenje. ■

Na dalje ćemo razmatrati totalne funkcije.

Definicija 2.1.35 Funkcija $f : A \mapsto B$ je:

- *injektivna*, ili $1 - 1$, ako iz $f(x) = f(y)$ sledi da je $x = y$,
- *surjektivna*, ili *na*, ako sa svaki $y \in B$ postoji $x \in A$ tako da je $f(x) = y$,
- *bijektivna* ako je *injektivna* i *surjektivna*. ■

Primetivmo da je za surjektivnu funkciju f, $\text{Kodom}(f) = \text{Im}(f)$.

Primer 2.1.36 Funkcija $f : \mathbb{Z} \mapsto \mathbb{N}$ definisana sa $f(x) = x^2$ nije ni injektivna, jer $f(-1) = f(1))$, ni surjektivna, jer za $2 \in \mathbb{N}$ ne postoji $x \in \mathbb{Z}$ tako da je $x^2 = 2$.

Funkcija $f : \{1, 2\} \mapsto \{1, 2, 3\}$ takva da je $f(1) = 2$ i $f(2) = 3$ jeste injektivna jer je vrednost funkcije za sve elemente njenog domena različita, ali nije surjektivna jer za $1 \in \{1, 2, 3\}$ ne postoji $x \in \{1, 2\}$, tako da je $f(x) = 1$. Funkcija $g : \{1, 2\} \mapsto \{1\}$ takva da je $g(1) = g(2) = 1$ jeste surjektivna, ali nije injektivna, dok je funkcija $h : \{1, 2\} \mapsto \{1, 2\}$ takva da je $h(1) = 2$, $h(2) = 1$ bijektivna.

Neka su A i B neprazni skupovi i $A \times B$ njihov Dekartov proizvod. Funkcije projekcije definisane sa:

- $\pi_1 : A \times B \mapsto A$, tako da $\pi_1(a, b) = a$ i
- $\pi_2 : A \times B \mapsto B$, tako da $\pi_2(a, b) = b$

su obe surjektivne, ali nisu injektivne (ako skupovi A i B imaju više od jednog člana). ■

Ako za neku funkciju važi da je $f(a) = b$, interesantno je definisati obrnuti postupak koji b preslikava u a. Taj postupak nije funkcija ako postoji bar jedno $a_1 \in \mathrm{Dom}(f)$, tako da $a_1 \neq a$ i $f(a_1) = b$. Ako za neko b ne postoji ni jedan element domena koji se slika u njega, postupak, ako i jeste funkcija, je paracijalan. Oba ova ograničenja ne postoje za bijektivne funkcije.

Definicija 2.1.37 Za bijektivnu funkcija $f : A \mapsto B$ njoj *inverzna funkcija* $f^{-1} : B \mapsto A$ je definisana sa $f^{-1}(b) = a$ ako i samo ako je $f(a) = b$. ■

Za bijektivnu funkciju f važi da je $f \circ f^{-1} = id_B$ i $f^{-1} \circ f = id_A$ i $f^{-1^{-1}} = f$, jer, ako je $f(a) = b$, onda $(f^{-1} \circ f)(a) = f^{-1}(f(a)) = f^{-1}(b) = a$.

Primer 2.1.38 Za svaki skup A, trivijalno važi da je identička funkcija id_A inverzna samoj sebi, jer $(id_A \circ id_A)(x) = id_A(id_A(x)) = id_A(x) = x$.

Neka su dati $A = \{a, b, c\}$, $B = \{1, 2, 3\}$ i $f : A \mapsto B$ tako da je $f(a) = 1$, $f(b) = 2$ i $f(c) = 3$. Funkcija f je bijektivna, a njoj inverzna je funkcija $f^{-1} : B \mapsto A$, za koju je $f^{-1}(1) = a$, $f^{-1}(2) = b$ i $f^{-1}(3) = c$. ■

Definicija 2.1.39 Funkcija $f : A^n \mapsto A$ se naziva *n-arna operacija skupa* A. ■

U slučaju binarnih operacija, uglavnom se koristi infiksna notacija, tj. umesto $*(x, y)$ piše se $x * y$.

Definicija 2.1.40 Za binarne operacija $*$ i $\star$ skupa A kažemo:

- $*$ je *komutativna* ako je za sve $B, C \in A$, $B * C = C * B$,
- $*$ je *asocijativna* ako je za sve $B, C, D \in A$, $(B * C) * D = B * (C * D)$,
- $*$ je *distributivna u odnosu na* $\star$ ako je za sve $B, C, D \in A$, $B*(C \star D) = (B * C) \star (B * D)$. ■

U nastavku će nam posebno biti značajne neke unarne i binarne operacije na skupovima.

2.1.4 Operacije nad skupovima

Neka je A neki neprazan skup i $\mathbb{P}(A)$ njegov partitivni skup. Ramatraćemo sledeće operacije definisane na $\mathbb{P}(A)$:

- operacija *komplementa*, $\mathbb{C} : \mathbb{P}(A) \mapsto \mathbb{P}(A)$, za koju je $\mathbb{C}(B) = \{x : x \in A \text{ i } x \notin B\}$,
- operacija *unije*, $\cup : \mathbb{P}(A) \times \mathbb{P}(A) \mapsto \mathbb{P}(A)$, za koju je $B \cup C = \{x : x \in B, \text{ ili } x \in C\}$,
- operacija *preseka*, $\cap : \mathbb{P}(A) \times \mathbb{P}(A) \mapsto \mathbb{P}(A)$, za koju je $B \cap C = \{x : x \in B \text{ i } x \in C\}$ i
- operacija *razlike*, $\setminus : \mathbb{P}(A) \times \mathbb{P}(A) \mapsto \mathbb{P}(A)$, za koju je $B \setminus C = \{x : x \in B \text{ i } x \notin C\}$,
- operacija *simetrične razlike*, $\triangle : \mathbb{P}(A) \times \mathbb{P}(A) \mapsto \mathbb{P}(A)$, za koju je $B \triangle C = \{x : x \in B \text{ i } x \notin C, \text{ ili } x \notin B \text{ i } x \in C\}$.

sve od navedenih operacija su totalne, odnosno definisane za sve elemente svojih domena. Pri tome je komplement unarna, a druge navedene operacije binarne. Za neku konačnu familiju B_1, B_2, $\ldots$, B_k podskupova skupa A često se koriste uopštene unije i preseci:

- $\cup_{i=1}^k B_i = \{x : x \in B_1, \text{ ili } x \in B_2, \ldots, \text{ ili } x \in B_k\}$, odnosno
- $\cap_{i=1}^k B_i = \{x : x \in B_1 \text{ i } x \in B_2 \text{ i } \ldots \text{ i } x \in B_k\}$.

Ako je familija B_1, B_2, $\ldots$ beskonačna, koriste se i oznake $A = \cup_i B_i$, odnosno $A = \cap_i B_i$.

Neke od osnovnih osobina operacija nad skupovima formulisane su teoremom 2.1.41.

Teorema 2.1.41 Neka je A neki neprazan skup, $\mathbb{P}(A)$ njegov partitivni skup i $B, C, D \in \mathbb{P}(A)$. Tada važe sleeće jednakosti:

- $\mathbb{C}(\mathbb{C}(B)) = B$,
- *zakoni idempotencije*
  - $B \cup B = B$ i
  - $B \cap B = B$,
- *zakoni komutativnosti*
  - $B \cup C = C \cup B$ i
  - $B \cap C = C \cap B$,
- *zakoni asocijativnosti*
  - $B \cup (C \cup D) = (B \cup C) \cup D$ i
  - $B \cap (C \cap D) = (B \cap C) \cap D$,
- *zakoni apsorpcije*
  - $B \cup (B \cap C) = B$ i
  - $B \cap (B \cup C) = B$,
- *zakoni distributivnosti*
  - $B \cap (C \cup D) = (B \cap C) \cup (B \cap D)$
  - $B \cup (C \cap D) = (B \cup C) \cap (B \cup D)$,
- *De Morganovi zakoni*
  - $\mathbb{C}(B \cup C) = \mathbb{C}B \cap \mathbb{C}C$ i
  - $\mathbb{C}(B \cap C) = \mathbb{C}B \cup \mathbb{C}C$,
- $B \cup \emptyset = B$,
- $B \cap \emptyset = \emptyset$,
- $B \cap A = B$,
- $B \cup A = A$,
- $B \cup \mathbb{C}(B) = A$,
- $B \cap \mathbb{C}(B) = \emptyset$,
- $\mathbb{C}(A) = \emptyset$,
- $\mathbb{C}(\emptyset) = A$,

- $B \setminus C = B \cap \complement C$,
- $B \triangle C = (B \setminus C) \cup (C \setminus B)$.

Dokaz. Kao primer dokazaćemo De Morganov zakon za uniju, dok se ostali dokazi slično sprovode. Dakle, za proizvoljan x važi

- $x \in \complement(B \cup C)$ ako i samo ako
- $x \notin B \cup C$ ako i samo ako
- $x \notin B$ i $x \notin C$ ako i samo ako
- $x \in \complement B$ i $x \in \complement C$ ako i samo ako
- $x \in \complement B \cap \complement C$,

odakle sledi tvrđenje. ■

Pretpostavimo da smo u nekom izrazu koji se sastoji od simbola operacija i skupova zamenimo $\cup$ sa $\cap$ i obrnuto, a $\emptyset$ sa A i obrnuto. Takvom zemenom dobija se *dual* polaznog izraza. Lako se vidi da da je dual duala polazni izraz, kao i da duali jednakosti iz teoreme 2.1.41 takođe važe za proizvoljne $B, C, D \in \mathbb{P}(A)$. Uopšte, važu *princip dualnosti*: ako neki izraz važi iza sve proizvoljne skupove, elemente $\mathbb{P}(A)$, onda je to slučaj i sa njegovim dualom.

Definicija 2.1.42 Skupovi A i B su *disjunktni* ako je $A \cap B = \emptyset$. ■

Definicija 2.1.43 *Particija* skupa A je familija B_1, B_2, $\ldots$ podskupova skupa A za koju važi:

- $A = \cup_i B_i$ i
- za $i \neq j$, disjunktni su B_i i B_j. ■

Particija, zavisno od familije B_i, može biti konačna i beskonačna.

2.1.5 Kardinalnost skupova

Često je u analizi i rešavanju problema bitno koliko neki skup ima elemenata. Ovde je korisno najpre razmotriti jedan način definisanja prirodnih brojeva.

Primer 2.1.44 Prirodni brojevi se u teoriji skupova definišu na sledeći način:

- $0 = \emptyset$,

- $1 = \emptyset \cup \{\emptyset\} = \{0\}$,
- $2 = 1 \cup \{1\} = \{0\} \cup \{1\} = \{0, 1\}$,
- $3 = 2 \cup \{2\} = \{0, 1\} \cup \{2\} = \{0, 1, 2\}, \ldots,$
- $n + 1 = n \cup \{n\} = \{0, 1, 2, \ldots, n-1, n\}, \ldots,$

tako da je svaki prirodan broj skup svojih prethodnika. Tada je skup $\mathbb{N}$ prirodnih brojeva najmanji skup sa osobinama da sadrži 0 i, ako mu pripada n, onda mu pripada i $n+1 = n \cup \{n\}$ i u teoriji skupova se dokazuje njegovo postojanje. Relacija $\leq \subset \mathbb{N}^2$ se definiše tako da $x \leq y$ ako i samo ako je $x = y$ ili $x \in y$ (neformalno, x je jedan od prethodnika od y). Lako se pokazuje da je $\leq$ relacija totalnog poretka (svaka dva prirodna broja su uporedivi relacijom) i da je $\langle \mathbb{N}, \leq \rangle$ dobro uređenje (svaki neprazan podskup od $\mathbb{N}$ ima minimum). ■

Definicija 2.1.45 Skupovi A i B imaju *istu kardinalnost*, u oznaci $|A| = |B|$, ako postoji bijektivna funkcija $f : A \mapsto B$.

Skup A *konačan* ako postoje prirodan broj n i bijektivna funkcija $f : A \mapsto n$. Tada je *broj elemenata (kardinalost) skupa* A, u oznaci $|A|$ jednak n. Ako skup nije konačan, onda je *beskonačan*.

Skup A je *prebrojiv*, odnosno *prebrojivo beskonačan*, ako postoji bijektivna funkcija $f : A \mapsto \mathbb{N}$. Tada $|A| = \aleph_0$.

Skup A je *neprebrojiv*, ako postoji injektivna funkcija $f : \mathbb{N} \mapsto A$, ali A i $\mathbb{N}$ nemaju istu kardinalnost. ■

Primer 2.1.46 Prazan skup je konačan jer je 0 jedini prirodan broj za koji postoji bijekcija iz definicije 2.1.45, pa je i $|\emptyset| = 0$. Skup $\{1, 3, 5, 7, 9, 11, 13\}$ je konačan, kardinalnosti 7. Skupovi $\mathbb{N}$, $\mathbb{Z}$ i $\mathbb{Q}$, skup svih parnih i skup svih neparnih prirodnih brojeva su svi prebrojivo beskonačni. ■

U nastavku će pre svega biti reči o konačnim i prebrojivim skupovima. Teorema 2.1.47 sadrži nekoliko tvrđenja o kardinalnosti konačnih skupova.

Teorema 2.1.47 Neka su A, B, C, A_1 i A_2 konačni skupovi. Tada,

1. Ako $|A| = k \in \mathbb{N}$, onda je $|\mathbb{P}(A)| = 2^k$.
2. Ako je $|A_1| = k_1$, a $|A_2| = k_2$, $k_1, k_2 \in \mathbb{N}$, onda je $|A_1 \times A_2| = k_1 \cdot k_2$.
3. Ako postoji injektivna funkcija $f : A \mapsto B$, onda je $|A| \leq |B|$. Podskup konačnog skupa je konačan.
4. Ako postoji surjektivna funkcija $f : A \mapsto B$, onda je $|A| \geq |B|$.

5. Ako je $A \cap B = \emptyset$, onda je $|A \cup B| = |A| + |B|$.

6. $|A \cup B| = |A| + |B| - |A \cap B|$.

7. $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$.

8. Ako je B^A skup svih funkcija iz A u B, onda je $|B^A| = |B|^{|A|}$.

Dokaz. Ilustrovaćemo dokaze nekoliko tvrđenja.
(1) Neka je $A = \{a_1, \ldots, a_k\}$. Proizvoljan element B partitivnog skupa $\mathbb{P}(A)$ se može prikazati kao uređena k-torka nula i jedinica, pri čemu je na i-tom mestu 1 ako je $a_i \in B$, a 0 $a_i \notin B$. Na primer, $\emptyset$ je na taj način predstavljen sa $\langle 0, 0, \ldots, 0 \rangle$, a celi skup A sa $\langle 1, 1, \ldots, 1 \rangle$. Odatle, broj elemenata skupa $\mathbb{P}(A)$ jednak je broju različitih k-torki nula i jedinica. Lako se vidi da je taj poslednji broj jednak 2^k. Naime, na svakoj od k-pozicija moguće je izbor jedne od dve binarne cifre, pa se indukcijom tvrđenje lako pokazuje.
(5) Neka je $|A| = k$ i $|B| = l$. To znači da postoje bijektivne funkcije $f : A \mapsto k$ i $g : B \mapsto l$. Označimo elemente skupova A i B tako da je $f(a_i) = i$, $i = 0, \ldots, k-1$ i $g(b_j) = j$, $j = 0, \ldots, l-1$. Pošto je $A \cap B = \emptyset$, onda je $A \cup B = \{a_0, \ldots, a_{k-1}, b_0, \ldots, b_{l-1}\}$. Posmatrajmo funkciju $h : A \cup B \mapsto k + l$, definisanu sa:

- $h(a_i) = f(a_i) = i$, $i = 0, \ldots, k-1$,
- $h(b_j) = g(b_j) + k = j + k$, $j = 0, \ldots, l-1$,

za koju se lako proverava da je bijekcija. Prema tome, $|A \cup B| = k + l = |A| + |B|$.
(6) Uočimo da je $A \cup B = (A \setminus B) \cup (B \setminus A) \cup (A \cap B)$, gde su $A \setminus B$, $B \setminus A$ i $A \cap B$ međusobni disjunktni skupovi. Zbog toga je

$$\begin{aligned} |A \cup B| &= |A \setminus B| + |B \setminus A| + |A \cap B| \\ &= (|A| - |A \cap B|) + (|B| - |A \cap B|) + |A \cap B| \\ &= |A| + |B| - |A \cap B|. \end{aligned}$$

■

Opšti slučaj tvrđenja 2.1.47.2 je

$$|A_1 \times \cdots \times A_n| = |A_1| \cdot |A_2| \cdots |A_n|.$$

Slično, uopštavanjem tvrđenja 2.1.47.6 i 2.1.47.7 može se formulisati i sledeće pravilo:

$$\begin{aligned} |A_1 \cup \ldots \cup A_n| &= |A_1| + \cdots + |A_n| \\ &- \text{ brojevi elemenata dvočlanih preseka} \\ &+ \text{ brojevi elemenata tročlanih preseka } \ldots \end{aligned}$$

Pored teorijskog značaja, prethodno tvrđenje će imati i konkretne primene u nekim od postupaka koje ćemo predstaviti u nastavku.

2.2 Skupovi u programskim jezicima

Slično dokazu tvrđenja iz teoreme 2.1.47, u većini programskih jezika (konačni) skupovi se prikazuju u obliku niza bitova čija dužina je jednaka broju elemenata skupa. Skupovne operacije se simuliraju kao logičke operacije nad bitovima, na primer preseku odgovara konjukcija po bitovima. Zbog toga je ponekad u konkretnim implementacijama ograničena dužina skupa na broj bitova u registru konkretnog računara. Ovakav način predstavljanja kao posledicu ima i uređenje elemenata (računarskog) skupa. Na primer jednim bitom možemo predstaviti logičke vrednosti, i to nulom netačno ($\perp$), a jedinicom tačno ($\top$) pri čemu onda važi i $\perp < \top$.

Binarne relacije nad konačnim skupovima se pogodno predstavljaju matricama nula i jedinica. Na primer, neka je $R \subset A \times B$, $A = \{a_1, \ldots, a_k\}$ i $B = \{b_1, \ldots, b_m\}$. Relaciju R tada možemo prikazati binarnom matricom $M_{k\times m}$ tako da je $M_{i,j} = 1$ ako je $R(a_i, b_j)$, odnosno $M_{i,j} = 0$ ako je $\neg R(a_i, b_j)$. Ako je $R \subset A \times A$, odgovarajuća matrica je kvadratna.

Refleksivna, simetrična relacija, tranzitivno zatvorenje ...

2.2.1 Tipovi

U savremenim programskim jezicima pojedine programske celine, odnosno konkretni objekti, na primer funkcije, promenljive i datoteke, obično moraju biti deklarisane da su nekog tipa. Takođe, često se insistira na deklarisanju tipova argumenata funkcija i procedura i prilikom prevođenja programa se vrši provera usklađenosti tih tipova sa tipovima argumenata u pozivu programske celine. Primeri tipova su: celi brojevi, realni brojevi, logički tip, stringovi itd. Ovako posmatrani tipovi imaju sličnosti i razlike u odnosu na (matematičke, apstraktne) skupove.

Primetimo najpre da upotreba naziva tipa, poput celi (ili realni) brojevi, ovde treba da asocira na odgovarajuće (matematičke, apstraktne) skupove, ali da može i da zavede. Naime, celi brojevi predstavljeni u nekom konkretnom računarskom sistemu su (samo) konačni podskup skupa $\mathbb{Z}$. Na primer, standard programskog jezika C, definiše standarne konstante (čija vrednost zavisi od konkretne implementacije):

- INT_MAX, kao največi podržani celi broj i
- INT_MIN, kao najmanji podržani celi broj

sa garantovanim vrednostima od (makar) $+32767$, odnosno -32767 itd. Slično, realni brojevi u računaru predstavljaju konačan podskup skupa $\mathbb{Q}$, i to su zapravo neki racionalni brojevi ograničene preciznosti. Druga restrikcija koja se odnosi na tipove je da, u opštem slučaju, neki (računarski) skup sadrži elemente samo istog tipa. Tako nisu dopušteni skupovi koji, na primer, sadrže realni broj i string.

U opštem slučaju pojam tipa u programskom jeziku obuhvata i neke unapred zadate operacije nad objektima tog tipa. Tako su na konkretnim tipovima definisane pojedine operacije, dok neke druge operacije nisu smislene. Na primer, cele brojeve je dozvoljeno sabirati, množiti, oduzimati, ali (sem kod nekih posebnih programskih jezika) besmislena je primena logičkih operatora na njih. Takođe, zbog ograničenja konkretnih implementacija, pojedine operacije nisu istovetne matematičkim (apstraktnim) pandanima. Recimo, vrednost (računarskog) sabiranja celih brojeva, zbog prekoračenja, može biti različita od (matematičke, apstraktne) vrednosti. Ovo se često zaboravlja i dovodi do neželjenih rezultata pri izvršavanju programa.

S obzirom da su (računarske) operacije definisane samo nad određenim tipovima, u opštem slučaju se prilikom prevođenja programa vrši provera ispravnosti tipova argumenata na koje se operacije primenjuju i, u slučaju neslaganja javlja poruka o grešci. Na primer, besmislena je operacija korenovanja stringa. Kod nekih operacija, recimo sabiranja, ako je jedan argument celi, a drugi realni broj vrši se automatsko prevođenje celog u realni broj.

2.3 Relacione baze podataka

Ovaj odeljak je posvećen prikazu teorijske osnove nekih osnovnih koncepata i operacija u relacionim bazama podataka.

Neki podatak (*slog*) karakterišu njegove komponente ili *atributi*. Na primer, u telefonskom imeniku jedan podatak može imati atribute: ime, adresa, telefonski broj. Svaki od atributa ima svoj poseban tip, koji svi zajedno određuju složeni, ili *slogovni* tip podatka. *Tabela* je skup podataka istog slogovnog tipa. Svaka od kolona u tabeli odgovara jednom atributu, dok su redovi konkretni podaci odgovarajućeg slogovnog tipa. Nije teško videti da se jedna tabela može formalno predstaviti na sledeći način:

- neka su atributi u tabeli redom A_1, A_2, …, A_k, a njihovi tipovi T_1, T_2, …, T_k,

- neka je X_i skup mogućih vrednosti atributa A_i, za $i = 1, 2, \ldots, k$,

- relaciji $R \subset X_1 \times X_2 \times \cdots \times X_k$ tada odgovara cela tabela, dok

- neki slog, odnosno red tabele,s predstavljamo konkretnom k-torkom $\langle x_1, x_2, \dots, x_k \rangle \in R$.

O relaciji R se može razmišljati i kao da je promenljiva u vremenu, do čega dolazi brisanjem postojećih ili dodavanjem novih slogova.

Prateći prethodno označavanje, *relaciona baza podataka* sa atributima A_1, A_2, $\dots$, A_k je kolekcija relacija (tabela) R_1, R_2, $\dots$, R_m, gde je svaka od relacija $R_i \subset X_{i_1} \times X_{i_2} \times \cdots \times X_{i_n}$, definisana nad Dekartovim proizvodom odgovarajućih skupova vrednosti nekih od atributa, a indeksi $i_1, \dots, i_n$ međusobno različiti članovi skupa $\{1, 2, \dots, k\}$.

Ključevi kandidati su takvi skupovi atributa čije vrednosti na jedinstven način određuju vrednosti ostalih atributa slogova u tabeli, dok to ne važi za prave podskupove ključevi kandidata. Jedan od ključevi kandidata, *primarni ključ*, se bira da bude aktuelni ključ. Na primer, matični broj je uveden da na jedinstven način odredi osobu koja ga poseduje, ali treba obratiti pažnju da, iako je matični broj jedan atribut, u opštem slučaju ključ sadrži više atributa. Atribut koji pripada nekom ključu kandidatu je *primitivan*, a *nije primitivan* atribut koji ne pripada ni jednom ključu kandidatu.

Primer 2.3.1 Posmatrajmo atribute:

- A_1, matični broj,
- A_2, ime i prezime,
- A_3, ulica i broj,
- A_4, grad,
- A_5, datum i
- A_6, težina,

njima odgovarajuće skupove vrednosti $X_1, \dots, X_6$ i relacionu bazu podataka koja sadrži dve tabele predstavljene relacijama:

- $R_1 \subset X_1 \times X_2 \times X_3 \times X_4$ i
- $R_1 \subset X_1 \times X_5 \times X_6$,

o ličnim podacima osoba i podacima o njihovim težinama merenim nekih datuma. Atribut A_1, matični broj, biće primarni ključ za R_1, dok je par A_1 i A_5, matični broj i datum, primarni ključ za R_2. ■

Osnovni postupci koje možemo sprovesti radeći sa relacionom bazom odnose se na:

- izdvajanje onih slogova koji zadovoljavaju neki uslov i
- kreiranje novih tabela na osnovu postojećih.

Postupak *selekcije* je zapravo izbor onih slogova iz neke tabele $R \subset X_1 \times X_2 \times \cdots \times X_k$ koji imaju neke specificirane vrednosti atributa, na primer formiranje podskupa oblika

$$\{\langle x_1, x_2, \ldots, x_k \rangle : x_{i_1} = r_{i_1}, \ldots, x_{i_m} = r_{i_m}\}.$$

Prirodnim spajanjem dve tabele $R \subset X_1 \times \cdots \times X_k \times Y_1 \times \cdots \times Y_m$ i $S \subset X_1 \times \cdots \times X_k \times Z_1 \times \cdots \times Z_n$ dobija se tabela:

$$\begin{aligned}\{\langle x_1, \ldots, x_k, y_1, \ldots, y_m, z_1, \ldots, z_n, \rangle \quad : \quad & \langle x_1, \ldots, x_k, y_1, \ldots, y_m \rangle \in R, \\ & \langle x_1, \ldots, x_k, z_1, \ldots, z_n \rangle \in S\}.\end{aligned}$$

Na dve tabele istog slogovnog tipa mogu se primenjivati uobičajane skupovne operacije: unija, presek, razlika.

Projekcijom se izdvajaju kolone neke tabele. Recimo, projekcijom tabele $R \subset X_1 \times X_2 \times \cdots \times X_k$ po atributima $A_2, \ldots, A_k$, dobija se tabela $R' = \{\langle x_2, \ldots, x_k \rangle :$ postoji $x_1 \in X_1, \langle x_1, \ldots, x_k \rangle \in R\}$. Drugim rečima, R' je kodomen funkcije:

$$f_I : X_1 \times X_2 \times \cdots \times X_k \mapsto X_2 \times \cdots \times X_k$$

kod koje je $\mathrm{Dom}(f_I) = R$, $f_I(x_1, x_2, \ldots, x_k) = \langle x_2, \ldots, x_k \rangle$, a skup indeksa atributa $I = \{2, \ldots, k\}$.

Primer 2.3.2 Posmatrajmo relacionu bazu iz primera 2.3.1. Neka su konkretne tabele R_1 sa ličnim podacima i R_2 sa podacima o merenjima kao u tabelama 2.1, odnosno 2.2.

Selekcijom iz tabele R_1 po kriterijumu da je vrednost atributa 'Grad' jednaka ' Novi Sad' dobijamo R_1^{NoviSad} u tabeli 2.3, dok prirodnim spajanjem tabela R_1 i R_2 dobijamo R_3 u tabeli 2.4. ■

2.3.1 Normalne forme

Relacione baze podataka su dinamičke, tj. tokom vremena podaci se menjaju, brišu i dodaju. *Normalne forme* se definišu kako bi se tokom rada izbegla pojava nekonzistentnosti do kojih može doći zbog postojanja zavisnosti između atributa u tabelama. Najpre, svaka tabela definisana kao skup podataka istog slogovnog tipa je u *prvoj normalnoj formi (1NF)*.

Matični broj	Ime i prezime	Ulica i broj	Grad
$JMBG_1$	Pera Perić	Glavna 7a	Beograd
$JMBG_2$	Žika Žikić	Nova b.b.	Beograd
$JMBG_3$	Mika Mikić	Dunavska 21	Novi Sad
$JMBG_4$	Stevo Stević	R. Domanovića 14	Kragujevac

Tabela 2.1. Tabela R_1 sa ličnim podacima.

Matični broj	Datum	Težina
$JMBG_1$	12. 01. 2009.	80
$JMBG_1$	12. 07. 2009.	81
$JMBG_2$	12. 01. 2009.	87
$JMBG_2$	12. 07. 2009.	86
$JMBG_3$	12. 01. 2009.	77
$JMBG_4$	12. 07. 2009.	80

Tabela 2.2. Tabela R_2 sa podacima o merenjima.

Matični broj	Ime i prezime	Ulica i broj	Grad
$JMBG_3$	Mika Mikić	Dunavska 21	Novi Sad

Tabela 2.3. Tabela R_1^{NoviSad} sa ličnim podacima stanovnika Novog Sada.

Matični broj	Ime i prezime	Ulica i broj	Grad	Datum	Težina
$JMBG_1$	Pera Perić	Glavna 7a	Beograd	12. 01. 2009.	80
$JMBG_1$	Pera Perić	Glavna 7a	Beograd	12. 07. 2009.	81
$JMBG_2$	Žika Žikić	Nova b.b.	Beograd	12. 01. 2009.	87
$JMBG_2$	Žika Žikić	Nova b.b.	Beograd	12. 07. 2009.	86
$JMBG_3$	Mika Mikić	Dunavska 21	Novi Sad	12. 01. 2009.	77
$JMBG_4$	Stevo Stević	R. Domanovića 14	Kragujevac	12. 07. 2009.	80

Tabela 2.4. Tabela R_3 dobijena prirodnim spajanjem tabela R_1 i R_2.

Primer 2.3.3 Posmatrajmo tabelu R_3 (datu u tabeli 2.4) iz primera 2.3.2. Lako je primetiti umnogostručavanje podataka u njoj: imena osoba i njihove adrese se neprestano ponavljaju. Veći problem predstavljaju izmene. Na primer, neka je Pera Perić promenio adresu. Ako bi se ta izmena svuda unosila, u principu je potrebno izvršiti dosta ispitivanja i upisivanja. Ako se izmena ne bi unosila, u različitim slogovima adresa Pere Perića bi bila različita. Projekcija po imenima i odgovarajućim adresama, koja bi trebalo da izlista sve osobe i njihove adrese, bi bila u najmanju ruku zbunjujuća. Sledeći problem se pojavljuje ako imamo novu osobu čija merenja treba pratiti. Nju nije moguće ubaciti u tabelu, dok god se ne izvrši prvo merenje. Slično, ako se iz bilo kog razloga obriše red tabele, a u tom redu je jedina pojava neke osobe (na primer poslednji red tabele R_3), automatski se gube i podaci o toj osobi. ■

Funkcionalna zavisnost se može pojaviti kada vrednosti jednog skupa atributa jednoznačno određuju vrednosti drugog, disjunktnog, skupa atributa. Preciznije, neka su:

- $I = \{i_1, \ldots, i_k\}$ i $J = \{j_1, \ldots, j_m\}$ dva skupa indeksa atributa tako da je $I \cap J = \emptyset$ (jednostavnosti radi, pretpostavimo da su u indeksi rastućem poretku $i_1 < \ldots < i_k < j_1 < \ldots < j_m$),

- skupovi atributa $A_I = \{A_{i_1}, \ldots, A_{i_k}\}$, $A_J = \{A_{j_1}, \ldots, A_{j_m}\}$ i $A_{I \cup J} = A_I \cup A_J$,

- R tabela u kojoj skup atributa sadrži atribute $A_{I \cup J}$ i

- $R_{I \cup J}$ projekcija tabele R po $A_{I \cup J}$.

Tada je skup atributa A_J *funkcionalno zavisan* od skupa atributa A_I, ako je $R_{I \cup J}$ funkcija iz R_I u R_J, odnosno za svaku k-torku $r_k \in R_I$ postoji tačno jedna m-torka $r_m \in R_J$, tako da je $\langle r_k, r_m \rangle \in R_{I \cup J}$.

U nastavku ćemo, kao ilustraciju, razmotriti takozvanu *drugu normalu formu (2NF)* u kojoj se tabela nalazi ako i samo ako svaki ne-primitivni atribut nije funkcionalno zavisan od nekog pravog podskupa nekog ključa kandidata. Pored ove, postoje i druge normalne forme čije razmatranje prevazilazi opseg ove knjige.

Lako se uočava da je tabela u 2NF ako su svi ključevi kandidati jednočlani. Međutim, ako postoje višečlani ključevi kandidati moguće je da neki njihov pravi podskup na jedinstven određuje vrednosti nekih od ne-primitivnih atributa.

Primer 2.3.4 Nastavimo analizu iz primera 2.3.3. Iako matični broj na jedinstven način određuje osobu, u tabeli R_3 on nije ključ kandidat, jer je na primer $JMBG_1$ vrednost tog atributa u dva sloga. Međutim, skup atributa {Matični broj, Datum} jeste ključ kandidat.

Narušavanje druge normalne forme se ovde javlja pošto atribut 'Matični broj' na jedinstven način određuje vrednost atributa 'Ime i prezime'.

Normalizacija, ili transformacija u drugu normalnu formu, se vrši delenjem tabele R_3 u dve tabele R_1 i R_2 iz primera 2.3.2. U prvoj tabeli primarni ključ je 'Matični broj', a u drugoj {Matični broj, Datum}. ■

3

Prebrojavanje

Samo ime sugeriše da se prebrojavanje odnosi na utvrđivanje broja elemenata nekog skupa, na primer na broj raspoloživih sedmocifrenih telefonskih brojeva. U matematici i računarstvu postupci prebrojavanje se koriste u rešavanju raznovsnih problema (na primer, u diskretnoj verovatnoći i izračunavanju složenosti algoritama) i kao osnova moćnih tehnika dokazivanja (kombinatorijalni dokazi). U nastavku ćemo spomenuti nekoliko tehnika prebrojavanja koje se uglavnom sastoje od:

- transformacija polaznog problema (uglavnom u neki problem nad uređenim k-torkama, tj. konačnim sekvencama, ili nizovima) i
- dekompozicija problema u jednostavnije.

Za očekivati je da u svemu tome kombinatorika, grana matematike koja proučava rasporede elemenata u skupovima, ima veliki značaj, pa je značajan deo ovog poglavlja posvećen nekim kombinatornim pojmovima. Nakon toga biće predstavljena još dva postupka koji imaju primene i u prebrojavanju - rekurentne relacije i generatorne funkcije. Svi ovi alati se mogu, okvirno govoreći, primenjivati na dva tipa zadataka, a to su:

- problemi u kojima se traži dokaz postojanja rešenja i
- problemi u kojima se traži broj rešenja.

3.1 Postupci transformacije problema

Prilikom transformacije problema često se koriste delovi tvrđenja 2.1.47:

1. skupovi između kojih postoji bijekcija su iste kardinalnosti, tj. ako postoji bijekcija $f : A \mapsto B$, onda je $|A| = |B|$,

2. domen injektivnog preslikavanja nije veće kardinalnosti od kodomena, tj. ako postoji injekcija $f : A \mapsto B$, onda je $|A| \leq |B|$, i

3. kodomen surjektivnog preslikavanja nije veće kardinalnosti od domena, tj. ako postoji surjekcija $f : A \mapsto B$, onda je $|A| \geq |B|$,

kojima se u vezu dozvode kardinalnosti skupova rešenja polaznih problema i problema u koje se oni transformišu. Na primer, u dokazu tvrđenja 2.1.47.1 iskorištena je jedna bijekcija koja elemente partitivnog skupa $\mathbb{P}(A)$ preslikava u k-torke nula i jedinica, pa se prebrojavanjem potonjih dolazi do broja podskupova posmatranog skupa.

Kontrapozicijom navedenog pravila (2), poznata i pod nazivom *princip golubarnika*[1] glasi:

Ako je $|A| > |B|$ onda ni jedna funkcija $f : A \mapsto B$ nije injektivna.

Drugim rečima, ako je $|A| > |B|$, za svaku funkciju $f : A \mapsto B$ moraju postojati bar dva elementa $a_1, a_2 \in A$, tako da je $a_1 \neq a_2$ i $f(a_1) = f(a_2)$.

Primer 3.1.1 Ako u nekom golubarniku postoji k pregrada, a imamo $k+1$ goluba, onda da bi svi golubovi bili smešteni, u bar jednoj od pregrada moraju biti smeštena bar dva goluba.

Posmatrajmo skup A od 90 dekadnih brojeva od po 25 cifara. Tada je $|\mathbb{P}(A)| = 2^{90}$. Najveća moguća suma brojeva u bilo kom podsupu skupa A sigurno ne premašuje $90 \cdot 10^{25}$, pošto su svi brojevi u skupu A manji od 10^{25}. Pokazuje se da je $2^{90} > 90 \cdot 10^{25}$, pa možemo zaključiti da u skupu A moraju postojati dva različita podskupa sa istim zbirovima elemenata. ■

Jedno uopštenje principa golubarnika glasi: Ako je $|A| > k \cdot |B|$ tada za svaku funkciju $f : A \mapsto B$ postoji bar $k+1$ elemenata skupa A koji se preslikavaju u isti element skupa B.

Primer 3.1.2 Neka u Beogradu postoji $n = 100000$ bradatih muškaraca i neka je broj dlaka u njihovoj bradi najviše $k = 40000$. Koristeći uopšteni princip golubarnika, pošto je $n > 2 \cdot k$, zaključujemo da bar trojica od njih moraju imati isti broj dlaka u bradi. ■

Još jedno variranje pravila (2) dovodi do uopštavanja pojma injektivne (odnosno $1-1$) funkcije na k–*u*–1-*funkciju* koja u svaki element kodomena slika tačno k elemenata domena:

Ako je $f : A \mapsto B$ k–u–1-funkcija, onda je $|A| = k \cdot |B|$.

[1] Pigeohhole principle. Veruje se da je Johann Dirichlet, 1805 – 1859, nemački matematičar, prvi dao formalizaciju ovog principa, pa se ovaj princip naziva i Dirichletovim.

Primer 3.1.3 Odredimo na koliko različitih načina se na šahovskoj tabli mogu postaviti dva topa tako da se ne nalaze ni u istom redu, ni u istoj koloni. Najpre poziciju topa i (za $i = 1, 2$) predstavimo kao uređeni par koordinata $\langle r_i, k_i \rangle$, gde su $r_i, k_i \in \{1, \ldots, 8\}$ i r_i označava red, a k_i kolonu u kojoj se top i nalazi. Na ovaj način skup pozicija dva topa se preslikava na skup $\langle r_1, k_1, r_2, k_2 \rangle$. Na primer, jedno rešenje problema je $\langle 1, 1, 2, 2 \rangle$. Treba uočiti i da je $\langle 2, 2, 1, 1 \rangle$, iako formalno različita uređena četvorka, zapravo zapis istog tog rešenja. Odatle, ako skup svih pozicija koje jesu rešenja problema označimo sa A, a sa B skup svih njima odgovarajućih uređenih četvorki, prethodno opisano preslikavanje iz B u A je $2-$u-1-funkcija, i zaključujemo da je $|A| = \frac{|B|}{2}$.

Pošto za rešenje problema mora da važi $r_1 \neq r_2$ i $k_1 \neq k_2$, lako se uočava da par r_1, k_1 možemo izabrati na jedan od $8 \cdot 8 = 64$ načina, dok za izbor para r_2, k_2 na raspolaganju imamo $7 \cdot 7 = 49$ načina (po jedan red i jednu kolonu zauzima prvi top, tako da su za drugi top na raspolaganju 7 redova i 7 kolona). Prema tome, broj rešenja problema je $|A| = \frac{8^2 \cdot 7^2}{2} = 1568$. ■

3.2 Postupci dekompozicije problema

Dekompozicija problema na jednostavnije, tako da se kombinovanjem njihovih dobija rešenja polaznog problema, je jedna opšta tehnika rešavanja zadataka. *Postupcima dekompozicije problema* ovde nazivamo:

- pravilo proizvoda
- pravilo zbira
- pravilo uključenja-isključenja

pomoću kojih se izračunava broj članova nekog skupa. Ova pravila su zapravo delovi tvrđenja 2.1.47.

Pravilo proizvoda se odnosi na veličinu proizvoda skupova:

$$|A_1 \times \cdots \times A_n| = |A_1| \cdot |A_2| \cdots |A_n|.$$

Primer 3.2.1 Neka su redovi u nekoj sali numerisani rimskim brojevima I, II, ..., XXV, a sedišta u svakom redu arapskim brojevima 1, 2, ..., 30. Ukupan broj sedišta je tada, prema pravilu proizvoda jednak $25 \cdot 30 = 750$.

Slično, broj različitih binarnih reči dužine n jednak je $|\{0,1\} \times \ldots \times \{0,1\}|$ (broj članova proizvoda je upravo n), što iznosi $|\{0,1\}|^n = 2^n$. ■

Pravilo zbira se odnosi na veličinu unije uzajamno disjunktnih skupova:

$$|A_1 \cup \cdots \cup A_n| = |A_1| + |A_2| + \cdots + |A_n|,$$

Primer 3.2.2 Neka od grada A do grada B postoje autobuska i železnička veza, i to svakog dana 3 polaska autobusa i 2 polaska voza. Broj načina da neko iz A stigne u B je prema pravilu zbira jednak $3 + 2 = 5$. ■

Pravilo uključenja-isključenja odnosi se na veličinu unije proizvoljnih skupova:

$$\begin{aligned} |A_1 \cup \ldots \cup A_n| &= |A_1| + \cdots + |A_n| \\ &- \text{ brojevi elemenata dvočlanih preseka} \\ &+ \text{ brojevi elemenata tročlanih preseka } \ldots \end{aligned}$$

što je za $n = 2$ izraz oblika:

$$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|,$$

a za $n = 3$:

$$|A_1 \cup A_2 \cup A_3| = |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|.$$

Primer 3.2.3 Posmatrajmo sve binarne reči dužine 4 koje bilo počinju sa 1 bilo završavaju sa 0. Binarnih reči oblika $1x_2x_3x_4$ ima 8, kao i binarnih reči oblika $y_1y_2y_30$. Ovde su reči oblika $1z_2z_30$ brojane dva puta, a njih ima 4. Odatle, broj traženih reči je prema pravilu uključenja-isključenja jednak $8 + 8 - 4 = 12$.

Odredimo koliko je prirodnih brojeva između 1 i 100 deljivo sa 3 ili 7. Primetimo da su brojevi deljivi i sa 3 i sa 7 zapravo deljivi sa 21 i ti brojevi u posmatranom skupu su 21, 42, 63 i 84, odnosno ima ih 4. Brojeva deljivih sa 3 ima 33, dok brojeva deljivih sa 7 ima 14. Prema pravilu uključenja-isključenja traženi broj je jednak $33 + 14 - 4 = 43$.

Pretpostavimo da je u sobi 12-oro ljudi, od kojih je 10 visokih i 5 vitkih. Prema pravilu uključenja-isključenja, ako je x broj visokih i vitkih, onda je $12 = 10 + 5 - x$, pa je $x = 3$. ■

Sledeći primer ilustruje situaciju u kojoj se prethodna pravila kombinuju.

Primer 3.2.4 Neka se šifra računarskog sistema formira kao reč dužine od 7 ili 8 znaka, pri čemu prvi znak mora biti slovo, a dozvoljeni znaci su mala i velika ASCII-slova i dekadne cifre. Posmatrajmo skupove

- $P = \{a, \ldots, z, A, \ldots, Z\}$ (iz kog se bira prvi znak u šifri) i
- $O = \{a, \ldots, z, A, \ldots, Z, 0, \ldots 9\}$ (iz kog se biraju ostali znaci u šifri),

za koje je $|P| = 52$ i $|O| = 62$. Ukupan broj mogućih šifara je tada

$$
\begin{aligned}
|(P \times O^6) \cup (P \times O^7)| &= |P \times O^6| + |P \times O^7|, \text{ prema pravilu zbira,}\\
&= |P| \cdot |O^6| + |P| \cdot |O^7|, \text{ prema pravilu proizvoda,}\\
&= 52 \cdot 62^6 + 52 \cdot 62^7.
\end{aligned}
$$

■

3.3 Permutacije i kombinacije

Definicija 3.3.1 *k-permutacija skupa od n elemenata* je svaka uređena k-torka različitih elemenata iz skupa od n elemenata. *Permutacija skupa od n elemenata* je svaka n-permutacija tog skupa. ■

Pored naziva k-permutacija koristi se i termin *varijacija bez ponavljanja k-te klase od n elemenata.*

Definicija 3.3.2 *k-kombinacija skupa od n elemenata* je svaki podskup od k (različitih) elemenata iz skupa od n elemenata. ■

U definicijama 3.3.1 i 3.3.2 treba obratiti pažnju na to da se kod permutacija vodi računa o redosledu elemenata, dok to nije slučaj sa kombinacijama. Takođe, u obe definicije naglasak je na *različitim* elementima polaznog skupa (što je u definiciji 3.3.2, iako suvišno, naglašeno). Kasnije ćemo odbaciti to ograničenje.

Primer 3.3.3 Neka je skup $A = \{1, 2, 3\}$. Sve 2-permutacije skupa A su:

$$
\begin{array}{ccc}
\langle 1,2\rangle & , & \langle 1,3\rangle,\\
\langle 2,1\rangle & , & \langle 2,3\rangle,\\
\langle 3,1\rangle & , & \langle 3,2\rangle
\end{array}
$$

i ima ih 6. Primetimo da su prema definiciji 3.3.1 različite 2-permutacije, na primer, $\langle 1, 2\rangle$ i $\langle 2, 1\rangle$.

Sve permutacije skupa A su

$$
\begin{array}{ccc}
\langle 1,2,3\rangle & , & \langle 1,3,2\rangle,\\
\langle 2,1,3\rangle & , & \langle 2,3,1\rangle,\\
\langle 3,1,2\rangle & , & \langle 3,2,1\rangle.
\end{array}
$$

U ovom slučaju postoji i 6 permutacija kupa A.

Sve 2-kombinacije skupa A su:

$$\{1, 2\}, \{1, 3\}, \{2, 3\}$$

i ima ih 3. Pošto skupovi nisu uređeni, ista 2-kombinacija je predstavljena i sa $\{1, 2\}$ i sa $\{2, 1\}$. ■

Primetimo da ako za svaku 2-kombinaciju skupa A posmatramo njene permutacije, onda ćemo dobiti sve 2-permutacije skupa A, što se može uopštiti i na proizvoljno k. Drugim rečima, sve k-permutacije skupa od n elemenata su zapravo permutacije odgovarajućih k-kombinacija skupa od n elemenata.

Broj k-permutacija skupa od n elemenata se označava sa $P(n, k)$. Broj permutacija skupa od n elemenata se označava sa $P(n, n)$. Broj k-kombinacija skupa od n elemenata se označava sa $C(n, k)$.

Teorema 3.3.4 Neka su $P(n, k)$, $P(n, n)$ i $C(n, k)$ redom broj k-permutacija skupa od n elemenata, broj permutacija skupa od n elemenata i broj k-kombinacija skupa od n elemenata. Tada:

1. $P(n, k) = n \cdot (n-1) \cdots (n-k+1) = \dfrac{n!}{(n-k)!}$,
2. $P(n, n) = n \cdot (n-1) \cdots 2 \cdot 1 = n!$
3. $C(n, k) = \dfrac{n!}{k!(n-k)!} = \dbinom{n}{k}$.

Dokaz. (1) Prvi član k-permutacije skupa od n elemenata može biti izabran na n načina. Pošto je izabran jedan element skupa, a u permutaciji se nalaze samo različiti elementi, preostaje $n-1$ element, tako da se sledeći član bira na $n-1$ način. Prema tome, prva dva člana se mogu izabrati na $n \cdot (n-1)$ način. Postupak produžavamo, tako da se poslednji, k-ti, član permutacije bira iz skupa koji sadrži $n-(k-1)$ element, pa postoji upravo toliko načina da i on bude izabran. Prema pravilu proizvoda, $P(n, k) = n \cdot (n-1) \cdots (n-k+1)$.
(2) Direktna posledica (1) za $k = n$.
(3) Proizvoljna k-kombinacija od n elemenata je jedan skup sa k elemenata. Broj permutacija tog skupa je $k!$. Ako je $C(n, k)$ broj k-kombinacija od n elemenata, onda je $k! \cdot C(n, k)$ ukupan broj k-permutacija od n elemenata, pa je $k! \cdot C(n, k) = \frac{n!}{(n-k)!}$, odnosno $C(n, k) = \frac{n!}{k!(n-k)!}$. ■

Izraz $\binom{n}{k}$ se naziva *binomni koeficijent.*

Primer 3.3.5 Lako se vidi da su u primeru 3.3.3, u kome je $|A| = |\{1, 2, 3\}| = 3$:

- broj 2-permutacija $P(3, 2) = \frac{3!}{(3-2)!} = \frac{6}{1} = 6$,
- broj permutacija $P(3, 3) = 3! = 6$ i
- broj 2-kombinacija $C(n, k) = \binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{3!}{2!(3-2)!} = \frac{6}{2} = 3$. ■

Definicijom 3.3.2 se kaže da je broj k-kombinacija od n elemenata zapravo broj k-to članih podskupova skupa od n elemenata, pa onda $C(n, k)$ određuje i taj broj.

Primer 3.3.6 Broj načina na koji možemo izabrati 3 od 10 knjiga je $C(10, 3) = \binom{10}{3} = \frac{10!}{3! \cdot 7!} = \frac{10 \cdot 9 \cdot 8}{1 \cdot 2 \cdot 3} = \frac{720}{6} = 120$.

Koristeći metodologiju iz teoreme 2.1.47, svaki podskup od k elemenata nekog n-to članog skupa se može prikazati binarnom reči dužine n koja ima tačno k jedinica. Broj podskupova sa tačno k elemenata skupa sa n elemenata, je upravo $C(n, k) = \binom{n}{k}$, pa je to i broj binarnih reči dužine n koje imaju tačno k jedinica. ■

Primetimo da smo u poslednjem delu primera 3.3.6 iskoristili transformaciju originalnog problema (za koji imamo rešenje) jednom bijekcijom u problem nad konačnim nizovima binarnih cifara (odnosno nad binarnim rečima) i onda dobili rešenje tog novog problema.

U sledećem primeru ćemo iskombinovati postupke za izračunavanje broja permutacija i transformaciju problema pomoću $k-$u-1-funkcije.

Primer 3.3.7 Neka 12 vitezova treba da sedne za okrugli sto sa isto toliko stolica. Smatramo da, ako je u neka dva konkretna razmeštaja vitezova ispunjeno:

- svaki od 12 vitezova ima istog suseda sa leve, odnosno sa desne strane,

da je reč o istom rasporedu, jer se posmatrani razmeštaji dobijaju rotacijom oko stola. Imajući to u vidu, broj različitih rasporeda na koji 12 vitezova okruglog stola mogu zauzeti mesta se određuje na sledeći način. Najpre, jasno je da su rasporedi vitezova neke od permutacije tog skupa.

Neka je A skup traženih rasporeda sedenja vitezova i neka je B skup svih permutacija skupa vitezova. Pošto je broj vitezova 12, to je $|B| = P(12, 12) = 12!$. Jednom rasporedu odgovara tačno 12 (zarotiranih) razmeštaja vitezova, dobijenih pomeranjem proizvoljno određenog prvog viteza za po jedno mesto. Dakle, postoji preslikavanje $f : B \mapsto A$ koje je $12-$u-1-funkcija. Odatle je traženi broj rasporeda $|A| = \frac{|B|}{12} = \frac{12!}{12} = 11!$. ■

3.4 Permutacije i kombinacije skupova sa višestrukim elementima

Iako suprotno dosadašjem pristupu, ponekad je pogodno posmatrati skupove u kojima postoji više istih elemenata, takozvane n-skupove. Na primer, time bismo mogli razlikovati skup $\{a, a, b\}$ od skupa $\{a, b\}$. Formalno gledano, mogli bismo te iste elemente indeksirati, pa umesto skupa $\{a, a, b\}$ posmatrati skup $\{a_1, a_2, b\}$, pri čemu bismo imali u vidu da su elementi a_1 i a_2 isti, ali bismo formalno radili kao da su različiti. Ovo može biti praktično kada posmatramo neke realne situacije u kojima brojimo određene resurse iste vrste, recimo nije nam svejedno da li imamo skup od 4 automobilske gume, ili je on isto što i skup koji ima samo jedan element - gumu za auto, a pri svemu tome ne želimo da označavanje nepotrebno komplikujemo. Imajući ovo u vidu, pogodno je uopštiti pristup i posmatrati i permutacije i kombinacije skupova sa višestrukim elementima. Pod permutacijama n (ne nužno različitih) objekata u ovakvim slučajevima podrazumevaćemo n-torke objekata bez obzira da li su oni različiti, ili ne.

Primer 3.4.1 Posmatrajmo reč *aparat* u kojoj imamo po jedno slovo p, r i t, ali i tri slova a. Permutacijama ovih slova dobijaju se reči poput *aaaprt*, *aaaptr*, itd. ■

Postavlja se pitanje koliko se različitih reči (permutacija) može dobiti od slova reči *aparat*.

Ako bismo slova a posmatrali kao različita i indeksirali ih kao a_1, a_2 i a_3, onda bi reči u kojima su prve tri pozicije zauzete njima, a poslednje tri pozicije oblika prt, bile:

$$\begin{array}{ccc}
\langle a_1, a_2, a_3, p, r, t\rangle & , & \langle a_1, a_3, a_2, p, r, t\rangle \\
\langle a_2, a_1, a_3, p, r, t\rangle & , & \langle a_2, a_3, a_1, p, r, t\rangle \\
\langle a_3, a_1, a_2, p, r, t\rangle & , & \langle a_3, a_2, a_1, p, r, t\rangle
\end{array}$$

Od permutacije *aaaprt* tako bismo dobili $3! = 6$ novih. Isto važi i za bilo koji drugi fiksiran raspored slova p, r i t.

Prema tome, ako sa x označimo traženi broj permutacija, onda je broj $x \cdot 3!$ jednak broju permutacija u kojima smo iste objekte indeksirali i smatramo ih za različite. U razmatranom slučaju bi bilo $x \cdot 3! = P(6, 6) = 6!$ (jer je $|\{a_1, a_2, a_3, p, r, t\}| = 6$), pa je $x = \frac{6!}{3!}$.

Sledeća teorema uopštava ovaj zaključak.

Teorema 3.4.2 Neka su je dato k vrsta objekata, tako da i-te vrste ima n_i objekata, i neka je $n = \sum_{i=1}^{k} n_i$. Tada je broj permutacija tih n objekata jednak

$$\frac{n!}{n_1! \cdots n_k!}.$$

Dokaz. Slično kao u prethodnom razmatranju, neka je x traženi broj. Za neku fiksiranu permutaciju posmatranih n objekata, ako indeksiramo objekte prve vrste, dobijamo $x \cdot n_1!$ permutacija u kojima se svi objekti prve vrste posmatraju kao različiti. Dalje, isto možemo ponoviti za objekte druge i ostalih vrsta, odakle dobijamo $x \cdot n_1! \cdots n_k! = P(n,n) = n!$, pa je traženi broj jednak $\frac{n!}{n_1! \cdots n_k!}$. ■

Primer 3.4.3 Broj različitih reči koje se dobijaju od slova reči *poplava* je prema teoremi 3.4.2 jednak $\frac{7!}{2! \cdot 2!}$, jer je dužina reči 7, a slova p i a se u njoj javljaju po 2 puta. ■

Kada se govori o k-permutacijama skupova sa višestrukim elementima, obično se podrazumeva da u svakoj vrsti postoji neograničen broj objekata, što se naziva i *k-permutacije sa ponavljanjem*. Jasno je da ovde može biti i $k > n$, gde je n broj vrsta objekata. k-permutacije sa ponavljanjem se mogu posmatrati i na sledeći način: u svakoj vrsti postoji samo jedan objekat, ali se on može birati neograničen broj puta. Tada je broj n vrsta objekata ujedno i broj objekata.

Teorema 3.4.4 Neka su je dato n objekata. Broj k-permutacija sa ponavljanjem je jednak n^k.

Dokaz. Pošto u svakom od k koraka izbora možemo izabrati jedan od n elemenata, rezultat neposredno sledi prema pravilu proizvoda. ■

Primer 3.4.5 Broj različitih binarnih reči dužine 8 je $2^8 = 256$.

Reči dužine 2 sastavljenih od slova $\{a, b, c\}$ ima $3^2 = 9$: aa, ab, ac, ba, bb, bc, ca, cb i cc. ■

Slično k-permutacijama sa ponavljanjem, mogu se posmatrati i *k-kombinacije sa ponavljanjem*. Kao i ranije, kod kombinacija se posmatraju mogući izbori skupova, ali sada su to skupovi sa višestrukim elementima. Dakle, razmatraju se situacije kada je na raspolaganju n objekata, ali se svaki od njih može birati više puta.

Teorema 3.4.6 Neka su je dato n objekata. Broj k-kombinacija sa ponavljanjem je jednak

$$\binom{n+k-1}{k} = \frac{(n+k-1)!}{k!(n-1)!}.$$

Dokaz. Bez gubitka opštosti, posmatrajmo skup $C = \{1, 2, \ldots, n\}$ i njegove k-kombinacije sa ponavljanjem. Neka je $\{c_1, \ldots, c_k\}$ jedna takva kombinacija, pri čemu pretpostavimo i da je $c_i \leq c_{i+1}$, za $i = 1, k-1$. Ovo je moguće jer svaku kombinaciju posmatramo kao skup, pa redosled nije od značaja. Svakoj takvoj kombinaciji pridružimo kombinaciju (bez ponavljanja) $b_1 \dots b_k$ brojeva iz skupa $B = \{1, 2, \dots, n, n+1, \dots, n+k-1\}$, tako da je

$$b_1 = c_1,\ b_2 = c_2 + 1,\ \dots,\ b_k = c_k + k - 1.$$

Uočimo da između k-kombinacija sa ponavljanjem skupa C i k-kombinacija (bez ponavljanja) skupa B postoji bijekcija. Naime, svakoj kombinaciji $c_1 \dots c_k$ odgovara tačno jedna kombinacija $b_1 \dots b_k$ i obrnuto. Pošto je broj k-kombinacija (bez ponavljanja) skupa B jednak $\binom{n+k-1}{k}$, tvrđenje sledi neposredno. ■

Poredeći ovaj rezultat sa tvrđenjem 3.3.4.3, lako se vidi da je broj k-kombinacija sa ponavljanjem n elemenata jednak broju k-kombinacija od $n+k-1$ elementa, tj. $C(n+k-1, k)$.

Primer 3.4.7 Razmotrimo na koliko je različitih načina moguće obojiti 5 loptica plavom, žutom i crvenom boje. Pošto koristimo $n = 3$ boje (uz pretpostavku da je njihova količina dovoljna), koristeći formulu iz tvrđenja 3.4.6, odgovor je $\binom{3+5-1}{5} = \binom{7}{5} = \frac{7!}{5!(7-5)!} = 21$. ■

3.5 Kombinatorijalni dokazi

U ovom odeljku ćemo razmotriti primenu tehnika prebrojavanja u dokazivanju nekih identiteta, što se naziva i *kombinatorijani dokaz*. Takvi dokazi često su u sledećoj formi. Posmatra se neki skup A i pomoću postupaka prebrojavanja se utvrdi da je $|A| = n$ i $|A| = m$, odakle se zaključuje da je $n = m$. Primer za to je:

Teorema 3.5.1 (Pascal-ov identitet) [2]

$$\binom{n-1}{k-1} + \binom{n-1}{k} = \binom{n}{k}.$$

Dokaz. Podsetimo se da je $\binom{n}{k}$ broj k-točlanih podskupova n-točlanog skupa A. Fiksirajmo a kao jedan od elemenata skupa. Svi k-točlani podskupovi skupa A se dele u dve grupe:

- skupovi koji sadrže a i njih ima $\binom{n-1}{k-1}$ i
- skupovi koji ne sadrže a i njih ima $\binom{n-1}{k}$,

odakle tvrđenje sledi direktno. ■

[2] Blaise Pascal, 1623 – 1662.

Binomni koeficijenti $\binom{n}{k}$ se određuju iz *Pascal-ovog trougla*:

$$\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1 \\
 & & & & \cdots & & & &
\end{array}$$

koji zapravo predstavlja zapis za

$$\begin{array}{ccccccccc}
 & & & & \binom{0}{0} & & & & \\
 & & & \binom{1}{0} & & \binom{1}{1} & & & \\
 & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & \\
 & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & \\
\binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4} \\
 & & & & \cdots & & & &
\end{array}$$

koji se lako obrazuje pomoću Pascal-ovog identiteta, a $\binom{n}{k}$ se nalazi u $n+1$ redu na $k+1$ poziciji. Drugi primer primene binomnih koeficijenata je:

Teorema 3.5.2 (Binomna teorema)

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k.$$

Dokaz. Najpre je

$$(x+y)^n = \underbrace{(x+y)\cdots(x+y)}_{n},$$

gde proizvod sa desne strane jednakosti ima tačno n faktora. Primenom distributivnih zakona dobija se zbir od 2^n proizvoda oblika $e_1 \cdot e_2 \cdots e_n$, gde je svaki od e_i dobijen izborom iz skupa $\{x, y\}$ u i-tom faktoru proizvoda. Proizvode $e_1 \cdot e_2 \cdots e_n$ možemo posmatrati kao reči nad dvočlanim skupom, pa proizvoda $e_1 \cdot e_2 \cdots e_n$ koji se svode na $x^{n-k}y^k$ ima upravo $\binom{n}{k}$, odakle sledi tvrđenje. ∎

Lako se vidi da jednostavnim čitanjem redova iz Pascal-ovog možemo dati pun zapis nekog stepena izraza $(x+y)$. Pri tome, za traženi stepen n treba čitati red $n+1$, na primer za $n=4$ je

$$(x+y)^4 = 1 \cdot x^4 + 4 \cdot x^3 \cdot y + 6 \cdot x^2 \cdot y^2 + 4 \cdot x \cdot y^3 + 1 \cdot y^4.$$

Jednostavne posledice tvrđenja 3.5.2 su (za $x = y = 1$):

$$2^n = \sum_{k=0}^{n} \binom{n}{k} = \binom{n}{0} + \binom{n}{1} + \ldots + \binom{n}{n}$$

i (za $y = 1$):

$$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k.$$

3.6 Rekurentne relacije

Rekurentna relacija je jednačina u kojoj se n-ti član niza definiše preko svojih prethodnika. Verovatno najpoznatiji primer niza definisanog rekurentnom relacijom predstavljaju *Fibonačijevi brojevi*[3] određeni:

- inicijalnim vrednostima $F_0 = 0$ i $F_1 = 1$ i
- rekurentnom relacijom $F_n = F_{n-1} + F_{n-2}$,

tako da je:

- $F_2 = F_1 + F_0 = 1$,
- $F_3 = F_2 + F_1 = F_1 + F_0 + F_1 = 2 \cdot F_1 + F_0 = 2$ itd.

Prvih nekoliko članova niza su: $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

Definicija 3.6.1 *Linearna homogena rekurentna relacija sa konstantnim simbolima reda* k je jednačina oblika:

$$a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}$$

gde su $c_1, \ldots, c_k$ konstante, $c_k \neq 0$, i $n \geq k$. Niz $\{x_n\}_n$ je *rešenje* ove relacije ako je za svako $n \geq k$ ispunjeno da je $x_n = \sum_{i=1}^{k} c_i \cdot x_{n-i}$. $\blacksquare$

[3] Leonardo Pisano Bigollo, Leonardo Fibonacci (filius Bonacci, "sin Bonaccio-a"), oko 1170 - oko 1250. Fibonači je u tekstu "Liber Abaci" ("Knjiga o abakusu", ili: "Knjiga o izračunavanju") iz 1202. godine, predstavljajući istočnjačke matematičke rezultate zapadno-evropskoj naučnoj publici, opisao niz brojeva kasnije nazvan po njemu. Niz se pojavljuje u sledećem zadatku: ako se par zečeva razmnožava svakog meseca i dobija novi par (ženku i mužjaka), a novi par se kroz dva meseca razmnožava na isti način, koliko zečeva će biti na kraju godine ako je na početku godine bio jedan par zečeva spremnih za reprodukciju? Pravilnost koja određuje članove ovog niza se često javlja u prirodi. Na primer, količnik uzastopnih članova niza teži odnosu zlatnog preseka $\frac{1+\sqrt{5}}{2} = 1,618\ldots$

Primetimo da tek davanje inicijalnih vrednosti $a_0, \ldots, a_{k-1}$ za početne članove na jedinstveni način određuje niz $\{x_n\}_n$ iz definicije 3.6.1, a da bez tih vrednosti postoji beskonačno mnogo rešenja, tj. nizova koji zadovoljavaju konkretnu rekurentnu relaciju.

Pored homogenih razmatraju se i *linearne nehomogene rekurentne relacije sa konstantnim simbolima reda* k koje su oblika

$$a_n = f(n) + \sum_{i=1}^{k} c_i \cdot a_{n-i},$$

gde se pored ostalih uslova iz definicije 3.6.1, još zatheva da $f(n)$ nije nula-funkcija. Za ovakve relacije nije poznat opšti metod za pronalaženje rešenja. U nastavku ćemo razmatrati samo homogene relacije, tako da to neće više biti posebno naglašavano.

Primer 3.6.2 Fibonačijev niz brojeva je određen linearnom rekurentnom relacijom sa konstantnim simbolima reda 2 u kojoj su $c_1 = c_2 = 1$, $F_n = F_{n-1} + F_{n-2}$ i inicijalne vrednosti $F_0 = 0$ i $F_1 = 1$.

Označimo sa s_n broj podskupova skupa sa n elemenata. Tada je:

- $s_0 = 1$, a
- $s_{n+1} = 2 \cdot s_n$. ■

Kada razmatramo neku rekurentnu relaciju postavlja se pitanje kako odrediti njena rešenja, tj. nizove koji je zadovoljavaju.

Definicija 3.6.3 *Opšte rešenje* linearne rekurentne relacije $a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}$ reda k je niz koji zavisi od k parametara $\alpha_1, \ldots, \alpha_k$ čijim pogodnim izborom se dobija svako drugo rešenje posmatrane relacije. ■

Dakle, cilj nam je da polazeći od neke rekurentne relacije konstruišemo njeno opšte rešenje iz koga se kasnije dobijaju svi konkretni nizovi koji zadovoljavaju relaciju.

Definicija 3.6.4 *Karakteristični polinom* linearne rekurentne relacije sa konstantnim simbolima reda k

$$a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k}$$

je polinom oblika

$$P(x) = x^k - c_1 \cdot x^{k-1} - c_2 \cdot x^{k-2} - \cdots - c_k$$

sa koeficijentima $c_1, \ldots, c_k$ koji se javljaju u rekurentnoj relaciji. ■

U nastavku ćemo razmotriti veze rešenja nekog karakterističnog polinoma sa nizovima koji zadovoljavaju odgovarajuću rekurentnu relaciju. Jednostavnosti radi, pretpostavimo da je red rekurentne relacije $k = 2$.

Teorema 3.6.5 Neka je $a_{n+2} = c_1 \cdot a_{n-1} + c_2 \cdot a_n$ rekurentna relacija i $x^2 - c_1 \cdot x - c_2$ njoj odgovarajući karakteristični polinom. Ako je r koren karakterističnog polinoma, onda niz $\{r^i\}_{i=0}^{\infty}$ zadovoljava rekurentnu relaciju.

Dokaz. Ako je r rešenje jednačine $x^2 - c_1 \cdot x - c_2 = 0$, onda su opšti članovi posmatranog niza oblika $a_n = r^{n-1}$, $a_{n+1} = r^n$ i $a_{n+2} = r^{n+1}$. Prema rekurentnoj relaciji važi

$$r^{n+1} = c_1 \cdot r^n + c_2 \cdot r^{n-1},$$

odnosno ekvivalentno

$$0 = r^{n-1}(r^2 - c_1 \cdot r - c_2),$$

što je tačno s obzirom da je r rešenje jednačine $x^2 - c_1 \cdot x - c_2 = 0$. ■

Teorema 3.6.6 Neka je $a_{n+2} = c_1 \cdot a_{n+1} + c_2 \cdot a_n$ rekurentna relacija i neka su nizovi $\{x_i\}_i$ i $\{y_i\}_i$ njena rešenja i A i B proizvoljni realni brojevi. Tada je i niz

$$\{A \cdot x_i + B \cdot y_i\}_i$$

rešenje posmatrane relacije.

Dokaz. Pošto su $\{x_i\}_i$ i $\{y_i\}_i$ rešenja važi:

- $x_{n+2} = c_1 \cdot x_{n+1} + c_2 \cdot x_n$,
- $A \cdot x_{n+2} = c_1 \cdot A \cdot x_{n+1} + c_2 \cdot A \cdot x_n$,
- $y_{n+2} = c_1 \cdot y_{n+1} + c_2 \cdot y_n$,
- $B \cdot y_{n+2} = c_1 \cdot B \cdot y_{n+s1} + c_2 \cdot B \cdot y_n$ i
- $A \cdot x_{n+2} + B \cdot y_{n+2} = c_1(A \cdot x_{n+1} + B \cdot y_{n+1}) + c_2(A \cdot x_n + B \cdot y_n)$,

pa je i posmatrana linearna kombinacija rešenje rekurentne relacije. ■

Za karakteristični polinom $x^2 - c_1 \cdot x - c_2$ rekurentne relacije $a_{n+2} = c_1 \cdot a_{n-1} + c_2 \cdot a_n$, razmotrićemo dva slučaja:

- kada je $c_1^2 + 4c_2 > 0$, tj. kada karakteristični polinom ima dva realna različita korena i

- kada je $c_1^2 + 4c_2 = 0$, tj. kada karakteristični polinom ima dvostruki realni koren.

Teorema 3.6.7 Neka su r_1 i r_2 dva različita realna rešenje jednačine $x^2 - c_1 \cdot x - c_2 = 0$. Tada je niz

$$\{d_1 \cdot r_1^n + d_2 \cdot r_2^n\}_n,$$

gde su d_1 i d_2 parametri (proizvoljne konstante), opšte rešenje rekurentne relacije $a_{n+2} = c_1 \cdot a_{n+1} + c_2 \cdot a_n$.

Dokaz. Najpre, prema tvrđenju 3.6.5, nizovi $\{r_1^n\}_n$ i $\{r_2^n\}_n$ jesu rešenja posmatrane rekurentne relacije, a prema tvrđenju 3.6.6 to je i njihova proizvolja linearna kombinacija.

Neka je $\{b_n\}_n$ proizvoljno rešenje posmatrane rekurentne relacije. Pokazaćemo da se d_1 i d_2 mogu odrediti tako da je $b_n = d_1 \cdot r_1^n + d_2 \cdot r_2^n$. Pošto je svako konkretno rešenje jednoznačno odeđeno vrednostima prva dva člana niza koje označavamo sa b_0 i b_1, zamenom dobijamo sistem:

$$\begin{array}{rcl} d_1 + d_2 & = & b_0 \\ d_1 r_1 + d_2 r_2 & = & b_1 \end{array}$$

koji ima jedinstvena rešenja po d_1 i d_2:

$$d_1 = \frac{b_1 - b_0 r_2}{r_1 - r_2},\ d_2 = \frac{b_1 r_1 - b_2}{r_1 - r_2}$$

jer je $r_1 \neq r_2$. Odatle je rešenje $\{b_n\}_n$ zaista dobijeno iz opšteg rešenja. ■

Teorema 3.6.8 Neka je r jedinstveno realno rešenje jednačine $x^2 - c_1 \cdot x - c_2 = 0$. Tada je niz

$$\{d_1 \cdot r^n + d_2 \cdot (n+1) \cdot r^n\}_n,$$

gde su d_1 i d_2 parametri (proizvoljne konstante), opšte rešenje rekurentne relacije $a_{n+2} = c_1 \cdot a_{n+1} + c_2 \cdot a_n$.

Dokaz. Pošto karakteristični polinoim $x^2 - c_1 \cdot x - c_2$ ima dvostruki realni koren r, sledi da je $c_1^2 + 4 \cdot c_2 = 0$ i $r = \frac{c_1}{2}$, odnosno $2 \cdot r - c_1 = 0$. Zbog toga, kao i zbog činjenice da je r koren polinoma važi jednakost:

- $r^{n-1}[n(r^2 - c_1 \cdot r - c_2) + r(2 \cdot r - c_1)] = 0$

koja se drugačije zapisuje sa:

- $(n+2)r^{n+1} = c_1(n+1)r^n + c_2 \cdot n \cdot r^{n-1}$, pa je niz $\{(n+1)\cdot r^n\}_n$ rešenje posmatrane rekurentne relacije. Kao i ranije, isto važi i za $\{r^n\}_n$ i za njihovu linearnu kombinaciju $\{d_1\cdot r^n+d_2(n+1)\cdot r^n\}_n$. Kao i u dokazu tvrđenja 3.6.7, pokazuje se da je ovo opšte rešenje jer sistem

$$\begin{array}{rcl} d_1 + d_2 & = & b_0 \\ d_1 r + d_2 2r & = & b_1 \end{array}$$

ima jedinstvena rešenja $d_1 = 2b_0 - \dfrac{b_1}{r}$, $d_2 = \dfrac{b_1}{r} - b_0$. ■

Opšte tvrdjenje se može formulisati na sledeći način. Neka su dati linearna rekurentna relacija sa konstantnim simbolima reda k $a_n = c_1 \cdot a_{n-1} + c_2 \cdot a_{n-2} + \cdots + c_k \cdot a_{n-k}$ i njen karakteristični polinom $P(x) = x^k - c_1 \cdot x^{k-1} - c_2 \cdot x^{k-2} - \cdots - c_k$. Neka su svi koreni polinoma $r_1, \ldots, r_k$ međusobno različiti, onda je opšte rešenje rekurentne relacije oblika:

$$\{d_1 r_1^n + d_2 r_2^n + \cdots + d_k r_k^n\}_n.$$

Ako je neki koren r_a polinoma višestruk (reda l), onda će njemu odgovarati član u zbiru (koji daje opšte rešenje) oblika

$$r_a^n[d_1 + d_2(n+1)\cdots + d_l(n+1)^{l-1}].$$

Primer 3.6.9 Linearna rekurentna relacija za Fibonačijev niz brojeva je oblika

$$F_n = c_1 \cdot F_{n-1} + c_2 \cdot F_{n-2},$$

pri čemu važi $c_1 = c_2 = 1$. Ponovimo da bez zadavanja vrednosti prva dva člana niza ova relacija određuje beskonačno mnogo nizova. Karakteristični polinom je oblika

$$x^2 - x - 1$$

i njegovi koreni su

$$r_1 = \frac{1+\sqrt{5}}{2} \quad \text{i} \quad r_2 = \frac{1-\sqrt{5}}{2}.$$

Prema tvrđenju 3.6.7,

$$\left\{ d_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + d_2 \left(\frac{1-\sqrt{5}}{2}\right)^n \right\}_n$$

je opšte rešenje relacije. S obzirom da su inicijalne vrednosti Fibonačijevog niza $F_0 = 0$ i $F_1 = 1$, dobija se da je

$$d_1 = \frac{1}{\sqrt{5}} \quad \text{i} \quad d_2 = -\frac{1}{\sqrt{5}}.$$

Tada je formula opšteg člana Fibonačijevog niza

$$F_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right].$$

Zanimljivo je uočiti da su vrednosti ovih izraza uvek prirodni brojevi. ■

Sledeći primer ilustruje jednu primenu rekurentnih relacija u prebrojavanju.

Primer 3.6.10 Neka je s_n broj binarnih reči dužine n koje ne sadrže podreč 00. Lako se vidi da je $s_1 = 2$, jer obe reči dužine 1 ispunjavaju traženo svojstvo, kao i da je $s_2 = 3$, jer je jedina od 4 binarne reči dužine 2 koja ne ispunjava svojstvo upravo 00.

Ako sa F_n označimo n-ti član Fibonačijevog niza, uočimo da je $s_1 = F_3$, a $s_3 = F_4$. Dalje, kao i kod Fibonačijevog niza, rekurentna relacija za niz $\{s_n\}_n$ je oblika $s_n = s_{n-1} + s_{n-2}$. Ovo se pokazuje na sledeći način. Svaka binarna reč koja ne sadrži podreč 00 završava se sa 1 ili 10. Broj reči prvog tipa (dužine n koje se završavaju sa 1) je upravo s_{n-1} jer brisanjem poslednjeg znaka 1 iz binarnih reči dužine n dobijamo reči dužine $n-1$ u kojima se ne nalazi podreč 00. Slično, broj reči drugog tipa (dužine n koje se završavaju sa 10) je s_{n-2}.

Odavde zaključujemo da je broj binarnih reči dužine n koje ne sadrže podreč 00 određen sa $s_n = F_{n+2}$. ■

3.7 Generatorne funkcije

Pretpostavimo da rešenje nekog problema predstavlja niz $\{a_n\}_n$. To rešenje možemo opisati:

- formulom za opšti član niza a_n, ili
- formulom za sumu stepenog reda $\sum_{n=0}^{\infty} a_n \cdot x^n$ čiji su koeficijenti članovi traženog niza.

U drugom slučaju, odgovor bi bio sledećeg oblika: n-ti član Fibonačijevog niza je koeficijent uz x^n u razvoju funkcije $\dfrac{x}{1-x-x^2}$ u stepeni red. U ovakvom pristupu koriste se *generatorne funkcije*[4] koje su formalni zapisi stepenih redova čiji koeficijenti kodiraju informaciju o nizovima brojeva. Generatorne funkcije predstavljaju svojevrsnu sponu diskretne matematike i analize i često omogućavaju:

[4]Generating functions. Koristi se i naziv *proizvodne funkcije*. Uveo ih je Abraham de Moivre, 1667 – 1754, kao sredstvo za rešavanje problema rekurentnih relacija.

- rešavanje problema u kojima drugi pristupi nisu efikasni i
- analizu eventualnog rešenja (recimo, asimptotsko ponašanje formule traženog niza) čak i kada samo rešenje nije poznato (na primer, za veliko n, n-ti prost broj se može aproksimirati sa $n \ln n$).

Pridev 'formalni' koji se pojavljuje u objašnjenju šta su generatorne funkcije koristi se u sledećem smislu: najčešće se domeni i kodomeni ovih funkcija ne određuju, kao što se ni njihove vrednosti uglavnom ne izračunavaju za konretne vrednosti argumenata. Naglasak se stavlja na operacije u algebarskoj strukturi stepenih redova koje se primenjuju prilikom manipulacije koeficijentima odgovarajućih redova, pri čemu se ne koriste osobine funkcija kojima eventualno ti redovi konvergiraju. Tako će se, na primer, u postupku vršiti diferenciranje, a tek na kraju će se videti da li dobijeni red konvergira nekoj funkciji ili ne.

Definicija 3.7.1 Stepeni red

$$G(a) = \sum_{n=0}^{\infty} a_n \cdot x^n$$

je *generatorna funkcija* niza brojeva $\{a_n\}_n$. ■

Pored ovako definisanih, takozvanih *običnih* postoje i druge vrste generatornih funkcija: eksponencijalne, Lamber-ove, Bell-ove, Dirichlet-ove itd., ali se u ovom tekstu njima nećemo baviti, tako da neće biti potrebno ni da se upotrebljava pridev 'obične'. Razmotrimo sada nekoliko primera generatornih funkcija.

Primer 3.7.2 Ako je niz $\{a_n\}_n$ oblika $\langle 1, 0, 0, \ldots \rangle$, tj. dat sa $a_0 = 1$ i $a_n = 0$, za $n > 0$, onda je generatorna funkcija:

- $G(a) = 1$.

Ako je niz $\{a_n\}_n$ oblika $\langle c, 0, 0, \ldots \rangle$, tj. dat sa $a_0 = c$, za neku konstantu c, a $a_n = 0$, za $n > 0$, onda je generatorna funkcija:

- $G(a) = c$, tj. konstanta funkcija.

Ako je niz $\{a_n\}_n$ oblika $\langle 0, 1, 0, \ldots \rangle$, tj. dat sa $a_0 = 0$, $a_1 = 1$ i $a_n = 0$, za $n > 1$, onda je generatorna funkcija:

- $G(a) = x$, tj. identička funkcija.

Ako je niz $\{a_n\}_n$ oblika $\langle 1, 1, 1, \ldots \rangle$, tj. dat sa $a_n = 1$, za $n \geq 0$, onda je generatorna funkcija geometrijski red:

- $G(a) = 1 + x + x^2 + \cdots + x^n + \cdots$ ■

Primetimo da za $x \in \mathbb{R}$, za koje je $|x| < 1$, geometrijski red poslednje generatorne funkcije iz primera 3.7.2 konvergira i da je on zapravo razvoj funkcije

$$\frac{1}{1-x}$$

u Tejlorov red[5]. Imajući na umu početne napomene o formalnom radu sa generatornim funkcijama, ovde se ne vodi računa o konvergentnosti, a izraz $\frac{1}{1-x}$ se naziva *zatvorena forma* generatorne funkcije $1+x+x^2+\cdots+x^n+\cdots$.

U sledećem primeru biće ilustrovan metod korištenja generatornih funkcija.

Primer 3.7.3 Razmotrimo najpre niz $\{a_n\}_n$ za koji važi:

- $a_0 = 1$,
- članovi su rekurentno definisani sa $a_{n+1} = a_n$, $n \in \mathbb{N}$, odnosno $a_n = 1$ za $n \in \mathbb{N}$,

i neka je $G(a) = \sum_{n=0}^{\infty} a_n \cdot x^n$. Najpre pomnožimo obe strane rekurentne relacije sa x^n i sumirajmo. Sa leve strane ćemo dobiti

$$\sum_{n\geq 0} a_{n+1}x^n = \frac{1}{x}[(a_0 + a_1x + a_2x^2 + \cdots) - a_0] = \frac{G(a)-1}{x},$$

jer je $a_0 = 1$, dok će desna strana dati

$$\sum_{n\geq 0} a_n x^n = G(a).$$

Izjednačavanjem se dobija

$$\frac{G(a)-1}{x} = G(a),$$

tj. zatvorena forma generatorne funkcije niza a je

$$G(a) = \frac{1}{1-x}.$$

Na sličan način se pokazuje da su zatvorene forme generatornih funkcija nizova $\{c^n\}_n$ i $\{n+1\}_n$ redom jednake $\dfrac{1}{1-cx}$, odnosno $\dfrac{1}{(1-x)^2}$. Dalje, pretpostavimo da tražimo niz $\{b_n\}_n$ za koji važi:

---

[5] Red $f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f^{(3)}(a)}{3!}(x-a)^3 + \cdots$ kojim se aproksimira vrednost funkcije u okolini tačke nosi naziv po Brook-u Taylor-u (1685 - 1731), mada se smatra da je prvi do tog resultata došao James Gregory (1638 - 1675).

- $b_0 = 0$,
- članovi su rekurentno definisani sa $b_{n+1} = 2b_n + 1$, $n \in \mathbb{N}$ i
- $G(b) = \sum_{n=0}^{\infty} b_n \cdot x^n$.

Kao i malopre, pomnožimo obe strane rekurentne relacije sa x^n i sumirajmo. Dobićemo:

$$\sum_{n\geq 0} b_{n+1}x^n = \frac{1}{x}[(b_0 + b_1x + b_2x + \cdots) - b_0] = \frac{G(b)}{x},$$

odnosno

$$\sum_{n\geq 0}(2b_n + 1)x^n = 2G(b) + \sum_{n\geq 0} x^n = 2G(b) + \frac{1}{1-x},$$

gde je iskorišteno da je $b_0 = 0$, a da je zatvorena forma generatorne funkcije niza $\{1\}_n$ jednaka $\frac{1}{1-x}$. Izjednačavanjem se dobija

$$\frac{G(b)}{x} = 2G(b) + \frac{1}{1-x},$$

odnosno

$$G(b) = \frac{x}{(1-x)(1-2x)}.$$

Konačno, članove niza $\{b_n\}_n$ određujemo iz:

$$\begin{aligned}
G(b) &= x\left(\frac{2}{1-2x} - \frac{1}{1-x}\right)\\
&= x\Big(2\cdot(1 + 2x + 2^2x^2 + 2^3x^3 + \cdots) - (1 + x + x^2 + x^3 + \cdots)\Big)\\
&= x\Big(2 + 2^2x + 2^3x^2 + 2^4x^3 + \cdots - 1 + x + x^2 + x^3 + \cdots\Big)\\
&= x\Big((2^1-1) + (2^2-1)x + (2^3-1)x^2 + (2^4-1)x^3 + \cdots\Big)\\
&= (2^1-1)x + (2^2-1)x^2 + (2^3-1)x^3 + (2^4-1)x^4 + \cdots
\end{aligned}$$

odakle je $b_n = 2^n - 1$ za $n \in \mathbb{N}$. ■

Slede primeri još nekoliko zatvorenih formi.

Primer 3.7.4 Ako je niz $\{a_n\}_n$ oblika $\langle 1, c, c^2, \ldots\rangle$, tj. dat sa $a_n = c^n$, za $n \geq 0$ i neku konstantu c, onda je za generatornu funkciju:

- $G(a) = 1 + c\cdot x + \cdots + c^n \cdot x^n + \cdots$ zatvorena forma izraz $\frac{1}{1 - c\cdot x}$.

Ako je niz $\{a_n\}_n$ oblika $\langle 1, -1, 1, \ldots\rangle$, tj. dat sa $a_n = (-1)^n$ za $n \geq 0$, onda je za generatornu funkciju:

- $G(a) = 1 - x + x^2 - x^3 + x^4 - \cdots$ zatvorena forma izraz $\dfrac{1}{1+x}$.

Ako je niz $\{a_n\}_n$ oblika $\langle 1, 0, 1, 0, \ldots\rangle$, tj. dat sa $a_{2n} = 1$ i $a_{2n+1} = 0$ za $n \geq 0$, onda je za generatornu funkciju:

- $G(a) = 1 + x^2 + x^4 + \cdots$ zatvorena forma izraz $\dfrac{1}{1-x^2}$.

Ako je niz $\{a_n\}_n$ dat sa $a_n = \frac{1}{n!}$ za $n \geq 0$, onda je za generatornu funkciju:

- $G(a) = 1 + x + \frac{1}{2}x^2 + \cdots$ zatvorena forma izraz e^x.

Ako je niz $\{a_n\}_n$ dat sa $a_i = \binom{k}{i}$ za neko fiksirano k i $i = 0, 1, \ldots, k$, a $a_i = 0$ za $i > k$, odnosno

$$a = \left\langle \binom{k}{0}, \binom{k}{1}, \ldots, \binom{k}{k-1}, \binom{k}{k}, 0, 0, \ldots \right\rangle,$$

onda je za generatornu funkciju:

- $G(a) = \binom{k}{0} + \binom{k}{1}x + \ldots + \binom{k}{k}x^k$ zatvorena forma izraz $(1+x)^k$. ■

3.7.1 Operacije sa generatornim funkcijama

Kao što je na početku rečeno, sa generatornim funkcijama se radi formalno, ne vodeći računa da li je reč o konveregentim redovima.

Definicija 3.7.5 Generatorne funkcije $G(a)$ i $G(b)$ nizova $\{a_n\}_n$ i $\{b_n\}_n$ su jednake ako su jednaki odgovarajući nizovi, tj. ako je za svako $n \in \mathbb{N}$, $a_n = b_n$. ■

U nastavku ćemo definisati nekoliko operacija sa generatornim funkcijama.

Definicija 3.7.6 (Pravilo skaliranja) Neka je $G(a)$ generatorna funkcija niza $\{a_n\}_n$ i neka je c neka konstanta. Tada je $G(c \cdot a) = c \cdot G(a)$ generatorna funkcija niza $\{c \cdot a_n\}_n$. ■

Primer 3.7.7 U primeru 3.7.2 je ilustrovano da je $G(a) = 1$ generatorna funkcija niza $a = \langle 1, 0, 0, \ldots\rangle$, dok je $G(c \cdot a) = c$ generatorna funkcija niza $c \cdot a = \langle c, 0, 0, \ldots\rangle$.

Slično, u primeru 3.7.4 je data generatorna funkcija $G(a) = 1 + x^2 + x^4 + \cdots$ niza $a = \langle 1, 0, 1, 0, \ldots\rangle$, za koju je zatvorena forma $\dfrac{1}{1-x^2}$. Tada je za niz $2 \cdot a = \langle 2, 0, 2, 0, \ldots\rangle$ generatorna funkcija $G(a) = 2 + 2x^2 + 2x^4 + \cdots$, za koju je zatvorena forma $\dfrac{2}{1-x^2}$. ■

Definicija 3.7.8 (Pravilo sabiranja) Neka su $G(a)$ i $G(b)$ generatorne funkcije nizova $\{a_n\}_n$ i $\{b_n\}_n$. Tada je $G(a \pm b) = G(a) \pm G(b)$ generatorna funkcija niza $\{a_n \pm b_n\}_n$. ■

Primer 3.7.9 Generatorne funkcije nizova $a = \langle 1, 1, 1, 1, \ldots \rangle$ i $b = \langle 1, -1, 1, -1, \ldots \rangle$ su redom $G(a) = 1 + x + x^2 + x^3 + x^4 + \cdots$ i $G(b) = 1 - x + x^2 - x^3 + x^4 - \cdots$, dok su njima odgovarajuće zatvorene forme izrazi redom $\frac{1}{1-x}$ i $\frac{1}{1+x}$. Prema pravilu sabiranja, generatorna funkcija zbira nizova $a + b = \langle 2, 0, 2, 0, \ldots \rangle$ je $G(a + b) = 2 + 2 \cdot x^2 + 2 \cdot x^4 + \cdots$. Pri tome je i odgovarajuća zatvorena forma oblika

$$\frac{1}{1-x} + \frac{1}{1+x} = \frac{2}{1-x^2}.$$

Ovde treba obratiti pažnju da su, prema primeru 3.7.7, zatvorene forme generatornih funkcija nizova $c = \langle 1, 0, 1, 0, \ldots \rangle$ i $2 \cdot c = \langle 2, 0, 2, 0, \ldots \rangle$ redom $\dfrac{1}{1-x^2}$, odnosno $\dfrac{2}{1-x^2}$. Očigledno je da su u ovom primeru nizovi $a + b$ i $2 \cdot c$ jednaki, i nije čudno što su jednake i odgovarajuće zatvorene forme, od kojih je jedna dobijena pravilom sabiranja, a druga pravilom skaliranja. ■

Definicija 3.7.10 (Pravilo proizvoda) Neka su $G(a)$ i $G(b)$ generatorne funkcije nizova $\{a_n\}_n$ i $\{b_n\}_n$. Tada je

$$G(a \cdot b) = \sum_{n=0}^{\infty} (\sum_{k=0}^{n} a_k \cdot b_{n-k}) \cdot x^n$$

generatorna funkcija niza koji se dobija proizvodom nizova $\{a_n\}$ i $\{b_n\}_n$. ■

Imajući u vidu pravilo proizvoda za generatorne funkcije, dobijamo da je:

$$(1 - x)(1 + x + x^2 + x^3 + \cdots) = 1,$$

pa je stepeni red čiji su koeficijenti $a_0 = 1$, $a_1 = -1$, $a_k = 0$, za $k \geq 2$ *inverzan* stepenom redu sa koeficijentima određenim nizom $b = \langle 1, 1, 1, 1, \ldots \rangle$.

Definicija 3.7.11 (Pravilo pomeranja) Neka je $G(a)$ generatorna funkcija niza $\{a_n\}_n$ i neka je niz $\{b_n\}_n$ oblika $b_0 = \ldots = b_{k-1} = 0$, a $b_{k+i} = a_i$, za $i \geq 0$. Tada je $G(b) = x^k G(a)$ generatorna funkcija niza $\{b_n\}_n$. ■

Primer 3.7.12 Prema primeru 3.7.2 zatvorena forma generatorne funkcije niza $a = \langle 1, 1, 1, \ldots \rangle$ je $\frac{1}{1-x}$. Ako posmatramo niz b dobijen pomeranjm niza a za k mesta u desno, preciznije $b = \langle \underbrace{0, \ldots 0}_{k}, 1, 1, \ldots \rangle$, za njemu odgovarajuću generatornu funkciju $x^k + x^{k+1} + \cdots$ izraz

$$\frac{x^k}{1-x}$$

predstavlja zatvorenu formu. ■

Definicija 3.7.13 (Pravilo diferenciranja) Neka je $G(a)$ generatorna funkcija niza $\{a_n\}_n$. Generatorna funkcija $G'(a)$ niza

$$\{(n+1)a_{n+1}\}_n = \langle a_1, 2a_2, 3a_3, \ldots\rangle$$

je *izvod* generatorne funkcije $G(a)$. ■

Primer 3.7.14 Neka je niz $a = \langle 1, 1, 1, \ldots\rangle$. Prema primeru 3.7.2 njemu odgovarajuća zatvorena forma generatorne funkcije $1 + x^2 + x^3 + \cdots$ je $\frac{1}{1-x}$, dok se pravilom diferenciranja izraz

$$\frac{d}{dx}\left(\frac{1}{1-x}\right) = \frac{1}{(1-x)^2}$$

dobija kao zatvorena forma generatorne funkcije $1 + 2\cdot x + 3\cdot x^2 + \cdots$ niza $b = \langle 1, 2, 3, \ldots\rangle$ ■

U sledećem primeru ćemo ilustrovati primenu navedenih operacija u pronalaženju zatvorene forme generatorne funkcije Fibonačijevog niza brojeva.

Primer 3.7.15 Fibonačijev niz $\{F_n\}_n$ je oblika $F_0 = 0$, $F_1 = 1$, $F_2 = F_1 + F_0 = 1$, $F_3 = F_2 + F_1 = 2$, $F_4 = F_3 + F_2 = 3$, $F_5 = F_4 + F_3 = 5$, tako da je generatorna funkcija niza:

$$\begin{aligned} G(F) &= G(\langle 0, 1, 1, 2, 3, 5, \ldots\rangle) = 0 + 1\cdot x + 1\cdot x^2 + 2\cdot x^3 + 3\cdot x^4 + 5\cdot x^5 + \ldots \\ &= F_0 + F_1\cdot x + F_2\cdot x^2 + \ldots \end{aligned}$$

Istovremeno, članovi Fibonačijevog niza se mogu zapisati i kao:

$$\langle 0, 1 + F_0, F_0 + F_1, F_1 + F_2, F_2 + F_3, F_3 + F_4, \ldots\rangle,$$

pa generatorna funkcija može biti data i na sledeći način:

$$G(F) = 0 + (F_0 + 1)x + (F_0 + F_1)x^2 + (F_1 + F_2)x^3 + (F_2 + F_3)x^4 + (F_3 + F_4)x^5 + \ldots$$

Sada ćemo ovaj poslednji zapis prikazati pomoću jednostavnijih generatornih funkcija za koje su već određene zatvorene forme:

$$\begin{array}{cllllll} & \langle & 0, & 1, & 0, & 0, & \ldots\rangle & x \\ & \langle & 0, & F_0, & F_1, & F_2, & \ldots\rangle & x\cdot G(F) \\ + & \langle & 0, & 0, & F_0, & F_1, & \ldots\rangle & x^2\cdot G(F) \\ \hline & \langle & 0, & 1 + F_0, & F_0 + F_1, & F_1 + F_2, & \ldots\rangle & x + x\cdot G(F) + x^2\cdot G(F) \end{array}$$

Sada se vidi da je

$$G(F) = x + x\cdot G(F) + x^2\cdot G(F),$$

što daje rešenje

$$G(F) = \frac{x}{1 - x - x^2}$$

za zatvorenu formu generatorne funkcije Fibonačijevog niza brojeva. ■

Prethodni primer se može generalizovati do sledećeg tvrđenja.

Teorema 3.7.16 Neka su dati niz $\{a_n\}_n$ i konstante $c_1, \ldots, c_k$. Tada je ekvivalentno:

1. Niz $\{a_n\}_n$ je rešenje linearne rekuretne relacije,

$$a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}.$$

2. Zatvorena forma generatorne funkcije niza $\{a_n\}_n$ je racionalna funkcija oblika

$$G(a) = \frac{g(x)}{1 - \sum_{i=1}^{k} c_i \cdot x^i}$$

gde je $g(x)$ polinom stepena najviše $k - 1$.

3. Ako važi

$$1 - \sum_{i=1}^{k} c_i \cdot x^i = (1 - r_1 x)(1 - r_2 x) \cdots (1 - r_k x),$$

gde su svi r_i međusobno različiti, onda je

$$\{d_1 r_1^n + d_2 r_2^n + \cdots + d_k r_k^n\}_n$$

opšte rešenje linearne rekurentne relacije

$$a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}.$$

Dokaz. $(1 \Rightarrow 2)$ Slično postupku u primeru 3.7.14 posmatraćemo generatornu funkciju:

$$G(a) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k + \ldots$$

i sumirati sledeće jednakosti:

$$\begin{array}{rcl}
c_1 x G(a) & = & c_1(a_0 x + a_1 x^2 + a_2 x^2 + \ldots + a_k x^{k+1} + \ldots) \\
c_2 x^2 G(a) & = & c_2(a_0 x^2 + a_1 x^3 + a_2 x^4 + \ldots + a_k x^{k+2} + \ldots) \\
\ldots & & \\
c_k x^k G(a) & = & c_k(a_0 x^k + a_1 x^{k+1} + a_2 x^{k+2} + \ldots + a_k x^{k+k} + \ldots)
\end{array}$$

Tada se za $(c_1x+\ldots+c_kx^k)G(a)$ dobija:

$$\begin{array}{ccccccc} c_1a_0x & + & c_1a_1x^2 & + & c_1a_2x^2+\ldots & + & c_1a_{k-1}x^k+\ldots+ \\ & & c_2a_0x^2 & + & c_2a_1x^3+\ldots & + & c_2a_{k-2}x^k+\ldots+ \\ \ldots & & & & & & \\ & & & & & & c_ka_0x^k+\ldots \end{array}$$

što je jednako

$$\begin{array}{ll} & c_1a_0x+(c_1a_1+c_2a_0)x^2+\ldots+(c_1a_{k-2}+\ldots+c_{k-1}a_0)x^{k-1} \\ + & (c_1a_{k-1}+c_2a_{k-2}+\ldots+c_ka_0)x^k \\ + & (c_1a_k+c_2a_{k-1}+\ldots+c_ka_1)x^{k+1}+\ldots \end{array}$$

Kako je

$$\begin{array}{rcl} a_k & = & c_1a_{k-1}+c_2a_{k-2}+\ldots+c_ka_0 \\ a_{k+1} & = & c_1a_k+c_2a_{k-1}+\ldots+c_ka_1 \\ \ldots & & \end{array}$$

dobijamo da je $G(a)-(c_1x+\ldots+c_kx^k)G(a)$ jednako

$$a_0+(a_1-c_1a_0)x+\ldots+(a_k-c_1a_{k-1}-\ldots-c_{k-1}a_1)x^{k-1},$$

pa je zatvorena forma za generatornu funkciju

$$G(a)=\frac{g(x)}{1-(c_1x+\ldots+c_kx^k)},$$

gde je $g(x)$ polinom stepena najviše $k-1$, kao što je i traženo.
$(2\Rightarrow 3)$ Posmatrajmo polinom

$$h(x)=1-(c_1x+\ldots+c_kx^k)$$

koji se nalazi u imeniocu zatvorene forme generatorne funkcije i karakteristični polinom

$$p(x)=x^k-c_1x^{k-1}-\ldots-c_k$$

rekurentne relacije $a_n=\sum_{i=1}^{k}c_i\cdot a_{n-i}$. Tada je:

$$h(x)=x^k(\frac{1}{x^k}-c_1\frac{1}{x^{k-1}}-\ldots-c_k)=x^kp(\frac{1}{x}).$$

Sada je r koren polinoma $h(x)$ ako i samo ako je njegova recipročna vrednost koren polinoma $p(x)$. Naime, s obzirom da 0 nije koren posmatranih polinoma, pretpostavimo da je $r\neq 0$ i $h(r)=0$. Pošto je $h(r)=r^kp(\frac{1}{r})$, onda je i $p(\frac{1}{r})=0$, a slično važi i obrnuto.

Neka je sada $h(x) = (1 - r_1x)(1 - r_2x)\cdots(1 - r_kx)$ gde su svi r_i međusobno različiti. Tada su $\frac{1}{r_1}, \ldots \frac{1}{r_k}$ međusobno različiti koreni polinoma $h(x)$, pa su $r_1, \ldots, r_k$ međusobno različiti koreni karakterističnog polinoma

$$p(x) = x^k - c_1x^{k-1} - \ldots - c_k$$

rekurentne relacije $a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}$ i po uopštenju teoreme 3.6.7

$$\{d_1r_1^n + d_2r_2^n + \cdots + d_kr_k^n\}_n$$

jeste opšte rešenje linearne rekurentne relacije $a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}$.
$(3 \Rightarrow 1)$ Pošto su r_i međusobno različiti koreni karakterističnog polinoma, onda je niz $\{a_n\}_n$ dat sa $a_n = d_1r_1^n + d_2r_2^n + \cdots + d_kr_k^n$ pod pretpostavljenim uslovima rešenje rekurentne relacije $a_n = \sum_{i=1}^{k} c_i \cdot a_{n-i}$. ■

Teorema 3.7.16 se može uopštiti i za slučaj kada polinom $h(x)$ u imeniocu zatvorene forme generatorne funkcije ima višestruke korene, tako da u opštem slučaju važi teorema:

Teorema 3.7.17 Niz zadovoljava linearnu rekuretnu relaciju ako i samo ako je zatvorena forma njegove generatorne funkcije racionalna.

Određivanje zatvorenih formi generatornih funkcija ima i praktičan značaj. Naime, polazeći od neke zatvorene forme često je relativno jednostavno odrediti koeficijente odgovarajućih stepenih redova (generatornih funkcija), odnosno članove njima odgovarajućih nizova. Upravo na toj ideji je zasnovana primena o kojoj će biti reči u odeljku 3.8.

3.8 Primene generatornih funkcija u prebrojavanju kombinacija

Interesantna primena generatornih funkcija se odnosi na prebrojavanje načina izbora elemenata nekog skupa, pri čemu koeficijent uz stepen x^k odgovara broju načina na koji je moguće birati k elemenata.

Kao motivaciju razmotrimo jednočlani skup $\{a_1\}$. Generatorna funkcija za izbor k elemenata iz tog skupa je $1 + x$ jer postoji po 1 način za izbor 0 i 1 elementa skupa, a 0 načina za izbor više od jednog elementa. U opštijem slučaju, posmatrajmo proizvod

$$\begin{array}{ll} & (1 + a_1x)(1 + a_2x)(1 + a_3x) = \\ 1 & +(a_1 + a_2 + a_3)x + (a_1a_2 + a_1a_3 + a_2a_3)x^2 + (a_1a_2a_3)x^3 \end{array}$$

u kome su:

- koeficijent uz x zapis svih mogućih 1-kombinacija,
- koeficijent uz x^2 zapis svih mogućih 2-kombinacija i
- koeficijent uz x^3 zapis svih mogućih 3-kombinacija

objekata a_1, a_2 i a_3. Brojevi 1, 2 i 3-kombinacija ovih objekata su redom $\binom{3}{1} = 3$, $\binom{3}{2} = 3$ i $\binom{3}{3} = 1$, i mogu se dobiti brojanjem sabiraka uz odgovarajući stepen x^i. Isto dobijamo i zamenom $a_1 = a_2 = a_3 = 1$ i računanjem odgovarajućih koefijenata. Tada se početni izraz svodi na funkciju

$$(1 + x)(1 + x)(1 + x)$$

čijim razvijanjem po stepenima dobijamo traženi broj 1, 2 i 3-kombinacija tri proizvoljna objekta. Prema tome, funkcija $(1 + x)^3$ je zatvorena forma generatorne funkcije koja broji kombinacije 3 objekta. U primeru 3.7.4 ovo zaključivanje je uopšteno na slučaj k objekata, gde je $(1 + x)^k$ je zatvorena forma generatorne funkcije koja broji kombinacije tih objekata.

Slično, generatorna funkcija kombinacija sa ponavljanjem elemenata jednočlanog skupa $\{a_1\}$ je $1+x+x^2+\ldots$ jer postoji samo jedan način za izbor k elemenata iz tog skupa. Kako je $\frac{1}{1-x} = 1+x+x^2+\ldots$, to je $\frac{1}{1-x}$ zatvorena forma generatorne funkcije kombinacija sa ponavljanjem elemenata jednočlanog skupa.

Dalje razmatrimo slučaj od n različitih objekata a_1, a_2 $\ldots$, a_n kojih ima po k komada i njihove kombinacije sa ponavljanjem. Slično malopređašnjem, posmatrajmo formulu:

$$\begin{array}{rl} & (1 + a_1x + a_1^2x^2 + \ldots + a_1^kx^k)\cdots(1 + a_nx + a_n^2x^2 + \ldots + a_n^kx^k) = \\ 1 & +(a_1 + a_2 + \ldots + a_n)x \\ & +(a_1^2 + a_2^2 + \ldots + a_n^2 + a_1a_2 + a_1a_3 + \ldots + a_1a_n + \ldots + a_{n-1}a_n)x^2 + \\ & \ldots \\ & +(a_1^k + a_2^k + \ldots + a_n^k + a_1^{k-1}a_2 + \ldots \\ & +a_1a_2\cdots a_k + \ldots + a_{n-k+1}\cdots a_{n-1}a_n)x^k + \\ & \ldots \\ & +(a_1^ka_2^k\cdots a_n^k)x^{nk} \end{array}$$

kojom nalazimo sve i-kombinacije sa ponavljanjem od n objekata, za $i = 1, 2, \ldots, nk$. Ponovo, zamenom $a_1 = a_2 = \ldots = a_n = 1$ i računanjem odgovarajućih koefijenata uz stepen oblika x^i mogu se odreditit odgovarajući brojevi kombinacija sa ponavljanjem.

I ovo se može uopštiti na slučaj kada različitih objekata ima različiti broj, ili kada je broj svakog od objekata neograničen. U tom poslednjem slučaju je

$$\frac{1}{(1-x)^n}$$

zatvorena forma generatorne funkcije kombinacija sa ponavljanjem n različitih objekata. Razvojem funkcije $f(x) = \dfrac{1}{(1-x)^n}$ u Tejlorov red $f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2 + \frac{f^{(3)}(0)}{3!}x^3 + \cdots$ i izračunavanjem izvoda funkcije f:

- $f'(x) = n(1-x)^{-(n+1)}$,
- $f''(x) = n(n+1)(1-x)^{-(n+2)}, \ldots,$
- $f^{(k)}(x) = n(n+1)\cdots(n+k-1)(1-x)^{-(n+k)}, \ldots,$

dobija se da je koeficijent uz x^k oblika

$$\frac{f^{(k)}(0)}{k!} = \binom{n+k-1}{k}$$

koji je upravo broj k-kombinacija sa ponavljanjem od n objekata (kojih imamo na raspolaganju neograničeni broj).

Primetimo i da je

$$\frac{1}{(1-x)^n} = \prod_{i=1}^{n} \frac{1}{1-x},$$

gde je $\frac{1}{1-x}$ zatvorena forma generatorne funkcije kombinacija sa ponavljanjem elemenata jednočlanog skupa.

Prethodno razmatranje se formalizuje kao još jedna operacija nad generatornim funkcijama.

Definicija 3.8.1 (Pravilo konvolucije) Neka su $G(a)$ i $G(b)$ generatorne funkcije nizova $\{a_n\}_n$ i $\{b_n\}_n$ koji predstavljaju broj kombinacija elemenata skupa A, odnosno B i neka je $A \cap B = \emptyset$. Tada je

$$G(c) = G(a \cdot b)$$

generatorna funkcija niza $\{c_n\}_n$ koji predstavlja broj kombinacija elemenata skupa $A \cup B$. ■

Pri ovome nije preciziran način formiranja kombinacija, recimo da li neki element može biti biran jednom, konačan ili neograničen broj puta. Jedina ograničenja su da se radi sa kombinacijama, odnosno da redosled izbora nije bitan, i da su A i B disjuktni skupovi.

Pravilo konvolucije obrazlažemo na sledeći način. Imajući u vidu da su skupovi A i B disjunktni, n elemenata iz skupa C možemo izabrati tako što biramo i elemenata iz A i $n-i$ elemenata iz B, gde je $i = 0, 1, \ldots, n$. Kako je a_i broj načina za izbor i elemenata iz skupa A i $n-i$ broj načina za izbor

$n-i$ elemenata iz skupa B, broj načina za izbor n elemenata iz skupa C, odnosno koeficijent uz x^n u $G(c)$ je:

$$c_n = a_0 b_n + a_1 b_{n-1} + \cdots + a_{n-1} b_1 + a_n b_0,$$

što se upravo i dobija kao izraz uz x^n u $G(a) \cdot G(b)$.

Primer 3.8.2 U prodavnici se jedna vrsta majica prodaje u tri boje: plavoj, sivoj i beloj. Kupac želi da kupi 3 komada, pri čemu sivih i belih može biti najviše jedan komad, dok za plave majice ograničenje ne postoji. Postavlja se pitanje na koliko načina se ova kupovina može ostvariti.

U duhu pravila konvolucije, posmatrajmo funkciju

$$\begin{aligned}(1+x+x^2+x^3)(1+x)(1+x) &= (1+x+x^2+x^3)(1+2x+x^2)\\ &= 1+3x+4x^2+4x^3+\ldots,\end{aligned}$$

u kojoj je prvi faktor generatorna funkcija niza koji predstavlja broj 0, 1, 2, odnosno 3-kombinacija elemenata skupa plavih majica, dok se drugi i treći faktor odnose na 0 i 1-kombinacija elemenata skupa sivih, odnosno belih, majica.

Lako se vidi da je traženi broj kombinacija sa ponavljanjem koeficijent uz x^3 i jednak je 4 i odgovara sledećim kobinacijama:

- plava, plava i plava majica,
- plava, plava i siva majica,
- plava, plava i bela majica i
- plava, siva i bela majica. ■

Primer 3.8.3 Odredimo na koliko se različitih načina može u vreću spakovati n komada odeće ako su data sledeća ograničenja:

- broj majica mora biti paran,
- broj košulja mora biti oblika $3k$,
- broj džempera je najviše 2 i
- broj šalova je najviše 1,

pri čemu vodimo samo računa o broju pojedinačnih vrsta odeće, a ne i o različitim izborima svake pojedinačne vrste odeće. Drugim rečima, razmatramo kombinacije sa ponavljanjem jednočlanih skupova.

Odgovarajući nizovi čiji koeficijenti prikazuju broj kombinacija su tada:

- za paran broj majica $M = \langle 1, 0, 1, 0, \ldots \rangle$,
- za broj košulja oblika $3k$, $K = \langle 1, 0, 0, 1, 0, 0, 1 \ldots \rangle$,
- za najviše dva džempera $D = \langle 1, 1, 1, 0, 0 \ldots \rangle$ i
- za najviše 1 šal $S = \langle 1, 1, 0, 0 \ldots \rangle$.

Ovim nizovima odgovaraju redom generatorne funkcije i njihove zatvorene forme:

- za majica $1 + x^2 + x^4 + \ldots = \dfrac{1}{1 - x^2} = G(M)$,
- za košulje $1 + x^3 + x^6 + \ldots = \dfrac{1}{1 - x^3} = G(K)$,
- za džempere $1 + x + x^2 = \dfrac{1 - x^3}{1 - x} = G(D)$ i
- za šalove $1 + x = G(S)$.

Prema pravilu konvolucije posmatramo proizvod zatvorenih formi generatornih funkcija:

$$\frac{1}{1 - x^2} \cdot \frac{1}{1 - x^3} \cdot \frac{1 - x^3}{1 - x} \cdot (1 + x) = \frac{1}{(1 - x)^2}$$

Pošto je funkcija $\dfrac{1}{(1 - x)^2}$ zatvorena forma stepenog reda $\sum_{i=0}^{\infty}(i+1)x^i$ koji je generatorna funkcija niza $\langle 1, 2, 3, 4, \ldots \rangle$, onda je koeficijent uz x^n uvek $n + 1$. Odatle, pod zadatim uslovima, broja načina pakovanja n komada odeće je uvek $n + 1$. ■

4

Algebarske strukture

Definicija 4.0.4 *Algebarska struktura* je uređena n-torka

$$\langle A, f_1, \ldots, f_k, c_1, \ldots, c_m \rangle$$

gde su:

- $n = 1 + k + m$,
- A neprazan skup, *domen*,
- $f_1, \ldots, f_k$ operacije domena A i
- $c_1, \ldots, c_m \in A$ konstante. ■

U ovom poglavlju ćemo razmotriti sledeće interesantne algebarske strukture:

- grupe
- prstene i polja i
- Bulove algebre

i neke njihove primene u oblastima računarstva poput korektivnih kodova, kriptologije, dizajna logičkih kola itd.

4.1 Grupe

Definicija 4.1.1 *Grupa*[1] je algebarska struktura $\langle A, \cdot, ^{-1}, 1 \rangle$ za koju važi:

[1] Termin *grupa* za grupe permutacija je prvi upotrebio Évariste Galois (1811 – 1832). On je postavio temelje apstraktne algebre. Bio je radikalni republikanac. Preminuo je od posledica ranjavanja u dvoboju do koga je došlo pod sumnjivim okolnostima u vreme vladavine kralja Louis-Philippe I.

- $\cdot$ je binarna, a $^{-1}$ unarna operacija,
- zakon asocijativnosti, $(x \cdot y) \cdot z = x \cdot (y \cdot z)$,
- 1 je neutral za operaciju $\cdot$, $x \cdot 1 = x$, $1 \cdot x = x$ i
- $x \cdot x^{-1} = 1$ i $x^{-1} \cdot x = 1$.

Grupa je *komutativna* ili *Abelova*[2] ako za nju važi:

- $x \cdot y = y \cdot x$.

Red grupe $G = \langle A, \cdot, ^{-1}, 1\rangle$, u oznaci $|G|$ je kardinalnost skupa A. ■

Sledi nekoliko primera algebarskih struktura koje (ni)su grupe.

Primer 4.1.2 Primer komutativne grupe je struktura $\langle \mathbb{Z}, +, -, 0\rangle$ u kojoj je sabiranje komutativno i asocijativno, 0 je neutral, a $-x$ je inverzni element elementa x. ■

Primer 4.1.3 Neka je A konačan neprazan skup simbola koji nazivamo azbuka. Reč je konačan niz simbola. Prazna reč, u oznaci ϵ, ne sadrži ni jedan simbol. A^* je skup svih reči azbuke A. Binarnom operacijom nadovezivanja (konkatenacije), u oznaci $*$, od reči $x, y \in A^*$ dobija se nova reč koja počinje as x, nakon čega sledi y. Preciznije:

$$x * y = xy.$$

Na primer, za azbuku $A = \{a, b, c\}$, neke reči su: ϵ, a, b, c, aa, aba, ac, ba, ccc, $abaccc$, ... Takođe važi: $a * b = ab$, $\epsilon * ab = ab$, $aba * ccc = abaccc$. Lako se vidi da je operacija $*$ asocijativna, kao i da je prazna reč neutral za nju, jer je $\epsilon * x = x * \epsilon = x$. Međutim, u opštem slučaju $*$ nije komutativna (sem za jednočlanu azbuku). Na primer $a * b \neq b * a$. Takođe, sem prazne reči ϵ za koju je $\epsilon * \epsilon = \epsilon$, reči iz A^* nemaju inverzne elemente, pa $\langle A^*, *, \epsilon\rangle$ nije grupa[3]. ■

Primer 4.1.4 Neka je $A = \{1, 2, 3\}$. U primeru 3.3.3 navedeno je svih 6 permutacija skupa A:

$$\begin{array}{lcl} p_1 = \langle 1,2,3\rangle & , & p_2 = \langle 1,3,2\rangle, \\ p_3 = \langle 2,1,3\rangle & , & p_4 = \langle 2,3,1\rangle, \\ p_5 = \langle 3,1,2\rangle & , & p_6 = \langle 3,2,1\rangle. \end{array}$$

[2] Niels Henrik Abel, 1802 – 1829, norveški matematičar. U njegovu čast je 2002. godine, kao pandan Nobelovoj nagradi, ustanovljena Abelova nagrada za vrhunska dostignuća u matematici.

[3] Struktura $\langle A^*, *, \epsilon\rangle$ je zapravo *monoid* generisan azbukom A.

Označimo skup ovih permutacija sa S_3. Permutacije možemo shvatiti i kao bijektivne funkcije koje redom elemente 1, 2 i 3 preslikavaju u elemente navedene u prethodnoj tabeli. Tako je, na primer, $p_6(1) = 3$, a $p_2(3) = 2$. Definišimo operaciju na skupu permutacija tako da je $p_i * p_j$ permutacija za koju važi:

$$p_i * p_j(x) = p_j(p_i(x)), \text{ za } x \in A.$$

Sada je $p_6 * p_2(1) = p_2(p_6(1)) = p_2(3) = 2$. Pošto su permutacije bijektivne funkcije, to su i njihove kompozicije. Tabelom[4]

$*$	p_1	p_2	p_3	p_4	p_5	p_6
p_1	p_1	p_2	p_3	p_4	p_5	p_6
p_2	p_2	p_1	p_4	p_3	p_6	p_5
p_3	p_3	p_5	p_1	p_6	p_2	p_4
p_4	p_4	p_6	p_2	p_5	p_1	p_3
p_5	p_5	p_3	p_6	p_1	p_4	p_2
p_6	p_6	p_4	p_5	p_2	p_3	p_1

je operacija $*$ u S_3 precizno opisana. Pemutacija p_1 je neutral operacije $*$: lako se proverava da je, na primer, $p_1 * p_2 = p_2 * p_1 = p_2$. Kako je kompozicija asocijativna operacija, a svaka od permutacija ima i inverz u odnosu na $*$ (recimo, $p_6 * p_6 = p_1$), struktura $\mathbb{S}_3 = \langle S_3, *, ^{-1}, p_1 \rangle$ je grupa koja nije komutativna. Uopšte, za proizvoljno $n \in \mathbb{N}$, $n > 0$, $\mathbb{S}_n$ je grupa. ■

Teorema 4.1.5 U grupi $\langle A, \cdot, ^{-1}, 1 \rangle$

1. jednačina $a \cdot x = b$ ima jedinstveno rešenje $x = a^{-1} \cdot b$ i

2. jednačina $y \cdot a = b$ ima jedinstveno rešenje $x = b \cdot a^{-1}$.

Dokaz. (1) Pošto je reč o grupi, za a postoji jedinstveni inverz a^{-1}. Za $x = a^{-1} \cdot b$ tada važi $a \cdot (a^{-1} \cdot b) = (a \cdot a^{-1}) \cdot b = b$, pa je $a^{-1} \cdot b$ jedno rešenje polazne jednačine.

Ako su x_1 i x_2 dva rešenja polazne jednačine, tada važi $a \cdot x_1 = b$ i $a \cdot x_2 = b$, pa je $a \cdot x_1 = a \cdot x_2$. Tada je i $a^{-1} \cdot a \cdot x_1 = a^{-1} \cdot a \cdot x_2$, odnosno $x_1 = x_2$, odakle sledi jedinstvenost rešenja. ■

Tvrđenja formulisana u teoremi 4.1.5 imaju zanimljivu posledicu na grupe sa konačnim brojem elemenata:

[4]Tabele ovog oblika se nazivaju *Kejlijevim tabelama*. Arthur Cayley (1821 – 1895) je jedan od pionira moderne matematike. Prvi je definisao koncept grupe kao algebarske strukture koja se sastoji od skupa i binarne operacije koja zadovoljava odgovarajuće zakone. Dao je veliku podršku uključivanju žena u univerzitetsko obrazovanje.

Posledica 4.1.6 U Kejlijevoj tabeli koja odgovara operaciji grupe u svakom redu, odnosno u svakoj koloni, svaki od elemenata grupe se pojavljuje tačno jednom. ■

Naime, neka je Kejlijeva tabela za neku grupu oblika:

$\cdot$	$\dots$	c	$\dots$
$\vdots$		$\vdots$	
a	$\dots$	b	$\dots$
$\vdots$		$\vdots$	

Element b se pojavljuje u redu u kome je a u onoj koloni u čijem zaglavlju se nalazi c ako i samo ako je c rešenje jednačine $a \cdot x = b$. Pošto je rešenje ovakve jednačine jedinstveno, b se može u redu u kome je a javiti samo jednom, i to u koloni koja odgovara jedinstvenom rešenju.

Posledica 4.1.6 se može upotrebiti prilikom utvrđivanja da li je neka algebarska struktura grupa: ako se u nekom redu (koloni) odgovarajuće Kejlijeve tabele bar jedan element javlja bar dva puta, struktura nije grupa. Primetimo i da obrat ne važi: postoje algebarske strukture koje ispunjavaju zahtev iskazan posledicom 4.1.6, a koje nisu grupe.

4.1.1 Korektivni kodovi

Prilikom prenosa podataka, koje shvatamo kao da su binarne reči - tj. kao konačne nizove bitova (0 ili 1), dolazi do grešaka zbog:

- nepouzdanosti samog kanala preko koga se prenos vrši i
- uticaja spoljnih izvora, takozvanog šuma.

U analizi ovakvih grešaka obično se pretpostavlja:

- greške prenosa su: prelazak 0 u 1, ili 1 u 0,
- obe konverzije su podjednako veovatne,
- pogreške na pojedinačnim bitovima su međusobno nezavisne i
- podjednako su verovatne greške na svim bitovima.

Poslednje dve pretpostavke, pored ostalog, znače da je verovatnije da se dogodi manje, nego više, grešaka, pa je najverovatniji broj grešaka 1.

Ovde su od interesa dva zadatka:

- otkrivanje da je došlo do greške, tj. *detekcija*, i

- ispravljanje otkrivene greške, tj. *korekcija*.

Do kraja ovog odeljka ćemo koristiti oznaku B^n za skup binarnih reči dužine n. Recimo $B^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$. Pod *težinom* binarne reči a, u oznaci $w(a)$, podrazumevaćemo broj bitova jednakih 1 koji se nalaze u a.

Sabiranje dvaju binarnih reči se realizuje primenom logičke operacije *ekskluzivno ili*, u oznaci $\veebar$, (tj. sabiranja po modulu 2 o kome se govori u primeru 4.2.6) definisane sa:

$\veebar$	0	1
0	0	1
1	1	0

Na primer:

$$\begin{array}{rr} & 1011001 \\ + & 1000111 \\ & 0011110 \end{array}$$

Primer 4.1.7 Neka se vrši prenos binarnih reči iz skupa B^3 i neka je poslata reč 010, a zbog greške primljena reč 011. Pošto je primljena reč u skupu reči koje se prenose, nije moguće detektovati grešku. ■

Primer 4.1.8 Neka se sada vrši prenos binarnih reči iz skupa $\{001, 010, 100, 111\}$ i neka je primljena reč 011. Ona ne pripada skupu i detekcija greška je laka.

Međutim, na osnovu pretpostavke da je najverovatnija greška na samo jednom bitu, postoje tri kandidata za reč koja je mogla biti poslata: 001, 010 i 111, tako da je korekciju nemoguće izvesti. ■

Primeri 4.1.7 i 4.1.8 ilustruju neophodnu osobinu koju komunikacioni sistem treba da poseduje: da bi se ostvarila detekcija greške, nekorektno preneta reč ne sme pripadati skupu reči koje se očekuju na prijemu. Ovo možemo interpretirati i na sledeći način:

- reči koje se šalju, odnosno, koje se očekuju na prijemu, moraju biti dovoljno *udaljene*, tj. različite, kako bi greška prilikom prenosa dovela do prijema reči koja ne pripada očekivanom skupu.

Primer 4.1.8, međutim ukazuje da je korekcija teži problem nego detekcija greške i da postoje situacije u kojima, iako je poznato da je došlo do greške prenosa, tu grešku nije moguće ispraviti. To potvrđuje i primer 4.1.9.

Primer 4.1.9 Neka se vrši prenos samo binarnih reči iz skupa $\{000, 111\}$ i neka je primljena reč 011. Ponovo, ona ne pripada skupu i detekcija greške je laka, ali sada, poštujući pretpostavku da je najverovatnija greška na samo jednom bitu, vršimo korekciju zaključujući da je poslata reč 111.

Međutim, ako je došlo do 2 greške prilikom prenosa reči 000, a primljena reč je 011, na osnovu pretpostavke o verovatnoći broja grešaka, korekcija će biti nekorektna. ■

Definicija 4.1.10 *Hamingovo rastojanje*[5] binarnih reči a i b dužine n, u oznaci $d(a, b)$ je broj bitova na kojima se a i b razlikuju. ■

Lako se vidi da je

$$d(a, b) = w(a + b)$$

gde je $+$ operacija sabiranja reči koja se realizuje primenom operacije ekskluzivnog ili na odgovarajućim bitovima reči a i b, a $w()$ je težina binarne reči.

Primer 4.1.11 Reči $x = 1011001$ i $y = 1000111$ se razllikuju na 4 mesta, pa je $d(x, y) = 4$. Analogno, pošto je

$$\begin{array}{rr} & 1011001 \\ + & 1000111 \\ & 0011110 \end{array}$$

i $w(0011110) = 4$, ponovo dobijamo $d(x, y) = 4$. ■

Razmotrićemo postupak (m, n)-*blok kodiranja* koji se primenjuje u realnim komunikacionim sistemima tako što se binarne reči kodiraju pre prenosa dodavanjem izvesnog broja bitova koji kasnije olakšavaju detekciju i korekciju grešaka. Preciznije, prenose se binarne reči čijih m bitova sadrže informacije, a preostalih r bitova se koriste prilikom detekcije i korekcije grešaka. Kodiranje se, prema tome vrši funkcijom:

- $E : B^m \mapsto B^n$

Kodne reči su elementi slike funkcije E, $\mathrm{Im}(E)$. Jasno je da E mora biti injektivna, tj. različite reči moraju imati različite kodove, kako bi bilo moguće sprovesti obrnuti proces, dekodiranje:

- $D : B^n \mapsto B^m \cup \{e\}$

[5] Richard Wesley Hamming (1915 – 1998), američki matematičar, koji je postavio osnove u oblasto detekcije i korekcije grešaka prenosa podataka.

gde e označava grešku[6].

Ako je primljena reč $y \in B^n \backslash \mathrm{Im}(E)$, detektuje se greška, a ako je moguća korekcija, onda je $D(y) = D(x)$, gde je x kodna reč koja je najbliža reči y. Ako ne postoji jedinstvena najbliža reč, javlja se greška, odnosno $D(y) = e$.

Primer 4.1.12 Kodirajuća funkcija za proveru parnosti (odnosno neparnosti)[7] je oblika $E : B^m \mapsto B^{m+1}$, za koju je $m+1$. bit slike takav da $E(x)$ ima paran (odnosno neparan) broj bitova 1. ■

(m, n)-blok kodiranje je *sistematsko* ako za svako $x \in B^m$, prvih m bitova u $E(x)$ čini upravo reč x. Ako je pri (m, n)-blok kodiranju moguće bilo koju kombinaciju od k ili manje grešaka detektovati (odnosno izvršiti korekciju), kodiranje je *k-detektibilno* (odnosno *k-korektibilno*). *Minimalna distanca* za neko kodiranje je $\min\{d(x, y) : x, y \in \mathrm{Im}(E)\}$.

Teorema 4.1.13 karakteriše mogućnost detekcije i korekcije greške u nekom kodiranju:

Teorema 4.1.13 Kodiranje je:

- k-detektibilno ako i samo ako je minimalna distanca kodiranje bar $k+1$ i
- k-korektibilno ako i samo ako je minimalna distanca kodiranje bar $2k+1$. ■

Primer 4.1.14 Pošto kodiranje $E : B^2 \mapsto B^6$, definisano sa:

- $E(00) = 001000$,
- $E(01) = 010100$,
- $E(10) = 100010$ i
- $E(11) = 110001$,

kao minimalnu distancu ima 3, to je ono 2-detektibilno i 1-korektibilno. ■

Funkcija kodiranja $E : B^m \mapsto B^n$ pri kojoj se na reč koja se kodira nadovezuju bitovi za proveru, pogodno se prikazuje *generatornom matricom* G koja sadrži samo 0 i 1. Kodiranje se obavlja matričnim množenjem u kome se operacije sabiranja i množenja vrše u binarnom brojnom sistemu.

[6]Engleski: error.
[7]Engleski: parity check code, odd parity check code.

Primer 4.1.15 Neka je kodiranje $E : B^3 \mapsto B^6$ definisano sa $E(x) = x \times G$, gde je generatorna matrica:

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}_{3\times 6}$$

Recimo, reč oblika $x_1x_2x_3$ se kodira na sledeći način:

$$\begin{aligned} E(x_1x_2x_3) &= \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} x_1 & x_2 & x_3 & x_1 + x_2 & x_2 + x_3 & x_1 + x_3 \end{bmatrix} \end{aligned}$$

Tako da je, recimo, $E(011) = 011101$.

E je sistematsko kodiranje, tj. prva tri bita kodne reči su zapravo reč koja se kodira, jer je levi blok generatorne matrice G jedinična matrica $I_{3\times 3}$. Preostala 3 bita služe za proveru parnosti parova bita originalne reči.

U ovom kodiranju važi da se svaka greška pri prenosu jednog bita jednoznačno određuje vrednostima kodne reči. Greška u prenosu na jednom od prva tri bita vidi se na dva od tri poslednja bita: $x_1 + x_2$, $x_2 + x_3$ i $x_1 + x_3$. Tako se greška na bitu 2 uočava na bitovima 4 i 5: $x_1 + x_2$, $x_2 + x_3$. Sa druge strane, ako se greška pojavi na bitu 4, $x_1 + x_2$, ona će biti vidljiva samo tu. Recimo, ako je prenosom 011 dobijena reč $w_1w_2w_3w_4w_5w_6 = 001101$, lako se vidi da je sa jedne strane $w_1 + w_2 = 0 \neq w_4$ i $w_2 + w_3 = 1 \neq w_5$, pa pošto se neslaganje javlja na bitovima 4 i 5 zaključujemo da je bit w_2 loše prenet. Sa druge strane, ako je prenosom 011 dobijena reč $w_1w_2w_3w_4w_5w_6 = 011001$, pošto je $w_1 + w_2 = 1 \neq w_4$, $w_2 + w_3 = 0 = w_5$ i $w_1 + w_3 = 1 = w_6$, zaključujemo da je bit w_4 loše prenet.

Generatorna matrica G je oblika $[IF]$, gde je I zapravo jedinična matrica $I_{3\times 3}$. Za matricu $H = \left[F^T I\right]_{3\times 6}$ i svaku kodnu reč $w \in B^6$ važi da je

$$H \times w^T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

gde su F^T i w^T transponovane matrice. ■

Matrice tipa matrice H iz primera 4.1.15 se nazivaju *matrice provere parnosti*. Poslednji deo ovog primera se uopštava tvrđenjem 4.1.16.

Teorema 4.1.16 Neka je generatorna matrica G oblika $[I_{m\times m}F_{m\times r}]_{m\times n}$, gde je $n = m + r$, i neka je matrica provere parnosti $H = \left[F^T_{m\times r} I_{r\times r}\right]_{r\times n}$.

Ako je funkcija kodiranja $E : B^m \mapsto B^n$ definisana sa $E(x) = x \times G$, onda za svaku kodnu reč $w \in B^n$ važi:

$$H \times w^T = 0_{r\times 1}$$

gde je $0_{r\times 1}$ nula matrica. ■

Druga važna karakteristika funkcija kodiranja definisanih generatornim matricama data je tvrđenjem 4.1.17.

Teorema 4.1.17 Ako je funkcija kodiranja $E : B^m \mapsto B^n$ definisana sa $E(x) = x\times G$, gde je G generatorna matrica, onda skup kodnih reči $\mathrm{Im}(E) \subset B^n$ čini grupu u odnosu na sabiranje reči koje se realizuje primenom logičke operacije ekskluzivno ili na bitovima reči. ■

Kodiranje za koje $\mathrm{Im}(E)$ u odnosu na operaciju sabiranja binarnih reči čini grupu, naziva se *grupovno kodiranje*. Ove vrste kodiranje imaju odgovarajuće prednosti. Na primer, dok se za pronalaženje minimalne distance nekog kodiranja moraju uporediti svi parovi reči, za grupovna kodiranja je to mnogo lakše, kao što se tvrdi u teoremi 4.1.18.

Teorema 4.1.18 Minimalna distanca grupovnog kodiranja je minimalna težina kodnih reči kod kojih svi bitovi nisu 0. ■

Koristeći ovaj rezultat odgovora se na pitanje koliko grešaka je moguće detektovati i korigovati za kodiranje E koje se vrši generatornom matricom G, odnosno njoj odgovarajućom matricom provere parnosti H. Neka je $H_{r\times n}$ matrica provere parnosti u kojoj su kolone označene sa $h_1, \ldots, h_n$ i neka su kolone $h_{i_1}, \ldots, h_{i_k}$ takve da su zbirovi njihovih elemenata po odgovarajućim redovima jednaki 0.

Primer 4.1.19 Za matricu H kažemo da su zbirovi elemenata po odgovarajućim redovima u kolonama h_1, h_3 i h_4 jednaki 0 ako:

$$H = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} & h_{1,5} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} & h_{2,5} \\ h_{3,1} & h_{3,2} & h_{3,3} & h_{3,4} & h_{3,5} \end{bmatrix} \qquad \begin{array}{ccccccc} h_{1,1} & \veebar & h_{1,3} & \veebar & h_{1,4} & = & 0 \\ h_{2,1} & \veebar & h_{2,3} & \veebar & h_{2,4} & = & 0 \\ h_{3,1} & \veebar & h_{3,3} & \veebar & h_{3,4} & = & 0 \end{array}$$

pri čemu je sabiranje realizovano u binarnom sistemu, odnosno kao operacija ekskluzivnog ili. ■

Ako su u binarnoj reči $w = w_1 w_2 \ldots w_n$ jedinice na pozicijama $i_1, \ldots, i_k$, a na svim ostalim 0, onda je $H \times w^T = 0$, pa je $w \in \mathrm{Im}(E)$, odnosno w je (ispravno preneta) kodna reč. Važi o obrnuto: ako je w kodna reč koja jedinice ima samo na pozicijama $i_1, \ldots, i_k$, onda su odgovarajući zbirovi kolona $h_{i_1}, \ldots, h_{i_k}$ jednaki 0.

Primer 4.1.20 Ako je matrica H kao u primeru 4.1.19, onda za reč $w = 10110$ važi

$$\begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} & h_{1,5} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} & h_{2,5} \\ h_{3,1} & h_{3,2} & h_{3,3} & h_{3,4} & h_{3,5} \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} h_{1,1} \veebar h_{1,3} \veebar h_{1,4} \\ h_{2,1} \veebar h_{2,3} \veebar h_{2,4} \\ h_{3,1} \veebar h_{3,3} \veebar h_{3,4} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

odnosno $H \times w^T = 0$, pa je w kodna reč. ■

Prema tome, u slučaju grupovnog kodiranja E definisanog generatornom matricom G, odnosno njom odgovarajućom matricom provere parnosti H, minimalna distanca jednaka je minimalnom broju kolona kod kojih su opisane sume elemenata po redovima jednake nuli. U praksi pronalaženje tog minimalnog broja kolona se realizuje na sledeći način:

- proverava se da li postoji nula-kolona, u kom slučaju je minimalna distanca jednaka 1, ako ne
- proverava se da li postoje dve jednake kolone, u kom slučaju je minimalna distanca jednaka 2, ako one
- proverava se da li postoje tri kolone (koje ne moraju biti jedinstvene), kod kojih su traženi zbirovi jednaki 0, u kom slučaju je minimalna distanca jednaka 3, ...

Za ovakva kodiranja, prilikom detekcije grešaka proverava se proizvod $H \times w^T$. Ako on daje 0-matricu, razumno je pretpostaviti da je w korektno preneta, a dekodiranje se sprovodi izdvajanjem početnih bitova reči w. Ako je $H \times w^T \neq 0$, došlo je do greške prenosa. Podrazumevajući da se pojavila samo jedna greška (što je, po pretpostavci sa početka odeljka, najverovatnije) pozicija greške se određuje na sledeći način:

- neka je w_p kodna reč čijim prenosom je dobijena reč w (w_p i w se razlikuju samo na jednom bitu, recimo na poziciji i)
- tada je $w_p = w + e_i$, gde je e_i binarna reč koja se sastoji od bitova 0 i samo jednog bita 1, na poziciji i,
- $H \times w^T = H \times (w_p + e_i)^T = H \times (w_p^T + e_i^T) = (H \times w_p^T) + (H \times e_i^T) = 0 + H \times e_i^T = H \times e_i^T$
- $H \times e_i^T = h_i$, gde je h_i i-ta kolona matrice H, a i pozicija bita na kome se pojavila greška.

Prema tome upoređivanjem rezultata $H \times w^T$ sa kolonama matrice H detektuje se pozicija greške koja se potom lako koriguje promenom vrednosti odgovarajućeg bita u reči w. Ako $H \times w^T \neq 0$, ali $H \times w^T$ nije jednako ni jednoj od kolona matrice H, zaključujemo da se pojavilo više od jedne greške (što se opisanim postupcima ne može pouzdano korigovati i za šta postoje druge složenije metode).

Primer 4.1.21 Neka je funkcija kodiranje $E : B^3 \mapsto B^6$ definisana generatornom matricom G, odnosno matricom provere parnosti H, kao u primeru 4.1.15 i neka je nakon prenosa dobijena reč $w = 100100$. Tada je:

$$H \times w^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Dobijena matrica nije 0-matrica, pa w nije kodna reč. Pošto se $H \times w^T$ poklapa sa kolonom h_6 matrice H, zaključujemo da je greška nastala u prenosu poslednjeg bita reči i da je on originalno bio 1, pa je poslata reč bila $w_p = 100101$. ■

4.2 Prsteni i polja i kongruencija po modulu

Definicija 4.2.1 *Prsten* je algebarska struktura $\langle A, +, \cdot, -, 0, 1\rangle$ za koju važi:

- $+$ i $\cdot$ su binarne operacije, a $-$ je unarna operacija,
- $\langle A, +, -, 0\rangle$ je komutativna grupa,
- zakon asocijativnosti, $(x \cdot y) \cdot z = x \cdot (y \cdot z)$,
- zakon distributivnosti, $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ i $(x + y) \cdot z = (x \cdot z) + (y \cdot z)$.

Element a prstena koji je različit od 0 je *levi (desni) delilac nule* ako postoji ne-nulti element prstena b tako da je $a \cdot b = 0$ $(b \cdot a = 0)$.

Prsten je *komutativan sa jedinicom* ako važi:

- $x \cdot y = y \cdot x$ i
- $x \cdot 1 = x$.

Komutativni prsten sa jedinicom je *polje* ako važi:

- za svaki $x \in A$, ako je $x \neq 0$, onda postoji njegov inverz $x^{-1} \in A$, tako da je $x \cdot x^{-1} = 1$. ■

4.2.1 Kongruencija po modulu

Neka je dat $n \in \mathbb{N}$, $n > 0$. Posmatraćemo relaciju *kongruencije po modulu* n u oznaci $x \equiv_n y$ ili $x \equiv y \pmod{n}$, na skupu celih brojeva $\mathbb{Z}$ definisanu sa

$$x \equiv_n y \text{ ako i samo ako } x - y = k \cdot n, \text{ za neki } k \in \mathbb{Z}.$$

Relacija kongruencije se često javlja u računarstvu, ali primer 4.2.2 ilustruje i jednu svakodnevnu situaciju.

Primer 4.2.2 Pretpostavimo da je trenutno ponoć. Koliko sati će biti za 50 sati? Pošto se vreme broji u 24-voro satnim ciklusima, posmatramo relaciju $\equiv_{24}$ i pošto je $50 \equiv_{24} 2$, odgovor je: biće 2 sata ujutro. ■

Za relaciju $\equiv_n$ je u primeru 2.1.18 pokazano da je relacija ekvivalencije. Tvrđenje 4.2.3 daje alternativnu karakterizaciju ove relacije.

Teorema 4.2.3 Za $x, y \in \mathbb{Z}$ je $x \equiv_n y$ ako i samo ako x i y imaju iste ostatke pri deljenju sa n.

Dokaz. ($\Leftarrow$) Ako važi $x = k_x n + r$ i $y = k_y n + r$, onda je $x - y = (k_x - k_y)n$, pa je $x \equiv_n y$.
($\Rightarrow$) Neka je $x \equiv_n y$, $x = k_x n + r_x$ i $y = k_y n + r_y$. Tada je $x - y$ deljivo sa n, pa je i

$$r_x - r_y = (x - y) + (k_y - k_x)n$$

deljivo sa n. Pošto su r_x, $r_y \in \{0, \ldots n - 1\}$, sledi da je $r_x - r_y = 0$, tj. $r_x = r_y$. ■

Primer 4.2.4 Pošto je $17 - 5 = 12$ i 6 deli 12, važi $17 \equiv_6 5$. Isto zaključujemo primenom teoreme 4.2.3, pošto je $17 = 2 \cdot 6 + 5$ i $5 = 0 \cdot 6 + 5$. ■

U primeru 2.1.21 za $n = 3$ klase ekvivalencije relacije $\equiv_3$ su zapravo poistovećene za celim brojevima 0, 1 i 2, za šta će opravdanje dati tvrđenje 4.2.5.

Teorema 4.2.5 Za $x, x', y, y' \in \mathbb{Z}$, ako je $x \equiv_n x'$ i $y \equiv_n y'$ onda važi:

1. $x + y \equiv_n x' + y'$,

2. $x - y \equiv_n x' - y'$,

3. $x \cdot y \equiv_n x' \cdot y'$ i

4. $x^k \equiv_n x'^k$ za $k \in \mathbb{N}$.

Dokaz. (1),(2) Pošto n deli i $x-x'$ i $y-y'$ tada je desna strana jednakosti:

$$(x + y) - (x' + y') = (x - x') + (y - y')$$

zapravo zbir dva broja deljiva sa n, pa je $x+y \equiv_n x'+y'$ i dobija se tvrđenje (1). Takođe je

$$(x - y) - (x' - y') = (x - x') + (y' - y)$$

i uz isto obrazloženje sledi tvrđenje (2).
(3), (4). Slično,

$$(x \cdot y) - (x' \cdot y') = (x \cdot y) - (x \cdot y') + (x \cdot y') - (x' \cdot y') = x(y - y') + y'(x - x')$$

i poslednji zbir čine dva sabirka deljiva sa n, pa je to slučaj i sa $(x \cdot y) - (x' \cdot y')$, odnosno važi $x \cdot y \equiv_n x' \cdot y'$. Za $x = y$ i $x' = y'$ tvrđenje (4) direktno sledi iz (3). ■

Na osnovu teoreme 4.2.5, ako u nekom aritmetičkom izrazu koji uključuje cele brojeve, sabiranje, oduzimanje i množenje (ali ne i deljenje!) zamenimo brojeve koji su međusobno kongruenti, rezultati polaznog i izraza dobijenog zamenom možda neće biti jednaki, ali će biti kongruentni. Na ovaj način se može efikasnije računati, kao što ilustruje primer 4.2.6.

Primer 4.2.6 Neka je:

- $n = 113 \cdot (167 + 484) + 192 \cdot 145$

i neka treba pronaći kojoj klasi ekvivalencije relacije $\equiv_{21}$ pripada n. Direktan način je izračunati $n = 113 \cdot (167 + 484) + 192 \cdot 145 = 101403$, i nakon delenja sa 21 dobiti ostatak 15, tako da je $n \equiv_{21} 15$. Međutim, problem će efikasnije biti rešen ako se uoči da je:

- $113 \equiv_{21} 8$,
- $167 \equiv_{21} 20$,
- $484 \equiv_{21} 1$,
- $192 \equiv_{21} 3$ i

- $145 \equiv_{21} 19$,

pa se odgovarajućom zamenom dobija izraz $8(20 + 1) + 3 \cdot 19$. Pošto je $8(20+1) \equiv_{21} 0$, izraz se redukuje na $3 \cdot 19 = 57$, pa kako se deljenjem 57 sa 21 dobija se ostatak 15, to je, prema očekivanju, $n \equiv_{21} 15$.

Neka treba pronaći kojoj klasi ekvivalencije relacije $\equiv_{10}$ pripada 9^{342}, odnosno koliki je ostatak prilikom deljenja 9^{342} sa 10. Umesto izračunavanja velikog broja 9^{342} i onda deljenja, pogodno je uočiti da je $9^2 \equiv_{10} 1$, pa je $9^{342} = (9^2)^{171} \equiv_{10} 1^{171} \equiv_{10} 1$.

Slično, na sledeći način je moguće izračunati čemu je kongruentno 5^8 u odnosu na relaciju $\equiv_{16}$:

- $5 \equiv_{16} 5$,
- $5^2 \equiv_{16} 25 \equiv_{16} 9$,
- $5^4 \equiv_{16} 5^2 \cdot 5^2 \equiv_{16} 9 \cdot 9 \equiv_{16} 81 \equiv_{16} 1$ i
- $5^8 \equiv_{16} 5^4 \cdot 5^4 \equiv_{16} 1 \cdot 1 \equiv_{16} 1$. ■

Prikazana tehnika se ne može primenjivati i na deljenje, što je ilustrovano u primeru 4.2.7.

Primer 4.2.7 Iako je $484 \equiv_{21} 1$ i $64 \equiv_{21} 1$, $484/64$ nije ceo broj i ne važi da je $484/64 \equiv_{21} 1/1$. ■

4.2.2 Modularna aritmetika

Istovremeno, teorema 4.2.5 sugeriše način na koji će biti pogodno definisati aritmetičke operacije na količničkom skupu $\mathbb{Z}_{/\equiv_n}$:

- $[i] + [j] = [i + j]$,
- $[i] - [j] = [i - j]$ i
- $[i] \cdot [j] = [i \cdot j]$.

Dakle, za klase $[i]$ i $[j]$ zbir je klasa čiji predstavnik je ostatak pri deljenju $i + j$ sa n. Teorema 4.2.5 garantuje da zbir klasa ekvivalencije $([i] + [j])$ ne zavisi od njihovih predstavnika, a slično je i u preostala dva slučaja, tako da su operacije korektno definisane.

Primer 4.2.8 Neka je $n = 4$, tada je $\mathbb{Z}_{/\equiv_n} = \{[0], [1], [2], [3]\}$. Prema prethodnom je $[3] + [2] = [5]$ i kako je $5 \equiv_4 1$, to je $[3] + [2] = [1]$. Slično, pošto je $[3] \cdot [2] = [6]$ i $6 \equiv_4 2$, onda je $[3] \cdot [2] = [2]$. ■

Imajući u vidu sve prethodno rečeno, na dalje ćemo skup $\mathbb{Z}_{/\equiv_n}$ poistovetiti sa skupom $\mathbb{Z}_n = \{0, 1, \dots, n-1\}$. Nije teško proveriti, na osnovu svojstava operacija $+$, $-$ i $\cdot$ na skupu $\mathbb{Z}$, da je algebarska struktura $\langle \mathbb{Z}_n, +, \cdot, -, 0, 1 \rangle$ komutativni prsten sa jedinicom.

Ako n nije prost broj, jasno je da uvek postoje prirodni brojevi x i y takvi da važi:

- $1 < x < n$, $1 < y < n$ i
- $x \cdot y = n$, odnosno $x \cdot y \equiv_n 0$.

Dakle, ako n nije prost broj, $\mathbb{Z}_n$ ima delioce nule. Tada $\langle \mathbb{Z}_n, +, \cdot, -, 0, 1 \rangle$ neće biti polje, jer pretpostavka da, na primer, uočeni element x ima inverz x^{-1} dovodi do sledećeg:

- $x \cdot y \equiv_n 0$,
- $x^{-1} \cdot (x \cdot y) \equiv_n x^{-1} \cdot 0$,
- $(x^{-1} \cdot x) \cdot y \equiv_n 0$, zbog asocijativnosti, pa je suprotno pretpostavci
- $1 \cdot y \equiv_n 0$, odnosno $y \equiv_n 0$.

Međutim, ako je broj, odnosno modul, p u odnosu na koga se definiše kongruencija prost broj, prethodno razmatranje ne prolazi i $\langle \mathbb{Z}_p, +, \cdot, -, 0, 1 \rangle$ jeste polje. U dokazivanju ovog tvrđenja koristićemo takozvanu Malu Fermaovu[8] teoremu 4.2.9.

Teorema 4.2.9 (Mala Fermaova teorema) Ako su p prost broj i a prirodan broj, onda p deli $a^p - a$, odnosno

$$a^p - a = k \cdot p,$$

za neki ceo broj k.

Dokaz. Najpre ćemo uočiti da važi sledeće: ako je p prost broj i l prirodan broj, za koga je $0 < l < p$, onda je $\binom{p}{l}$ deljivo sa p. Naime,

- $\binom{p}{l} = \dfrac{p!}{l!(p-l)!}$, pa je

[8] Pierre de Fermat, 1601 (ili 1607/8) – 1665), francuski advokat i pasionirani matematičar. Glavni rezultati koje je postigao odnose se na početni razvoj infinitezimalnog računa, teoriju brojeva, verovatnoću itd. Najpoznatiji je po Fermaovoj poslednjoj (ili velikoj) teoremi koju je zabeležio na marginama Diofantove aritmetike i u kojoj se tvrdi da ne postoje ne-nulti prirodni brojevi a, b i c koji za $n > 2$ zadovoljavaju jednačinu $a^n + b^n = c^n$. Dokaz teoreme je tek 1995. godine dao Andrew Wiles.

- $p! = \binom{p}{l} \cdot l! \cdot (p-l)!$.

Leva strana poslednje jednakosti je deljiva sa p, a na desnoj strani to nisu ni $l!$ ni $(p-l)!$, jer je p prost broj, a l i $p-l$ su manji od p. Odatle je $\binom{p}{l}$ deljivo sa p.

Dokaz dalje ide indukcijom po a. Za $a = 0$, tvrđenje trivijalno važi, pa pretpostavimo da je $a > 0$ i $a = b+1$, kao i da za b važi indukcijska pretpostavka da p deli $b^{p-1} - 1$. Tada je:

$$\begin{aligned} a^p - a &= (b+1)^p - (b+1) \\ &= b^p + \left(\sum_{l=1}^{p-1} \binom{p}{l} b^{p-l} \right) + 1 - b - 1 \\ &= (b^p - b) + \sum_{l=1}^{p-1} \binom{p}{l} b^{p-l}. \end{aligned}$$

Pošto na osnovu indukcijske pretpostavke p deli $b^p - b = b(b^{p-1} - 1)$, a na osnovu prethodnog važi i da p deli $\sum_{l=1}^{p-1} \binom{p}{l} b^{p-l}$, zaključujemo da p deli $a^p - a$. ■

Primetimo da se teorema 4.2.9 može formulisati i na sledeći način (za $0 < a < p$):

$$a^{p-1} - 1 = k \cdot p,$$

tj.

$$a^{p-1} \equiv_p 1$$

odnosno

$$a \cdot a^{p-2} \equiv_p 1.$$

Prema tome, za prost broj p i proizvoljan prirodan broj a, takav da je $0 < a < p$, $a^{-1} \equiv_p a^{p-2}$ je inverz od a u $\langle \mathbb{Z}_p, +, \cdot, -, 0, 1 \rangle$, pa na osnovu definicije 4.2.1 ova struktura jeste polje.

Primer 4.2.10 Inverzni elementi za 2, 3 i 4 u $\mathbb{Z}_5$ su redom:

- pošto je $2^{5-2} = 2^3 = 8$ i $8 \equiv_5 3$, to je $2^{-1} \equiv_5 3$,
- pošto je $3^{5-2} = 3^3 = 27$ i $27 \equiv_5 2$, to je $3^{-1} \equiv_5 2$ i
- pošto je $4^{5-2} = 4^3 = 64$ i $64 \equiv_5 4$, to je $4^{-1} \equiv_5 4$. ■

Primer 4.2.11 Sledećim tabelama

$+_2$	0	1
0	0	1
1	1	0

$\cdot_2$	0	1
0	0	0
1	0	1

definisane su operacije $+_2$ i $\cdot_2$ u polju $\langle \mathbb{Z}_2, +_2, \cdot_2, -_2, 0, 1\rangle$.

Za oduzimanje u ovom polju se lako vidi da je $x +_2 y = x -_2 y$. Ako je $y = 0$, ovo važi trivijalno. Ako je $y = 1$, za $x = 1$ je očigledno $1 +_2 1 = 0 = 1 -_2 1$. Ako je $x = 0$, onda je $1 +_2 0 = 1$, dok je $0 - 1 = -1$ (kao operacija nad celim brojevima), a pošto je $-1 \in [1]_{\equiv_2}$, to je i $0 -_2 1 = 1$.

U ovom slučaju je lako odrediti i rezultat deljenja: deljenje sa 0 nije definisano, dok deljenje sa 1 ne menja deljenik.

Za označavanje sabiranja, oduzimanje i množenja u $\mathbb{Z}_2$ koriste se i oznake $\oplus$, $\ominus$ i $\odot$. ■

Primetimo da smo, da bismo naglasili da se radi sa elementima skupa $\mathbb{Z}_2$ namerno koristili indeks $_2$ u označavanju operacija, ali u nastavku ćemo to izbegavati i kratko pisati $+$, $\cdot$ i $-$, ako to ne izaziva zabunu. Simboli $+$ će biti korišten i u slučaju sabiranja binarnih reči.

Posledica tvrđenja 4.2.9 je da možemo odrediti inverze ne-nultih elemenata polja $\mathbb{Z}_p$. Međutim, u opštem slučaju i za razliku od primera 4.2.11, ovaj postupak, kao i izračunavanje rezultata deljenja, nisu trivijalni. Najpre, izračunavanje $p-2$ stepena broja može biti zahtevno. Takođe, iako je jasno da za $y \neq 0$ važi $\frac{x}{y} = z$ ako i samo ako je $z \cdot y = x$ i da pretragom kroz skup $\mathbb{Z}_p$ uvek možemo odrediti z, u slučajevima kada je p veliko ovakav postupak nije efikasan. Pokazuje se da se problem deljenja svodi na problem nalaženja inverza:

- neka se traži vrednost $c \equiv_p \frac{a}{b}$,
- ako možemo naći d takav da je $b \cdot d \equiv_p 1$, tj. $d = b^{-1}$, tada je $\frac{1}{b} \equiv_p d$ i
- $c \equiv_p \frac{a}{b} \equiv_p a \cdot \frac{1}{b} \equiv_p a \cdot d$.

Na ovom mestu je pogodno koristiti Euklidov algoritam za izračunavanje najvećeg zajedničkog delioca[9].

Definicija 4.2.12 *Najveći zajednički delilac* prirodnih brojeva x i y (koji nisu oba jednaka 0), u oznaci $\gcd(x, y)$ je najveći prirodni broj koji deli i x i y. Dva prirodna broja x i y su *uzajamno prosti* ako je $\gcd(x, y) = 1$. ■

Primer 4.2.13 Lako se proverava da važi:

- $\gcd(1, 6) = 1$,
- $\gcd(2, 6) = 2$,
- $\gcd(4, 6) = 2$,

[9] The greatest common divisor, gcd.

- $\gcd(5, 6) = 1$,
- $\gcd(6, 6) = 6$. ■

Dobro je poznato sledeće tvrđenje o faktorizaciji prirodnih brojeva:

Teorema 4.2.14 Svaki prirodan broj veći od 1 se na jedinstven način može prikazati kao proizvod nekih stepenova nekih prostih brojeva. ■

i na osnovu njega se u principu uvek može izračunati $\gcd(x, y)$:

- $x = p_1^{a_1} \cdots p_k^{a_k}$ (gde neki od a_i mogu biti jednaki 0) i
- $y = p_1^{b_1} \cdots p_k^{b_k}$ (gde neki od b_i mogu biti jednaki 0), pa je
- $\gcd(x, y) = p_1^{\min(a_1,b_1)} \cdots p_k^{\min(a_k,b_k)}$.

Primer 4.2.15 Kako je:

- $24 = 2^3 \cdot 3$ i
- $36 = 2^2 \cdot 3^2$, to je
- $\gcd(24, 36) = 2^2 \cdot 3 = 12$. ■

Mana ovog postupka je potreba da se pronađu svi faktori brojeva čiji se najveći zajednički delilac traži, pa se u realnosti, kao efikasniji, obično koristi takozvani Euklidov algoritam[10] koji ne zahteva faktorizaciju. Algoritam se opisuje na sledeći način (uz pretpostavke da važi $x, y > 0$ i $x < y$)[11]:

1. $y = q_0 \cdot x + r_0$, gde su $q_0, r_0 \in \mathbb{N}$, $r_0 < x$,
2. ako je $r_0 = 0$, $\gcd(x, y) = x$, inače
3. $i = 1$, $x = q_i \cdot r_{i-1} + r_i$, gde su $q_1, r_1 \in \mathbb{N}$, $r_1 < r_0$,
4. ako je $r_i = 0$, $\gcd(x, y) = r_{i-1}$, inače
5. $i = i + 1$, $r_{i-2} = q_i \cdot r_{i-1} + r_i$, $q_i, r_i \in \mathbb{N}$, $r_i < r_{i-1}$; preći na korak (4).

Primer 4.2.16 Primenimo prethodni algoritam na izračunavanje $\gcd(91, 287)$:

- $287 = 3 \cdot 91 + 14$, pa je $r_0 = 14$,

[10] Ovo je jedan od najstarijih poznatih algoritama. Opisan je u sedmoj knjizi Euklidovih "Elemenata".

[11] Ove pretpostavke nisu suštinske. Ako je jedan od x ili y jednak 0, najveći zajednički delilac je onaj drugi. Ako je $x = y$, najveći zajednički delilac je x. Ako je $x > y$, x i y zamenjuju vrednosti.

- $91 = 6 \cdot 14 + 7$, pa je $r_1 = 7$, $i = 1$
- $i = 2$, $14 = 2 \cdot 7$, pa je $r_2 = 0$,

odakle je $\gcd(91, 287) = r_1 = 7$. ■

Za Euklidov algoritam se lako ustanovljava da uvek završava sa radom: vrednosti brojeva r_i opadaju i uvek su nenegativne, tako da celi postupak mora okončati. Korektnost algoritma se dokazuje u tvrđenju 4.2.17. U vezi sa Euklidovi algoritmom ovde ćemo još napomenuti da ćemo o njegovu složenost analizirati u odeljku 5.5.5.

Teorema 4.2.17 Euklidov algoritam izračunava $\gcd(x, y)$.

Dokaz. Neka su poslednji koraci algoritma oblika:

- $r_{k-2} = q_k r_{k-1} + r_k$, $r_k \neq 0$,
- $r_{k-1} = q_{k+1} r_k + r_{k+1}$, $r_{k+1} \neq 0$ i
- $r_k = q_{k+2} \cdot r_{k+1}$, tj. ostatak $r_{k+2} = 0$.

Potrebno je najpre dokazati da poslednji ne-nulti ostatak r_{k+1} zaista bez ostatka deli i x i y. Uočimo najpre da:

- r_{k+1} bez ostatka deli r_k.

Pošto je $r_{k-1} = q_{k+1} r_k + r_{k+1}$, to:

- r_{k+1} bez ostatka deli oba sabirka, pa deli bez ostatka i r_{k-1}.

Dalje, pošto je $r_{k-2} = q_{k+1} r_{k-1} + r_k$, to ponovo važi da:

- r_{k+1} bez ostatka deli i $q_{k+1} r_{k-1}$ i r_k, pa deli bez ostatka i r_{k-2}.

Ponavljajući postupak dobijamo da r_{k+1} deli r_0 i r_1, pa r_{k+1} deli najpre x, a zatim i y.

Još je potrebno dokazati da je r_{k+1} zaista najveći takav broj. Da bismo to pokazali uočimo da svaki drugi delilac c brojeva x i y mora da deli ostatak r_0, jer je $y = q_0 \cdot x + r_0$, a zatim analogno i preostale ostatke r_1, r_2 itd. Dakle, c mora deliti bez ostatka i r_{k+1}, pa je zaista $r_{k+1} = \gcd(x, y)$. ■

Pored neposrednog izračunavanja najvećog zajedničkog delioca $\gcd(x, y)$, Euklidov algoritam daje i dodatne informacije, što se ovde pre svega odnosi na mogućnost da se $\gcd(x, y)$ napiše u posebnom obliku koristeći polazne brojeve x i y.

Teorema 4.2.18 Neka je $d = \gcd(x, y)$. Tada je moguće d napisati u obliku

$$d = m \cdot x + n \cdot y$$

gde su m i n celi brojevi.

Dokaz. Pokazaćemo da se svaki od brojeva r_i koji se pojavljuje tokom izračunavanja $\gcd(x, y)$ može napisati kao zbir oblika $a \cdot x + b \cdot y$. Najpre, pošto je $y = q_0 \cdot x + r_0$ i $x = q_1 \cdot r_0 + r_1$, to važi za

- $r_0 = y - q_0 \cdot x$ i
- $r_1 = x - q_1 \cdot r_0 = x - q_1(y - q_0 \cdot x) = -q_1 \cdot y + (1 - q_1 \cdot q_0)x$.

Dalje, neka za bilo koje uzastopne r_{i-2} i r_{i-1} važi da $r_{i-2} = m' \cdot x + n' \cdot y$ i $r_{i-1} = m'' \cdot x + n'' \cdot y$. Pošto je:

1. $r_{i-2} = q_i \cdot r_{i-1} + r_i$, onda je
2. $r_i = r_{i-2} - q_i \cdot r_{i-1} = (m' \cdot x + n' \cdot y) - q_i(m'' \cdot x + n'' \cdot y) = (m' - q_i \cdot m'')x + (n' - q_i n'')y$,

pa se i r_i može zapisati u traženom obliku, čime je dokazano tvrđenje. ■

Primer 4.2.19 U primeru 4.2.19 izračunato je da je $\gcd(91, 287) = r_1 = 7$, pa je:

- iz $287 = 3 \cdot 91 + 14$, je $14 = 287 - 3 \cdot 91$,
- iz $91 = 6 \cdot 14 + 7$, je $7 = 91 - 6 \cdot (287 - 3 \cdot 91) = 19 \cdot 91 - 6 \cdot 287$. ■

Konačno, primenićemo prethodno rečeno da prikažemo postupak izračunavanja inverza elemenata $\mathbb{Z}_p$. Najpre, neka važi:

- p je prost broj i
- a je prirodan broj, takava da je $0 < a < p$

i neka je potrebno pronaći prirodan broj x za koji je $0 < x < p$, tako da je $a \cdot x \equiv_p 1$, odnosno $x = a^{-1}$ u $\mathbb{Z}_p$. Pošto su p i a uzajamno prosti, jasno je da je $\gcd(a, p) = 1$, ali primenom Euklidovog algoritma određuju se i celi brojevi u i v za koje je

$$\gcd(a, p) = 1 = u \cdot a + v \cdot p.$$

Drugim rečima dobija se u tako da je:

$$a \cdot u \equiv_p 1.$$

Konačno, pošto ne mora biti $u < p$, inverz elementa a je $a^{-1} \equiv_p u$.

Primer 4.2.20 Neka je $p = 234527$, i neka je potrebno izračunati 2^{-1} u $\mathbb{Z}_p$. Tada je:

- $234527 = 117263 \cdot 2 + 1$, pa je
- $-117263 \cdot 2 + 1 \cdot 234527 = 1$ i
- $-117263 \equiv_{234527} 117264$,

odnosno $2^{-1} \equiv_{234527} 117264$ i $\frac{1}{2} \equiv_{234527} 117264$. ∎

U nastavku ćemo razmotriti tri primera u kojima se direktno primenjuje modularna aritmetika:

- generatori pseudoslučajnih brojeva,
- heš-funkcije i
- kriptologija.

4.2.3 Generatori pseudoslučajnih brojeva

Generatori pseudoslučajnih brojeva se koriste za simulaciju slučajnosti u programima. Oni generišu niz brojeva u nekom intervalu tako da postoji uniformna verovatnoća izbora bilo kog broja. Pri tome se može koristiti relativno jednostavni postupak zasnovan na modularnoj aritmetici koji daje niz koji izgleda kao slučajan, ali se elementi izračunavaju pomoću determinističkih funkcija prethodnika. Jedan takav postupak, takozvani *metod linearne kongruencije*, sastoji se u sledećem:

- biraju se prirodni brojevi:
  - n, modul u odnosu na koga se posmatra kongruencija $\equiv_n$,
  - množilac a, takav da je $2 \leq a < n$,
  - pomeraj c, takav da $0 < c < n$ i
  - polazni element niza, x_0, koji se naziva i *seme*, takav da $0 \leq x_0 < n$.
- generiše se niz $\{x_i\}_i$ brojeva iz $\mathbb{Z}_n$ formulom

$$x_{n+1} \equiv_n (a \cdot x_n + c).$$

Primer 4.2.21 Neka je $n = 9$, $a = 7$, $c = 4$ i $x_0 = 3$. Tada je:

- $7 * 3 + 4 = 25$ i $25 \equiv_9 7$, pa je $x_1 = 7$,

- $7 * 7 + 4 = 53$ i $53 \equiv_9 8$, pa je $x_2 = 8$, i slično
- $x_3 = 60 \equiv_9 6$, $x_4 = 46 \equiv_9 1$ itd. ■

Jasno je da nakon ponavljanja jednog broja u nizu (a do takve situacije mora pre ili posle doći jer su članovi niza $0 \leq x_i < n$) ulazi u ciklus, odnosno da se članovi niza periodično ponavljaju. Za dobijanje dužeg perioda preporučljivo je da modul n i množilac a budu veliki, ali ne toliko da množenje $a \cdot (n-1)$ dovede do prekoračenja. Zbog brzog izračunavanja ostatka pri deljenju sa n je pogodno i da je n oblika 2^m, jer se deljenje realizuje jednostavnim odbacivanjem najlakših bitova. Napomenimo i da je ovaj postupak generisanja efikasan u smislu memorijskog zauzeća i vremena izvršavanja, ali da postoje i drugi, napredniji, postupci sa manjim stepenom korelacije generisanih pseudoslučajnih brojeva koji se primenjuju, na primer, u kriptografskim aplikacijama.

4.2.4 Heš-funkcije

U programiranju se često javlja potreba da se radi sa velikim brojem podataka koje je potrebno brzo pretraživati. Jedan pristup ovom problemu je zasnovan na korištenju *heš-funkcija*. Ove funkcije na osnovu identifikatora podatka određuju njegovu lokaciju. Jedna prosta heš-funkcija se uvodi na sledeći način:

- ako je k identifikator podatka i n prirodan broj koji odgovara broju raspoloživih memorijskih lokacija, onda je heš-funkcija h definisana sa
- $h(k) = a$, gde je $a \equiv_n k$ ostatak pri delenju k sa n.

Primer 4.2.22 Za $k_1 = 21$ i $k_2 = 35$ i $n = 9$ je

- kako je $3 \equiv_9 21$, onda je $h(k_1) = 3$ i
- kako je $8 \equiv_9 35$, onda je $h(k_2) = 8$. ■

U primeru 4.2.22 podacima sa identifikatorima k_1 i k_2 pristupa se u jednom koraku. Međutim, očigledan problem ovde može predstavljati situacija kada se dva podatka preslikavaju u istu adresu, tj. kada je $k_i \neq k_j$, ali $h(k_i) = h(k_j)$. Na primer, takva situacija bi se javila u primeru 4.2.22 za $k_3 = 39$, kada je $3 \equiv_9 39$, pa je $h(k_3) = 3 = h(k_1)$. Tu su dva od mogućih rešenja:

- kreira se dinamička lista na koju pokazuje sadržaj memorijske lokacije u koju se preslikava više podataka, a sama lista sadrži te podatke, ili
- ako je memorijska lokacija u čiju adresu se preslikava neki podatak već zauzeta, podatak se smešta u prvu sledeću slobodnu lokaciju.

$$\begin{array}{c} x \longrightarrow A \xrightarrow{E(e,x)} B \xrightarrow{D(d,E(e,x))} x \\ \uparrow \\ P \end{array}$$

Slika 4.1. Komunikacija dve strane u prisustvu prisluškivača.

4.2.5 Kriptologija

U ovom odeljku razmotrićemo jednu vrstu situacija, prikazanu na slici 4.1 u kojima se koristi kriptologija. Dve osobe[12] komuniciraju i treba da onemoguće da treća osoba, prisluškivač, razume njihove poruke. Recimo da osoba A želi da pošalje poruku osobi B. Poruku shvatamo kao binarnu reč. Osobe A i B se prethodno dogovaraju oko algoritama E i D za kodiranje (kriptovanje), odnosno dekodiranje, poruka koji imaju osobinu da za svaku poruku x važi $D(d, E(e, x)) = x$, gde su e i d dve binarne reči, *ključevi* za kodiranje i dekodiranje, koje se biraju tako da funkcije E i D postanu inverzne. U praksi je potrebno i da se funkcije E i D efikasno izračunavaju[13] dok se sigurnost komunikacije obezbeđuje time što su e i d *tajni*[14], tj. znaju ih samo osobe A i B, a x se ne može efikasno izračunati iz $E(e, x)$ bez poznavanja ključa d.

Primer 4.2.23 Za funkcije kodiranja i dekodiranja i odgovarajuće ključeve mogu se izabrati sabiranje po modulu 2 po bitovima, tj. ekskluzivno ili, i bilo koja binarna reč a koja ima istu dužinu kao i poruka koju treba preneti:

10101001 - tekst poruke x,

$+$ 11000111 - ključ a

01101110 - kodirani tekst dobijen kao $x + a$.

Pošto je $D(a, E(a, x)) = (x + a) + a = x$, dekodiranjem dobijamo:

01101110 - kodirani tekst $x + a$,

$+$ 11000111 - ključ a

10101001 - tekst poruke x, dobijen kao $(x + a) + a$,

[12] Tradicionalno se osobe nazivaju agenti koji komuniciraju i daju im se imena Alice i Bob.

[13] U poglavlju **??** ćemo videti da se smatra da su takve funkcije takozvane polinomijalne složenosti.

[14] Secret key.

čime je postignuta inverznost kodiranja i dekodiranja.

Ako prisluškivač uspe da sazna originalnu poruku x i odgovarajuću kodiranu poruku, biće u stanju da otkrije i ključ:

$$\begin{array}{rl} & 10101001 \text{ - tekst poruke } x, \\ + & 01101110 \text{ - kodirani tekst } x+a, \\ & 11000111 \text{ - ključ } a, \text{ dobijen kao } x+(x+a). \end{array}$$

Odavde sledi da prisluškivač može dekodirati kodiranu poruku ako i samo ako zna a, što je primer takozvane apsolutne sigurnosti kodiranja. ■

Primetimo da je u primeru 4.2.23 problematičan dogovor osoba A i B koji se obavlja pre komunikacije, jer je reč o razmeni ključa koji treba da ostane tajan i koji je iste dužine kao i poruka. Ovo nije praktično jer pretpostavlja rešavanje zadatka za čije rešavanje ključ treba da posluži. Problem predstavlja i što postoje postupci, takozvani napadi, koji:

- ako je ključ kratak, vrše pretragu kroz prostor svih mogućih ključeva ili
- ako se ključ koristi više puta u kodiranju, vrše razne analize

i time otkrivaju, ili kako se u žargonu kaže *razbijaju*, tajni ključ. Izlaz iz ovakve situacije je da se svaki bit ključa koristi samo jednom za kodiranje samo jednog bita informacije i zatim odbacuje, što takođe često nije ostvarljiv zahtev.

U teorijskom smislu apsolutna sigurnost kodiranja upotrebom ključa umerene dužine nije moguća, ali se izborom ključa dovoljne dužine, recimo reda veličine 100 bita, postiže praktična sigurnost koja predstavlja verovanje da se ključ ne može brzo otkriti.

U stvarnosti je često problematična i razmena tajnih ključeva, pa u kriptološkim sistemima sa *javnim ključem*[15] svaka osoba B ima par ključeva: javni, svima dostupni, ključ e_B za kodiranje i privatni ključ d_B za dekodiranje poruka. Bilo koja osoba koja želi da pošalje osobi B poruku za kodiranje koristi javni ključ e_B.

Primer sistema sa javnim ključem je *RSA*[16]. Sistem se zasniva na tvrđenjima iz teorije brojeva koje su prethodno navedena. Najpre, iz teoreme 4.2.18 dobijamo:

[15] Public key

[16] Naziv sistema dolazi od imena autora: Ron Ravist, Adi Shamir i Len Adleman.

- za proste brojeve p i q i broj e uzajamno prost sa $(p-1)(q-1)$ postoji broj d takav da je

$$1 = e \cdot d + u \cdot (p-1)(q-1)$$

pa je $ed \equiv_{(p-1)(q-1)} 1$, odnosno (ostatak pri delenju sa $(p-1)(q-1)$ od) d je inverz za e u odnosu na množenje po modulu $(p-1)(q-1)$.

Sledeće izvođenje sledi iz Male Fermaove teoremi 4.2.9. Za proste brojeve p i q, brojeve e i d uzajamno proste sa $(p-1)(q-1)$, takve de je $ed \equiv_{(p-1)(q-1)} 1$, i broj $x < pq$, je:

- $ed = (p-1)k + 1$, gde je k pozitivan celi broj,
- $x^{ed} - x = x(x^{(p-1)k} - 1)$
- pošto važi da je

$$x^{(p-1)k} - 1 = x^{(p-1)^k} - 1 = (x^{(p-1)} - 1)(x^{(p-1)^{k-1}} + \ldots + x^{(p-1)} + 1)$$

onda je $x^{(p-1)k} - 1$ deljivo sa $x^{(p-1)} - 1$, koje je opet po teoremi 4.2.9 deljivo sa p, pa zaključujemo

- $x^{ed} - x$ je deljivo sa p.

Slično, $x^{ed} - x$ je deljivo i sa q, pa i sa pq, odakle je:

- $x^{ed} \equiv_{pq} x$.

Postupak realizacije RSA-kodiranja je:

- biraju se dva prosta broja p i q, recimo $p = 47$ i $q = 71$,
- razmatraju se njihov proizvod, $n = pq = 47 \cdot 71 = 3337$, i $(p-1)(q-1) = 6 \cdot 70 = 3220$,
- iz skupa $\{1, \ldots, (p-1)(q-1)\}$ se bira broj e koji je uzajamno prost sa $(p-1)(q-1)$, recimo $e = 79$,
- nalazi se broj d takav da je $ed \equiv_{(p-1)(q-1)} 1$, $d = 79^{-1} \equiv_{3220} 1019$,
- funkcija kodiranja je

$$E(e, x) \equiv_{pq} x^e$$

- funkcija dekodiranja je

$$D(d, E(a, x)) \equiv_{pq} E(a, x)^d \equiv_{pq} x^{ed} \equiv_{pq} x.$$

Brojevi e i d će biti javni ključ za šifrovanje, odnosno tajni ključ za dešifrovanje. Preciznije, javni ključ je uređeni par (pq, e), dok je tajni ključ par (pq, d). Brojevi p i q treba da ostanu tajni.

U realnim primenama veličine brojeva p, q, e i d su reda stotina bita. Ako da su p i q poznati, broj d se efikasno izračunava, pa se nalaženje inverzne funkcije od $E(e, x)$ svodi na problem faktorisanja brojeva. RSA se zasniva upravo na pretpostavci da su p i q tajni, te da je problem nalaženja broja d težak. Ovde treba obratiti pažnju i na mogućnost efikasnog izračunavanja velikih brojeva koji se pojavljuju pri RSA kodiranju: na primer $x^e (\bmod\, pq)$. U osnovi to izračunavanje se sprovodi na način opisan u primeru 4.2.15:

- ako je broj na koji se stepenenuje neparan, rezultat x^{2k+1} dobijamo kao $x \cdot x^{2k}$,
- ako je broj na koji se stepenenuje paran, rezultat x^{2k} dobijamo kao $x^k \cdot x^k$ i
- postupak s ponavlja dok se ne stigne do x^1, a
- pri svakom koraku se računa samo sa ostacima u odnosu na pq

čime se broj operacija koja treba sprovesti i memorijsko zauzeće minimizuje.

U praksi je poruka koja se šifruje binarni niz. Taj niz se deli na blokove koji svaki za sebe predstavlja ceo broj veći do jednak od 0, a manji od $n = pq$. Zato je svaki blok binarni niz nešto kraći od dužine binarnog zapisa broja n. Na primer:

- neka je 688232 poruka koju treba šifrovati prethodno određenim ključem,
- poruka se deli u dva bloka $x_1 = 688$ i $x_2 = 232$,
- kodiranje: $c_1 \equiv_{pq} x_1^e \equiv_{pq} 688^{79} \equiv_{pq} 1570$ i $c_2 \equiv_{pq} 232^{79} \equiv_{pq}= 2756$,
- dekodiranje: $c_1^d \equiv_{pq} 1570^{1019} \equiv_{pq} 688$ i $c_2^d \equiv_{pq} 2756^{1019} \equiv_{pq} 232$.

Činjenica da važi $x^{ed} \equiv_{pq} x^{de}$, tj. postupci kodiranja i dekodiranja za RSA komutiraju, omogućava da se RSA može koristiti i za takozvani *digitalni potpis* kojim se osoba B uverava da je osoba A zaista poslala poruku x:

- Neka A i B koriste RSA i imaju redom javne ključeve e_A i e_B i tajne ključeve d_A i d_B,
- A šalje potpisanu poruku oblika $S_A(x) = (x, D(d_A, x)) = (x, x^d)$ koja sadrži originalnu poruku x na koju je nadovezan potpis, tj. vrednost $D(d_A, x) \equiv_{pq} x^d$

- B proverava potpis računajući $E(e_A, D(d_A, x)) \equiv_{pq} x^{de} \equiv_{pq} x$.

Ako je zbog sigurnosti potrebno i kodirati celu potpisanu poruku koristi se uobičajeni postupak nad ključevima e_B i d_B.

4.3 Bulove algebre

Definicija 4.3.1 *Bulova algebra*[17] je algebarska struktura $\langle B, +, \cdot, ', 1, 0\rangle$ za koju važi:

- Komutativnost:
  - $x \cdot y = y \cdot x$ i
  - $x + y = y + x$
- Asocijativnost:
  - $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ i
  - $x + (y + z) = (x + y) + z$
- Distributivnost:
  - $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ i
  - $x + (y \cdot z) = (x + y) \cdot (x + z)$
- Svojstva elemenata 0 i 1:
  - $x \cdot 1 = x$ i
  - $x + 0 = x$
- Svojstva komplementa:
  - $x \cdot x' = 0$ i
  - $x + x' = 1$. ■

Među najpoznatije Bulove algebre spadaju:

[17] George Boole, 1815 –1864, engleski matematičar. Najpoznatiji rezultati koje je dao sistematizovani su u knjizi An Investigation of the Laws of Thought, on Which are Founded the Mathematical Theories of Logic and Probabilities (1854). U knjizi je uveo algebarski sistem kojim je formalizovao logičko zaključivanje. Smatrao je da je verovatnoća deo logike i precizirao postupke kojima se na osnovu verovatnoća uslova izračunavaju verovatnoće logičkih posledica.

- *algebra skupova*[18] $\langle \mathbb{P}(A), \cup, \cap, \complement, A, \emptyset \rangle$, gde su A neprazan skup, $\mathbb{P}(A)$ partitivni skup skupa A i $\complement$ komplement u odnosu na skup A,
- *intervalna algebra* u kojoj su elementi skupa nosača konačne unije poluotvorenih intervala oblika $[a, b) \subset [0, +\infty)$, a operacije su unija, presek i komplement u odnosu na $[0, +\infty)$, $1 = [0, +\infty)$ i $0 = [0, 0)$ i
- *iskazna algebra* $BA_2 = \langle \{\top, \bot\}, \vee, \wedge, \neg, \top, \bot \rangle$,
- Lindenbaum-Tarski algebra u kojoj je skup nosač skup svih klasa ekvivalencije skupa iskaznih formula u odnosu na relaciju $\alpha \sim \beta$ definisanu sa $\vdash \alpha \leftrightarrow \beta$, a operacije su $[\alpha] \vee [\beta] = [\alpha \vee \beta]$, $[\alpha] \wedge [\beta] = [\alpha \wedge \beta]$, $\neg[\alpha] = [\neg\alpha]$, $1 = [\alpha \vee \neg\alpha]$ i $0 = [\alpha \wedge \neg\alpha]$.

Poznati rezultat

Teorema 4.3.2 (Stonova teorema o reprezentaciji) Svaka Bulova algebra je izomorfna nekoj algebri skupova. ■

govori da je jedina suštinska razlika između Bulovih algebri kardinalnost skupa nosača koji, opet, kod konačnih algebri mora biti stepen broja 2 zbog kardinalnosti partitivnog skupa konačnog skupa. Teoreme 4.3.3 i 4.3.4 ističu poseban značaj Bulove algebre BA_2.

Teorema 4.3.3 Jednakost $t_1 = t_2$ važi u svim Bulovim algebrama ako i samo ako važi u Bulovoj algebri BA_2. ■

Teorema 4.3.4 Iskazna formula α na jeziku $\{\neg, \wedge, \vee\}$ je tautologija ako i samo ako u Bulovoj algebri BA_2 važi jednakost $\alpha = 1$, pri čemu se iskazna slova shvataju kao promenljive. ■

Slično dualnim izrazima kod skupova, za svaki iskaz na jeziku neke Bulove algebre definiše se njemu *dualni iskaz* koji se od polaznog iskaza dobija sistematskom zamenom simbola $+$ i $\cdot$, odnosno 1 i 0. Lako se proverava da su svi parovi uslova iz definicije 4.3.1 međusobno dualni, što znači da kada je dokazano da neki iskaz posledica ovih uslova, njegov dual je posledica njima odgovarajućih dualnih uslova, pa važi teorema"

Teorema 4.3.5 (Princip dualnosti) Iskaz važi u svim Bulovim algebrama ako i samo ako u svim Bulovim algebrama važi i njegov dual. ■

U sledećoj teoremi sumiraćemo više važnih osobina Bulovih algebri.

[18] Uporediti osobine skupovnih operacija iskazane u tvrđenju 2.1.41 sa uslovima iz definicije 4.3.1.

Teorema 4.3.6 U svim Bulovim algebrama važi:

1. jedinstvenost komplementa: ako $x + y = 1$ i $x \cdot y = 0$, onda $x' = y$,
2. idempotencija: $x \cdot x = x$, $x + x = x$,
3. $x \cdot 0 = 0$, $x \cdot 1 = x$,
4. apsorpcija: $x + (x \cdot y) = x$, $x \cdot (x + y) = x$,
5. involucija: $(x')' = x$,
6. $0' = 1$, $1' = 0$,
7. $x \cdot (x' + y) = x \cdot y$, $x + (x' \cdot y) = x + y$,
8. De Morganovi[19] zakoni: $(x + y)' = x' \cdot y'$, $(x \cdot y)' = x' + y'$,
9. $x \cdot y + x \cdot y' = x$, $(x + y) \cdot (x + y') = x$,
10. konsenzus: $x \cdot y + x' \cdot z + y \cdot z = x \cdot y + x' \cdot z$, $(x + y) \cdot (x' + z) \cdot (y + z) = (x + y) \cdot (x' + z)$.

Dokaz. (1) Kako je $y = y \cdot 1 = y \cdot (x + x') = (y \cdot x) + (y \cdot x') = 0 + (y \cdot x') = (x' \cdot x) + (x' \cdot y) = x' \cdot (x + y) = x' \cdot 1 = x'$, dobija se traženo.
(2) Važi da je: $x = x + 0 = x + (x \cdot x') = (x + x) \cdot (x + x') = (x + x) \cdot 1 = x + x$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(3) Važi da je: $x \cdot 0 = x \cdot (x \cdot x') = (x \cdot x) \cdot x' = x \cdot x' = 0$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(4) Važi da je: $x + (x \cdot y) = (x \cdot 1) + (x \cdot y) = x \cdot (1 + y) = x \cdot 1 = x$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(5) Pošto je $x' + x = 1$ i $x' \cdot x = 0$, x je koplemenet od x', a kako je prema koraku (1) komplement jedinstven, sledi da je $(x')' = x$.
(6) Važi da je: $0' = 0' + 0 = 1$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(7) Važi da je: $x \cdot (x' + y) = x \cdot x' + x \cdot y = 0 + x \cdot y = x \cdot y$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(8) Važi da je: $(x + y) + (x' \cdot y') = ((x + y) + x') \cdot ((x + y) + y') = ((x + x') + y) \cdot (x + (y + y')) = 1 \cdot 1 = 1$, kao i $(x + y) \cdot (x' \cdot y') = (x' \cdot y') \cdot (x + y) = ((x' \cdot y') \cdot x) + ((x' \cdot y') \cdot y) = 0 + 0 = 0$, pa je na osnovu jedinstvenosti komplemeta $(x + y)' = x' \cdot y'$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.
(9) Važi da je: $x \cdot y + x \cdot y' = x \cdot (y + y') = x \cdot 1 = x$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti.

[19]Augustus De Morgan, 1806 – 1871, engleski matematičar.

(10) $x \cdot y + x' \cdot z + y \cdot z = x \cdot y + x' \cdot z + y \cdot z \cdot (x + x') = x \cdot y + x' \cdot z + x \cdot y \cdot z + x' \cdot y \cdot z = x \cdot y \cdot (1 + z) + x' \cdot z \cdot (1 + y) = x \cdot y + x' \cdot z$, dok drugi deo tvrđenja sledi na osnovu principa dualnosti. ∎

U Bulovoj algebri $\langle B, +, \cdot, ', 1, 0\rangle$ se relacija parcijalnog poretka $\leq$ definiše sa $x \leq y$ ako $x + y = y$, odnosno ako $x \cdot y = x$.

4.3.1 Bulove funkcije

Definicija 4.3.7 (Bulove funkcije) Za Bulovu algebru $\langle B, +, \cdot, ', 1, 0\rangle$ *klasa Bulovih funkcija sa n promenljivih* sadrži osnovne funkcije:

- konstantne funkcije $f(x_1, \ldots, x_n) = b$ za svaki $b \in B$ i
- funkcije projekcije $f(x_1, \ldots, x_n) = x_i$

i sve funkcije koje se od njih dobijaju konačnom primenom pravila:

- $(f + g)(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) + g(x_1, \ldots, x_n)$,
- $(f \cdot g)(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) \cdot g(x_1, \ldots, x_n)$ i
- $(f')(x_1, \ldots, x_n) = (f(x_1, \ldots, x_n))'$

na već definisane funkcije f i g. ∎

Ako su izrazi kojima se definišu dve Bulove funkcije međusobno jednaki, očigledno je da se radi o ekvivalentnim zapisima iste funkcije.

Primer 4.3.8 Posmatrajmo Bulovu algebru $\langle B, +, \cdot, ', 1, 0\rangle$ i:

- $f : B^2 \mapsto B$, definisanu sa $f(x, y) = x \cdot (x' + y)$ i
- $g : B^2 \mapsto B$, definisanu sa $f(x, y) = x \cdot y$.

Kako je prema tvrđenju 4.3.6.7 $x \cdot (x' + y) = x \cdot y$, ovo su dva zapisa iste funkcije. ∎

Pošto ispitivanje jednakosti dva izraza izvođenjima sličnim onima iz teoreme 4.3.6 nekada nije trivijalan zadatak, u nastavku ćemo opisati metod koji se može implementirati i sprovoditi automatski.

Definicija 4.3.9 Neka je $f(x_1, x_2, \ldots, x_n)$ Bulova funkcija i

- $f_{x_1'}(x_2, \ldots, x_n) = f(0, x_2, \ldots, x_n)$ i
- $f_{x_1}(x_2, \ldots, x_n) = f(1, x_2, \ldots, x_n)$.

Funkcije $f_{x_1'}(x_2, \ldots, x_n)$ i $f_{x_1}(x_2, \ldots, x_n)$ se nazivaju *kofaktori* funkcije $f(x_1, x_2, \ldots, x_n)$ u odnosu na x_1.

Izrazi $f(0, \ldots, 0, 0)$, $f(0, \ldots, 0, 1)$, $\ldots$, $f(1, \ldots, 1, 1)$ se nazivaju *diskriminante.* ■

Teorema 4.3.10 (Bul-Šenonova (Shannon) teorema ekspanzije) Za svaku Bulovu funkciju $f : B^n \to B$ i sve $(x_1, \ldots, x_n) \in B$ je

$$f(x_1, x_2, \ldots, x_n) = x_1' \cdot f(0, x_2, \ldots, x_n) + x_1 \cdot f(1, x_2, \ldots, x_n)$$

i dualno

$$f(x_1, x_2, \ldots, x_n) = (x_1' + f(1, x_2, \ldots, x_n)) \cdot (x_1 + f(0, x_2, \ldots, x_n)).$$

Dokaz. Dokaz jednakosti se sprovodi indukcijom po složenosti funkcija. Za konstantne funkcije i funkcije projekcije tvrđenje trivijalno važi. U indukcijskom koraku se pretpostavi da tvrđenje važi za funkcije f i g i lako proverava da važi za funkcije koje se od njih dobijaju. ■

Primer 4.3.11 Primenom teoreme ekspanzije 4.3.10 na funkciju

$$f = x_1 \cdot x_2 \cdot x_3' + x_2 \cdot (x_1' + x_3)$$

dobija se

$$f = x_1' \cdot x_2 + x_1 \cdot (x_2 \cdot x_3' + x_2 \cdot x_3) = (x_1' + x_1) \cdot x_2 = x_2,$$

odnosno dualnom varijantom teoreme

$$f = (x_1' + (x_2 \cdot x_3' + x_2 \cdot x_3)) \cdot (x_1 + x_2) = (x_1' + x_2) \cdot (x_1 + x_2) = x_2,$$

pri čemu je $f(0, x_2, \ldots, x_n) = x_2 \cdot (1 + x_3) = x_2$ i $f(1, x_2, \ldots, x_n) = (x_2 \cdot x_3' + x_2 \cdot x_3) = x_2$. Slično, primenom teoreme ekspanzije se može pokazati da je $f(x+y) \cdot f(x'+y) = x' \cdot (f(y) \cdot f(1)) + x \cdot (f(1) \cdot f(y)) = (x'+x) \cdot (f(y) \cdot f(1)) = f(1) \cdot f(y)$. ■

Razvijanje izraza koje se koristi u teoremi ekspanzije 4.3.10 može se nastaviti tako da se dobijaju *kanonske forme*:

- *kanonska disjunktivna forma* ili *kanonska minterm forma* za funkcije koje nisu identički jednake 0:

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) \quad = \quad & x_1' \cdot \ldots \cdot x_{n-1}' \cdot x_n' \cdot f(0, \ldots, 0, 0) + \\ & x_1' \cdot \ldots \cdot x_{n-1}' \cdot x_n \cdot f(0, \ldots, 0, 1) + \ldots + \\ & x_1 \cdot \ldots \cdot x_{n-1} \cdot x_n \cdot f(1, \ldots, 1, 1) \end{aligned}$$

- *kanonska konjunktivna forma* ili *kanonska maksterm forma* za funkcije koje nisu identički jednake 1:

$$\begin{aligned} f(x_1, x_2, \ldots, x_n) &= (x_1 + \ldots + x_{n-1} + x_n + f(0, \ldots, 0, 0)) \cdot \\ & \quad (x_1 + \ldots + x_{n-1} + x_n' + f(0, \ldots, 0, 1)) \cdot \ldots \cdot \\ & \quad (x_1' + \ldots + x_{n-1}' + x_n' + f(1, \ldots, 1, 1)). \end{aligned}$$

Primetimo da se za istaknute slučajeve kada je funkcija identički jednaka 0 (odnosno 1) kanonska disjunktivna forma (odnosno kanonska konjunktivna forma) upravo svode na 0 (odnosno 1).

Izrazi:

$$x_1' \cdot \ldots \cdot x_{n-1}' \cdot x_n',$$

$$x_1' \cdot \ldots \cdot x_{n-1}' \cdot x_n, \ \ldots$$

$$x_1 \cdot \ldots \cdot x_{n-1} \cdot x_n$$

se nazivaju *mintermi*, a izrazi

$$x_1' + \ldots + x_{n-1}' + x_n',$$

$$x_1' + \ldots + x_{n-1}' + x_n, \ \ldots$$

$$x_1 + \ldots + x_{n-1} + x_n$$

sse nazivaju *makstermi*.

Primer 4.3.12 Posmatrajmo funkciju

$$f(x, y) = x + y.$$

Kako je $f(0, 0) = 0$ i $f(1, 0) = f(0, 1) = f(1, 1) = 1$, to je disjunktivna kanonska forma oblika

$$x' \cdot y' \cdot f(0, 0) + x' \cdot y \cdot f(0, 1) + x \cdot y' \cdot f(1, 0) + x \cdot y \cdot f(1, 1) = x' \cdot y + x \cdot y' + x \cdot y.$$

Posmatrajmo funkciju

$$f = x_1 \cdot x_2 + a \cdot x_2'$$

nad Bulovom algebrom sa skupom nosačem $B = \{0, a, b, 1\}$. Njena disjunktivna kanonska forma je

$$a \cdot x_1' \cdot x_2' + 0 \cdot x_1' \cdot x_2 + a \cdot x_1 \cdot x_2' + 1 \cdot x_1 \cdot x_2,$$

dok je

$$(a + x_1' + x_2') \cdot (0 + x_1' + x_2) \cdot (a + x_1 + x_2') \cdot (1 + x_1 + x_2)$$

njena konjunktivna kanonska forma. ■

Na osnovu sledećeg tvrđenja kanonske forme se mogu koristiti u ispitivanju jednakosti Bulovih funkcija.

Teorema 4.3.13 Kanonske forme date Bulove funkcije su jedinstvene (do na redosled minterma/maxterma u zapisu).

Dokaz. Razmotrićemo slučaj kanonske disjunktivne forme. Pretpostavimo da funkcija $f(x_1, x_2, \ldots, x_n)$ ima dve takve različite forme. Primetimo najpre da vrednosti diskriminanti $f(0, \ldots, 0, 0)$, $f(0, \ldots, 0, 1)$, $\ldots$, $f(1, \ldots, 1, 1)$ mogu biti samo 0 ili 1, jer su i argumenti funkcije tog oblika. Zbog toga, nakon izračunavanja vrednosti diskriminanti, te dve forme se mogu zapisati kao:

- $f(x_1, x_2, \ldots, x_n) = M_1 + \ldots + M_m$ i
- $f(x_1, x_2, \ldots, x_n) = N_1 + \ldots + N_n$,

gde su M_i i N_j mintermi. Pošto su forme različite, postoji minterm koji jeste u jednoj, a nije u drugoj formi i neka to bude minterm M_k. On se razlikuje od svih minterma N_j, što znači da za svaki N_j postoji $x_{k(j)}$ takav da je $x_{k(j)}$ u M_k, a $x'_{k(j)}$ u N_j, ili obrnuto. Zbog toga će uvek važiti da je $M_k \cdot N_j = 0$. Dalje je

- $M_k \cdot f(x_1, x_2, \ldots, x_n) = M_k \cdot (M_1 + \ldots + M_m) = M_k \cdot M_k = M_k$ i
- $M_k \cdot f(x_1, x_2, \ldots, x_n) = M_k \cdot (N_1 + \ldots + N_n) = 0$,

što je očigledna kontradikcija. Prema tome kanonska disjunktivna forma svake Bulove funkcuje je jedinstvena.

Drugi deo tvrđenja o jedinstvenosti kanonske konjuktivne forme sledi na osnovu principa dualnosti. ■

Prema tome, ako dve funkcije imaju jednake kanonske forme, one su jednake. Uopšte, uvodi se pojam kanonskog sistema sa reprezentovanje Bulovih funkcija.

Definicija 4.3.14 Neki sistem za reprezentovanje Bulovih funkcija je *kanonski* ako za dve funkcije f i g važi da su jednake ako i samo ako im je reprezentacija jedinstvena. ■

Primer 4.3.15 Posmatrajmo funkcije

- $f(x, y) = x + y$ i
- $g(x, y) = x' \cdot y + x$.

Obe funkcije imaju istu kanonsku disjunktivnu formu: $x' \cdot y + x \cdot y' + x \cdot y$, pa je reč samo o različitim zapisima iste funkcije. ■

Druga bitna posledica teoreme 4.3.13 je da je svaka Bulova funkcija potpuno okarakterisana diskriminantama, odnosno vrednostima u ovim specijalnim tačkama, i bez obzira na skup nosač B koji ne mora biti $\{0, 1\}$.

Primer 4.3.16 Ako je za funkciju $f(x, y)$ ispunjeno da je $f(0, 0) = 0$ i $f(1, 0) = f(0, 1) = f(1, 1) = 1$, njena disjunktivna kanonska forma je $x' \cdot y + x \cdot y' + x \cdot y$. ■

Imajući u vidu karakterizaciju funkcija diskriminantama, moguće je izračunati broj različitih n-arnih Bulovih funkcija za neku Bulovu algebru B. Naime, broj preslikavanja n-torki nula i jedinica (kojih ima 2^n) u skup kardinalnosti $|B|$ je $|B|^{2^n}$, pa je i broj Bulovih funkcija takođe $|B|^{2^n}$. Kako svih n-arnih funkcija za Bulovu algebru kardinalnosti $|B|$ ima $|B|^{|B|^n}$ vidi se da nisu sve funkcije $f : B^n \to B$ Bulove u smislu definicije 4.3.7. Te funkcije se nazivaju *pseudo-Bulove funkcije*. Iako sve funkcije $f : B^n \to B$ nisu Bulove, ovih ima veoma mnogo, čak i za male dimenzije skupa B. Na primer, razmotrimo Bulovu algebru sa 4 elementa i Bulove funkcije arnosti 4. Ukupan broj takvih Bulovih funkcija iznosi $|B|^{2^4} = 4^{2^4} = 2^{32}$.

4.3.2 Minimizacija Bulovih funkcija

Digitalni sistemi su, bez obzira na raznovsnost i složenost, bazirani na jednostavnim *logičkim kapijama*[20] koje predstavljaju osnovne logičke operacije poput konjunkcije, disjunkcije, negacije itd. Kombinovanjem logičkih kapija dobijaju se *logička kola*. Jedan od važnih aspekata razvoja tih kola je dizajniranje optimizovanih rešenja koja štede materijal i energiju, a treba da efikasno i tačno rade. Pri tome se koriste tehnike bazirane na rezultatima u oblasti Bulovih algebri.

Primer 4.3.17 Primenom principa konsenzusa (tvrđenje 4.3.6.10), izraz $(x + y) \cdot (x' + z) = x \cdot x' + x \cdot z + x' \cdot y + y \cdot z = x \cdot z + x' \cdot y + y \cdot z$ se svodi na $x \cdot z + x' \cdot y$. ■

U sledećem primeru ilustrovana je primena kanonskih formi za opis jednog logičkog kola.

Primer 4.3.18 Puni sabirač[21] je elektronsko kolo sa:

[20]Engleski: *logic gates*.

[21]*Full adder*. Drugi naziv je *binarni sabirač*.

- tri ulaza x, y i c, gde x i y odgovaraju bitovima koji se sabiraju, a c prenosu pri sabiranju prethodnog para bitova, i
- dva izlaza r i s, gde je r bit rezultata, a s bit prenosa koji predstavlja ulazni podatak za sabiranje sledećeg para bitova.

Koristeći tabelu 4.1 sa vrednostima funkcija koje daju r i s, dobijaju se kanonska disjunktivna normalna forma za r:

$$(x' \cdot y' \cdot c) + (x' \cdot y \cdot c') + (x \cdot y' \cdot c') + (x \cdot y \cdot c)$$

i za s:

$$(x' \cdot y \cdot c) + (x \cdot y' \cdot c) + (x \cdot y \cdot c') + (x \cdot y \cdot c).$$

Ove forme se mogu optimizovati u

$$(x' \cdot ((y' \cdot c) + (y \cdot c'))) + (x \cdot ((y' \cdot c') + (y \cdot c))),$$

odnosno

$$(y \cdot c) + (x \cdot ((y' \cdot c) + (y \cdot c'))).$$

Ako uvedemo veznik za ekskluzivnu disjunkciju $\veebar$ tako da je $p \veebar q$ skraćenica za $(p' \cdot q) + (p \cdot q')$, onda se optimizovane forme mogu zapisati u obliku $x \veebar (y \veebar c)$ za r, odnosno $(y \cdot c) + (x \cdot (y \veebar c))$ za s. ■

x	y	c	r	s
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Tabela 4.1. Tablica punog sabirača.

Optimizacija broja elemenata logičkih kola, pored direktnih ušteda, smanjuje i mogućnost grešaka u razvoju. Imajući u vidu prednosti koje imaju jednostavniji izrazi koji predstavljaju logička kola, potrebno je precizirati relaciju "biti jednostavniji izraz":

- razmatraju se ekvivalentni izrazi predstavljeni u disjunktivnoj formi (kao zbirovi proizvoda *literala* - promenljivih ili komplementiranih promenljivijih),

- jednostavniji su izrazi imaju manji broj proizvoda koji se sabiraju i
- ako se dva izraza sastoje od istog broja proizvoda, jednostavniji je onaj koji ima ukupno manji broj literala (pri čemu se broje sve pojave literala, a ne samo različiti literali).

Primetimo da u skladu sa terminologijom iz odeljka 2.1.2, ne mora postojati jedinstveni najmanji (najjednostavniji) izraz, već je moguće da više međusobno ekvivalentnih izraza zadovoljava navedene uslove. Bez obzira na jedinstvenost, minimalnim ćemo zvati svaki izraz koji te uslove zadovoljava.

Karnuovi dijagrami

Karnuovi dijagrami[22] je naziv jedne tabelarne metode za optimizovanje izraza u disjunktivnoj normalnoj formi kojima se definišu Bulove funkcije. Za neki fiksirani skup promenljivih, polja tabele odgovaraju mintermima nad tim promenljivim. Tabela 4.2 prikazuje Karnuov dijagram koji odgovara promeljivima x, y i z. Redovi tabele su označeni jednom izabranom promenljivom, odnosno njenim komplementom (ovde je to x, odnosno x'), dok su kolone označene mintermima sastavljenim od preostalih promenljivih (ovde su to y i z).

Bitno je uočiti da su mintermi poređani tako da se mintermi u susednim ćelijama razlikuju samo na jednom mestu, odnosno svi literali sem jednog su im identični, dok je taj literal je u jednoj ćeliji sama promenljiva, a u drugoj njen komplement. Pod susednim ćelijama ovde se podrazumevaju i ćelije na početku i kraju jednog reda, odnosno jedne kolone, ali ne i ćelije susedne po nekoj dijagonali.

	$y \cdot z$	$y' \cdot z$	$y' \cdot z'$	$y \cdot z'$
x	$x \cdot y \cdot z$	$x \cdot y' \cdot z$	$x \cdot y' \cdot z'$	$x \cdot y \cdot z'$
x'	$x' \cdot y \cdot z$	$x' \cdot y' \cdot z$	$x' \cdot y' \cdot z'$	$x' \cdot y \cdot z'$

Tabela 4.2. Karnuov dijagram za promeljive x, y i z.

U slučaju većeg broja promenljivih, recimo označavanje redova i kolona je drugačije, ali tako da i dalje treba da važi da susedne ćelije sadrže minterme koji se razlikuju samo po jednom literalu. U praksi se Karnuovi dijagrami retko koriste za izraze sa više od 4 promenljive, a ako izraz sadrži tačno 4 promenljive, kolone se redom označavaju sa: $z \cdot t$, $z' \cdot t$, $z' \cdot t'$ i $z \cdot t'$, a redovi sa: $x \cdot y$, $x' \cdot y$, $x' \cdot y'$ i $x \cdot y'$.

Za konkretan izraz u ćelijama koje odgovaraju mintermima prisutnim u disjunktivnoj normalnoj formi izraza upisuje je 1, dok su ostale ćelije

[22] Engleski: *Karnaugh map.*

prazne. Na primer izraz $x \cdot y' \cdot z + x \cdot y' \cdot z' + x' \cdot y \cdot z + x' \cdot y \cdot z'$ je predstavljen Karnuovim dijagramom u tabeli 4.3.

	$y \cdot z$	$y' \cdot z$	$y' \cdot z'$	$y \cdot z'$
x		1	1	
x'	1			1

Tabela 4.3. Karnuov dijagram izraza $x \cdot y' \cdot z + x \cdot y' \cdot z' + x' \cdot y \cdot z + x' \cdot y \cdot z'$.

Prisutnost znaka 1 u (kako horizontalno, tako i vertikalno) susednim ćelijama nekog Karnuovog dijagrama znači da su disjunktivnoj normalnoj formi izraza prisutna dva minterma koja se razlikuju samo u jednom literalu. U tom slučaju ta promenljiva se može eliminisati primenom jednakosti formulisanih u teoremi 4.3.6. Na primer, pošto su u tabeli 4.3 u susednim ćelijama i prvom redu koje odgovaraju mintermima $x \cdot y' \cdot z$ i $x \cdot y' \cdot z'$ prisutni znaci 1, imamo da je:

$$x \cdot y' \cdot z + x \cdot y' \cdot z' = x \cdot y'(z + z') = x \cdot y'.$$

Slično, zbog prisutnih znaka 1 u ćelijama u drugom redu koje odgovaraju mintermima $x' \cdot y \cdot z$ i $x' \cdot y \cdot z'$ dobija se:

$$x' \cdot y \cdot z + x' \cdot y \cdot z' = x' \cdot y,$$

odakle je

$$x \cdot y' \cdot z + x \cdot y' \cdot z' + x' \cdot y \cdot z + x' \cdot y \cdot z' = x \cdot y' + x' \cdot y,$$

što je jedna minimalna forma polaznog izraza.

Slično, prisutnost znaka 1 u 4 (ili 8, ili 2^n) susedne ćelije, na primer kako je dato u tabeli 4.4 za 4 ćelije, omogućava eliminisanje većeg broja promenljivih:

$$\begin{array}{rcl} x \cdot y' \cdot z & + & x \cdot y' \cdot z' + x' \cdot y' \cdot z + x' \cdot y' \cdot z' \\ & = & x \cdot y'(z + z') + x' \cdot y'(z + z') \\ & = & x \cdot y' + x' \cdot y' = (x + x')y' = y' \end{array}$$

tako da preostanu samo zajednički literali u posmatranim mintermima.

Imajući u vidu kriterijume koji preciziraju šta su minimalni izrazi, jednostavna strategija minizacije bazirane na Karnuovim dijagramima je da se prvo redukuje broj minterma, a potom broj literala. Peciznije, treba:

- odrediti izolovana polja u kojima je znak 1, odnosno polja koja sadrže, a njihovi susedi ne sadrže 1; mintermi koji im odgovaraju ne mogu biti eliminisani, ni skraćeni

	$y \cdot z$	$y' \cdot z$	$y' \cdot z'$	$y \cdot z'$
x		1	1	
x'		1	1	

Tabela 4.4. Karnuov dijagram sa 4 susedne ćelije u koje je upisan znak 1.

- odrediti polja u kojima je znak 1, a koja imaju tačno jedno susedno polje u kome je takođe 1; 2 odgovarajuća minterma zamenjuju se jednim proizvodom literala koji je zajednički tim poljima,
- odrediti polja u kojima je znak 1, a koja na jedinstveni način formiraju blok od 4 susedna polja; 4 odgovarajuća minterma zamenjuju se jednim proizvodom literala koji je zajednički tim poljima, (postupak ponoviti za blokove od 8, 16, ..., polja),
- za preostala polja koja sadrže 1 formirati najveće kvadratne blokove, tako da je tih blokova što manje a da su sva takva polja u njih uključena.

Moguće je da pri ovom postupku isto polje bude u sklopu različitih blokova, što ne predstavlja problem jer se tumači kao da se isti minterm više puta pojavljuje u polaznom izrazu.

Primer 4.3.19 Razmotrimo izraz:

$$x \cdot y \cdot z + x \cdot y' \cdot z + x \cdot y' \cdot z' + x' \cdot y' \cdot z' + x \cdot y \cdot z'$$

i njemu odgovarajući Karnuov dijagram:

	$y \cdot z$	$y' \cdot z$	$y' \cdot z'$	$y \cdot z'$
x	1	1	1	
x'			1	1

U dijagramu ne postoje izolovana polja u kojima je znak 1.

U drugom koraku određujemo polja u kojima je znak 1, a koja imaju tačno jedno susedno polje u kome je takođe 1. To su polja $x \cdot y \cdot z$ (susedno je polje $x \cdot y' \cdot z$) i $x' \cdot y \cdot z'$ (susedno je polje $x' \cdot y' \cdot z'$). Ovim dobijamo proizvode: $x \cdot z$ i $x' \cdot z'$.

Pošto ne postoje blokovi od 4 susedna polja u kojima je upisan znak 1, preostaje samo poslednji korak u kome razmatramo blokove od 2 susedna polja u kojima je 1. To se može uraditi na dva načina:

- to je blok polja $x \cdot y' \cdot z'$ i njemu susednog polja $x' \cdot y' \cdot z'$ ili
- to je blok polja $x \cdot y' \cdot z'$ i njemu susednog polja $x \cdot y' \cdot z$.

Prvim izborom dobija se proizvod $y' \cdot z'$, a drugim $x \cdot y'$. Odatle su određene dve forme minimalnog izraza:

- $x \cdot z + x' \cdot z' + y' \cdot z'$, odnosno
- $x \cdot z + x' \cdot z' + x \cdot y'$. ■

4.3.3 Binarni dijagrami odlučivanja

U ovom odeljku ćemo opisati jeda postupak efikasnog predstavljanja Bulovih funkcija koji je baziran na korištenju *binarnih diagrama odlučivanja*[23] koji se skraćeno označavaju sa BDD. Pod određenim uslovima BDD je kanonski sistem predstavljanja Bulovih funkcija. U definiciji BDD biće korišten pojam direktnog acikličnog grafa koji se uvodi definicijom 6.8.1. Na slikama koje će predstavljati takve grafove će biti pretpostavljeno da su ivice usmerene na dole, odnosno na desno ako su ivice horizontalne, ps na njima neće biti ivica sa strelicama koje označavaju usmerenje.

Definicija 4.3.20 BDD za funkciju F sa više izlaza je direktni aciklični graf $\langle V \cup \Phi \cup \{0, 1\}, E\rangle$. Skup čvorova je podeljen u tri podskupa:

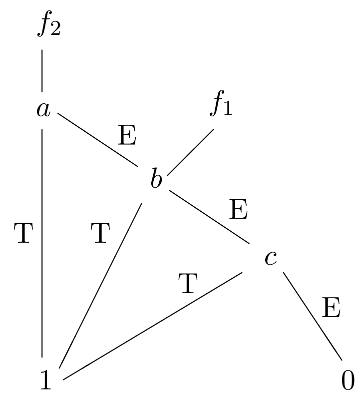
- V je skup unutrašnjih čvorova takav da svaki $v \in V$ ima stepen izlaznog grananja 2 i oznaku $l(v) \in S_F$, gde je $S_F = \{x_1, \ldots, x_n\}$ skup promenljivih od kojih zavisi funkcija F,
- $\{0, 1\}$ je skup završnih čvorova i
- Φ, skup funkcijskih čvorova, je skup polaznih čvorova.

Funkcijski čvorovi predstavljaju komponente funkcije F. Za svaki čvor $v \in V$ izlazne ivice su označene sa T i E[24]. Unutrašnji čvor v se opisuje sa $(l(v), T, E)$. Promenljive iz S_F su uređene tako da, ako je v_j naslednik od v_i, onda kažemo da je $l(v_i) < l(v_j)$. ■

BDD reprezentuje Bulovu funkciju F sa jednim ili više izlaza. Bulova funkcija sa više izlaza u stvari predstavlja složeno logičko kolo u kome se nad istim skupom promenljivih istovremeno računa više funkcija sa po jednim izlazom. Da bi se efikasnije realizovali, zajednički podizrazi tih funkcija fizički se postavljaju na isto mesto na čipu. Na slici 4.2 je prikazan jedan BDD koji odgovara funkciji $F(a, b, c) = (f_1, f_2) = (b + c, a + b + c)$ sa dva izlaza.

[23] Binary Decision Diagrams.

[24] Oznake su skraćenice od *then* i *else* i odgovaraju redom vrednostima 1 i 0 promenljive iz čvora repa.



Slika 4.2. BDD za funkciju $F(a, b, c) = (b + c, a + b + c)$.

Definicija 4.3.21 Funkcija F koju reprezentuje neki BDD definiše se induktivno:

1. Funkcija završnog čvora označenog sa 1 je konstantna funkcija 1.

2. Funkcija završnog čvora označenog sa 0 je konstantna funkcija 0.

3. Funkcija ivice je funkcija čvora glave.

4. Funkcija čvora $v \in V$ je data sa $l(v)f_T + l(v)'f_E$ gde su f_T i f_E redom funkcije izlaznih ivica T i E čvora v.

5. Funkcija funkcijskog čvora $\phi \in \Phi$ je funkcija njegove izlazne ivice. ■

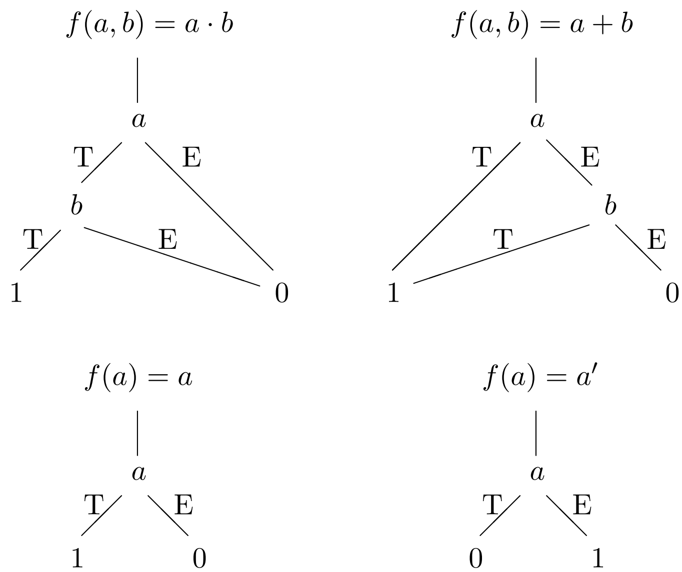
Na slici 4.3 su prikazane BDD-reprezentacije nekih tipičnih funkcija.

Primer 4.3.22 Na primer, izračunajmo funkciju čiji je BDD označen sa $a + b$ na slici 4.3. Najpre je $f(1) = 1$ i $f(0) = 0$. Zatim je za čvor označen sa b, $f_T = 1$ i $f_E = 0$, pa je $f(b) = b \cdot 1 + b' \cdot 0 = b$. Za čvor a je $f_T = 1$ i $f_E = a' \cdot b$, pa je $f(a) = a \cdot 1 + a' \cdot b = a + b$. Konačno, za čvor $f(a, b)$ funkcija izlazne ivice, a time i samog funkcijskog čvora, jednaka je $a + b$. ■

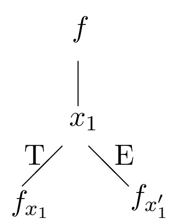
Konstrukcija BDD-reprezentacije Bulovih funkcija

Sledećim postupkom u kome se koristi teorema ekspanzije 4.3.10 polazeći od analitičkog opisa funkcije dolazi se do njene BDD-reprezentacije:

1. Uoči se jedan redosled promenljivih koje se javljaju u funkciji f: $x_1 < x_2 < \ldots < x_n$.



Slika 4.3. BDD reprezentacije za neke tipične Bulove funkcije.



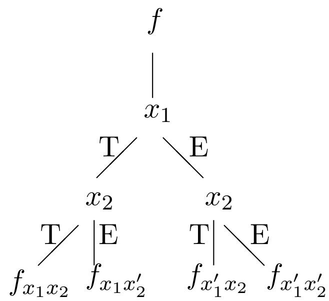
Slika 4.4. Parcijalni BDD nakon primene ekspanzije u odnosu na x_1.

2. Za prvu promenljiva u nizu se izračunaju njeni kofaktori f_{x_1} i $f_{x'_1}$ i konstruiše parcijalni BDD kao na slici 4.4.

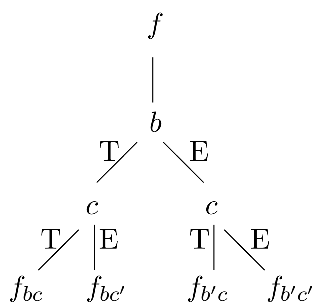
3. Računaju se kofaktori $(f_{x_1})_{x_2}$, $(f_{x_1})_{x'_2}$ i $(f_{x'_1})_{x_2}$, $(f_{x'_1})_{x'_2}$ u odnosu na sledeću promenljivu x_2 iz niza i konstruiše se parcijalni BDD kao na slici 4.5.

4. Prethodni korak se ponavlja dok god se ne dođe do konstantnih kofaktora 1 i 0 ili se ne iscrpe sve promenljive, pri čemu se koristi da je $(x_i)_{x_i} = 1$, a $(x_i)_{x'_i} = 0$ i $(x_i)_{x_j} = (x_i)_{x'_j} = x_i$, za $i \neq j$.

Primer 4.3.23 Neka je $f(a,b,c,d) = a \cdot b \cdot c + b' \cdot d + c' \cdot d$ funkcija za koju treba konstruisati BDD i neka je izabrani redosled promenljivih $b <$



Slika 4.5. Parcijalni BDD nakon primene ekspanzije u odnosu na x_2.

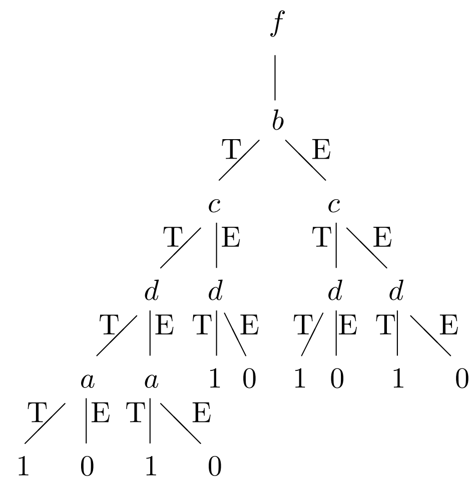


Slika 4.6. Parcijalni BDD za funkciju $a \cdot b \cdot c + b' \cdot d + c' \cdot d$.

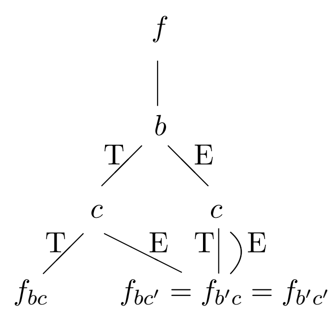
$c < d < a$. U prvom koraku izračunavaju se kofaktori od f u odnosu na b, $f_b = f(a, 1, c, d)_b = a \cdot c + c' \cdot d$ i $f(a, 0, c, d) = f_{b'} = d + c' \cdot d$. Zatim se izračunavaju $f_{b,c} = a$ i $f_{b,c'} = f_{b',c} = f_{b',c'} = d$. Sada parcijalni BDD izgleda kao na slici 4.6. Kofaktori u odnosu na d su $f_{b,c,d} = f_{b,c,d'} = a$, $f_{b,c',d} = f_{b',c,d} = f_{b',c',d} = 1$ i $f_{b,c',d'} = f_{b',c,d'} = f_{b',c',d'} = 0$. Poslednjih šest kofaktora su konstantne funkcije i na njima sa postupak prekida. Na prva dva kofaktora se primenjuje jedina preostala promenljiva i dobija $f_{b,c,d,a} = f_{b,c,d',a} = 1$, odnosno $f_{b,c,d,a'} = f_{b,c,d',a'} = 0$. Završni BDD za funkciju f izgleda kao na slici 4.7. ■

Redukovani BDD

U primeru 4.3.23 se lako uočava da su neki kofaktori koji se izračunavaju tokom konstrukcije BDD-reprezentacije funkcije međusobno jednaki. Na primer, $f_{b,c'} = f_{b',c} = f_{b',c'} = d$. U takvim slučajevima se može kreirati samo jedan čvor što će optimizovati veličinu BDD-reprezentacije. Ova situacija



Slika 4.7. Završeni BDD za funkciju $a \cdot b \cdot c + b' \cdot d + c' \cdot d$.



Slika 4.8. Optimizovani BDD.

je prikazana na slici 4.8. Sledećom definicijom se formalizuje šta se podrazumeva pod minimalnom reprezentacijom Bulove funkcije.

Definicija 4.3.24 BDD je *redukovan* ako je ispunjeno:

1. Svaki unutrašnji čvor $v \in V$ je potomak nekog funkcijskog čvora $f \in \Phi$.
2. U grafu ne postoje izomorfni podgrafovi.
3. Za svaki čvor v i njegove izlazne ivice T i E je $f_T \neq f_E$. ■

Uslovi 2 i 3 garantuju da je redukovani BDD jedna vrsta minimizovane reprezentacije odgovarajuće Bulove funkcije. Kanoničnost je još jedno važno svojstvo koje, poput potpunih formi, ima redukovani BDD.

Teorema 4.3.25 Za fiksirano uređenje promenljivih redukovani BDD je kanonski sistem za reprezentovanje Bulovih funkcija. ■

Sledećim sistematskim postupkom se polazeći od običnog dobija redukovani BDD u odnosu na jedinstveno uređenje promenljivih primenjeno prilikom konstrukcije:

1. Polazeći od završnih čvorova uočavaju se izomorfni podgrafovi. Za svaku klasu uočenih međusobno izomorfnih podgrafova, odbacuju se svi, sem jednog, na koga se usmeravaju sve ivice koje su pokazivale na preostale.

2. Svaki čvor čije obe izlazne ivice nakon prošlog koraka pokazuju na isti podgraf G se briše, dok se podgraf G direktno spaja sa čvorovima prethodnicima obrisanog čvora.

3. Pethodna dva koraka se ponavljaju dok god ima izmena na grafu.

Primetimo da korak 2 postupka predstavlja sledeću situaciju. Funkcija čvora $v \in V$ čije obe izlazne ivice pokazuju na isti podgraf G je oblika

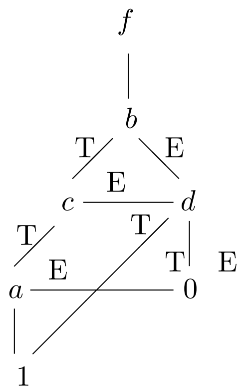
$$f(v) = v \cdot f(G) + v' \cdot f(G) = f(G)$$

što je istovremeno i funkcija ivice čija je čvor v glava, pa je očigledno da je ovaj čvor suvišan i da njegovo uklanjanje ne utiče na vrednost reprezentovane funkcije.

Primer 4.3.26 Primenimo opisani postupak na završni BDD koji odgovara funkciji iz primera 4.3.23. Najpre se po četiri čvora 1, odnosno 0, svedu na po jedan čvor te vrste, a odgovarajuće ivice preusmere. Slično, dva podgrafa sa korenom u čvorovima označenim sa a se svode na jedan na koji se usmeravaju obe ivice najlevljeg čvora označenog sa d. Prema koraku 2, taj čvor se eliminiše, a T ivica levog čvora označenog sa c sada pokazuje na podgraf čiji je koren označen sa a. Dalje, tri podgrafa sa korenom označenim sa d se svode na jedan, na koji pokazuju E ivica levog čvora označenog sa c i obe ivice desnog čvora označenog sa c. Ponovo, prema koraku 2, taj čvor se briše i E ivica čvora označenog sa b pokazuje na podgraf čiji koren je označen sa d. Tako se dobija redukovani BDD prikazan na slici 4.9. ■

ITE-algoritam za direktnu konstrukciju redukovanog BDD-a

Umesto da se prvo konstruiše neoptimizovani BDD, pa da se on potom redukuje, i vremenski i prostorno je pogodnije da se konstrukcija redukovanog BDD-a izvede direktno. To se ostvaruje tako što se pre dodavanja čvora u



Slika 4.9. Redukovani BDD.

graf proveri da li u grafu već postoji izomorfan podgraf. Jedna mogućnost je pretraživanje do tada konstruisanog grafa, ali efikasnija provera se postiže korištenjem *tabele jedinstvenosti*[25], realizovane kao heš tabela[26], u kojoj se čuvaju funkcije reprezentovane do tog trenutka konstrukcije. Upotreba tabele jedinstvenosti garantuje da će konstrukcija BDD-a u kome se kao komponente javljaju jednake funkcije završiti tako što funkcije imaju jedinstveni podgraf koji ih predstavlja. Posledica toga je da se i provera jednakosti funkcija vrši u konstantnom vremenu upoređivanjem na šta pokazuju izlazne ivice iz funkcijskih čvorova. Ključ koji se koristi za heš tabelu je oblika $(v, f_x, f_{x'})$, gde je v celi broj - redni broj promenljive po kojoj se izračunavaju kofaktori, a f_x i $f_{x'}$ pokazivači na podgrafove koji reprezentuju odgovarajuće kofaktore. Ako takav čvor već postoji u tabeli, prilikom konstrukcije BDD-a se samo pokazuje na njega, inače se on dodaje i u do tada konstruisani podgraf i u heš tabelu.

U nastavku teksta ćemo opisati osnovne ideje ITE-algoritma za direktnu konstrukciju redukovane BDD reprezentacije funkcije. Funkcija koja se zadaje kao ulaz sadrži pored standardnih operacija NOT, AND i OR[27] i operacije XOR, $NAND$, NOR, $XNOR$, $\leq$[28] i druge kojima se direktno,

[25] Unique table.

[26] Hash table. To je struktura podataka u kojoj se podatak locira prema nekom ključu. Funkcija heširanja preslikava ključ podatka u adresu. Ako više ključeva daju istu adresu, preklapanje podataka se izbegava upotrebom povezane liste. Brzina pristupa je povezana sa kvalitetom funkcije heširanja koja treba da obezbedi da se malo ključeva preslikava u istu adresu, a time i da povezane liste budu kratke. Ako je ovo postignuto, brizina pronalaženja podatka je (skoro) konstantna.

[27] Operacije $'$, $\cdot$ i $+$.

[28] $xXORy = (x \cdot y') + (x' \cdot y)$, $xNANDy = (x \cdot y)'$, $xNORy = (x + y)'$, $xXNORy = (xXORy)'$, $x \leq y = x' + y$.

zbog efikasnosti, realizuju razni iskazni logički veznici. Zanimljivo je da se sve unarne i binarne operacije izražavaju pomoću jedne ternarne operacije

$$ITE(f, g, h) = f \cdot g + f' \cdot h$$

od koje i dolazi naziv celog algoritma[29]. ITE-algoritam je rekurzivan i zasniva se na sledećoj transformaciji u kojoj se koristi teorema ekspanzije 4.3.10:

$$\begin{aligned} ITE(f, g, h) &= f \cdot g + f' \cdot h \\ &= x_1 \cdot (f_{x_1} \cdot g_{x_1} + f'_{x_1} \cdot h_{x_1}) + x'_1 \cdot (f_{x'_1} \cdot g_{x'_1} + f'_{x'_1} \cdot h_{x'_1}) \\ &= ITE(x_1, ITE(f_{x_1}, g_{x_1}, h_{x_1}), ITE(f_{x'_1}, g_{x'_1}, h'_{x_1})), \end{aligned}$$

pri čemu su završni koraci:

$$ITE(1, f, g) = ITE(0, g, f) = ITE(f, 1, 0) = ITE(g, f, f) = f.$$

ITE-algoritam se može opisati sledećom procedurom:

procedure ITE(f, g, h)
begin
    (rezultat, zavrsni_korak) := Zavrsni_Korak(f, g, h)
    if zavrsni_korak **then** return(rezultat)
    x := Prva_Promenljiva(f, g, h)
    T := ITE(f_x, g_x, h_x)
    E := ITE($f_{x'}$, $g_{x'}$, $h_{x'}$)
    if $T = E$ **then** return(T)
    R := Naci_ili_Dodati_u_Hes_Tabelu(x, T, E)
    return(R)
end

Dakle, funkcija se najpre zapiše u ITE-formi, nakon čega se poziva procedura. Ukoliko je reč o nekom od završnih slučajeva, postupak se odmah prekida. U suprotnom se nalazi prva promenljiva x i rekurzivno poziva ITE-procedura kojom se pronalaze T i E izlazne ivice čvora označenog sa x. Ako te ivice pokazuju na isti podgraf, čvor koji treba označiti sa x se odbacuje i pokazivač na podgraf za T se vraća kao rezultat. U suprotnom, ispituje se da li se podatak za ključ (x, T, E) nalazi u tabeli jedinstvenosti.

[29] Ime operacije ITE je skraćenica od *if-then-else* konstrukcije koju operacija u stvari simulira. $NOTx = ITE(x, 0, 1)$, $xANDy = ITE(x, y, 0) = x \cdot y + x' \cdot 0$, $xORy = ITE(x, 1, y)$, $xXORy = ITE(x, y', y)$, $\ldots$

Ako je podatak pronađen vraća se pokazivač na njega, a u suprotnom se novi podatak smešta u tabelu i vraća pokazivač na njega.

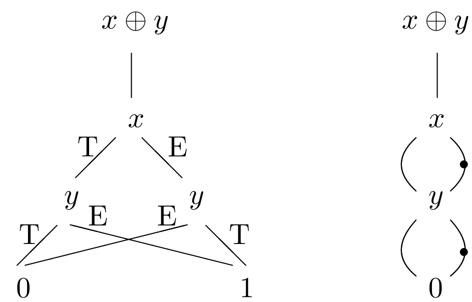
Ovako prezentovan algoritam ima eksponencijalan broj rekurzivnih poziva, a time i vreme izvršavanja, pošto, sem u završnom slučaju, svaki poziv generiše dva nova poziva. Ovaj problem se donekle rešava uvođenjem još jedne tabele u koju se smeštaju već izračunate funkcije. Ta tabela se naziva *tabela izračunatih funkcija*. Uočimo razliku između ove i tabele jedinstvenosti. Kod tabele jedinstvenosti se utvrđuje da li je već konstruisan podgraf određenog oblika, dok se kod tabele izračunatih funkcija utvrđuje da li je već obrađena neka funkcija, dakle provera se vrši pre nego se podgrafovi i generišu. Ako je tabela izračunatih funkcija dovoljno velika, ITE-procedura će biti pozvana po jednom za svaku različitu kombinaciju čvorova f, g i h, pa bismo dobili polinomijalnu složenost izračunavanja. Ovde je nerealna pretpostavka o veličini tabele izračunatih funkcija koja mora biti ograničena, pa se efikasnost postiže njenom pažljivom realizacijom.

Pored ove, postoje i druge metode za podizanje efikasnosti algoritma, ali njihovo izlaganje zbog velikog obima izostavljamo.

BDD sa komplementiranim ivicama

U do sada datim definicijama spominjane ivice u BDD-reprezentacijama funkcija su *regularne*. U cilju efikasnijeg reprezentovanja ponekada se koriste *komplementirane ivice*. Atribut komplementiranja mogu imati izlazne E-ivice čvorova $v \in V$ i izlazne ivice funkcijskih čvorova $f \in \Phi$. BDD sa komplementiranim ivicama je takođe kanonski sistem. Ako ivica ima atribut komplementiranja, njena funkcija je komplement funkcije čvora glave. Na slici 4.3 se to vidi za funkcije $f(a) = a$ i $f(a)' = a'$, a u opštem slučaju nije teško proveriti, da su BDD-reprezentacije za funkciju f i njen komplement f' veoma slične, zapravo da se jedna dobija od druge zamenom vrednosti završnih čvorova. Prema tome, BDD reprezentacija funkcija f i f' može biti data kao jedan BDD za funkciju $F = (f, f')$ sa dve izlazne vrednosti tako da BDD reprezentuje, recimo, funkciju f, a pored funkcijskog čvora za f postoji i funkcijski čvor za f' čija je izlazna ivica komplementirana i ulazi u isti čvor kao i izlazna ivica iz čvora za f. Odavde se konstrukcija komplementa funkcije i provera da li je neka funkcija komplement neke druge funkcije izvode u konstantnom vremenu. Recimo, funkcija g je komplement funkcije f ako im izlazne ivice pokazuju na isti podgraf i jedna je komplementirana, a druga nije.

Primer 4.3.27 Na slici 4.10 date su dve BDD-reprezentacije funkcije ekskluzivne disjunkcije. Prvi BDD ima regularne, a drugi i komplementirane grane označene tačkama. Očigledna je ušteda koju postiže drugi BDD. ■



Slika 4.10. BDD sa običnim i BDD sa komplementiranim granama za $x \oplus y$.

Komentar o metodi BDD

Veličina BDD-reprezentacija je eksponencijalna u odnosu na broj promenljivih u najgorem slučaju, ali se dobro ponaša za mnoge bitne funkcije, recimo BDD sa komplementiranim ivicama za AND, OR ili XOR je polinomijalno veliki u odnosu na veličinu zapisa funkcija. Ovo je značajno za XOR jer se ostvaruje znatna ušteda u odnosu na predstavljanje pomoću NOT, AND i/ili OR. Slično se i komplementiranje efikasno predstavlja upotrebom komplementiranih ivica. Koristeći BDD problem valjanosti se rešava u konstantnom vremenu - proverom da li se redukovani BDD za formulu sastoji samo od završnog čvora 1. Sa druge strane, problem čini to što efikasnost predstavljanja metodom BDD zavisi od izabranog redosleda promenljivih, a nalaženje dobrog redosleda nije uvek lako. Takođe, postoje funkcije koje se kompaktnije prikazuju nekim drugim postupcima i situacije u kojima su neke druge metode reprezentacije bliže konačnoj fizičkoj realizaciji.

5

Izračunljivost, odlučivost i složenost

Rešavanje problema razvojem algoritama[1] i pisanjem programa je jedan od osnovnih zadataka u matematici i računarstvu. Međutim, iako probleme treba rešavati, nisu samo oni mogući predmet razmatranja. Matematičkim sredstvima proučavaju se i sami postupci rešavanja, algoritmi. Formalni model izračunavanja koje ćemo razmatrati biće Tjuringove mašine. Rezultati do kojih se došlo u ovoj oblasti navode na tezu da taj (i svaki njemu ekvivalentan) model izračunavanja upravo određuje granicu mogućnosti mehaničkog izračunavanja. Ta granica razdvaja klase problema na one na za koje, u principu, postoji mogućnost programskog rešavanja i one za koje to nije slučaj, tesno povezujući pojmove izračunljivosti i odlučivosti. Teorija složenosti izračunavanja uvodi klasifikaciju među problemima koji jesu (barem u principu) rešivi, a koja se bazira na potrebnim sredstvima da se dođe do odgovora.

5.1 Izračunljivost

Teorija izračunljivosti (tj. teorija algoritama) je oblast koja je nastala između 1930. i 1940. godine, dakle pre razvoja digitalnih računara, kao rezultat pretresanja osnova matematike zbog paradoksa koji su se pojavili krajem XIX i početkom XX veka. U razmatranju strogog zasnivanja matematike, postavljalo se pitanje[2] da li postoji opšti postupak utvrđivanja

[1] Reč algoritam je nastala od imena persijskog matematičara Abu Džafar Mohamed ibn Musa al-Hovarizmij-a (780 – 850).

[2] Pitanje je poznato pod nemačkim nazivom *Entscheidungsproblem*, tj. problem odlučivanja. Nezavisno jedan od drugog, Čerč i Tjuring su negativno odgovorili na ovo pitanje, svodeći ga na probleme jednakosti λ-izraza, odnosno utvrđivanja da li će

istinitosti matematičkih iskaza. Ovo pitanje vodi poreklo još od Leibnitz-a[3] koji je u XVII veku, nakon uspešne konstrukcije mehaničke računske mašine, razmišljao o konstrukciji mašine koja bi mogla manipulisati simbolima jednog veštačkog univerzalnog jezika i na taj način odrediti istinitost iskaza. Problem je aktuelizovao Hilbert[4], najpre na Kongresu matematičara održanom 1900. godine u poznatom desetom problemu, a zatim zajedno sa Ackermann-om[5] 1928. godine. Da bi se na ovo pitanje moglo odgovoriti bilo je neophodno precizirati šta se podrazumeva pod postupkom mehaničkog izvođenja. Istraživanja i rezultati postignuti u ovoj oblasti imali su presudan uticaj kako na teorijske, tako i na praktične aspekte razvoja računarstva. To se odnosi na principe programibilnog digitalnog računara opšte namene, koncept pisanja programa kao liste naredbi u formalnom jeziku, interpretiranje i prevođenje programa, razvoj programskih jezika uopšte itd.

5.1.1 Intuitivni pojam algoritma

Pojam algoritama spada, poput geometrijskih pojmova kao što su tačka ili prava, u osnovne matematičke koncepte i kao takav se ne definiše. Međutim, sledeći opšti uslovi se, prihvataju kao kriterijumi za nazivanje nekog postupka algoritmom (efektivnom procedurom):

- postupak se opisuje konačnim nizom jednostavnih naredbi[6],
- postoji idealna mašina (računar) koja izvršava te naredbe prema unapred utvrđenim pravilima,

---

se proizvoljna Tjuringova mašina zaustaviti (halting problem).

[3]Gottfried Wilhelm Leibnitz (Lajbnić), 1646 – 1716, matematičar i filozof, verovatno slovenskog porekla. Jedan od kreatora diferencijalnog računa, mislilac koji je išao daleko ispred svog vremena. Njegove zamisli o beskonačno malim i infinitezimalnom računu u punoj meri su razvijene tek u drugoj polovini XX veka.

[4]David Hilbert, 1862 – 1943, nemački matematičar. Smatra se da je jedan od najuticajnijih matematičara svih vremena. Njegovi rezultati pokrivaju mnoge oblasti: od geometrije do funkcionalne analize i fizike. Posebno je značajan po radovima na osnovama matematike, gde je zasnovao pravac poznat pod nazivom formalizam. U oblasti matematičke logike jedan je od osnivača teorije dokaza. U istorijskom predavanju na Kongresu matematičara održanom 1900. godine u Parizu postavio je niz problema koji su obeležili razvoj matematike u 20. veku i doveli do razvoja mnogih novih naučnih oblasti, uključujući i računarstvo.

[5]Wilhelm Friedrich Ackermann, 1896 – 1962, nemački matematičar. Sarađivao je sa Hilbertom u pisanju knjige *Grundzüge der theoretischen Logik* (Principi matematičke logike) u kojoj je predstavljena formalizacija predikatske logika prvog reda i postavljena pitanja o njenoj kompletnosti i odlučivosti (Entscheidungsproblem). U teoriji izračunljivosti je poznat po funkciji nazvanoj po njemu.

[6]Primetimo da algoritmi mogu biti izraženi više ili manje detaljno.

- ta mašina započinje izračunavanje u nekom inicijalnom stanju; primenjena na ulazne podatke mašina izvršava naredbe u diskretnim koracima u kojima menja svoja stanja,
- izvršavanje svake naredbe se izvodi u konačnom vremenu pri čemu se koristi konačan memorijski prostor,
- izvršavanje naredbe je determinističko: iz jednog stanja izvršavanjem iste naredbe mašina uvek prelazi u isto stanje i
- prelaskom u završno stanje mašina prestaje sa izračunavanjem.

Osobina determinisanosti izvršavanja naredbi se drugačije može formulisati kao mogućnost ponavljanja izvršavanja algoritama. Ako ga prihvatimo, postupci koji uključuju slučajnost[7] ne spadaju u algoritme. U pojedinim slučajevima mi ćemo odbaciti ovaj uslov i razmatrati i nedeterminističke algoritme.

Primetimo da se među navedenim uslovima ne nalazi zahtev da se algoritam uvek završi, tj. da se rezultat uvek dobije u konačnom vremenu[8], odnosno da se ne zahteva da se dobije odgovor za sve moguće ulazne podatke, dok se taj zahtev postavlja za svaki pojedinačni korak izvršavanja. Slično je i sa zahtevom za ukupno memorijsko zauzeće. Kao što ćemo u nastavku teksta videti ovakav pristup u teorijskim razmatranjima pruža pogodnosti za elegantno opisivanje formalnih metoda.

Algoritam predstavlja opis funkcije koja ulazne podatke preslikava u odgovor. Funkcije za koje postoje algoritmi zato nazivamo algoritamskim funkcijama (efektivnim funkcijama, izračunljivim funkcijama).

5.1.2 Formalni modeli izračunavanja

Poznat je veliki broj algoritama. Na primer, to su postupak za množenje celih brojeva, tablični metod ispitivanja da li je neka iskazna formula tautologija, Euklidov algoritam nalaženja najvećeg zajedničkog delioca dva broja itd. Za probleme za koje poznajemo postupke rešavanja lako utvrđujemo da jesu algoritamski rešivi. Međutim, kako se napreduje u razvoju matematike, nailazi se na probleme za koje nismo u stanju da damo rešenje. Postavlja se pitanje da li je to samo posledica naše nesposobnosti ili je reč o principijelnoj nemogućnosti. Da bi se na to pitanje odgovorilo potrebno je formalno precizirati pojmove algoritma i izračunljivih funkcija, čime bi

[7] Recimo, postupci u kojima prelazak sa jednog na drugi korak zavisi od dogadjaja kao što su dobijena strana prilikom bacanja novčića. Postupci takvog tipa su nedeterministički.

[8] Ovakav zahtev bi se mogao nazvati konačnost, finitnost, algoritma. Videti s tim u vezi odeljak **??**.

se jasno odredila za sada dosta nejasna ideja o granicama efektivnosti, tj. dosega algoritama.

Problem postojanja efektivnog postupka za utvrđivanje da li proizvoljna diofantovska jednačina $p(x_1, \ldots, x_m) = 0$ ima nenegativna celobrojna rešenja je primer za ovakvu situaciju. U prethodnoj jednačini $p(x_1, \ldots, x_m)$ je polinom sa celobrojnim koeficijentima i promenljivim $x_1, \ldots, x_m$, recimo $x_1^4 x_3 - 3x_2^5 + 6$. Sa jedne strane, nabrajanjem svih m-torki prirodnih brojeva i proverom da li predstavljaju nule polinoma bi se, pre ili posle, stiglo do rešenja jednačine, ako ono postoji. Međutim, kako neke jednačine ovog tipa, recimo $x^2 - 2 = 0$, nemaju rešenja, prethodno opisani postupak se u takvim slučajevima ne bi nikada završio, zbog čega i ne predstavlja rešenje problema. Provera postojanja rešenja diofantovskih jednačina je zapravo ekvivalentna formulacija desetog Hilbertovog problema. Primetimo da svaki eventualni odgovor na ovo pitanje mora na neki način ponuditi i formalnu definiciju onoga što se podrazumeva pod efektivnim postupkom, bilo u smislu da ponuđeno rešenje potpada pod tu definiciju, bilo da ne postoji rešenje sa zahtevanim svojstvima. Formalna definicija efektivnog postupka pojavila se razvojem teorije algoritama, dok je Matijaševič[9] 1970. godine sredstvima razvijenim u okviru te teorije negativno rešio sam problem, o čemu će biti reči u odeljku 5.4.

U razvoju teorije algoritama ponuđeno je više pristupa formalizaciji ovih granica:

- Sistem izračunljivosti predstavljen u formalnom sistemu aritmetike je predložio Gedel[10] između 1931. i 1934. godine, pri čemu se funkcija f smatra izračunljivom ako za svako m i n za koje je $f(m) = n$, u formalnom sistemu važi $\vdash f(m) = n$.

- Prikazivanje izračunljivih funkcija kao jedinstvenih rešenja sistema funkcionalnih jednačina je u istom periodu opisao takođe Gedel, a prema ideji Erbrana[11].

- λ-račun koji je razvio Čerč[12] do 1936. godine je jednostavan formalni

[9] Юрий Матиясевич, ruski matematičar, rođen 1947. godine.

[10] Kurt Gödel, 1906 – 1978, austrijski logičar. Najpoznatiji je po rezultatima o esencijalnoj nepotpunosti aksiomatizacije Peanove aritmetike koji su imali dalekosežne posledice po razvoj ljudske misli. Zajedno sa A. Turing-om uvršten je u listu 100 najznačajnijih ličnosti 20. veka koju je 1999. godine objavio magazin TIME [19].

[11] Jacques Herbrand, 1908 – 1931, francuski matematičar. U kratkoj karijeri dao je fundamentalne priloge u oblasti izračunljivih funkcija, kao i logici, gde je tvrđenje nazvano po njemu postalo osnova za rad automatskih dokazivača teorema, poput onih zasnovanih na rezoluciji. Poginuo na planinarenju.

[12] Alonzo Church, 1903 –1995, američki matematičar. Najpoznatiji po doprinosima u logici i osnovama teorijskog računarstva. Uveo je λ-račun i formulisao čuvenu hipotezu

jezik za koji se definiše pojam redukcije koji predstavlja izračunavanje, a funkcija je izračunljiva ako se može opisati u jeziku.

- Aritmetički opis, takozvane parcijalno rekurzivne funkcije, zasnovao je Klini[13] takođe do 1936. godine, a baziran je na generalisanom pojmu definisanja indukcijom.

- Sistemi zasnovani na automatima, među kojima su:

  - Tjuringove[14] mašine iz 1936. godine,
  - Postova[15] mašine predstavljena takođe 1936. godine,
  - Neograničena registarska mašina[16] koju su Šeferdson i i Stargis[17] opisali 1963. godine,

  formalizuju pojam algoritma opisujući idealne modele računara[18]. Zanimljivo je da su neki sistemi dati pre nastanka digitalnih računara.

- Sistemi produkcija (nekad se nazivaju i sistemi sa prezapisivanjem[19]), među kojima su:

  - Postovi sistemi iz 1943. godine,
  - Markovljevi[20] algoritmi uvedeni 1954. godine i
  - Gramatike Čomskog[21] predložene 1956. godine,

nazvanu po njemu. Pokretač je i dugogodišnji urednik časopisa Journal of Symbolic Logic. Bio je mentor velikom broju kasnijih istaknutih logičara, kao A. Turing-u, M. Davis-u, L. Henkin-u, S. Kleene-ju, M. Rabin-u, D. Scott-u, R. Smullyan-u itd.

[13] Stephen Kleene, 1909 – 1994, američki matematičar. Smatra se jednim od osnivača teorije rekurzivnih funkcija.

[14] Alan Turing, 1912 – 1954, engleski logičar. Zaslužan za nastanak i razvoj računarstva. Razvio ideje formalizacije koncepta izračunavanja i računara kao univeralne mašine [20]. U radu [21] postavio osnove veštačke inteligencije. Tokom Drugog svetskog rata je rukovodio razbijanjem nemačke šifarske mašine Enigma, za šta je razvio elektro-mehaničku mašinu nazvanu Bombe 1940. godine, i dao osnovne ideje za elektronsku mašinu pod nazivom Colossus, koji je nastao 1942. godine kao prvi programibilni elektronski računar. Zajedno sa K. Gödelo-om uvršten je u listu 100 najznačajnijih ličnosti 20. veka koju je 1999. godine objavio magazin TIME [19].

[15] Emil Leon Post,1897 – 1954, matematičar poljskog porekla. Najpoznatiji je po radu u oblasti teorije izračunljivost.

[16] Engleski: Unlimited Register Machine, URM.

[17] John C. Shepherdson, Howard E. Sturgis, [18].

[18] Bez obzira na upotrebu reči *mašina* koja se javlja u nazivima ovih formalizama, uvek treba imati u vidu da se ovde radi o apstraktnim matematičkim konceptima, a ne realnim fizičkim objektima.

[19] Rewriting systems.

[20] Андрей Андреевич Марков, 1903 – 1979, ruski matematičar. Jedan je od osnivača ruske škole konstruktivne matematike.

[21] Avram Noam Chomsky, rođen 1928. godine, čuveni liberalni mislilac i aktivista. Poznat je kao osnivač moderne lingvistike.

su jedna vrsta formalnih sistema u kojima se opisuju moguće transformacije (pravila izvođenja) jednih u druge reči na unapred fiksiranom alfabetu. Funkcije se opisuju kao skupovi parova reči (u, v) za koje postoji niz reči koje se dobijaju počev od u primenama pravila izvođenja i koji završava rečju v.

- *while*-programi su vrsta notacije proizašle iz ideja Goldstine-a (Goldštajna) i von Neumann-a[22] o algoritamskim šemama[23] kao formalizmu za prikazivanje izračunljivih funkcija. *while*-programi se sastoje samo od naredbi dodeljivanja, nizanja naredbi i *while*-naredbi.

Njihovi izvori inspiracije se međusobno značajno razlikuju, ali se pokazuje da su sistemi međusobno ekvivalentni. Ovo, kao i neuspeh pokušaja konstrukcije zadatka i postupka njegovog rešavanja koji ne potpadaju pod ove klasifikacije daje za pravo verovanju da je postignut nekakav apsolutni koncept i da se svi algoritmi mogu izraziti u svakom od ovih sistema, što je formulisano Čerčovom tezom koja se razmatra u odeljku 5.3.

5.2 Tjuringova mašina

5.2.1 Model mašine za izračunavanje

Digitalni računar se na apstraktnom nivou obično prikazuje kao celina sastavljena od procesora, memorije i ulazno-izlaznih uređaja. Procesor iz memorije pribavlja naredbe i podatke nad kojima se vrši obrada u skladu sa značenjem naredbi, a dobijeni rezultati se vraćaju u memoriju. Preko ulazno-izlaznih uređaja podaci koji će biti obrađeni se unose u memoriju, odnosno iz memorije se preuzimaju rezultati obrade i prikazuju na odgovarajući način. Komunikacija delova računara se obavlja preko magistrala.

Tjuringova mašina je preteča ovakvog modela računara, pri čemu su neka svojstva idealizovana. To se pre svega odnosi na memoriju za koju se pretpostavlja da je potencijalno beskonačna. Preciznije, na početku izvršavanja

[22] János-a Neumann, John von Nojman, 1903 – 1957, matematičar mađarskog porekla. Jedan od najuticijnijih matematičara 20. veka. Veliki doprinos je dao u mnogim oblastima: teoriji skupova, funkcionalnoj analizi, teoriji igara i ekonomiji, kao i u fizici, posebno u kvantnoj mehanici. Istaknuti učesnik projekta Manhattan koji je doveo do stvaranja atomske bombe. Jedan je od prvih članova Institute for Advanced Study u Princeton-u. Bio je savetnik u timu u Moore School of Electrical Engineering na University of Pennsylvania koji je realizovao projekt EDVAC (Electronic Discrete Variable Automatic Computer, 1944–1946) za potrebe Balističke laboratorije američke vojske. Izveštaj pod nazivom First Draft of a Report on the EDVAC iz 1945. godine koji je potpisao von Neumann, ali za čije ko-autorstvo su se borili i drugi istraživači sa projekta (J. Eckert i J. Mauchly) sadrži opis logički načela koje su u osnovi arhitekture i današnjih računara, poznatih kao von Neumann-ovi principi.

[23] Flowchart.

Tjuringove mašine zauzet je samo konačan broj memorijskih registara, a isto važi i u svakom koraku izračunavanja. Ne postoji ograničenje koliki je taj konačan broj registara. U svakom koraku izračunavanja moguće je i zahtevati novi, do tada neiskorišteni memorijski registar i svaki takav zahtev se ispunjava. Sa druge strane, Tjuringova mašina je restrikcija koncepta savremenog računara u smislu operacija koje je u stanju da izvršava, a koje su elementarne. Kako ćemo videti, zanimljivo je da su te operacije ipak dovoljne za opisivanje proizvoljnih algoritama. Njihova prednost u odnosu na bogatije programske jezike je upravo u jednostavnosti koja olakšava analizu.

U teorijskom računarstvu se, inače, razmatraju i druge, slabije, klase mašina koje su restrikcije druge vrste u odnosu na aktuelne računare: neki modeli nemaju memoriju (konačni automati[24]) ili je memorija organizovana na poseban način (stek kod potisnih automata[25]), ulazno-izlazni podaci su ograničeni na reči[26], neki čak nemaju izlaz (konačni automati).

5.2.2 Alfabet

Svaki problem se izražava u nekom jeziku. Alfabet je skup znaka koji su nedeljive celine. Reč na nekom alfabetu je bilo koja konačna sekvenca znaka tog alfabeta. Sekvenca od nula znaka se naziva prazna reč. Reči se razdvajaju znakom blanko koji se ne smatra delom alfabeta već pomoćnim simbolom. Jezik je neki podskup skupa svih reči odgovarajućeg alfabeta. Reč t je podreč reči q ako postoje, možda i prazne, reči u i v tako da je $q = utv$.

Obično je alfabet konačan skup znaka, jer sve što se može iskazati beskonačnim prebrojivim alfabetom $\{a_1, a_2, \ldots\}$ može se iskazati i najjednostavnijim, unarnim, alfabetom $A = \{1\}$. Reči ovog alfabeta: 1, 11, 111, $\ldots$ se mogu identifikovati sa znacima proizvoljnog beskonačnog alfabeta. U nastavku teksta koji se odnosi na izračunljivost i odlučivost, sem ako se posebno ne naglasi, koristićemo unarni alfabet A=$\{1\}$. Pored simbola 1 koristićemo i blanko-znak za čije označavanje ćemo zbog preglednosti upotrebljavati znak 0. Kasnije, kada se bude govorilo o složenosti izračuvanja, koristićemo binarni alfabet $\{0, 1\}$.

5.2.3 Opis Tjuringove mašine

U ovom odeljku daćemo takozvani neformalni opis Tjuringove mašine. Pored njega moguće je ove automate i strogo formalno opisati, kao matematičke strukture što je od koristi u dokazivanju nekih tvrđenja o Tjuringovim

[24] Finite automata.
[25] Push-down automata.
[26] String.

mašinama koje prevazilaze opseg ovog materijala. (Neformalno,) Tjuringova mašina se sastoji od:

- *trake* podeljene u ćelije, memorijske registre, koja se neograničeno pruža na levo i desno; broj ćelija (tj. dužina trake) je neograničen; sadržaj svake ćelije je ili znak 1 ili blanko znak (znak 0),
- *glave* koja se uvek nalazi nad tačno jednom ćelijom trake i može:
  - pročitati sadržaj ćelije nad kojom se nalazi i
  - upisati u ćeliju nad kojom se nalazi znak 1 ili 0 (blanko znak, tj. obrisati ćeliju) ili pomeriti se za jedan korak u levo ili u desno u odnosu na trenutnu poziciju,
- *indikatora stanja mašine.*

Tjuringova mašina se u svakom trenutku nalazi u tačno jednom od konačno mnogo stanja koje se eventualno menja nakon svakog koraka izračunavanja. Skup svih stanja mašine označićemo sa $S = \{q_0, q_1, \ldots\}$. Izvršavanje mašine se izvodi pod dejstvom programa koji čini neki konačan niz naredbi. Svaka naredba je četvorka oblika:

$q_i \ s \ o \ q_j$

gde su q_i i q_j neka stanja iz skupa S, s je znak nad kojim se nalazi glava mašine, a $o \in \{1, 0, L, R\}$ je oznaka operacije. U svakom koraku rada mašina analizira stanje u kojem se nalazi i sadržaj ćelije nad kojom je glava, a zatim izvršava naredbu koja ima odgovarajuće vrednosti parametara q_i i s. Efekat izvršenja naredbe je dvojak. Najpre se, u zavisnosti od vrednosti parametra o obavi:

- ako je $o = 1$, u ćeliju nad kojom se nalazi glava upisuje se znak 1,
- ako je $o = 0$, u ćeliju nad kojom se nalazi glava upisuje se 0, tj. blanko znak,
- ako je $o = L$, glava se pomera ulevo za jednu ćeliju i
- ako je $o = R$, glava se pomera udesno za jednu ćeliju.

Potom mašina menja stanje i prelazi u stanje q_j.

Primeri naredbi su:
q_5 0 1 q_{17},
q_1 0 0 q_2 i
q_0 1 L q_0.
U prvoj naredbi, ako se mašina nalazi u stanju q_5, a glava nad znakom blanko, u ćeliju se upisuje znak 1 i prelazi u stanje q_{17}. U drugoj naredbi,

ako se mašina nalazi u stanju q_1, a glava nad znakom blanko, u ćeliju se upisuje blanko znak i prelazi u stanje q_2. Ovakva naredba služi samo za promenu stanja mašine. U trećoj naredbi, ako se mašina nalazi u stanju q_0, a glava nad znakom 1, glava se pomera ulevo, a mašina ostaje u istom stanju.

Primetimo da, ako se želi da mašina radi deterministički, program sme sadržati samo jednu naredbu za svaku kombinaciju stanja q_i i sadržaja s ćelije nad kojom je glava. Na primer, u jednom programu se ne smeju pojaviti sledeće naredbe:

q_4 1 1 q_5 i
q_4 1 L q_2

jer im se vrednosti parametara q_i i s poklapaju, a vrednosti parametara o i q_j razlikuju. U slučaju nedeterminističkih mašina ovaj zahtev ne postoji.

U vezi sa Tjuringovom mašinom prihvatićemo sledeće konvencije. Stanje $q_0 \in S$ nazvaćemo *početnim stanjem*. Inicijalno, mašina se uvek nalazi u početnom stanju. Pri tome traka sadrži samo konačno mnogo ćelija u koje je upisan znak 1, dok sve ostale ćelije sadrže znak 0. Reč se na traci prikazuje kao neprekidan niz ćelija koje sadrže znak 1, a sa leve i desne strane tog niza se nalazi najmanje po jedan blanko znak, tj. znak 0. Po pravilu, na početku i na kraju izvršavanja glava mašine se nalazi iznad najlevlje ćelije koja sadrži znak 1. Skup stanja S proširićemo jednim novi stanjem q_z koje ne pripada do sada razmatranom skupu stanja. Stanje q_z nazvaćemo *završnim stanjem*. Kada se mašina nađe u stanju q_z ona prekida sa izvršavanjem.

O Tjuringovoj mašini možemo razmišljati na dva načina:

- kao o jedinstvenoj mašini koja izvršava sve programe (u smislu savremenog računara, o čemu će kasnije biti viče reči u odeljku 5.2.7) i

- kao o posebnoj mašini za svaki program u kom slučaju se pojam mašine poistovećuje sa pojmom programa, tj. svaki program predstavlja posebnu mašinu

koja se u suštini međusobno ne razlikuju.

Definicija 5.2.1 Pod *konfiguracijom* Tjuringove mašine podrazumevamo opis koji sadrži: opis sadržaja trake, položaj glave i stanje mašine. *Standardna konfiguracija* je konfiguracija u kojoj je:

- ili traka prazna (tj. sve ćelije sadrže blanko znak) ili sadrži najviše konačno mnogo nepraznih reči razdvojenih po jednim blanko znakom,

- glava mašine je iznad prve (gledano sa leva) ćelije trake koja sadrži znak 1 i

$\ldots 01q_0 1000\ldots$	$\ldots 0111q_3 10\ldots$
$\ldots 011q_0 000\ldots$	$\ldots 011q_3 110\ldots$
$\ldots 0110q_0 00\ldots$	$\ldots 01q_3 1110\ldots$
$\ldots 0111q_1 00\ldots$	$\ldots 0q_3 11110\ldots$
$\ldots 01110q_2 0\ldots$	$\ldots 01q_z 1110\ldots$
$\ldots 01111q_3 0\ldots$	

Slika 5.1. Izvršavanje Tjuringove mašine iz primera 5.2.2.

- ako počinje sa izvršavanjem, mašina se nalazi u početnom stanju q_0, a ako završava sa radom u završnom stanju q_z. ■

Sada se programi mogu shvatiti kao funkcije koje preslikavaju skup konfiguracija mašine u samog sebe.

Primer 5.2.2 Neka je na traci data samo jedna reč sastavljena od jedinica (a sve ostale ćelije sadrže znak 0) nad čijim krajnjim levim znakom se nalazi glava. Sledeći program dopisuje dva znaka 1 sa desne strane reči, a zatim se glava vraća na levo, na početak reči, nakon čega mašina staje:

$q_0\ 1\ R\ q_0$	glava se pomera udesno, na kraj reči
$q_0\ 0\ 1\ q_1$	na mestu prve 0 upisuje se 1
$q_1\ 1\ R\ q_2$	glava se pomera udesno
$q_2\ 0\ 1\ q_3$	na mestu druge 0 upisuje se 1
$q_3\ 1\ L\ q_3$	glava se pomera ulevo
$q_3\ 0\ R\ q_z$	do prve 0, ide udesno i zaustavlja se

Izvršavanje ove Tjuringove mašine, pod pretpostavkom da je traka na početku sadržala binarnu reč 11, prikazano je na slici 5.1. Pozicija oznake stanja (q_i) na slici predstavlja položaj glave trake, tj. glava je iznad ćelije u kojoj se nalazi cifra levo od oznake stanja. ■

Primetimo da se u opisu Tjuringove mašine ne kaže šta se događa ako za sadržaj ćelije nad kojim se nalazi glava i tekuće stanje mašine u programu ne postoji odgovarajuća naredba. Ova situacija bi odgovarala 'zaglavljivanju' programa pisanih na standardnim programskim jezicima i može se formalizovati kompletiranjem programa naredbama koje u takvim situacijama ne menjaju ni stanje, ni poziciju glave, ni sadržaj ćelije nad kojom se glava nalazi. Recimo, ako u programu ne postoji naredba koja odgovara situaciji kada je mašina u stanju q_0, a sadržaj ćelije nad kojom se nalazi glava 0, možemo program proširiti naredbom:

$q_0\ 0\ 0\ q_0$

koja predstavlja jednu beskonačnu petlju. S obzirom na ovakvu mogućnost, na dalje nećemo voditi računa da program bude u opisanom smislu kompletan.

5.2.4 Tjuringove mašine i funkcije

U ovom odeljku ćemo opisati kako se Tjuringove mašine mogu iskoristiti kao algoritmi, tj. za izračunavanje funkcija koje preslikavaju prirodne brojeve u prirodne brojeve.

Definicija 5.2.3 *Aritmetička funkcija* je preslikavanje f za koje važi:

- domen preslikavanja, $Dom(f)$, je podskup skupa $\mathbb{N}^k$ $(k > 0)$ i
- kodomen preslikavanja, $Im(f)$, je podskup skupa $\mathbb{N}$.

Ako je za neki $k > 0$, $Dom(f) = \mathbb{N}^k$, f je *totalna funkcija*. Ako je $Dom(f) \subset \mathbb{N}^k$, za neki $k > 0$ i $Dom(f) \neq \mathbb{N}^k$, f je *parcijalna funkcija*. ■

Definicija 5.2.4 *Unarna reprezentacija prirodnog broja* n u unarnom alfabetu $A = \{1\}$ je reč koja sadrži $n+1$ znak 1. ■

Definicija 5.2.5 Neka je f aritmetička funkcija oblika $f : X \to \mathbb{N}$, gde je $X \subset \mathbb{N}$. Funkcija f je *Tjuring-izračunljiva* ako postoji program P za Tjuringovu mašinu tako da je za svaki $m \in X$:

- pre početka izvršavanja programa P Tjuringova mašina u standardnoj konfiguraciji, pri čemu je jedina reč zapisana na traci unarna reprezentacija broja m i
- po završetku rada programa P Tjuringova mašina u standardnoj konfiguraciji, pri čemu je jedina reč zapisana na traci unarna reprezentacija broja $f(m)$.

Program P tada *izračunava* funkciju f. ■

Primetimo da su prema definiciji 5.2.5 Tjuring-izračunljive funkcije parcijalne, odnosno ako se neki m ne nalazi u domenu Tjuring-izračunljive funkcije f, odgovarajući program P ne staje.

Primer 5.2.6 Sledeći program:

q_0 0 0 q_0
q_0 1 1 q_0

nikada ne staje, pa izračunava jedino funkciju čiji je domen prazan skup. ■

Analogno definiciji 5.2.5 moguće je definisati k-arne aritmetičke Tjuring-izračunljive funkcije. Jedina razlika je u tome što početna standardna konfiguracija mašine odgovara traci na kojoj je prikazano k argumenata funkcije.

Sa $P(x_1, x_2, \ldots, x_k) \downarrow y$ označavamo da program P polazeći od standardne konfiguracije u kojoj traka sadrži unarne reprezentacije prirodnih brojeva x_1, x_2, ..., x_k završava rad pri čemu se mašina nalazi u standardnoj konfiguraciji u kojoj traka sadrži unarnu reprezentaciju prirodnog broja y. Oznaka $P(x_1, x_2, \ldots, x_k) \downarrow$ znači da je za neko y ispunjeno $P(x_1, x_2, \ldots, x_k) \downarrow y$. Oznaka $P(x_1, x_2, \ldots, x_k) \uparrow$ znači da nije $P(x_1, x_2, \ldots, x_k) \downarrow$.

Definicija 5.2.7 Program P *konvergira* za ulaz x_1, x_2, ..., x_k ako je ispunjeno $P(x_1, x_2, \ldots, x_k) \downarrow$. Program P *divergira* za ulaz x_1, x_2, ..., x_k ako je ispunjeno $P(x_1, x_2, \ldots, x_k) \uparrow$. ■

Sledi nekoliko primera programa i Tjuring-izračunljivih funkcija o kojima će biti reči u kasnijim odeljcima.

Primer 5.2.8 Sledeći program izračunava funkciju $f(x) = 0$.

q_0 1 0 q_1
q_0 0 1 q_z
q_1 0 R q_0

Sadržaj trake se briše, pri čemu se glava pomera na desno. Kada se naiđe na prvi znak 0, upisuje se znak 1 i završava rad. Dakle, $P(x) \downarrow 0$. ■

Primer 5.2.9 Sledeći program izračunava funkciju naslednika prirodnog broja u nizu prirodnih brojeva, $f(x) = x'$.

q_0 1 L q_0
q_0 0 1 q_z

U programu se glava najpre pomera na levo, nakon čega se nalazi iznad ćelije koja sadrži znak 0. U tu ćeliju se upisuje znak 1 i prelazi u završno stanje. Dakle, $P(x) \downarrow x'$. ■

Primer 5.2.10 Sledeći program za fiksirane k i i ($k \geq i \geq 1$) izračunava funkciju koja se naziva i-ta projekcija, $f(x_1, \ldots, x_k) = x_i$.

$q_0\ 1\ 0\ q_1$	briše zapisa broja x_1
$q_0\ 0\ R\ q_2$	
$q_1\ 0\ R\ q_0$	
...	
$q_j\ 1\ 0\ q_{j+1}$	briše zapisa broja x_{i-1}
$q_j\ 0\ R\ q_{j+2}$	
$q_{j+1}\ 0\ R\ q_j$	
$q_{j+2}\ 1\ R\ q_{j+2}$	prelazi zapis broja x_i
$q_{j+2}\ 0\ R\ q_{j+3}$	
$q_{j+3}\ 1\ 0\ q_{j+4}$	briše zapisa broja x_{i+1}
$q_{j+3}\ 0\ R\ q_{j+5}$	
$q_{j+4}\ 0\ R\ q_{j+3}$	
...	
$q_l\ 1\ 0\ q_{l+1}$	briše zapisa broja x_k
$q_l\ 0\ L\ q_s$	
$q_{l+1}\ 0\ R\ q_l$	
$q_s\ 0\ L\ q_s$	vraća se na početak zapisa broja x_i
$q_s\ 1\ L\ q_{s+1}$	
$q_{s+1}\ 1\ L\ q_{s+1}$	
$q_{s+1}\ 0\ R\ q_z$	

Na početku izvršavanja unarne reprezentacije brojeva $x_1, \ldots, x_k$ su na traci razdvojene jednim blanko znakom. Glava se najpre pomera do kraja zapisa broja x_{i-1} i pri tom briše sve jedinice, zatim prelazi preko zapisa broja x_i i ponovo briše zapise brojeva $x_{i+1}, \ldots, x_k$. Konačno, mašina se vraća na početak zapisa broja x_i i staje. U programu je dato rešenje u kojem se podrazumeva da je $k > i > 1$, ali se on jednostavno prerađuje za preostale slučajeve. ∎

5.2.5 Tjuring-neizračunljive funkcije

Definicijom 5.2.5 povezana je jedna klasa funkcija nazvanih Tjuring-izračunljivim sa programima za Tjuringovu mašinu. Kako je svaki program konačan niz naredbi, a svaka naredba konačan niz simbola iz nekog prebrojivog skupa, to postoji samo prebrojivo mnogo programa. Kako svih aritmetičkih funkcija ima neprebrojivo mnogo, to znači da postoje funkcije koje nisu Tjuring-izračunljive. Prethodno obrazloženje je u stvari dokaz sledećeg tvrđenja:

Teorema 5.2.11 Tjuring-izračunljivih funkcija ima prebrojivo mnogo. Postoje funkcije koje nisu Tjuring-izračunljive. ∎

5.2.6 Varijante Tjuringove mašine

Ovde izabran pristup u definisanju Tjuringove mašine je samo jedan od mogućih. Recimo, posmatraju se Tjuringove mašine u kojima:

- alfabet kojim se zapisuje sadržaj ćelija trake ne mora biti unarni,
- pored završnog stanja q_z uvode se i neka specijalna završna stanja, recimo q_{da} i q_{ne} koja, intuitivno, znače pozitivan, odnosno, negativan odgovor na postavljeni problem,
- dozvoljena je traka koja je beskonačna samo na jednu stranu, tj. postoji krajnja leva ćelija, dok se na desno traka pruža neograničeno,
- umesto samo jedne postoji više traka, a za svaku traku postoji posebna glava,
- nad jednom trakom postoji više glava umesto samo jedne,
- traka je dvodimenzionalna, a ne jednodimenzionalna, tj. traka podseća na beskonačnu šahovsku ploču,
- u jednoj naredbi mašine moguće je i upisati znak u ćeliju i pomerati glavu,
- ne važi zahtev za determinisanošću, tj. dozvoljeno je da postoje naredbe koje odgovaraju istom stanju i znaku u ćeliji nad kojom se nalazi glava, a koje se razlikuju po dejstvu (operaciji koja se izvršava i/ili stanju u koje se prelazi) itd.

Zanimljivo je da u smislu izračunljivosti gotovo sve od ovih varijanti Tjuringove mašine odgovaraju istoj klasi funkcija, tj. klasi Tjuring-izračunljivih funkcija, kao i osnovna verzija mašine. Izuzetak predstavljaju neki slabiji, restriktivni slučajevi: recimo, mašina čija traka je ograničena sa jedne strane i koristi unarni alfabet ili mašina koja ima samo dva stanja i koristi alfabet od dva znaka. Ekvivalencija varijanti Tjuringove mašine se dokazuje tako što se pokaže da za svaki program P za neku od varijanti Tjuringove mašine postoje programi za preostale varijante koji simuliraju izvršenje programa P i izračunavaju istu funkciju. Skiciraćemo neke od ovih postupaka.

Izbor varijante Tjuringove mašine zavisi od primene kojom se bavimo. Recimo, u analizi složenosti algoritama se koristi više varijanti mašina zavisno od klase složenosti koja se proučava.

Tjuringova mašina sa bogatijim alfabetom

Kao što je napomenuto u odeljku 5.2.2 reči bilo kog prebrojivog alfabeta se mogu prikazati pomoću unarnog alfabeta, tako da se u slučaju Tjuringove mašine sa trakom koja nije ograničena ni sa jedne strane i u većini drugih slučajeva alfabet može, zavisno od potrebe, slobodno određivati. Na primer, kada u odeljku 5.5 bude analizirana složenost algoritama, prirodni brojevi će biti dati u binarnoj, a ne kao do sada u unarnoj, reprezentaciji. Ostvarena ušteda će biti značajna pošto je unarna reprezentacija prirodnih brojeva eksponencijalno duža od binarne.

Tjuringova mašina sa trakom koja ima početak sa leve strane

Alfabet kod mašina ove vrste sadrži još jedan specijalni znak, recimo $\triangleright$, koji služi da se prepozna početna ćelija sa leve strane. Preko tog simbola se nikada ne prepisuje ni jedan drugi simbol, niti se glava sme pomeriti levo, tako da je jedina moguća naredba kada je znak $\triangleright$ u ćeliji ispod glave oblika:

$q_i \triangleright R\ q_j$

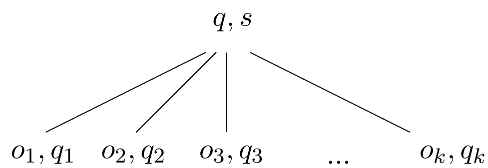
za neka stanja q_i i q_j. Očigledno je da se sve funkcije koje se izračunavaju pomoću Tjuringove mašine čija traka ima početak sa leve strane mogu izračunati i pomoću standardne varijante mašine: jednostavno se neće koristiti deo trake koji se nalazi sa leve strane ćelije iznad koje je pozicionirana glava pre početka izvršavanja. Takođe, važi i obrnuto: za svaku Tjuringovu mašinu M_1 koja na traci neograničenoj u oba smera izračunava neku funkciju f može se konstruisati mašina M_2 čija traka ima početak sa leve strane.

Tjuringova mašina sa više traka

Pretpostavićemo da je ova varijanta mašina organizovana na sledeći način. Za neki $k > 0$ Tjuringova mašina sa k traka sastoji se od traka označenih sa 1, 2, 3, $\ldots$, k. Sve trake imaju kraj sa leve strane, a nad svakom trakom nalazi se posebna glava. U svakom koraku čita se tekuća ćelija na svakoj od traka i preduzima odgovarajuća akcija, tj. neke glave upisuju znak, neke se pomeraju levo, a neke desno. Na početku izvršavanja ulazni podaci se smeštaju na prvu traku, dok su sve ostale trake prazne. Ako izvršavanje mašine shvatimo kao izračunavanje neke funkcije, rezultat se smešta u poslednju, k-tu traku.

U nekim situacijama mašina sa više glava olakšava programiranje. Takav slučaj je, recimo, sa ispitivanjem da li je neka reč palindrom[27]. Najpre se reč iz prve trake iskopira na drugu, glava prve trake se vrati na levo, dok

[27]Reč je palindrom ako se čitanjem sa leva na desno i sa desna na levo dobija isti niz znaka.



Slika 5.2. Nedeterministički korak u izvršavanju Tjuringove mašine.

glava druge ostaje u krajnjoj desnoj poziciji. Konačno, dve glave se kreću u suprotnim smerovima i ispituju da li se nalaze nad jednakim znacima.

Može se pokazati da važi da da svaku Tjuringovu mašinu sa k traka koja izračunava neku funkciju f postoji Tjuringova mašina sa jednom trakom koja simulira njeno izvršavanje. Pri tome važi sledeća teorema:

Teorema 5.2.12 Za datu determinističku Tjuringovu mašinu M_1 sa k-traka može se konstruisati deterministička Tjuringova mašina M_2 sa jednom trakom koja simulira rad mašine M_1. Dužina rada mašine M_2 je ograničena polinomijalnom funkcijom dužine rada mašine M_1. ■

Nedeterministička Tjuringova mašina

U komentaru u odeljku 5.2.3, ponašanje do sada korištenih verzija Tjuringove mašine je okarakterisano kao determinističko, tj. za svaku kombinaciju tekućeg stanja i znaka bila je predviđena samo jedna akcija. Kod *nedeterminističke Tjuringove mašine* ovaj zahtev ne postoji, odnosno

- za tekuće stanje i simbol u ćeliji iznad koje se nalazi glava mašine, može postojati više različitih operacija i/ili stanja u koja mašina prelazi nakon izvršavanja naredbe programa.

U izvršavanju nedeterminističke Tjuringove mašine postoji svojevrsna mogućnost izbora: u slučaju da za neko stanje q neki znak s postoji više mogućih naredbi treba izabrati neku od njih i nastaviti izvršavanje, što je šematski prikazano kao deo jednog drveta na slici 5.2. Grananje u drvetu je konačno, što znači da u svakom koraku izvršavanja postoji samo konačno mnogo opcija za izbor, dok grane predstavljaju moguće redoslede izvršavanja programa.

Nedeterminističke mašine su pre svega pogodne za davanje odgovora 'da' ili 'ne' na pitanja oblika 'da li za ulazne podatke važi ...?' Imajući u vidu ideju o uvođenju novih stanja q_{da} i q_{ne} zaustavljanje u nekom od ovih stanja ima značenje pozitivnog, odnosno negativnog, odgovora. Snaga, odnosno na jeziku savremenih računara - brzina, nedeterminističkih Tjuringovih mašina je posledica sledeće asimetrične konvencije:

- mašina potvrdno odgovara na pitanje ako se bar jedno od mogućih izračunavanja završava u stanju q_{da}, dok jedino okončanje svih mogućih izračunavanja u stanju q_{ne} znači da je odgovor 'ne' i

- ako ni jedno izračunavanje ne dovodi do stanja q_{da} i bar jedno izračunavanje ne dovodi ni do kog završnog stanja, nedeterministička mašina divergira.

Na osnovu ovog dogovora, nedeterministička mašina se može zamisliti kao višeprocesorski sistem koji se ponaša na sledeći način. U svakom koraku svaki od procesora kreira onoliko novih procesora koliko ima različitih konfiguracija u koje taj procesor može preći izvršavanjem tekuće naredbe. Ako mu u nastavku izvršavanja bilo koji od njegovih potomaka vrati informaciju o potvrdnom odgovoru, procesor tu informaciju prosleđuje svom neposrednom pretku. Negativan odgovor se prosleđuje samo ako je dobijen od svih neposrednih potomaka. Zapravo, svaki procesor izračunava disjunkciju odgovora svojih potomaka.

Ovakav model mašine je pogodan za rešavanje nekih složenih problema, o čemu će više reči biti u odeljku 5.5. Na primer, pretpostavimo da želimo ispitati da li je neki prirodan broj n složen ili prost. Običnom Tjuringovom mašinom problem bi se mogao rešiti na sledeći način: delili bismo broj svim prirodnim brojevima između 2 i $\frac{n}{2}$ i na osnovu toga dali odgovor. U slučaju nedeterminističke Tjuringove mašine na jednom mestu bismo imali mogućnost izbora broja kojim delimo broj n, pa ako je n složen, a izabrani broj delilac, mogli bismo dati odgovor u jednom koraku, što bi bio značajan dobitak u odnosu na deterministički postupak. Lako je uočiti da ovaj postupak nije realan, u smislu da izbor delioca podrazumeva da mi već znamo da je n složen, tj. da nam je poznat bar jedan njegov činilac. Međutim, i pored toga, nedeterministička Tjuringova mašina se može simulirati determinističkom mašinom, tako da se izražajnost u smislu onoga šta mašina može odgovoriti ne menja.

Pretpostavimo da je M_1 nedeterministička Tjuringova mašina. Odgovarajuća deterministička Tjuringova mašina M_2 će sistematski prelaziti sve moguće redoslede izvršavanja mašine M_1, najpre dužine 1, pa dužine 2 itd[28]. Ovo obezbeđuje da ni jedno moguće konačno izvršavanje neće biti preskočeno. Zato, ako bi se mašina M_1 u nekom trenutku izvršavanja našla u stanju q_{da}, to isto će pre ili posle biti slučaj i sa mašinom M_2. Ako svi mogući redosledi izvršavanja mašine M_1 dovode do stanja q_{ne}, i mašina M_2 će se naći u tom stanju kada iscrpi sve mogućnosti. Konačno ako mašina M_1 divergira za date ulazne podatke x i mašina M_2 se neće zaustaviti. Očigledno je da deterministička mašina M_2 u najgorem slučaju bar jednom posećuje

[28]Ovo podseća na mehanizam pretrage po širini.

svaki čvor drveta koje prikazuje izvršavanje nedeterminističke mašine M_1. Ovih čvorova može biti eksponencijalno više nego što je dužina najkraćeg mogućeg izračunavanja mašine M_1 koje dovodi do stanja q_{da}, ako takvo uopšte postoji.

Teorema 5.2.13 Za datu nedeterminističku Tjuringovu mašinu M_1 sa k-traka može se konstruisati deterministička Tjuringova mašina M_2 sa jednom trakom koja simulira rad mašine M_1. Dužina rada mašine M_2 je ograničena eksponencijalnom funkcijom dužine rada mašine M_1. ■

Za sada nije poznato da li je determinističku simulaciju rada nedeterminističkih mašina moguće izvesti u polinomijalnom vremenu. U vezi sa tim je čuveni problem da li je $P = NP$ o čemu će biti reči u odeljku 5.5.

5.2.7 Univerzalna Tjuringova mašina

Univerzalna Tjuringova mašina, u oznaci UTM, je svojevrstan primer programibilnog digitalnog računara opšte namene sa programom i podacima smeštenim u memoriju koji simulira izvršavanje ostalih Tjuringovih mašina. Ulazni podaci koji se smeštaju na traku univerzalne Tjuringove mašine su opis neke posebne mašine, tj. njen program, i ulazni podaci te mašine, a rezultat izvršavanja je rezultat rada simulirane posebne mašine. UTM je tako jedinstvena apstraktna mašina koji može samostalno uraditi sve što je u stanju da izvede bilo koja druga Tjuringova mašina.

Zamisao o postojanju ovakve mašine Tjuring je i konkretizovao: napisao je njen program. Univerzalna logička povezanost pojmova programa, podataka i automata koji izvršava dati program nad odgovarajućim podacima, potpuno revolucionarna u to vreme, a danas tako uobičajena, predstavlja temelj savremenog računarstva. Kao ilustraciju o načinu tadašnjeg razmišljanja navodimo dva citata (prema [7]):

- "...Smatrao bih kao najčudniju koincidenciju na koju sam ikada naišao ako bi se ispostavilo da se osnovna logika mašine dizajnirane za numeričko rešavanje diferencijalnih jednačina polkapa sa logikom mašine koja proizvodi račune u nekoj robnoj kući[29] ..."
- "...Vratimo se sada na analogiju sa teorijskim mašinama za izračunavanje ...Može se pokazati da jedna specijalna mašina tog tipa može obavljati posao svih njih. U stvari, ona može raditi kao model bilo koje

[29]Izjavio je 1956. godine Howard Aiken, 1900 – 1973, jedan od pionira u konstrukciji računara, osnivač računarske laboratorije Univerziteta Harvard, konceptualni dizajner računara Harvard Mark I iz 1944. godine.

> druge mašine. Ta specijalna mašina se može nazvati univerzalnom mašinom[30] ...”

Dalje razvijajući ideju o univerzalnom računaru, Tjuring je radio na njenoj realnoj implementaciji, ali i na suštinskim pitanjima, kao što je: u kojoj meri računari mogu oponašati ljudske aktivnosti, iz čega je proistekla danas široko rasprostranjena naučna disciplina veštačka inteligencija.

5.3 Čerčova teza

Čuvena Čerčova teza je iskaz da

svaki algoritam definiše funkciju koja
pripada jednoj dobro definisanoj klasi funkcija

(klasa Tjuring-izračunljivih funkcija, klasa parcijalno rekurzivnih funkcija, klasa λ-definabilnih funkcija ili neka druga ekvivalentna klasa), odnosno da se klasa intuitivno izračunljivih funkcija poklapa sa svakom od nabrojanih klasa. Iako je više istraživača skoro u isto vreme imalo slične ideje, tezu je prvi formulisao Čerč, pa je po njemu i dobila ime.

Intuitivni pojam algoritma je zasnovan na iskustvenom znanju o ljudskim umnim sposobnostima, dok su klase izračunljivih funkcija precizno definisane u odgovarajućim formalnim modelima izračunavanja. Čerčova teza izjednačava neformalni i formalni pristup pojmu efektivne izračunljivosti, te se ne može u strogom smislu smatrati matematičkim tvrđenjem, već je sličnija formulacijama raznih fizičkih zakona. Teza se ne može dokazati u okviru neke formalne teorije, ali može biti opovrgnuta ako bi bila pronađena funkcija koja jeste intuitivno izračunljiva, a nije, recimo, Tjuring-izračunljiva. Činjenica da se tako nešto nije dogodilo od njenog formulisanja govori u prilog tezi. Drugi važan argument u korist teze je međusobna ekvivalentnost raznorodnih formalnih modela izračunavanja do koje teško da bi došlo da neka od intuitivnih karakteristika algoritama nije njima obuhvaćena. Zbog svega toga, pošto višedecenijsko istraživanje nije uspelo da je obori, Čerčova teza se može prihvatiti i kao definicija izračunljivosti.

Tokom svih tih decenija razvijane su moćne tehnike za dokazivanje međusobne ekvivalentnosti formalnih modela izračunavanja i proučavanje široke klase intuitivno izračunljivih funkcija. Nagomilano iskustvo omogućava relativno lako prepoznavanje da li nekom neformalno opisanom postupku odgovara parcijalno izračunljiva funkcija i rutinski prelazak sa intuitivnog na strogi opis algoritma. To za posledicu ima primenu Čerčove teze u nešto širem

[30] Izjavio je 1947. godine Alan Turing na predavanju u Londonskom Matematičkom društvu.

smislu nego što je prethodno naznačeno. Naime, da bi se u raznim dokazima istakle suštinske ideje i izbegli tehnički detalji često se pribegava formulaciji oblika: 'funkcija je intuitivno izračunljiva, pa je prema Čerčovoj tezi Tjuring-izračunljiva'. Time se dokazi skraćuju i čine preglednijim, no treba skrenuti pažnju da ovakvo pozivanje na Čerčovu tezu ne znači suštinski gubitak strogosti jer za svaki takav korak mora postojati formalno opravdanje koje se i iznosi u slučaju potrebe.

Čerčova teza se koristi i kao argument pri objašnjavanju zašto neki problem nije rešiv. Naime, ako pokažemo da se postupak za rešavanje problema ne nalazi u nekoj od formalizovanih klasa izračunljivih funkcija, na osnovu Čerčove teze zaključujemo i da ne postoji efektivni postupak za rešavanje tog problema. U odeljku 5.4 će ovaj tip obrazloženja biti osnova za razdvajanje problema na one na koje smo u stanju da odgovorimo i one kod kojih to nije slučaj, tako da često neće ni biti eksplicitno istaknut.

Za kraj ovog odeljka spomenimo i jedan aspekt intuitivne izračunljivosti kome do sada nije bila posvećena pažnja. Naime, razmotrimo i šta se intuitivno podrazumeva pod algoritamskom izračunljivošću. Kao što će se videti u glavi 5.5, postoje rekurzivne funkcije za čije izračunavanje je potrebno vreme duže od vremena proteklog od pretpostavljenog nastanka kosmosa, i/ili se zahteva veći broj memorijskih registara nego što je broj atoma od kojih je sastavljena naša planeta. Postavlja se pitanje da li su takve funkcije zaista izračunljive, jer je očigledno da se bar neke njihove vrednosti praktično ne mogu izračunati. Ako se pod intuitivnom izračunljivošću podrazumeva ono što se stvarno može izračunati, uglavnom se prihvata da su izračunljive funkcije za čije izračunavanje je potrebno ne više od broja koraka koji je polinomijalna funkcija dužine ulaznih podataka, što je očigledno prava potklasa klase rekurzivnih funkcija. U tom slučaju, Čerčova teza predstavlja korisnu granicu klase funkcija izvan koje sigurno nema izračunljivih funkcija.

5.4 Odlučivost

U odeljku 5.1.2 je kao razlog uvođenja formalnih modela izračunavanja navedeno utvrđivanje da li za neki problem postoji algoritam koji ga rešava. Pošto definicija Tjuring-izračunljivih (ili njima ekvivalentnih, recimo parcijalno rekurzivnih) funkcija prema Čerčovoj tezi određuju jasnu granicu dosega algoritama, postojanje takvih funkcija biće kriterijum da njima odgovarajuće probleme smatramo algoritamski rešivim.

Definicija 5.4.1 *Predikat* je relacija, odnosno neki podskup skupa $\mathbb{N}^k$ za neki prirodan broj $k > 0$. Predikat R je *odlučiv* (*rekurzivan*) ako je njegova karakteristična funkcija $C_R(x_1, \ldots, x_n)$:

$$C_R(x_1, \ldots, x_n) = \begin{cases} 1 & \text{ako važi } R(x_1, \ldots, x_n) \\ 0 & \text{ako ne važi } R(x_1, \ldots, x_n) \end{cases}$$

totalna Tjuring izračunljiva funkcija.

Ako predikat nije odlučiv, on je *neodlučiv*. ■

Pošto je relacija nekakav podskup skupa nad kojim je definisana, prirodno je da se razmatra i odlučivost skupova. Zapravo, analogno se kaže za neki skup A da je odlučiv, odnosno neodlučiv, ako mu karakteristična funkcija jeste (odnosno nije) totalna Tjuring izračunljiva funkcija. Slično je i sa pojmom *problem*, koji možemo shvatiti kao skup k-torki koje su njegovo rešenje, pa se i tu koristi ista terminologija. Primeri odlučivih skupova (predikata i problema) su:

- skup $\mathbb{N}$ prirodnih brojeva,
- svaki njegov konačan podskup[31],
- skup parnih i skup neparnih brojeva,
- zadovoljivost i valjanost iskaznih formula,
- teorija[32] Bulovih algebri, teorija množenja prirodnih brojeva, teorija Abelovih grupa, teorija realno zatvorenih polja, elementarna euklidska geometrija itd.

Klasa odlučivih skupova (a time i predikata i problema) je zatvorena za osnovne operacije komplementiranja (u odnosu na skup $\mathbb{N}^k$), unije, preseka i razlike, što trivijalno proizilazi iz razmatranja karakterističnih funkcija za skupove dobijene tim operacijama.

Međutim, metodologijom koja je razvijena u teoriji izračunljivosti, pokazalo se da je odlučivost izuzetak, tj. da su neodlučivi problemi mnogo prisutniji. Neki od poznatih primera za to su:

- problem zaustavljanja - da li proizvoljna Tjuringova mašina za proizvoljan ulaz završava rad u konačno mnogo koraka,
- da li je proizvoljna Tjuring-izračunljiva funkcija totalna,
- da li su dve proizvoljne Tjuring-izračunljive funkcije jednake,

[31] Za konačan skup $A = \{a_1, \ldots, a_n\}$ prirodnih brojeva karakteristična funkcija $x = a_1 \vee \ldots \vee x = a_n$ je očigledno rekurzivna.

[32] Za neku teoriju T kažemo da je odlučiva ako je odlučiv problem 'formula α je teorema teorije T'.

- problem reči za grupe, tj. ako je grupa G sa jediničnim elementom e generisana skupom elemenata $Gen_G = \{g_1, g_2, \ldots\}$, da li za proizvoljan izraz t_1 sastavljan od elemenata iz Gen_G (recimo $t_1 = g_2^3 g_1^{-1} g_5$) važi $t_1 = e$,
- rešivost diofantskih jednačina,
- problemi zadovoljivosti i valjanosti formula predikatskog računa prvog reda,
- problem pokrivanja[33] ravni u kome je dat konačan broj proizvoljnih oblika poligona, a postavlja se pitanje da li je moguće u potpunosti, bez preklapanja, pokriti ravan poligonima samo tih oblika itd.
- Peanova aritmetika, teorija grupa, teorija prstena, teorija polja, ZF teorija skupova itd.

U istom duhu je i teorema 5.4.2 koja zapravo kaže da su svi netrivijalni skupovi Tjuring-izračunljivih funkcija neodlučivi:

Teorema 5.4.2 (Rajsova teorema) Neka je $\mathbb{B}$ neprazna prava potklasa klase svih Tjuring-izračunljivih funkcija. Problem da li proizvoljna Tjuring-izračunljiva funkcija pripada $\mathbb{B}$ nije odlučiv. ■

a na osnovu koje direktno sledi da sledeći problemi nisu odlučivi:

- domen funkcije je konačan,
- domen funkcije je beskonačan,
- kodomen funkcije je konačan i
- kodomen funkcije je beskonačan.

U teoriji izračunljivosti i složenosti izračunavanja se proučavaju klase (ne)odlučivih problema i uvode odgovarajuće hjerarhije. O jednoj klasifikaciji odlučivih problema biće reči u odeljku 5.5, a ovde ćemo kao ilustraciju dati definiciju jedne važne klase neodlučivih predikata i navesti neke njene osnovne osobine.

Definicija 5.4.3 Predikat (odnosno skup ili problem) R je *parcijalno odlučiv* (*rekurzivno nabrojiv*) ako je njegova karakteristična funkcija oblika

$$C_R(x_1, \ldots, x_n) = \begin{cases} 1 & \text{ako važi } R(x_1, \ldots, x_n) \\ \text{nedefinisano} & \text{inače.} \end{cases}$$

[33] Tiling problem.

parcijalna Tjuring-izračunljiva[34] funkcija. ■

Može se pokazati da važi sledeće:

- Predikat P je parcijalno odlučiv ako i samo ako postoji parcijalna Tjuring-izračunljiva funkcija f čiji je domen P.
- Predikat $P(x_1, \ldots, x_n)$ je parcijalno odlučiv ako i samo ako postoji odlučiv predikat $R(x_1, \ldots, x_n, y)$ tako da važi $P(x_1, \ldots, x_n)$ ako i samo ako važi $(\exists y)R(x_1, \ldots, x_n, y)$.
- Predikat P je odlučiv ako i samo ako su predikati P i $\mathbb{C}P$ parcijalno odlučivi[35].

Primeri parcijalno odlučivih problema su:

- problem zaustavljanja proizvoljne Tjuringova mašina za proizvoljan ulaz i
- rešivost diofantskih jednačina[36].

Sada direkntno sledi da komplementi ovih problema nisu ni odlučivi, ni parcijalno odlučivi jer bi u suprotnom svi problemi bili odlučivi. Takođe, ni problem da li je proizvoljna Tjuring-izračunljiva funkcija totalna nije parcijalno odlučiv. Dalje se pokazuje da postoji čitava jedna hijerarhija skupova prirodnih brojeva koja se naziva *aritmetička hijerarhija* tako da su skupovi na višim nivoima u nekom smislu više neodlučivi od skupova sa nižih nivoa.

5.5 Složenost izračunavanja

Razmatrajući Čerčovu tezu u odeljku 5.3 skrenuli smo pažnju da postoji suštinska razlika između praktično izračunljivih funkcija i funkcija koje se mogu izračunati u principu. U nastavku ćemo prikazati jedan pristup klasifikaciji složenosti odlučivih problema merenoj računarskim resursima poput

[34] Ova definicija se može oslabiti tako što se dozvoli da za neke, ali ne nužno sve, $x_1, \ldots, x_n$ za koje $R(x_1, \ldots, x_n)$ ne važi, bude $C_R(x_1, \ldots, x_n) = 0$.

[35] Intuitivno rečeno, predikat $P(x)$ je parcijalno odlučiv ako postoji program koji odgovara potvrdno u slučaju da predikat važi za argumente programa, inače program ne mora da se zaustavi. Ako bi postojali takvi programi $Prog_P$ za predikat P i $Prog_{\mathbb{C}P}$ za njegov komplement $\mathbb{C}P$, mogli bismo na dva računara da ih pokrenemo paralelno. Pošto za svaki x važi ili $P(x)$ ili $\mathbb{C}P(x)$ jedan od programa će se posle izvesnog vremena zaustaviti. Ako se zaustavi program $Prog_P$ odgovor bi bio 'važi $P(x)$', a ako se zaustavi program $Prog_{\mathbb{C}P}$ odgovor bi bio 'ne važi $P(x)$', pa bi predikat P bio odlučiv.

[36] Matijašević je pokazao da su svi parcijalno odlučivi predikati ekvivalentni problemima rešavanja nekih diofantskih jednačina, odakle direktno sledi da je taj problem nije odlučiv.

vremena i memorijskog zauzeća koji se koriste tokom rešavanja problema. Istraživanja o kojima je reč još nisu u potpunosti odgovorila na pitanja koliko su i zašto neki zadaci teški, tj. koliko je vremena i prostora potrebno da bi bili rešeni, ali su dovela do jedne elegentne hijerarhije problema koja pruža argumente da se sa velikom pravom veruje da su neki problemi jako teški za izračunavanje, mada to, možda nismo u stanju da precizno dokažemo.

Danas je dosta široko prihvaćeno stanovište da se pod praktično izračunljivim problemima podrazumevaju oni kod kojih je dužina rada odgovarajućih programa limitirana nekom stepenom funkcijom dužine ulaznih podataka. Preostali odlučivi problemi se smatraju praktično neizračunljivim, tj. izračunljivim samo u principu. Za takve probleme se ne preporučuje konstrukcija opštih algoritama za rešavanje, već se pokušava pronalaženje efikasnih rešavača za neke posebne potprobleme. I pored upornog istraživanja, granica između praktično izračunljivih i praktično neizračunljivih problema nije precizno određena kao što je to slučaj sa odlučivim i neodlučivim predikatima. Tako je, recimo, problem zadovoljivosti iskaznih formula u izvesnom smislu reprezent klase praktično neizračunljivih problema. Za sada još nije pokazano, iako se u to duboko veruje, da ovaj problem ne pripada klasi praktično izračunljivih problema. Ako bi se dokazalo da problem zadovoljivosti ipak pripada i ovoj klasi problema, onda bi granica koja razdvaja praktično izračunljive od praktično neizračunljivih problema morala biti znatno podignuta.

5.5.1 Opis problema

Formalni model izračunavanja koji se koristi u analizi složenosti su deterministička i nedeterministička Tjuringove mašine sa konačnim brojem $k \geq 1$ traka koje su ograničene sa leve strane i od kojih prva traka sadrži ulazne podatke, a poslednja eventualni rezultat. Odgovarajući konačni skupovi stanja sadrže početno stanje q_0, i završna stanja $\{q_z, q_{da}, q_{ne}\}$ koja redom označavaju završetak rada (koristi se pri analizi složenosti izračunavanja funkcija, kada prelazak u to stanje označava završetak rada programa), pozitivan odgovor na pitanje i negativan odogovor na pitanje. Kao pogodan za rad se u ovom kontekstu pokazao binarni alfabet $\{0, 1\}$, pri se čemu kao pomoćni znaci upotrebljavaju i marker levog kraja trake $\triangleright$ i blanko znak.

Koristiće se i takozvane Tjuringove mašine sa ulazom i izlazom koje imaju dodatni zahtev da se prva traka sa ulaznim podacima može samo čitati, a da se u poslednju traku sme samo upisivati (ovo poslednje se obezbeđuje tako što se glava poslednje trake ne sme kretati ulevo).

Problemi koji se analiziraju u teoriji složenosti izračunavanja uglavnom se karakterišu pitanjima na koja se odgovara sa 'da' ili 'ne'. Recimo, jedan problem se odnosi na ispitivanje da li je graf povezan. Svaki konkretan graf za koji se postavi ovo pitanje je *primerak* problema. U nekim situacijama, kao kod optimizacije, rešenje problema je neki numerički rezultat. Ovaj slučaj se može svesti na prethodni tako što se pitanje preformuliše u oblik: 'ako je data konstanta c, da li je x rešenje problema za koje je vrednost funkcije koja se optimizuje jednaka sa (veća od, manja od) c?', ali se može rešavati i direktno konstrukcijom odgovarajuće funkcije. Predstavljanje problema se vrši u nekom formalnom jeziku na alfabetu neke Tjuringove mašine, što se formalizuje sledećom definicijom.

Definicija 5.5.1 *Problem* L za koji se ispituje složenost je podskup skupa svih reči nekog alfabeta[37]. *Komplement problema* L, u oznaci $\overline{L}$ na nekom alfabetu je skup svih reči na tom alfabetu koje nisu u L. ■

Neka je dat alfabet A i neka je L problem. Tjuringova mašina *prihvata* ulazni podatak, tj. reč, x ako postoji izračunavanje[38] u kome se, polazeći od reči x upisane na ulaznoj traci u početnom stanju q_0, dolazi do završnog stanje q_{da}, a *odbacuje* x ako uvek dolazi do završnog stanje q_{ne}. Ako Tjuringova mašina M prihvata sve reči x jezika koje pripadaju problemu L, a odbacuje svaku reč koja nije u problemu L, kaže se da M *odlučuje* problem L.

Ako Tjuringova mašina M za ulazni podatak x u izračunavanju dolazi do stanja q_z, onda je sadržaj poslednje trake rezultat rada mašine i označava se sa $M(x)$. Sa $L(x)$ ćemo označavati primerak problema L za ulazni podatak x, odnosno pitanje da li je $x \in L$.

Primer 5.5.2 Neka je L problem ispitivanja povezanosti dva čvora grafa i x opis nekog grafa i njegova dva izabrana čvora. Tada je $L(x)$ primerak problema L u kome se ispituje da li su u grafu opisanom sa x povezani izabrani čvorovi. ■

Ako je $\overline{L}$ komplement problema L, onda je za svaki primerak x problema odgovor na pitanje da li je $\overline{L}(x)$ pozitivan, odnosno negativan, ako i samo ako je odgovor na pitanje $L(x)$ negativan, odnosno pozitivan.

Primer 5.5.3 Komplement problema ispitivanja zadovoljivosti formule je ispitivanje nezadovoljivosti formule. ■

Da bi opis problema koji koristimo bio univerzalan potrebno je proizvoljan zadatak predstaviti kao niz reči u nekom alfabetu. Recimo, graf bez izolovanih čvorova se može prikazati kao niz ivica, odnosno niz uređenih parova čvorova, elementi konačnog skupa se prikazuju kao prirodni brojevi koji se opet prikazuju u binarnom obliku itd.

[37]Imajući u vidu rečeno u odeljku 5.2.2 problem je zapravo jezik na nekom alfabetu.

[38]U slučaju determinističkih Tjuringovih mašina, to izračunavanje je jedinstveno.

5.5.2 O-notacija

U teoriji složenosti izračunavanja se razmatra brzina rasta nekih funkcija. Brzina rasta se analizira asimptotski, pri čemu se često koriste različite aproksimacije koje opisujemo narednim definicijama i tvrđenjima.

Definicija 5.5.4 Neka su f i g aritmetičke funkcije. Tada je *funkcija f u velikom O od g* (u oznaci $f(x) = O(g(x))$) ako postoje brojevi c i n takvi da za svaki $x > n$ važi $f(x) \leq c \cdot g(x)$. Ako takvi brojevi ne postoje, onda je $f(x) \neq O(g(x))$. ■

Umesto funkcija f je u velikom O od g kaže se *funkcija f je reda funkcije g* ili *funkcija g je asimptotska gornja granica funkcije f*.

Definicija 5.5.5 Funkcija f *raste brže* od funkcije g ako $f(x) \neq O(g(x))$. Funkcije f i g *rastu istom brzinom*, u oznaci $f(x) = \Theta(g(x))$, ako važi $f(x) = O(g(x))$ i $g(x) = O(f(x))$. ■

Primer 5.5.6 Jednostavnom analizom se zaključuje da funkcije n^4 i $1345 \cdot n^4 + 2007 \cdot n^3 - 7n + 5$ rastu istom brzinom, dok funkcija $0.0001 \cdot n^5$ raste brže od funkcije $1345 \cdot n^4 + 2007 \cdot n^3 - 7n + 5$. ■

Sledeće teoreme formulišu neke kriterijume za upoređivanje brzina rasta funkcija, a mogu se dokazati sredstvima koja se standardno koriste u realnoj analizi.

Teorema 5.5.7 Neka su f i g aritmetičke funkcije i neka je $\lim_{n\to\infty} \dfrac{f(n)}{g(n)} = \beta$. Ako je β pozitivan realan broj, onda funkcije f i g rastu istom brzinom. Ako je $\beta = \infty$, onda važi $g(x) = O(f(x))$ i $f(x) \neq O(g(x))$, tj. funkcija f raste brže od g. ■

Teorema 5.5.8 Neka je $P(n) = a_0 + a_1 \cdot n + \ldots + a_r \cdot n^r$, $a_r \neq 0$, polinom stepena r sa celobrojim koeficijentima. Tada za $P(n)$ i n^m važi:

1. ako je $m = r$, $P(n)$ i n^m rastu istom brzinom,
2. ako je $m < r$, $P(n)$ raste brže od n^m i
3. ako je $m > r$, n^m raste brže od $P(n)$. ■

Teorema 5.5.9 Neka je $k > 1$. Funkcije k^n raste brže od bilo kog polinoma sa celobrojnim koeficijentima. Svaki polinom sa celobrojnim koeficijentima raste brže od bilo koje logaritamske funkcije. ■

Kako je $\log_c x = \log_c d \cdot \log_d x$, direktno sledi sledeće tvrđenje.

Teorema 5.5.10 Za svake dve realne konstante $c, d > 1$ važi $\log_c(x) = \Theta(\log_d(x))$. ■

Funkcije koje se obično javljaju u O-notaciji prilikom analize složenosti su: logaritamska funkcija[39] $\log_2 n$, linearna funkcija $k \cdot n$, njihov proizvod $n \log_2 n$, stepena funkcija n^k, eksponencijalna funkcija k^n itd. Svima njima je zajedničko:

- $\lim_{n\to\infty} f(n) = \infty$, što ima razumljivo intuitivno opravdanje: što je problem većih dimenzija složenost izračunavanja je veća,
- funkcije su neopadajuće i
- postoje Tjuringove mašine koje ih izračunavaju u prostoru i vremenu koji su proporcionalni vrednostima funkcija.

Ovakve funkcije se nazivaju *prave funkcije složenosti*[40] i upotrebljavaju se u analizi složenosti izračunavanja.

5.5.3 Klase složenosti

Najčešće korištene mere složenosti se odnose na vreme, tj. broj koraka izvršavanja programa, i prostor, tj. količinu memorije koju koristi program. Uobičajeno je da se složenost izražava kao funkcija veličine ulaznog podatka. Ako je x ulazni podatak programa, njegova veličina se označava sa $|x|$. U slučaju da je ulazni podatak opis grafa, pogodno je da $|x|$ bude broj čvorova grafa. Slično, ako je ulazni podatak reč, $|x|$ označava dužinu, tj. broj znakova reči.

Definicija 5.5.11 *Vreme izvršavanja* izračunavanja Tjuringove mašine M koja kao ulaz dobija podatak x jednako je dužini niza konfiguracija koje predstavljaju to izračunavanje.

Neka je f unarna aritmetička funkcija, koja zadovoljava uslove za pravu funkciju složenosti. Tjuringova mašina M radi *u vremenu* $f(n)$, ako je za bilo koji ulazni podatak x vreme izvršavanja izračunavanja mašine najviše $f(|x|)$. Za nedeterminističku Tjuringovu mašinu M se kaže da radi u vremenu $f(n)$, ako je za bilo koji ulazni podatak x vreme izvršavanja bilo kog izračunavanja mašine najviše $f(|x|)$. Funkcija f je *vremenska granica složenosti* za M.

[39] Prema teoremi 5.5.10 konstanta koja je baza logaritma nije bitna, pa je moguće ravnopravno koristiti i druge logaritamske funkcije bez promene klase koja se njima određuje. Ponekad se u literaturi ovo ogleda u tome što se piše samo $\log n$.

[40] Proper complexity functions.

$TIME(f(n))$ je skup problema za koje postoje determinističke Tjuringove mašine koje ih odlučuju, a za koje je vremenska granica složenosti $f(n)$. $NTIME(f(n))$ se definiše analogno, u odnosu na nedeterminističke Tjuringove mašine. ■

Prostorna složenost nekog problema se definiše na donekle izmenjen način u odnosu na vremensku složenost. Razlog za to je želja da se izbegne uključivanje prostora u koji je upisan ulazni podatak, odnosno u koji se smešta rezultat, u razmatranje prostorne složenosti. Za to su pogodne Tjuringove mašine sa ulazom i izlazom. Restrikcija o kojoj je reč ne smanjuje izražajne sposobnosti, pošto je trivijalno da za svaku Tjuringovu mašinu sa k traka koja radi u vemenu $f(n)$ postoji Tjuringova mašina sa ulazom i izlazom sa $k+2$ trake koja rešava isti problem u vremenu $O(f(n))$.

Definicija 5.5.12 *Prostor izvršavanja* izračunavanja Tjuringove mašine M sa ulazom i izlazom koja kao ulaz dobija podatak x jednak je broju različitih ćelija traka, sem prve - ulazne i poslednje - izlazne trake, nad kojima se tokom izračunavanje nađu glave traka.

Neka je f unarna aritmetička funkcija koja zadovoljava uslove za pravu funkciju složenosti. Tjuringova mašina M radi *u prostoru* $f(n)$, ako je za bilo koji ulazni podatak x prostor izvršavanja izračunavanja mašine najviše $f(|x|)$. Za nedeterminističku Tjuringovu mašinu M se kaže da radi u prostoru $f(n)$, ako je za bilo koji ulazni podatak x prostor izvršavanja bilo kog izračunavanja mašine najviše $f(|x|)$. Funkcija f je *prostorna granica složenosti* za M.

$SPACE(f(n))$ je skup problema za koje postoje determinističke Tjuringove mašine koje ih odlučuju, a za koje je prostorna granica složenosti $f(n)$. Skup problema $NSPACE(f(n))$ se definiše analogno, u odnosu na nedeterminističke Tjuringove mašine. ■

Ovakav pristup prostornom zauzeću omogućava razmatranje mašina koje koriste manje od $|x|$, recimo $\log_2 |x|$, ćelija, gde se pod korištenjem podrazumeva da su te ćelije radni prostor, odnosno da se u njih privremeno smeštaju podaci koji se upotrebljavaju tokom izračunavanja. Pri tome se u prostor čija se veličina meri ne uključuju, recimo, ćelije prve trake koje sadrže ulazni podatak.

Primer 5.5.13 Neka je M mašina sa ulazom i izlazom koja sadrži četiri trake i ispituje da li je ulazna reč palindrom. Prva traka sadrži ulaznu reč i moguće ju je samo čitati. Druga traka sadrži binarni zapis indeksa i koji označava redni broj ciklusa rada, treća binarni zapis indeksa j, dok se četvrta traka ne koristi. Na početku rada indeks i se postavlja na 1, a zatim se rad obavlja u ciklusima. Svaki ciklus počinje inicijalizovanjem indeksa j na 1 i postavljanjem glave prve trake nad najlevlju ćeliju ulazne reči. Ako je $j < i$, uvećava se j za 1 i pomera glava prve trake nadesno. Ako je $j = i$ pamti se simbol prve trake koji se trenutno čita i ponovo postavlja j na 1. Zatim se analogno pronalazi i-ti znak ulazne reči brojano sa desne strane i upoređuje sa zapamćenim simbolom. Postupak se prekida kada su upoređeni znaci različiti, kom prilikom se prelazi u stanje q_{ne}, odnosno kada je i-ti znak ulazne reči blanko znak, pri čemu se prelazi u stanje q_{da}. U prvom slučaju reč nije, a u drugom reč jeste palindrom. Prostor izvršavanja je u $O(\log_2 n)$ koliko je potrebno za binarno predstavljanje indeksa i i j. ■

Definicija 5.5.14 *Klasa složenosti* je skup problema sa zajedničkom vremenskom ili prostornom granicom. ■

Primer 5.5.15 Skupovi problema $TIME(f(n))$, $NTIME(f(n))$, $SPACE(f(n))$ i $NSPACE(f(n))$ su neke klase složenosti. ■

U definisanju klase složenosti se pretpostavlja da za granice složenosti $f(n)$ važi:

- $f(n) \geq n$, ako je reč o vremenskoj složenosti i
- $f(n) \geq \log_2 n$, ako je reč o prostornoj složenosti.

Intuitivno rečeno, nedeterminističke klase složenosti sadrže probleme kod kojih je broj kandidata za rešenje veliki, ali kada se kandidat izabere, onda je problem njegovog testiranja, (verifikacije, provere) u okviru odgovarajuće determinističke klase problema. Pri tome za svaki x koji je primerak problema postoji izračunavanje koje dovodi do prihvatanja, a problem predstavlja izbor izračunavanja kojim se x prihvata. Ni za jedan x koji nije primerak problema ne postoji takvo izračunavanje.

Primer 5.5.16 Primer problema koji se nalazi u nedeterminističkoj klasi je testiranje zadovoljivosti iskaznih formula. Za proizvoljnu formulu postoji relativno veliki broj interpretacija koje treba ispitati, ali ako se izabere pogodna interpretacija pri kojoj je formula zadovoljena, sama provera nije komplikovana. Slično, i problem trgovačkog putnika u kome se ispituje da li postoji put u grafu koji kroz svaki čvor prolazi tačno jednom i koji je kraći od neke unapred zadate konstante se nedeterministički lako rešava. Nedeterministička Tjuringova mašina treba da izabere jednu permutaciju čvorova grafa i proveri dužinu odgovarajućeg puta. Iako je broj permutacija n čvorova jednak $n!$, nedeterministički postupak ima polinomijalnu vremensku granicu složenosti. ■

Definicija 5.5.17 Neka je $\mathcal{C}$ neka klasa složenosti, njen komplement, u oznaci *co*-$\mathcal{C}$ je skup problema oblika $\{\overline{L} : L \in \mathcal{C}\}$. ■

Očigledno je da za sve determinističke klase složenosti važi $\mathcal{C} = co\text{-}\mathcal{C}$ jer se komplement svakog problema iz klase $\mathcal{C}$ rešava istom Tjuringovom mašinom koja dodatno menja završno stanje q_{da} u q_{ne} i obrnuto. Zato se kaže da su determinističke klase složenosti zatvorene za komplement. Nije poznato da li u opštem slučaju isto važi i za nedeterminističke klase složenosti.

5.5.4 Odnosi između klasa složenosti

Određivanje odnosa između klasa složenosti je jedno od osnovnih pitanja kojima se bavi teorija složenosti izračunavanja. Tvrđenje 5.5.18 govori da se iz granice složenosti $f(n)$ može eliminisati konstantni faktor kojim se množi najsloženiji deo funkcije, tj. da je red brzine rasta $O(f(n))$ ono što je u granici složenosti $f(n)$ bitno.

Teorema 5.5.18 Neka je problem $L \in TIME(f(n))$. Tada je za proizvoljno $\epsilon > 0$, $L \in TIME(\epsilon f(n) + n + 2)$.

Neka je problem $L \in SPACE(f(n))$. Tada je za proizvoljno $\epsilon > 0$, $L \in SPACE(\epsilon f(n) + 2)$. ■

Sa druge strane, teorema 5.5.19, takozvana *teorema hijerarhije* govori da sa dovoljnim povećanjem granice složenosti klase složenosti šire.

Teorema 5.5.19 Neka je $f(n)$ prava funkcija složenosti. Tada važi:

1. ako je $f(n) \geq n$, onda je[41] $TIME(f(n)) \subsetneqq TIME((f(2n+1))^3)$ i
2. $SPACE(f(n)) \subsetneqq SPACE(f(n) \cdot \log_2 f(n))$. ■

U opisu granica složenosti O-notacija se koristi na sledeći način: $TIME(O(f(n)) = \cup_{c>0} TIME(c \cdot f(n))$ ili $SPACE(2^{O(n)}) = \cup_{c>0} SPACE(2^{c \cdot n})$. Često se umesto neke posebne funkcije koja definiše granicu složenosti koristi familija funkcija, recimo familija svih stepenih funkcija, svih eksponencijalnih funkcija itd., što podrazumeva da je takva klasa složenosti unija klasa određenih elementima familije. Neke od važnijih klasa složenosti su:

- $L = SPACE(O(\log_2 n))$
- $NL = NSPACE(O(\log_2 n))$
- $P = \cup_i TIME(n^i)$

[41] Može se pokazati i da je gustina različitih klasa veća. Recimo, svaka klasa $TIME(f(n))$ je pravi podskup kalse $TIME(f(n) \log_2^2 f(n))$. Nama je ovde dovoljno i slabije tvrđenje.

- $NP = \cup_i NTIME(n^i)$
- $PSPACE = \cup_i SPACE(n^i)$
- $NPSPACE = \cup_i NSPACE(n^i)$
- $EXP = \cup_i TIME(2^{n^i})$,
- $NEXP = \cup_i NTIME(2^{n^i})$,
- $EXPSPACE = \cup_i SPACE(2^{n^i})$,
- $2 - EXP = \cup_i TIME(2^{2^{n^i}})$,
- $2 - NEXP = \cup_i NTIME(2^{2^{n^i}})$ itd.

Dakle, P je klasa složenosti koja sadrži one probleme za koje je vremenska granica složenosti programa koje ih rešavaju neka stepena[42] funkcija. Primetimo da su, zbog određenih tehničkih pogodnosti, u klasama složenosti EXP i $NEXP$ stepeni funkcije polinomi. Nazivima klasa koje odgovaraju determinističkim Tjuringovim mašinama ponekada se dodaje slovo D, tako da se umesto $TIME$ koristi $DTIME$, a umesto $SPACE$, $DSPACE$.

U hijerarhiji klasa složenosti ima otvorenih pitanja o tome da li je neki stepen hijerarhije jednak nekom drugom stepenu. Često se zna da je jedan stepen sadržan u drugom, ali se ne zna da li, ili ne, važi i obrnuto, tj. da li se stepeni poklapaju, ili je jedan pravi podskup od drugog. Neke od dokazanih relacija su iskazane sledećom hijerarhijom klasa složenosti:

$$L \subset NL \subset P \subset NP \subset PSPACE \subset EXP \subset NEXP.$$

Poznato je da važi $NL \neq PSPACE$ i $P \neq EXP$, pa na bar nekim mestima u hijerarhiji relacija podskup mora biti striktna. Međutim, čitav niz problema je ostao za sada nerešen, bez obzira na intenzivna istraživanja koja se sprovode. Neka od tih pitanja su:

- Da li je $P = NP$[43]?
- Da li je $P = PSPACE$?
- Da li je $L = NL$?

[42] Pošto je $O(n^k) = O(a_n \cdot n^k + \ldots + a_1 \cdot n + a_0)$, preciznije je reći neka polinomijalna funkcija.

[43] Smatra se da je ovo glavni nerešeni problem teroijskog računarstva. Postavio ga je 1971.godine S. Cook u [Coo71]. Uvršten je na listu problema za čije rešavanje je Clay Mathematics Institute 2000. godine ponudio nagradu od po milion dolara.

- Da li je $EXP = NEXP$?

Prvo pitanje je od posebnog značaja s obzirom na ranije spomenutu granicu između praktično izračunljivih problema i onih koji su to samo u principu. Dokaz da je $P \neq NP$ bio bi potvrda takvih shvatanja, dok bi suprotan rezultat, mada malo verovatan, doveo do prave revolucije u razvoju algoritama. Zanimljivo je da iz $P = NP$ sledi $EXP = NEXP$.

5.5.5 Pozicioniranje složenosti problema

Među pitanja kojima se bavi teorija složenosti izračunavanja spada i određivanje kojoj klasi složenosti pripada neki problem. Pri tome se obično određuju neke gornje i donje granice složenosti tako da budu što bliže jedna drugoj, ali se dešava da su one međusobno dosta udaljene, te da se složenost problema ne može uvek precizno odrediti. *Gornja granica*[44] složenosti nekog odlučivog problema se određuje konstrukcijom algoritma za njegovo rešavanje i analizom koliko vremena i/ili memorije taj algoritam koristi. Ovde treba primetiti da različiti algoritmi za rešavanje nekog problema mogu dati i različite gornje granice složenosti.

Primer 5.5.20 Razmotrimo Euklidov algoritam za nalaženje najvećeg zajedničkog delioca dva prirodna broja:

```
function Euklid(m,l)
begin
    while m > 0 do
        t := l mod m
        l := m
        m := t
    return l
end
```

Ako je $l \geq m$, uvek je $l \bmod m < \frac{l}{2}$. Neka k predstavlja ukupan broj prolazaka funkcije kroz petlju za ulazne podatke m i l, i neka su za $i \leq k$, m_i i l_i vrednosti od m i l na kraju i-te petlje. Uslov za izlazak iz petlje u koraku k je da je $m_k = 0$ i $m_i \geq 1$, za $i < k$. Vrednosti m_i i l_i su definisane na sledeći način: $l_i = m_{i-1}$ i $m_i \equiv_{m_{i-1}} l_{i-1}$, za $1 \leq i \leq k$. Očigledno je da je za svaki $i \geq 1$, $l_i > m_i$. Zbog toga je

$$m_i(\equiv_{m_{i-1}} l_{i-1}) < \frac{l_{i-1}}{2} = \frac{m_{i-2}}{2}$$

[44] Upper bound.

za svaki $i \geq 2$. Ako je $k = 2d + 1$ imamo

$$m_{k-1} < \frac{m_{k-3}}{2} < \frac{m_{k-5}}{4} < \ldots < \frac{m_0}{2^d}.$$

Pošto je $m_{k-1} \geq 1$, važi $m_0 \geq 2^d$. Odatle sledi $k = 2d + 1 \leq 1 + 2 \log_2 m_0$. Slično se analizira i slučaj za $k = 2d$, imajući u vidu da je $m_1 \equiv_{m_0} l_0 < m_0$.

Dakle, broj prolazaka kroz petlju je reda $\log_2 m$. Kako je dužina binarnog zapisa broja m upravo reda $\log_2 m$, broj prolazaka kroz petlju je u $O(n)$, gde je $n = |m|$ veličina binarne reprezentacije ulaznog podatka. Potrebno vreme za operaciju deljenja koje se vrši u petlji prilikom izračunavanja modula je u $O(\log_2^2 m)$, pa je složenost celog postupka u $O(n^3)$, za $n = |m|$. ■

Donja granica[45] složenosti nekog odlučivog problema se određuje tako što se pokaže da su izvesno vreme i/ili memorijski prostor neophodni za rešavanje tog problema bilo kojim algoritmom. Određivanje donje granice složenosti je često teško i nije poznat neki univerzalni postupak za to. Jedna od metoda koja se primenjuje je *metoda brojanja* u kojoj se definiše neka karakteristika ponašanja mašine, pa se analizira koliko puta se ta karakteristika mora ispuniti prilikom prihvatanja ulaza veličine n. Napomenimo i to da su česte situacije, recimo u logičkim teorijama, u kojima donja granica nije valjana za sve ili za skoro sve ulazne podatke, već samo neograničeno mnogo puta, što dalje komplikuje problem.

5.5.6 Kompletni problemi

Druga vrsta pozicioniranja u hijerarhiji složenosti je relativna i izvodi se poređenjem odnosa složenosti problema u čemu postupak redukcije ima značajnu ulogu.

Definicija 5.5.21 Problem A se *redukuje* na problem B, u oznaci $A \leq B$, ako postoji izračunljiva funkcija f takva da je $A(x)$ tačno ako i samo ako je tačno i $B(f(x))$. Funkcija f se tada naziva *funkcija redukcije.* ■

Primetimo da redukovanje ima smisla samo ako je složenost izračunavanja funkcije redukcije zanemarljiva u odnosu na složenost problema B. Složenost izračunavanja funkcije redukcije se ograničava tako da pripada klasi L. Prema opisanoj hijerarhiji, funkcije redukcije pripadaju i klasi P, što je pogodnije za analizu problema u klasama sa vremenskim granicama složenosti. Uz to, ova klasa složenosti obezbeđuje da je veličina rezultata $f(x)$ takođe polinomijalno ograničena u odnosu na $|x|$.

[45] Lower bound.

Definicija 5.5.22 Funkcija redukcije f problema A na problem B je *efikasna*, a problem A je *efikasno reducibalan* na problem B, u oznaci $A \leq_{ef} B$, ako je složenost funkcije f u klasi L. ■

Primer 5.5.23 Problem postojanja Hamiltonovog puta u grafu, tj. put koji kroz svaki čvor grafa prolazi tačno jednom se efikasno redukuje na problem SAT koji se odnosi na ispitivanje da li je proizvoljna klasična iskazna formula zadovoljiva. Neka graf G sadrži n čvorova. Odgovarajuća iskazna formula $R(G)$ će sadržati n^2 iskaznih slova $x_{i,j}$, $1 \leq i, j \leq n$ čije značenje je 'čvor j je i-ti čvor u Hamiltonovom putu'. $R(G)$ je formula u konjunktivnoj formi čije konjunkti su oblika:

- $x_{1,j} \vee \ldots \vee x_{n,j}$, za svako j, što znači da se svaki čvor mora pojaviti u putu,
- $\neg x_{i,j} \vee \neg x_{k,j}$, za svako j i $i \neq k$, što znači da se svaki čvor pojavljuje tačno jednom u putu,
- $x_{i,1} \vee \ldots \vee x_{i,n}$, za svako i, što znači da jedan čvor mora biti i-ti u putu,
- $\neg x_{i,j} \vee \neg x_{i,k}$, za svako i i $j \neq k$, što znači da se samo jedan čvor može biti i-ti u putu i
- $\neg x_{k,i} \vee \neg x_{k+1,j}$, za sve čvorove i i j koji nisu susedni u grafu G i $1 \leq k \leq n-1$.

Lako se pokazuje da interpretacija koja zadovoljava formulu $R(G)$ opisuje jedan Hamiltonov put u grafu. Slično, svaki Hamiltonov put u grafu definiše jednu interpretaciju koja zadovoljava $R(G)$. Sledeća Tjuringova mašina koja za ulaz ima opis grafa G i generiše na izlaznoj traci $R(G)$ pripada klasi L. Mašina na radnoj traci najpre predstavi u binarnoj formi broj n. Na osnovu toga se izgenerišu svi konjunkti formule $R(G)$ koji ne zavise od grafa G, za šta su potrebna tri indeksa i, j i k. U poslednjem koraku, pomoću istih indeksa se generišu na radnoj traci redom formule $\neg x_{k,i} \vee \neg x_{k+1,j}$, za sve čvorove i i j i $1 \leq k \leq n-1$, a zatim se proverava da li su odgovarajući čvorovi povezani u grafu G. Ako to nije slučaj, formula se prepiše na izlaznu traku. ■

Definicija 5.5.24 Klasa problema $\mathcal{C}$ je *zatvorena za* $\leq_{ef}$ ako za svaki problem $B \in \mathcal{C}$ i svaki problem A važi da ako je $A \leq_{ef} B$, onda je i $A \in \mathcal{C}$. ■

Može se pokazati da su klase složenosti L, NL, P, NP, *co*-NP, $PSPACE$ i EXP zatvorene za redukciju.

Pod pretpostavkoma da je $A \leq_{ef} B$ upotrebom funkcije f, složenost problema A je odozgo ograničena zbirom složenosti problema B i funkcije redukcije f. Naime, za ispitivanje da li važi $A(x)$ najpre se x preslika u $f(x)$, a zatim se primeni program za utvrđivanje da li je $B(f(x))$. Dakle, ako su poznate složenosti problema B i funkcije f moguće je odrediti jednu gornju granicu složenosti problema A. Redukcija se može iskoristiti i za utvrđivanje donje granice složenosti. Ako je poznato da je složenost problema A veća od nekog zadatog nivoa, onda se kontrapozicijom može odrediti i jedna donja granica složenosti problema B.

Definicija 5.5.25 Neka je B problem i $\mathcal{C}$ klasa složenosti. Tada kažemo:

- problem B je *$\mathcal{C}$-težak*[46], u oznaci $\mathcal{C} \leq_{ef} B$, ako je za svaki problem $A \in \mathcal{C}$ ispunjeno $A \leq_{ef} B$ i
- problem B je *$\mathcal{C}$-kompletan*[47] ako je $\mathcal{C} \leq_{ef} B$ i $B \in \mathcal{C}$. ■

Pojam kompletnog problema je značajan pošto svaki takav problem predstavlja klasu u onosu na koju je kompletan. Tako, na primer, NP-kompletan problem pripada klasi P ako i samo ako $P = NP$. Ovo je posledica tvrđenja 5.5.26 i činjenice da je klasa P zatvorena za $\leq_{ef}$.

Teorema 5.5.26 Neka su $\mathcal{C}$ i $\mathcal{D}$ klase složenosti, takve da je $\mathcal{D} \subset \mathcal{C}$ i $\mathcal{D}$ zatvorena za $\leq_{ef}$ i neka je B jedan $\mathcal{C}$-kompletan problem. Tada važi $B \in \mathcal{D}$ ako i samo ako $\mathcal{C} = \mathcal{D}$. ■

Postojanje prirodnih problema koji su kompletni za neku klasu složenosti daje klasi odgovarajući značaj, iako on možda nije jasan samo na osnovu njene definicije. Takav slučaj je, na primer, sa raznim nedeterminističkim klasama. U nastavku ćemo prikazati primere kompletnih problema za neke klase složenosti. Videćemo da su ti problemi proistekli iz stvarnih istraživanja, odakle proističe i značaj odgovarajućih klasa složenosti.

Primeri kompletnih problema za najznačajnije klase složenosti:

- u klasi složenosti L se nalazi problem koji sadrži sve reči koje su palindromi, kao i svi problemi za grafove koji se mogu formulisati u klasičnom jeziku prvog reda[48]. Problem kompletnosti u ovoj klasi složenosti nije značajan pošto redukcija ima smisla samo u klasi koja je složenija od same redukcije. L je najmanja prirodna klasa složenosti jer je veličina binarne reprezentacije pokazivača na ulazni podatak x reda $\log_2 |x|$,

[46] *$\mathcal{C}$-hard.*
[47] *$\mathcal{C}$-complete.*
[48] Recimo simetričnost grafa se opisuje sa $(\forall x)(\forall y)(G(x,y) \to G(y,x))$.

- problem GAP[49] koji se odnosi na utvrđivanje da li postoji put između dva zadata čvora grafa je NL-kompletan problem,
- problem CV[50] koji se odnosi na izračunavanje vrednosti izlaza logičkog kola u kome ulazne promenljive imaju fiksirane vrednosti je P-kompletan problem,
- problem SAT koji se definiše kao skup svih zadovoljivih klasičnih iskaznih formula je NP-kompletan problem,
- $PSPACE$-kompletan je problem RD[51] u kome se ispituje da li je za dati sistem procesa koji komuniciraju i neko inicijalno stanje moguće stići u stanje u kome su svi procesi zaglavljeni čekajući međusobno jedan drugog.

Za problem SAT, uprkos obimnom istraživanju, još uvek nije pokazano da li je, ili nije, u klasi P. Na ovom primeru se lako uočava razlika između determinističkog i nedeteminističkog izračunavanja. Umesto razmatranja cele istinitosne tablice koje se vrši u determinističkom slučaju, izborom interpretacije (tj. reda tablice) pri kojoj formula važi, se lako (u polinomijalnom vremenu) pokazuje da je formula zadovoljiva.

5.5.7 Komentar o pristupu analizi složenosti

Razdvajanje problema na praktično izračunljive i izračunljive u principu, zavisno od toga jesu li, ili ne, u klasi P nije uvek opravdano. Recimo, algoritam sa eksponencijalnom vremenskom granicom složenosti u kojoj je eksponent mali je u nekim praktičnim slučajevima, u kojima ulaz relativno nije veliki, pogodniji od algoritma sa polinomijalnom vremenskom granicom složenosti. U tabeli 5.1 su prikazana dva slučaja. U svakom od redova je prikazana dužina rada računara koji u sekundi obavlja 10^9 koraka izračunavanja za po dva algoritma sa polinomijalnom, odnosno eksponencijalnom, vremenskom granicom složenosti. Razlika između redova je u stepenima polinoma, odnosno eksponetima i ilustruje relativnost ove vrste podele na praktično i samo u principu izračunljive probleme.

Slična situacija se javlja kod linearnog programiranja i simpleks algoritma koji je eksponencijalan, ali dobrih performansi u praksi, odnosno nekih polinomijalnih algoritama za ovaj problem koji su u praksi veoma spori. Teorijski gledano, broj slučajeva u kojima bi neki eksponencijalni algoritam mogao biti bolji od polinomijalnog je obavezano konačan, ali sa

[49] Graph accessibility problem.
[50] Circuit value problem.
[51] Reachable deadlock.

Vremenska granica	Vreme izvršavanja	Vremenska granica	Vreme izvršavanja
n^2	$3.6 \cdot 10^{-6} sec$	2^n	36.3 godine
n^{10}	19.4 godine	$2^{\sqrt[4]{n}}$	$8 \cdot 10^{-9} sec$

Tabela 5.1. Poređenje vremena izvršavanja algoritama za ulazne veličine $n = 60$.

stanovišta praktičnog programiranja, primerci problema koji su interesantni mogu biti upravo u tom skupu. Ipak treba reći da polinomijalnih algoritama sa ogromnim stepenima nema puno, kao ni eksponencijalnih algoritama sa jako malim eksponentom, pa spomenute sitacije nisu pravilo.

Druga primedba u vezi teorije složenosti izračunavanja se odnosi na to što se analiziraju slučajevi u kojima se algoritmi najgore ponašaju. Moguće je da se algoritam sa lošim najgorim slučajem prihvatljivo, pa čak i superiorno u odnosu na ostale, ponaša u proseku. Primer za to je *quick-sort* algoritam sortiranja koji za slučajan niz ima složenost $O(n \log_2 n)$, dok je složenost za najgori slučaj $O(n^2)$. Analiza očekivanog, a ne najgoreg, slučaja je u takvim situacijama mnogo informativnija. Međutim, da bi se ovakva analiza sprovela potrebno je poznavanje distibucije ulaznih problema, što je često teško ostvarljivo.

Konačno, polinomijalne funkcije su pogodne za analizu: njihova klasa je zatvorena za sabiranje i množenje, logaritmi svih polinomijalnih funkcija se razlikuju u konstantnom faktoru, pa su svi u $\theta(\log_2 n)$ itd. Zbog svega ovoga je izbor pristupu prihvatanja statusa praktične izračunljivosti za probleme koji su u najgorem slučaju polinomijalni nužno pojednostavljenje koje dovodi do primenljive i elegantne teorije koja govori o stvarnim izračunavanjima.

6

Teorija grafova

Teorija grafova je jedna od matematičkih disciplina koja poslednjih decenija izaziva veliko interesovanje koje potiče kako od teorijskih problema na koje se nailazi, tako i od primenljivosti rezultata do kojih se došlo. Bez obzira na aktuelnost, prvi rad[1] koji se odnosi na teoriju grafova pojavio se još u prvoj polovini XVIII veka. Danas se ova disciplina koristi u mnogobrojnim drugim oblastima, poput računarstva, hemije, fizike, za modeliranje saobraćajnih, električnih ili računarskih mreža, zapravo kad god se razmatraju neki objekti i na njima definisane relacije, itd. U ovom poglavlju biće date definicije osnovnih pojmova u teroji grafova i fomulisana neka osnovna tvrđenja sa naznakama oblasti u kojima nalaze primene.

6.1 Osnovne definicije

Definicija 6.1.1 *Graf* je uređeni par $G = \langle V_G, E_G \rangle$, gde su:

- V_G skup *čvorova*[2] i
- E_G skup *ivica*[3] (*grana, rebara*) oblika $\{u, v\}$, $u, v \in V_G$.

Ivica $\{u, v\}$ povezuje čvorove u i v, pri čemu su oni *susedni*[4].

Stepen čvora $u \in V_G$ je broj ivica $\{u, v\} \in E_G$[5].

Graf je *regularan* ako svi njegovi čvorovi imaju isti stepen.

[1]L. Ejler je 1736. godine u tom radu, iako nije imao razvijenu terminologiju o grafovima, rešavao takovani problem mostova u Königsberg-u. Naziv *graf* u ovom kontekstu je uveo Silvester 1878. godine

[2]Vertex, node.

[3]Edge, arc.

[4]Adjacent, neighbor.

[5]Ako se razmatraju grafovi koji imaju ivicu oblika $\{u, u\}$, ona se u određivanju stepena čvora u broji dva puta.

Graf H je *podgraf* grafa G ako je dobijen brisanjem iz G nekih čvorova i ivica, pri čemu se obavezno brišu sve ivice u kojima se nalaze obrisani čvorovi.

Komplement grafa G, u oznaci $\mathbb{C}G = \langle V_G, E_{\mathbb{C}G}\rangle$, ima isti skup čvorova kao i G, ali $\{u, v\} \in E_{\mathbb{C}G}$ ako i samo ako $\{u, v\} \notin E_G$ (pretpostavljamo da je $u \neq v$).

Težinski graf[6] je uređena trojka $G = \langle V_G, E_G, w\rangle$, gde su

- $G = \langle V_G, E_G\rangle$, graf i
- $w : E_G \mapsto [0, \infty)$ funkcija koja ivicama pridružuje težinu. ■

Graf može biti *konačan*, ili *beskonačan*, u zavisnosti od kardinalnosti njegovog skupa čvorova. U ovom poglavlju biće razmatrani konačni grafovi. U literaturi se opisuju i grafovi kod kojih između dva čvora može postojati više ivica. Takvi grafovi se nazivaju *multigrafovi*, dok su *obični* grafovi oni koji zadovoljavaju:

- u skladu sa definicijom 6.1.1, ne postoje višestruke ivice koje povezuju 2 čvora i
- ne sadrže ivice oblika $\{u, u\}$, takozvanu *petlju*[7].

U nastavku će uglavnom biti reči o običnim grafovima, pa to neće biti posebno naglašavano.

Očigledno je da je svaki graf $G = \langle V_G, E_G\rangle$ jedan način predstavljanja izvesne binarne relacije na skupu V_G. Primetimo da, ako graf definišemo kao što je urađeno u definiciji 6.1.1, relacija koju on predstavlja je simetrična.

Primer 6.1.2 Najjednostavniji primer grafa je $\langle V_G, \emptyset\rangle$ u kome ne postoji ni jedna ivica. Njegov komplement je *kompletni graf*[8], u oznaci K_n, gde je $n = |V_G|$.

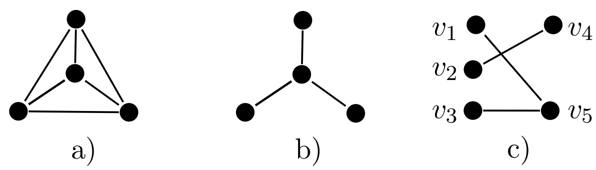
Zvezda je graf $\langle V_G, \{\{u, v\} : v \in V_G\}\rangle$ u kome sve ivice povezuju jedan čvor u sa ostalim čvorovima grafa.

Kod *bipartitnog grafa* skup čvorova ima particiju $\{V_G^1, V_G^2\}$ pri čemu svaka ivica povezuje čvor iz V_G^1 sa čvorom iz V_G^2. *Kompletni* bipartitni graf je bipartitni graf u kome je svaki čvor iz V_G^1 povezan sa svakim čvorom iz V_G^2.

Na slici 6.1 prikazani su kompletni graf K_4 (slika a), jedan graf u obliku zvezde (slika b) i nekompletni bipartitni graf (slika c) za koga je $V_G^1 = \{v_1, v_2, v_3\}$ i $V_G^4 = \{v_4, v_5\}$, pa je $|V_G^1| = 3$ i $|V_G^2| = 2$. ■

Drugi način za prikazivanje grafova je pomoću matrice susedstva.

[6] Engleski: weighted graph.
[7] Loop.
[8] Complete graph, clique.



Slika 6.1. Slikovne reprezentacije grafova.

Definicija 6.1.3 *Matrica susedstva* grafa $G = \langle V_G, E_G \rangle$ je kvadratna matrica $S(G)_{|V_G|\times|V_G|}$ u kojoj je $S(G)_{i,j}$ broj ivica koje povezuju čvorove v_i i v_j. ■

Primer 6.1.4 Matrica

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

predstavlja matricu susedstva za bipartitni graf iz primera 6.1.2 dat slikom 6.1(c). Očigledno je da je matrica simetrična. ■

Posmatrajmo jedan niz ivica $e_1 = \{v_0, v_1\}$, $e_2 = \{v_1, v_2\}$, ..., $e_n = \{v_{n-1}, v_n\}$ u grafu u kome za svako i ivice e_i i e_{i+1} imaju zajednički čvor ($e_i \cap e_{i+1} = \{v_i\}$, $i = 1, n-1$). Svake dve uzastopne ivice u ovom nizu su *susedne*, a nizu ivica e_1, ..., e_n odgovara niz čvorova v_0, v_1, ..., v_n.

Definicija 6.1.5 *Put* u grafu je niz međusobno različitih susednih ivica $e_1 = \{v_0, v_1\}$, $e_2 = \{v_1, v_2\}$, ..., $e_n = \{v_{n-1}, v_n\}$ takvih da u odgovarajućem nizu čvorova nema jednakih, sem eventualno čvorova v_0 i v_n. *Šetnja* je svaki niz međusobno različitih susednih ivica kod kojih u odgovarajućem nizu čvorova može biti i jednakih.

Dužina puta je broj ivica koje ga čine.

Ciklus (*kružni, zatvoreni put*) je put za koji važi $v_0 = v_n$. ■

Primetimo da:

- uslov da među ivicama u putu nema istih znači da put ne sadrži kao potput ni jedan ciklus, a

- uslov da među odgovarajućim čvorovima u nizu nema jednakih (sem eventualno v_0 i v_n) znači da put ne seče samog sebe.

Drugim rečima, prelazeći neki put obilazićemo različite čvorove (uz eventualni izuzetak početka i kraja puta), dok kod šetnje to ne mora biti slučaj.

Definicija 6.1.6 Čvorovi u i v u grafu G su povezani putem $e_1, e_2, \ldots, e_n$ ako je $e_1 = \{u, x\}$ i $e_n = \{y, v\}$.

Graf je *povezan* ako za svaka dva čvora postoji put koji ih povezuje.

H je *povezana komponenta* grafa G ako je to maksimalan podgraf grafa G koji je povezan. ∎

Svaki čvor grafa pripada tačno jednoj povezanoj komponenti jer:

- očigledno pripada bar jednoj povezanoj komponenti, a
- svake dve povezane komponente su disjunktne, jer ako bi imale zajednički čvor i njihova unija bi bila povezana komponenta.

Definicija 6.1.7 Dva grafa $G = \langle V_G, E_G \rangle$ i $H = \langle V_H, E_H \rangle$ su *izomorfna* ako postoji bijektivna funkcija $f : G \mapsto H$ takva da $\{u, v\} \in E_G$ ako i samo ako je $\{f(u), f(v)\} \in E_H$.

Dva grafa $G = \langle V_G, E_G \rangle$ i $H = \langle V_H, E_S \rangle$ su *homeomorfna* ako se izomorfna slika jednog može dobiti iz izomorfne slike drugog grafa dodavanjem na neke ivice, ili brisanjem sa nekih ivica, čvorova stepena 2. ∎

Ovde ćemo samo napomenuti da je problem složenosti ispitivanja (ne)izomorfnosti grafova otvoren, tj. nije poznato da li pripada klasi P ili je NP-kompletan.

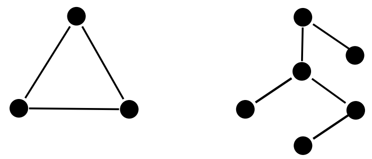
6.2 Planarnost grafova

Na prethodnim slikama predstavljeni su neki grafovi, tako što su im čvorovi prikazani kao tačke, a ivice kao linije koje ih povezuju. Neki od grafova imaju osobinu da su *predstavljivi u ravni*, tj. da su nacrtani u ravni tako da se linije koje povezuju njihove čvorove ne seku (sem što se dodiruju u temenima).

Definicija 6.2.1 Graf je *planarni* ako je izomorfan nekom grafu predstavljivom u ravni. ∎

Najpre ćemo formulisati tvrđenje 6.2.2 o slabijem zahtevu za predstavljivost grafova.

Teorema 6.2.2 Svaki graf se može predstaviti u prostoru dimenzije 3 (u E^3).



a) b)

Slika 6.2. Planarni grafovi i oblasti u ravni.

Dokaz. Neka je dat graf $G = \langle V_G, E_G \rangle$ u kome je broj čvorova $|V_G| = m$, i broj ivica $|E_G| = k$. Posmatraćemo proizvoljnu pravu l i k različitih ravni $\alpha_1, \alpha_2, \ldots, \alpha_k$ iz pramena ravni koje se seku po pravoj l. Na pravoj l zatim biramo m tačaka $A_1, A_2, \ldots, A_m$ koje će predstavljati čvorove grafa, dok svakoj od k ivica pridružimo tačno jednu od izabranih ravni. Ako je ivica oblika $e_n = \{v_i, v_j\}$, onda ćemo u ravni α_n tačke A_i i A_j povezati lukom. Time se dobija predstavljanje grafa u E^3. ■

Sada se prirodno postavlja pitanje da li se dimenzija prostora u kome su predstavljivi svi grafovi može spustiti na 2, a ako to nije slučaj - da li se za proizvoljan graf može proveriti da li je planaran. Pored teorijskog, odgovor ima značaj i za primene, recimo da li neko elektronsko kolo prikazano grafom može biti odštampano na jednom nivou štampane ploče, ili se (ako graf nije planaran) mora premostiti nekoliko nivoa štampe da bi se izbeglo da se veze elemenata seku.

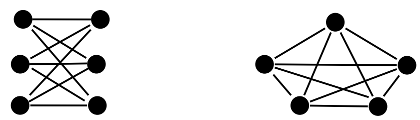
Uočimo najpre da ako je graf planaran, on deli ravan na *oblasti* od kojih je nula ili više njih konačnih zatvorenih, i tačno jedna neograničena.

Primer 6.2.3 Na slici 6.2 su prikazana dva planarna grafa. Prvi od njih, K_3, koji je u obliku trougla, (slika a) deli ravan na jednu konačnu zatvorenu i jednu neograničenu oblast, dok kod drugog koji je u obliku stabla (slika b) postoji samo jedna neograničena oblast. ■

Teorema 6.2.4 povezuje svojstvo planarnosti sa karakteristikama grafa datim brojevima čvorova, ivica i oblasti koje graf određuje u ravni.

Teorema 6.2.4 (Ojlerova teorema) Povezani planarni graf $G = \langle V_G, E_G \rangle$ deli ravan u $f = |E_G| - |V_G| + 2$ oblasti. ■

Ovo tvrđenje daje kriterijum za utvrđivanje da neki graf nije planaran.



a) b)

Slika 6.3. Kompletan bipartitni graf $K_{3,3}$ i kompletan graf K_5.

Primer 6.2.5 Razmotrimo kompletan bipartitni graf dat na slici 6.3(a). Ovaj graf se označava sa $K_{3,3}$. Za njega je $|E_G| = 9$ i $|V_G| = 6$. Primenom teoreme 6.2.4 se pokazuje da $K_{3,3}$ nije planaran.

Ako bi graf bio planaran, važilo bi da su granice oblasti neki ciklusi u grafu. Za zatvorene oblasti to je trivijalno, dok za neograničenu oblast kao intuicija može poslužiti prvi graf sa slike 6.2 u kome ciklus deli ravan na jednu konačnu zatvorenu i jednu neograničenu oblast. Svaka ivica pripada granici tačno dve oblasti. Odatle je broj ivica koje pripadaju granicama oblasti jednak $2|E_G|$. U grafu $K_{3,3}$ najkraći ciklus ima 4 ivice, pa i svaka oblast mora imati granicu sa najmanje toliko ivica. Pošto svaka ivica pripada nekom ciklusu, sledi da broj ivica koje pripadaju granicama oblasti nije manji od $4 \cdot f$, odnosno:

$$2|E_G| \geq 4 \cdot f.$$

Kada se ovo zameni u Ojlerovu formulu dobija se kontradikcija

$$2|E_G| = 18 \geq 4 \cdot (|E_G| - |V_G| + 2) = 4 \cdot (9 - 6 + 2) = 20,$$

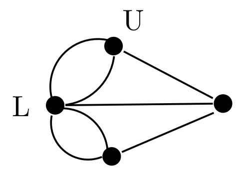
pa zaključujemo da graf $K_{3,3}$ nije planaran.

Slično razmatranje, uz ograničenje da su najkraći ciklusi dužine 3, pa je $2|E_G| \geq 3 \cdot f$, dovodi do zaključka da ni kompletni graf K_5 dat na slici 6.3(b) nije planaran. ■

Jasno je da ni jedan graf koji kao podgraf ima graf izomorfan bilo sa $K_{3,3}$, bilo sa K_5, ne može biti planaran. Međutim, obrnuto ne mora da važi, već se tu koristi nešto slabiji pojam - homeomorfizam grafova - uveden u definiciji 6.1.7. Intuicija je sledeća: posmatrajmo graf dobijen od K_5 tako što je na ivici $\{v_1, v_2\}$ dodat čvor v_6, tako da umesto $\{v_1, v_2\}$, postoje ivice $\{v_1, v_6\}$ i $\{v_6, v_2\}$. Taj novi graf i dalje nije planaran, ali nije ni izomorfan (već samo homeomorfan) sa K_5. Tada važi tvrđenje:

Teorema 6.2.6 (Kuratosvki) [9] Graf je planaran ako i samo ako ni jedan njegov podgraf nije homeomorfan grafovima $K_{3,3}$, ili K_5. ■

[9] Kazimierz Kuratowski, 1896 – 1980, poljski matematičar.



Slika 6.4. Grafovska reprezentacija mape Königsberg-a.

6.3 Ojlerova šetnja

Kao što je ranije napomenuto, Ojlerov je analizirajući problem mostova u Königsberg-u (danas Kaliningrad, u Rusiji) dao prvi poznati rad u oblasti teorije grafova. Na slici 6.4 data je grafovska reprezentacija mape grada u kojoj su čvorovi kvartovi razdvojeni rekom, a ivice označavaju mostove koji ih povezuju. Na primer, gornji čvor (odnosno kvart, označen sa U) 3 mosta povezuju sa ostalim kvartovima, dok je za najlevlji čvor (označen sa L) to slučaj sa 5 mostova. Primetimo da se ovde radi o multigrafu, a ne o običnom grafu, pošto, na primer, gornji i levi čvor povezuje 2 mosta. Problem o kome je reč odnosi se na ispitivanje da li je moguće izvesti šetnju, nazvanu kasnije *Ojlerova šetnja*, u kojoj se svaki most prelazi tačno jednom. Treba obratiti pažnju da je, pošto je reč o šetnji, dozvoljeno da se isti čvor poseti više puta.

Ojler je na problem odgovorio negativno, tj. da takva šetnja ne postoji, uz intuitivno jasno obrazloženje. Pretpostavimo da šetnju ne počinjemo u čvoru L. U nekom trenutku, prelazeći neku od ivica stići ćemo do njega, pa ga napustiti drugom ivicom, zatim se na njega vratiti trećom, pa ga ponovu napustiti četvrtom i konačno pomoću pete ivice vraćamo se u L. Tu moramo da se zaustavimo jer smo iskoristili svih 5 ivica. Dakle, ako šetnja nije započela u L, tu mora da se završi. Slično se objašnjava i konstatacija da, ako šetnja počinje u L, u njemu ne može da se i završi. Isto važi i za sve druge čvorove, pri čemu je jedina razlika da je stepen svakog od njih 3, pa je broj poseta tim čvorovima manji nego za L. Dakle, za svaki čvor zaključujemo da šetnja ili polazi iz njega, ili se u njemu završava. Pošto graf sadrži 4 čvora, ispuniti takav zahtev nije moguće. Ojler je kostatovao da se isti zaključak može dobiti ispitivanjem svih mogućih šetnji, ali je (primećujući da ta provera može biti jako duga) iskazao i opšti kriterijum za (ne)postojanje ovakve šetnje, a koja formulisana u savremenoj terminologiji glasi:

Teorema 6.3.1 Ako povezani graf ima više od dva čvora neparnog stepena, u njemu nije moguće izvesti Ojlerovu šetnju. Ako povezani graf ima tačno dva čvora neparnog stepena, u njemu je moguće izvesti Ojlerovu šetnju, a svaka od tih šetnji mora početi u jednom od tih čvorova i završiti u drugom.

Povezani graf ima zatvorenu Ojlerovu šetnju ako i samo ako su mu svi čvorovi parnog stepena. ■

6.4 Hamiltonov ciklus i problem trgovačkog putnika

Problem poznat pod nazivom *Hamiltonov ciklus*[10] je u izvesnom smislu dualan upravo opisanom problemu Ojlerove šetnje, s tim da se ovde ispituje postojanje ciklusa koji sadrži sve čvorove grafa, a ne šetnje koja sadrži sve ivice. Podsetimo i da, kao što u šetnji nema ponavljanja ivica, tako se ni u ciklusu ne smeju isti čvorovi pojavljivati više puta. Međutim, za razliku od jednostavne karakterizacije postojona Ojlerove šetnje kakvu daje teorema 6.3.1, do sada nisu poznati potrebni i dovoljni uslovi za postojanje Hamiltonovog ciklusa u proizvoljnom grafu.

Jedan od dovoljnih uslova za postojanje Hamiltonovog ciklusa formulian je tvrđenjem 6.4.1.

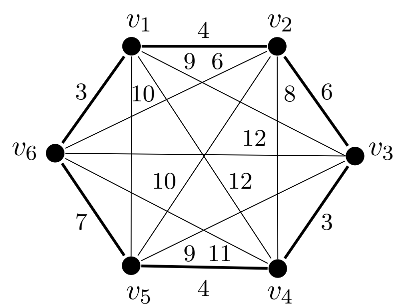
Teorema 6.4.1 Povezani graf sa $n \geq 3$ čvorova u kome je stepen svakog čvora barem $\frac{n}{2}$ sadrži Hamiltonov ciklus. ■

U bliskoj vezi sa Hamiltonovim ciklusima je jedan od najvažnijih problema u kombinatorijalnoj optimizaciji - takozvani *problem trgovačkog putnika*. Ovde se posmatraju kompletni težinski grafovi u kojima se traže Hamiltonovi ciklusi sa minimalnim zbirom težina ivica. Značaj problema trgovačkog putnika prepoznat je u mnogim oblastima u kojima služi za modeliranje različitih realnih situacija. Na primer, težine ivica mogu biti rastojanja koja treba preći na putu, vremena ili količine goriva koje treba potrošiti na obavljanja nekih operacija itd.

Primer 6.4.2 Razmotrimo težinski kompletan graf dat na slici 6.5. On može biti predstavljen i matricom

$$\begin{bmatrix} 0 & 4 & 9 & 12 & 10 & 3 \\ 4 & 0 & 6 & 8 & 10 & 10 \\ 9 & 6 & 0 & 3 & 9 & 12 \\ 12 & 8 & 3 & 0 & 4 & 11 \\ 10 & 10 & 9 & 4 & 0 & 7 \\ 3 & 10 & 12 & 11 & 7 & 0 \end{bmatrix}$$

[10] William Rowan Hamilton, 1805 – 1865, irski matematičar, fizičar i astronom. Uveo je kvaternione, jednu vrstu generalizacije kompleksnih brojeva.



Slika 6.5. Ciklus trgovačkog putnika.

u kojoj su odgovarajuće koordinate težine pridružene ivicama. Recimo, $w(v_1, v_2) = 4$ i $w(v_2, v_5) = 10$. Može se pokazati da je rešenje problema trgovačkog putnika u ovom grafu ciklus koji sadrži ivice $\langle v_1, v_2\rangle$, $\langle v_2, v_3\rangle$, $\langle v_3, v_4\rangle$, $\langle v_4, v_5\rangle$, $\langle v_5, v_5\rangle$, $\langle v_6, v_1\rangle$, koje su na slici prikazane debljim linijama, ukupne težine 29. ■

Primetimo da, pošto razmatramo konačne grafove u kojima postoji samo konačno mnogo Hamiltonovih ciklusa, uvek postoji i bar jedan minimalan, pa je problem trgovačkog putnika odlučiv. Međutim, za sada nije poznat i efikasan algoritam za njegovo rešavanje, tj. pokazano je da je ovaj problem NP-kompletan. Zbog toga se intenzivno radi na konstrukciji heurističkih algoritama za njegovo rešavanje.

6.5 Uparivanje u bipartitnim grafovima

Pretpostavimo da na raspolaganju imamo izvestan broj različitih fotokopir-uređaja i nekoliko tipova jediničnih punjenja tonera, tako da se neke vrste tonera mogu sipati u neke vrste fotokopira (neke tonere možemo iskoristiti za neke, ne nužno sve fotokopire, dok različitim fotokopirima ne moraju odgovarati iste vrste tonera) i da želimo da ustanovima da li je moguće tako rasporediti tonere da:

- svi fotokopiri budu napunjeni i
- sav toner potrošen.

Ako je tako nešto izvodljivo, ostvareno je *savršeno uparivanje*[11].

Situacije poput navedene se mogu predstaviti pomoću bipartitnih grafova. Recimo, i fotokopire i tonere bismo predstavili pomoću čvorova, dok bi ivice

[11] Engleski: perfect matching

koje ih povezuju ukazivale na kompatibilnost. Sada se problem može formulisati terminologijom grafova na sledeći način: da li se može pronaći skup ivica u bipartitnom grafu tako da je svaki čvor jedne particiji povezan sa tačno jednim čvorom druge particije. Tvrđenje 6.5.1 daje uslove za rešivost ovog problema.

Teorema 6.5.1 Neka je $G = \langle V_G^1 \cup V_G^2, E_G \rangle$ bipartitini graf. Tada:

- ako svaki čvor ima isti pozitivni stepen, u grafu G postoji savršeno uparivanje,
- u grafu G postoji savršeno uparivanje ako i samo ako $|V_G^1| = |V_G^2|$ i za svako k i svaki podskup $A \subset V_G^1$, takav da $|A| = k$, postoji $B \subset V_G^2$, takav da $|B| = k$, pri čemu su čvorovi iz B povezani sa barem jednim čvorom iz A. ■

Za utvrđivanje postojanja savršenog uparivanja u bipartitinom grafu postoje efikasni algoritmi sa polinomijalnom vremenskom složenošću.

6.6 Hromatski broj grafa

Prilikom bojenja graf svakom čvoru se pridružuje jedna boja, tako da susedni čvorovi nisu iste boje. *Hromatski broj grafa G* iznosi k, ako je k najmanji broj boja kojima se G može obojiti. Pored praktične primene u ispitivanju logičkih kola, problem utvrđivanja hromatskog broja ima i istorijsku pozadinu. Naime, engleski matematičar Kejli je 1879. godine postavio problem *četiri boje*: da li je moguće obojiti svaku kartu upotrebom četiri boje, pri čemu je svaka država obojena tačno jednom bojom i ni koje dve susedne države (koje imaju zajedničku graničnu liniju) nisu obojene istom bojom. Skoro vek kasnije, 1976. godine, problem je pozitivno rešen. U rešavanju problema su po prvi put u matematici ozbiljno iskorišteni računari pomoću kojih je testiran veliki broj relevantnih slučajeva.

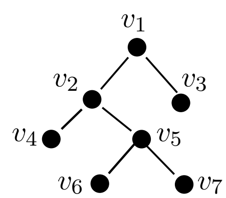
6.7 Stabla

Definicija 6.7.1 *Stablo*[12] je povezan graf bez ciklusa.

Razapinjuće stablo[13] za povezani graf $G = \langle V_G, E_G \rangle$ je stablo koje je podgraf grafa G i sadrži sve čvorove iz V_G.

Stablo sa korenom je uređena trojka $T = \langle V_T, E_T, v \rangle$, gde je $T = \langle V_T, E_T \rangle$ stablo, a $v \in V_T$ izabrani čvor koji se naziva *koren*[14]. Čvorovi

[12]Engleski: Tree.
[13]Engleski: Spanning tree.
[14]Engleski: root.



Slika 6.6. Graf koji je puno binarno stablo.

stepena 1 iz V_T, različiti od korena, se nazivaju *listovi*[15]. Svi ostali čvorovi iz V_T se nazivaju *unutrašnji*[16].

U stablu sa korenom $T = \langle V_T, E_T, v\rangle$ *nivo* čvora w je dužina jedinstvenog puta, tj. broj ivica, od korena v do w. *Visina* je maksimalni nivo čvorova u stablu. *Roditelj* čvora w nivoa k je jedinstveni susedni čvor v nivoa $k-1$. Ako je v roditelj čvora w, onda je w *potomak* čvora v.

Stablo sa korenom u kome svaki čvor ima najviše m potomaka, a bar jedan čvor ima tačno m potomaka je m-arno stablo. Ako je $m = 2$, reč je o *binarnom stablu*, a ako je $m = 3$ o *ternarnom*. m-arno stablo u kome svaki roditelj ima tačno m potomaka se naziva *puno*. ■

Primer 6.7.2 Slika 6.6 prikazuje jedno puno binarno stablo. Čvor v_1 je koren, a listovi su čvorovi v_3, v_4, v_6 i v_7. Čvor v_5 je roditelj čvorova v_6 i v_7. Nivo čvora v_4 iznosi 2, dok je nivo čvorova v_6 i v_7 jednak 3, što je istovremeno i visina stabla. ■

Tvrđenje 6.7.3 prikazuje neka osnovna svojstva stabala.

Teorema 6.7.3 Za svako stablo $T = \langle V_T, E_T\rangle$ važi:

- svaki par različitih čvorova je povezan tačno jednim putem,
- brisanje bilo koje ivice iz E_T proizvodi dva grafa koja su oba stabla i
- $|E_T| = |V_T| - 1$. ■

Nabrojaćemo nekoliko primena u kojima se razvijaju algoritmi zasnovani na strukturama podataka baziranim na stablima:

- sortiranje podataka pomoću binarnih stabala čiji čvorovi sadrže vrednosti tako da za svaki čvor važi da su sve vrednosti u čvorovima u levom podstablu manje do jednake od vrednosti u samom čvoru, koja je manja do jednaka od vrednosyi u čvorovima desnog podstabla,

[15]Engleski: leaf.
[16]Engleski: internal vertex.

- pretraživanje podataka po širini[17] ili po dubini[18] razapinjućeg stabla grafa koji sadrži neke podatke,

- konstrukcija minimalnog razapinjućeg stabla težinskog grafa (koji može modelirati, na primer, telefonsku ili mrežu puteva) ili najkraćeg puta između njegovih čvorova itd.

6.8 Direktni grafovi

U prethodnom tekstu ivice grafova su bile neusmerene, tj. ivice su uvedene kao skupovi. U nekim slučajevima je pogodno ivicama dodati usmerenje i time razlikovati čvorove iz kojih ivice polaze, od onih u koje ivice dolaze. Moguća interpretacija ovakvog tipa ivica je da tokom obilaska grafa nije dozvoljeno kretanje ivicama suprotno njihovim smerovima. Usmerenje se može definisati ako ivice, umesto kao skupove, shvatimo kao uređene parove.

Definicija 6.8.1 *Direktan graf* (*digraf*) je uređeni par $G = \langle V_G, E_G \rangle$, gde je V_G skup *čvorova* i $E_G \subset V_G^2$ skup *ivica*.

Svaka ivica $e = \langle u, v \rangle \in E_G$ je uređeni par čvorova $u, v \in V_G$, gde je u *čvor repa*[19] (*početni čvor*), a v *čvor glave*[20] (*ulazni, završni čvor*) ivice e.

Direktan acikličan graf[21] je direktan graf u kojem nema ciklusa. ■

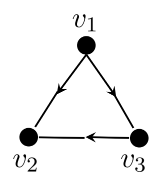
Većina pojmova, poput puta, ciklusa, povezanosti itd., se definišu analogno kao kod neorijentisanih grafova, pri čemu se jedino vodi računa o usmerenju ivica. Na primer, susedne ivice u putu u digrafu moraju biti oblika $e_i = \langle v_{i-1}, v_i \rangle$ i $e_{i+1} = \langle v_i, v_{i+1} \rangle$, tj. čvor glave ivice e_i mora biti čvor repa ivice e_{i+1}. Međutim, ovde relacije koje predstavljaju digrafovi ne moraju biti simetrične.

Prilikom slikovnog prikazivanja digrafova usmerenje ivica prikazuju strelice usmerene od polaznih ka završnim čvorovima, kao što je prikazano na slici 6.7.

Primer 6.8.2 Na slici 6.7 je prikazan digraf $\langle \{v_1, v_2, v_3\}, \{\langle v_1, v_2 \rangle, \langle v_1, v_3 \rangle, \langle v_3, v_2 \rangle\} \rangle$. ■

Sledeći problemi ilustruju moguće primene digrafova:

[17] Engleski: breadth-first search.
[18] Engleski: depth-first search.
[19] Tail vertex.
[20] Head vertex.
[21] Direct acyclic graph, DAG.



Slika 6.7. Direktan graf.

- binarni diagrami odlučivanja[22] koji se koriste u efikasnom predstavljanju Bulovih funkcija,
- maksimizacija protoka kroz transportne mreže[23] modelirne težinskim digrafovima (gde transportna mreža može biti mreža optičkih ili električnih kablova, cevovod, ...) itd.

[22]Engleski: binary decision diagrams, BDD.
[23]Engleski: maximum flow problem

Literatura

[1] M. Agrawal, N. Kayal, N. Saxena, PRIMES is in P, Annals of Mathematics 160, no. 2, 781–793, 2004. http://www.cse.iitk.ac.in/users/manindra/algebra/primality_v6.pdf

[2] E. F. Codd, A Relational Model of Data for Large Shared Data Banks, Communication of the ACM, vol. 13, no 6, 377–387, 1970. http://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf

[3] S. Cook, The complexity of theorem proving procedures, Proceedings of the Third Annual ACM Symposium on Theory of Computing, 151 – 158, 1971.

[4] D. Cvetković, S. Simić, Diskretna matematika, Prosveta, Niš, 1996.

[5] D. Cvetković, Applications of Graph Spectra: an Introduction to the Literature, Zbornik Radova 13(21), Matematički institut SANU, Beograd, 2009. http://elib.mi.sanu.ac.rs/files/journals/zr/21/n021p007.pdf

[6] R. Dacić, Elementarna kombinatorika, Matematički institut, Beograd, 1977.
http://elibrary.matf.bg.ac.rs/bitstream/handle/123456789/482/RadeDacicElementarnaKombinatorika.PDF?sequence=1

[7] M. Davis, Engines of Logic: Mathematicians and the Origin of the Computer, W.W. Norton and comp., New York, 2001. (Prevod pod nazivom: Na logički pogon. Podrijetlo ideja računala, Naklada Jasenski i Turk, Zagreb, 2003.)

[8] R. Garnier, J. Taylor, Discrete mathematics for new technology, Institute of Physics Publishing, 2002.

[9] K. Ghilezan, B. Latinović, Bulova algebra i primene, Matematički institut, Beograd, 1977.

http://elibrary.matf.bg.ac.rs/bitstream/handle/123456789/434/KoriolanGilezanBulovaAlgebraIPrimene.PDF?sequence=1

[10] A. Kron, Elementarna teorija skupova, Matematički institut, Beograd, 1992.

[11] L. Lovász, J. Pelikán, K. Vesztergombi, Discrete Mathematics: Elementary and Beyond, Springer, 2003.

[12] B. Marion, D. Baldwin. SIGCSE Committee Report On the Implementation of a Discrete Mathematics Course, 2007.

[13] Ž. Mijajlović, Algebra 1, MILGOR, Beograd, 1998.
http://elibrary.matf.bg.ac.rs/bitstream/handle/123456789/462/ZarkoMijajlovicAlgebra.pdf?sequence=1

[14] Z. Ognjanović, N. Krdžavac, Uvod u teorijsko racčunarstvo, FON, Beograd, 2004.
http://www.mi.sanu.ac.rs/˜zorano/ti/TeorijskoRacunarstvo.pdf

[15] C. Papadimitriou, Computational complexity, Addison-Wesley, 1995.

[16] A. Perović, A.jovanović, B. Veličković, Teorija skupova, Matematički fakultet, Beograd, 2007.

[17] M. Rašković, N. Ikodinović, Priče o malim i velikim brojevima. O brojanju, merenju, zaključivanju ..., Matematički institut, Zavod za udžbenike, Društvo matematičara Srbije, Beograd, 2010.

[18] J. Shepherdson, H. Sturgis, Computability of Recursive Functions, Journal of the ACM 10(2), 217–255, 1963.

[19] Time 100: The Most Important People of the Century, Special issue titled "TIME 100: Heroes & Icons of the 20th Century", June 14, 1999.
http://205.188.238.181/time/time100/index_2000_time100.html

[20] A. Turing, On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, (2) 42, 230 – 265, 1936. Korekcije objavljene u broju 43, 544 – 546, 1937. http://web.comlab.ox.ac.uk/oucl/research/areas/ieg/e-library/sources/tp2-ie.pdf

[21] A. Turing, Computing Machinery and Intelligence, Mind 49, 433 – 460, 1950. http://loebner.net/Prizef/TuringArticle.html

[22] J. von Neumann, First Draft of a Report on the EDVAC, Contract No.W-670-ORD-4926, between the United States Army Ordnance Department and the University of Pennsylvania. Moore School of Electrical Engineering, University of Pennsylvania, June 30, 1945. http://www.virtualtravelog.net/entries/2003-08-TheFirstDraft.pdf

[23] H. S. Wilf, Generatingfunctionology, third edition, A. K. Peters, Ltd. Wellesley, Massachusetts, 2006. http://www.math.upenn.edu/ wilf/DownldGF.html