# KONTEKSTNO-NEODVISNE GRAMATIKE ZA KODIRANJE IN STISKANJE PODATKOV

#### JANEZ PODLOGAR

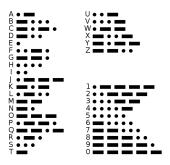
POVZETEK. V delu podamo definicje kodiranja, dekodiranja in stiskanja podtakov ter predstavimo primere, ki motivirajo stiskanje podatkov s kontekstno-neodvisnimi gramatikami.

#### 1. Kodiranje podatkov

Zapis informacije v neki obliki ni primeren za vsakršno rabo. Besedilo, zapisano z pismenkami, je neberljivo za slepe osebe, saj je komunikacijski kanal v tem primeru vid. Prav tako pisanega besedila v prvotni obliki ni mogoče poslati s telegrafom. V tem primeru je komunikacijski kanal žica in pismenke se po njej ne morejo sprehoditi. V obeh primerih je informacija, ki bi jo radi prenesli, zapisana v neprimerni obliki. V prvem primeru je potrebno besedilo zapisati z Braillovo pisavo. V drugem primeru pa je potrebno besedilo pretvoriti v električni signal. Spreminjanje zapisa sporočila imenujemo kodiranje, sistemu pravil, po katerem se kodiranje opravi, pa kod.

**Primer 1.1.** *Morsejeva abeceda* je kodiranje črk, števil in ločil s pomočjo zaporedja kratkih in dolgih signalov:

- Dolžina kratkega signala je ena enota.
- Dolgi signal je trikrat daljši od kratkega signala.
- Razmik med signali znotraj črke je tišina dolžine kratkega signala.
- Razmik med črkami je tišina dolga tri kratke signale oziroma en dolgi signal.
- Presledek med besedami je tišina dolga sedmih kratkih signalov.



Slika 1. Mednarodna Morsejeva abeceda

Prvotni namen Morsejeve abecede je komunikacija preko telegrama, saj komunikacijski kanal dovoljuje le električne signale in tišino med njimi. Kodiranje črk je takšno, da imajo črke z višjo frekvenco (v angleškem jeziku) krajši zapis. Tako se dolžina kodiranega sporočila skrajša in posledično tudi čas njegovega prenosa.

 $\Diamond$ 

**Definicija 1.2.** Abeceda je končna neprazna množica  $\Sigma$ . Elementom abecede pravimo *črke. Množica vseh končnih nizov abecede*  $\Sigma$  je

$$\Sigma^* = \{ a_1 a_2 a_3 \cdots a_n \mid n \in \mathbb{N}_0 \land \forall i : a_i \in \Sigma \},\$$

kjer za n=0 dobimo prazen niz, ki ga označimo z  $\varepsilon$ . Dolžino niza w označimo z |w| in je enaka številu črk v nizu  $w \in \Sigma^*$ . Jezik na abecedi  $\Sigma$  je poljubna podmnožica množice  $\Sigma^*$ .

Opomba 1.3. Kleenejeva zvezdica ali Kleenejevo zaprtje je operacija, ki abecedi  $\Sigma$  priredi najmanjšo nadmnožico  $\Sigma^*$ , ki vsebuje prazen niz  $\varepsilon$  in je zaprta za konkatenacijo oziroma veriženje. Z drugimi besedami,  $\Sigma^*$  je množica vseh končnih nizov, ki jih lahko generiramo z veriženjem črk abecede  $\Sigma$ . Za abecedo  $\Sigma$  definirajmo

$$\Sigma^0 = \{\varepsilon\}, \quad \Sigma^1 = \Sigma$$

ter za vsak i > 0 rekurzivno

$$\Sigma^{i+1} = \{ wa \mid w \in \Sigma^i \text{ in } a \in \Sigma \}.$$

Potem je Kleenejeva zvezdico na  $\Sigma$  enaka

$$\Sigma^* = \bigcup_{i \ge 0} \Sigma^i.$$

**Primer 1.4.** Naj bo  $\Sigma = \{a, b, c\}$  abeceda, potem so ab, ccc in cababcccababcccab končni nizi abecede  $\Sigma$  in potemtakem elementi  $\Sigma^*$ .

 $\Diamond$ 

**Definicija 1.5.** Kodiranje nizov abecede  $\Sigma$  je injektivna funkcija  $\kappa \colon \Sigma^* \to \Sigma_c^*$ , kjer je  $\Sigma_c$  kodirna abeceda in  $\kappa(w)$  imenujemo koda niza w. Dokodiranje kodiranja  $\kappa$  je funkcija  $\kappa^{-1} \colon C \subseteq \Sigma_c^* \to \Sigma^*$ , da velja

$$\forall w \in \Sigma^* \colon \kappa^{-1}(\kappa(w)) = w.$$

**Opomba 1.6.** Zožitev kodomene kodirne funkcije  $\kappa$  na  $C \subseteq \Sigma_c^*$  je bijektivna funkcija.

Primer 1.7. Formalizirajmo Morsejevo abecedo iz Primera 1.1. Abecedi sta

$$\Sigma = \{A, B, \dots, Z\} \cup \{0, 1, \dots, 9\} \cup \{\_\}, \quad \Sigma_c = \{\cdot, -, \_\}.$$

Definirajmo kodirno funkcijo črk abecede  $\kappa \colon \Sigma \to \Sigma_c^*$ , ki vsaki črki iz abecede  $\Sigma$  priredi niz črk kodirne abecede  $\Sigma_c$ . Vrednosti funkcije  $\kappa$  so določene s tabelo iz Slike 1, dodatno presledek med besedami \_ kodiramo v šest kratkih enot tišine

$$\kappa(\underline{\phantom{a}}) = \underline{\phantom{a}}$$

Za niz  $w = a_1 a_2 \dots a_n \in \Sigma^*$  definiramo kodirno funkcijo K po črkah

$$K(w) = \kappa(a_1) \underline{\hspace{0.1cm}} \kappa(a_2) \underline{\hspace{0.1cm}} \cdots \kappa(a_n).$$

Poglejmo si dva primera kodiranja in v Morsejevi abecedi

$$K(SOS) = \cdots --- \cdots,$$
  
 $K(AD\ HOC) = \cdots -\cdots \cdots --- \cdots$ 

Recimo, da smo dobili sporočilo, a se je pošiljatelj zmotil in je namesto kode  $--\cdot-\cdot$ , ki bi se dekodirala v

$$K^{-1}(--\cdot-\cdot--\cdot) = QED,$$

poslali kodo

\_ - · - - · - · ·.

Sporočila ne znamo dekodirati, saj se ne nahaja v domeni C dekodirne funkcije  $K^{-1}$ . Natančneje ne obstaja dekodiranje signala  $\kappa^{-1}(--\cdot--)$ .

#### 2. Stiskanje podatkov

Eden izmed namenov kodiranja je tudi doseči čim večjo ekonomičnost zapisa. Želimo, da bi bilo naše sporočilo čim krajše. Kodiranje, ki skrajša zapis podatkov, imenujemo stiskanje podatkov.

**Definicija 2.1.** Stiskanje je kodiranje K za katerega velja

$$\exists n \in \mathbb{N} \ \forall w \in \Sigma^* \colon |w| \ge n \implies |K(w)| \ll |w|.$$

Opomba 2.2. Ločimo kodiranje brez izgube, kjer velja

$$\forall w \in \Sigma^* : \kappa^{-1}(\kappa(w)) = w$$

in kodiranje z izgubo, kjer kodiranje ni reverzibilen proces in v grobem velja

$$\forall w \in \Sigma^* : \kappa^{-1}(\kappa(w)) \approx w.$$

**Primer 2.3.** Za abecedo vzemimo  $\Sigma = \{a, b, c\}$  in poglejmo niz

w = cababcccababcccab.

Opazimo, da se nam v nizu w večkrat ponovita vzorca ab in ccc. Zato uvedemo novi oznaki A=ab in B=ccc. Sedaj lahko zapišemo w kot

$$w = cAABAABA$$
.

Ponovno se nam pojavi vzorec, tokrat AAB. Uvedemo novo oznako C = AAB in zapišemo w kot

$$w = cCCA$$
.

Prvotni niz smo z novimi oznakami skrajšali. Kot bomo videli, smo pretvorili niz w v kontekstno neodvisno gramatiko  $G_w$  s produkcijskimi pravili

$$S \to cCCA$$
.

 $A \rightarrow ab$ ,

 $B \to ccc$ ,

 $C \rightarrow AAB$ .

## $\Diamond$

### 3. Kontekstno-neodvisne gramatike

**Definicija 3.1.** Formalna gramatika G so pravila, ki nam iz abecede  $\Sigma$  tvorijo jezik, označimo ga z L(G).

**Definicija 3.2.** Kontektsno-neodvisna gramatika je četverica  $G = (V, \Sigma, P, S)$ , kjer je V končna množica spremenljivk, abeceda  $\Sigma$  množica končnih simbolov tako, da  $\Sigma \cap V = \emptyset$ ,  $P \subseteq V \times (V \cup \Sigma)^*$  relacija, ki ji pravimo produkcijsko pravilo in  $S \in V$  začetna spremenljivka.

**Definicija 3.3.** Naj bo $G = (V, \Sigma, P, S)$  kontekstno-neodvisna gramatika. Naj bodo  $\alpha, \beta, \gamma \in (V \cup \Sigma)^*$  nizi spremenljivk in končnih simbolov,  $A \in V$  spremenljivka ter naj bo  $(A, \beta) \in P$  produkcijsko pravilo, označimo ga z $A \to \beta$ . Pravimo, da se  $\alpha A \gamma$  prepiše s pravilom A v $\alpha \beta \gamma$ , pišemo  $\alpha A \gamma \Rightarrow \alpha \beta \gamma$ . Pravimo, da  $\alpha$  porodi $\beta$ , če je  $\alpha = \beta$  ali če za  $k \geq 0$  obstaja zaporedje  $\alpha_1, \alpha_2, \ldots \alpha_n \in (V \cup \Sigma)^*$  tako, da

$$\alpha \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_n \Rightarrow \beta$$

in pišemo  $\alpha \stackrel{*}{\Rightarrow} \beta$ .

Posledica 3.4. Jezik kontekstno neodvisne gramatike G je

$$L(G) = \{ w \in \Sigma^* \mid S \stackrel{*}{\Longrightarrow} w \}.$$

**Opomba 3.5.** Ime kontekstno-neodvisna gramatika izvira iz oblike produkcijskih pravil. Na levi strani produkcijskega pravila mora vedno stati samo spremenljivka. Torej vsebuje samo pravila oblike

$$A \to \alpha$$
,

kjer je  $A \in V$  in  $\alpha \in (V \cup \Sigma)^*$ . Ne sme pa vsebovati pravila oblike

$$\alpha A \gamma \to \alpha \beta \gamma$$
,

kjer je  $A \in V$  in so  $\alpha, \beta, \gamma \in (V \cup \Sigma)^*$ , saj je pravilo odvisno od predhodnega konteksta. Odvisno je od tega ali pred njim stoji  $\alpha$  in za njim  $\beta$ .

**Primer 3.6.** Formalizirajmo gramatiko iz Primera 2.3, ki smo jo generirali z nizom w = cababcccababcccab. Označimo jo z  $G_w = (V, \Sigma, P, S)$ , kjer je

$$\begin{split} V &= \{S,A,B,C\},\\ \Sigma &= \{a,b,c\},\\ P &= \{S \rightarrow cCCA,A \rightarrow ab,B \rightarrow ccc,C \rightarrow AAB\},\\ S &= S. \end{split}$$

Vidimo, da  $G_w$  ustreza naši definiciji kontekstno-neodvisne gramatike in res kodira w, saj je

$$L(G_w) = \{w\}.$$

Dolžina gramatike  $G_w$  je enaka številu črk abecede  $\{S, A, B, C, \rightarrow\}$ , ki smo jih porabili za opis gramatike in je enaka |P| = 20. Vidimo, da niza w nismo skrajšali, saj je |w| = 17. Niz w je bil prekratek, da bi ga lahko zares stisnili. Naj bo sedaj

w = cababcccababcccabababccc.

Gramatika, ki jo generira novi niz se od prejšnje gramatike razlikuje le v

$$P = \{S \to cCCAC, A \to ab, B \to ccc, C \to AAB\}.$$

Sedaj smo stisnili  $w z G_w$ , saj je

$$|w| = 24 > 21 = |P|$$
.

 $\Diamond$ 

Za razliko od klasičnih metod stiskanja, naša metoda ne izvaja procesa stiskanja direktno na nizu w, temveč poiščemo gramatiko  $G_w$ , ki generira enojec  $\{w\}$  za svoj jezik. Med njimi poiščemo "najmanjšo", ki jo nato kodiramo. Ker gramatike  $G_w$  generira w in ker je gramatika "majhna" jo bomo kodirali v "kratko" kodo. Tako bomo preko gramatike "dobro" stisnili niz w.