Informacija in kodi

UN2-1-AV 2024/2025

Kodiranje vira

Simon Dobrišek

november 2024

Teme predavanja

Uvod

Znaki, ki jih oddaja vir informacije, pogosto ne zadoščajo tehničnim zahtevam, ki jih postavlja kanal.

Na primer, pri pisni komunikaciji uporabljamo približno sto znakov (velike in male črke abecede, številčni znaki med nič in devet ter ločila).

Če želimo napisano besedilo (sporočilo) shraniti v pomnilniku računalnika ali ga prenesti na oddaljeno mesto po elektronski pošti, moramo znake, ki smo jih uporabljali pri pisanju besedila, zamenjati z ustreznimi nizi dvojiških[1] znakov.

Prirejanju znakov ene abecede znakom druge abecede pravimo *kodiranje*.

[1] Velja za današnjo stopnjo razvoja tovrstne tehnologije.

Uvod

S kodiranjem, ki mu pravimo *kodiranje vira*, prirejamo znakom iz abecede vira znake (nize znakov) iz abecede koda[2]

Vzemimo, da je dan diskreten vir informacije s

- končno abecedo $A = \{x_1, x_2, ..., x_a\}$ ter
- končna množica znakov $B = \{z_1, z_2, ..., z_b\}$, ki ji pravimo abeceda koda.

[2] Abecedo koda sestavljajo znaki, ki jih kanal lahko "sprejme".

Uvod - kod vira

DEFINICIJA 5.1 *Kod vira je trojica (A, B, f), kjer so:*

- *A abeceda vira moči a,*
- *B abeceda koda moči b in*
- *f injektivna[3] preslikava*

$$f: A \longrightarrow E,$$

kjer je $E = B^m$ ali $E = B^1 \cup B^2 \cup \cdots \cup B^m$ množica kodnih zamenjav $f(x_i)$, m pa naravno število.

[3] Preslikava f je injektivna, če je $f(x)$ slika kvečjemu enega $x \in A$.

Uvod - primer

Primer 5.1

Vzemimo, da je $B = \{0, 1\}$ ter $m = 3$. V tem primeru je množica kodnih zamenjav

$$E = B^3 = \{000\,, 001\,, 010\,, 011\,, 100\,, 101\,, 110\,, 111\}$$

ali

$$\begin{aligned} E &= B^1 \cup B^2 \cup B^3 = \\ &= \{0\,, 1\,, 00\,, 01\,, 10\,, 11\,, 000\,, 001\,, 010\,, 011\,, 100\,, 101\,, 110\,, 111\} \end{aligned}$$

$\square$

Uvod - enakomerni in neenakomerni kod

DEFINICIJA 5.2 *Če je $E = B^m$, to je, če je množica kodnih zamenjav $f(x_i)$; $i = 1, 2, ..., a$, sestavljena le iz nizov znakov iz množice B dolžine m, pravimo takšnemu kodu vira* ***enakomerni kod****.*

DEFINICIJA 5.3 *Če je množica kodnih zamenjav sestavljena iz enočlenih, dvočlenih, tročlenih itn. nizov iz množice B, to je, če je $E = B^1 \cup B^2 \cup \cdots B^m$, pravimo takšnemu kodu vira* ***neenakomerni kod****.*

Neenakomerni kod lahko vsebuje kodne zamenjave, ki so predpone drugim kodnim zamenjavam koda.

DEFINICIJA 5.4 *Pravimo, da je k-člena kodna zamenjava $f(x_i)$ predpona l-členi kodni zamenjavi $f(x_j)$, $k < l$, če je $f(x_i)$ enaka prvim k znakom $f(x_j)$. Zadnjim $l - k$ znakom $f(x_j)$ pravimo pripona.*

Uvod - primer

Primer 5.2

Na primer, če sta kodni zamenjavi neenakomernega koda $f(x_i) = 1$ in $f(x_j) = 101$, je $f(x_i)$ predpona $f(x_j)$, niz 01 pa je pripona $f(x_j)$. $\square$

Uporabni so le tisti kodi, ki jih je možno enolično dekodirati.

DEFINICIJA 5.5 *Kod vira (A, B, f) je možno enolično dekodirati, če je možno enolično dekodirati vsak niz kodnih zamenjav iz množice E.*

Enakomerne kode je možno enolično dekodirati, če je $a \leq b^m$, medtem ko je neenakomerne kode možno enolično dekodirati, če prestanejo preizkus z algoritmom **Sardinasa in Pattersona**.

Uvod - algoritem Sardinasa in Pattersona

Začetek postopka

1. **korak** Postavi $E' \leftarrow E$.
2. **korak** V E' poišči par nizov znakov iz abecede koda, kjer je en niz predpona drugemu nizu. Dodaj v E' njuno pripono, vendar le v primeru, da je različna od pripon, ki si jih dodal v E' v predhodnih ponovitvah **2. koraka**.
3. **korak** Ponavljaj **2. korak**, dokler:
   a) v E' ne moreš najti para nizov, kjer je en niz predpona drugemu nizu (končaj preizkus – kod *je* možno enolično dekodirati),
   ali
   b) v E' si dodal pripono, ki je kodna zamenjava iz E (končaj preizkus – kod *ni* možno enolično dekodirati).

Konec postopka

Uvod - trenutni in netrenutni kodi

Kode, ki jih je možno enolično dekodirati, delimo na trenutne in netrenutne kode.

Sporočila, ki so kodirana s trenutnim kodom, lahko sprotno (to je, med sprejemanjem) dekodiramo. Sporočila, ki so kodirana z netrenutnim kodom, lahko dekodiramo šele po sprejetju koda celotnega sporočila.

Vsi enakomerni kodi so trenutni. Neenakomerni kodi so lahko trenutni ali netrenutni.

DEFINICIJA 5.6 *Neenakomerni kod je trenuten kod, če ne vsebuje kodne zamenjave, ki bi bila predpona kakšne druge kodne zamenjave.*

Uvod - primer

Primer 5.3

Vzemimo, da je dan vir informacije, ki oddaja znake iz abecede $A = \{x_1, x_2, x_3\}$, ter abeceda koda $B = \{0, 1\}$.

Oglejmo si naslednje kode vira:

A	kod $\mathcal{A}$	kod $\mathcal{B}_1$	kod $\mathcal{B}_2$	kod $\mathcal{C}$	kod $\mathcal{D}$
x_1	00	0	0	0	0
x_2	01	1	01	10	01
x_3	10	01	10	11	11

Preslikava, ki preslika x_i v $f(x_i)$, je za vse kode injektivna.

Kod $\mathcal{A}$ je enakomerni kod, kodi $\mathcal{B}_1, \mathcal{B}_2, \mathcal{C}$ in $\mathcal{D}$ so neenakomerni.

Enakomeren kod $\mathcal{A}$ lahko enolično dekodiramo, ker je $3 < 2^2$.

Uvod - primer

Ali je možno enolično dekodirati tudi neenakomerne kode $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{C}$ in $\mathcal{D}$ se prepričamo tako, da jih podvržemo preizkusu s postopkom Sardinasa in Pattersona.

kod	1. korak	2. korak	2. korak	3. korak
$\mathcal{B}_1$	$E' = \{0,\ 1, 01\}$	$E' = \{0,\ 1, 01, 1\}$		3 b)
$\mathcal{B}_2$	$E' = \{0, 01, 10\}$	$E' = \{0, 01, 10, 1\}$	$E' = \{0, 01, 10, 1, 0\}$	3 b)
$\mathcal{C}$	$E' = \{0, 10, 11\}$			3 a)
$\mathcal{D}$	$E' = \{0, 01, 11\}$	$E' = \{0, 01, 11, 1\}$		3 a)

Opazimo, da obeh neenakomernih kodov $\mathcal{B}$ ni mogoče enolično dekodirati.

Na primer, niz znakov $x_1 x_2 x_3$ kodiramo s kodom $\mathcal{B}_1$ z nizom znakov 0101, ki ga lahko dekodiramo kot $x_1 x_2 x_3$, $x_1 x_2 x_1 x_2$, $x_3 x_3$ ali $x_3 x_1 x_2$.

Podobno, niz znakov $x_1 x_3 x_3 x_3$ kodiramo s kodom $\mathcal{B}_2$ z nizom znakov 0101010, ki ga lahko dekodiramo kot $x_1 x_3 x_3 x_3$ ali $x_2 x_2 x_2 x_1$.

Ker koda $\mathcal{B}_1$ in $\mathcal{B}_2$ ne omogočata enoličnega dekodiranja vseh sporočil, ki jih odda vir, sta neuporabna.

Uvod - primer

Za neenakomerna koda $\mathcal{C}$ in $\mathcal{D}$ ni možno najti niza kodnih zamenjav, ki ga ne bi bilo možno enolično dekodirati, pri čemer kod $\mathcal{C}$ omogoča sprotno dekodiranje, kod $\mathcal{D}$ pa dekodiranje po sprejemu koda celotnega sporočila.

Vzemimo za primer, da moramo dekodirati niz znakov 0111111, ki ga je ustvaril kod $\mathcal{C}$.

V nizu je prvi znak kodna zamenjava znaka x_1, drugi znak ni kodna zamenjava koda $\mathcal{C}$, zato počakamo na sprejem tretjega znaka koda sporočila.

Drugi in tretji znak koda sporočila sta kodna zamenjava znaka x_3.

Četrti znak koda sporočila ni kodna zamenjava koda $\mathcal{C}$, zato četrti in peti znak, ter podobno šesti in sedmi znak, koda sporočila enolično dekodiramo kot znaka sporočila $x_3 x_3$.

Uvod - primer

Vzemimo sedaj, da moramo dekodirati niz znakov iz abecede koda 0111111, ki ga je ustvaril kod $\mathcal{D}$.

V sprejetem nizu je prvi znak kodna zamenjava znaka x_1, prva dva znaka pa kodna zamenjava znaka x_2.

Da bi odločili, ali je prvi dekodirani znak x_1 ali x_2, moramo počakati na konec sprejema celotnega sporočila.

Če dekodiramo prvi znak sporočila kot x_2, bomo dekodirali naslednja dva znaka kot x_3, zadnji znak sprejetega niza 1 pa ne bomo mogli dekodirati, ker 1 ni kodna zamenjava koda $\mathcal{D}$.

Uvod - primer

Če pa dekodiramo prvi znak sporočila kot x_1, bomo lahko dekodirali naslednjih šest znakov sprejetega niza kot kodne zamenjave niza treh znakov sporočila $x_3x_3x_3$.

Kod $\mathcal{D}$ očitno ni trenuten kod, vendar ga je možno enolično dekodirati.

Kod $\mathcal{C}$ smo lahko sprotno dekodirali, ker ne vsebuje kodne zamenjave, ki bi bila predpona kakšne druge kodne zamenjave.

V kodu $\mathcal{D}$ pa je kodna zamenjava $f(x_1)$ predpona kodne zamenjave $f(x_2)$, zato ga lahko enolično dekodiramo šele po sprejetju koda celotnega sporočila. $\square$

Kodna drevesa

Opazimo, da je možno ločevati trenutne kode od ostalih neenakomernih kodov, ki jih je možno enolično dekodirati, le z ugotavljanjem, da kod ne vsebuje kodne zamenjave, ki bi bila predpona kakšne druge kodne zamenjave.

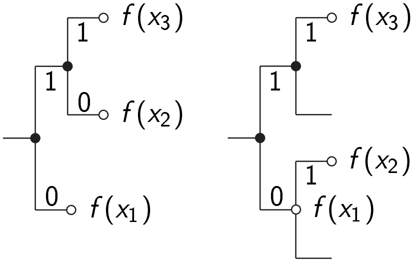
To najlažje naredimo, če kod ponazorimo s *kodnim drevesom*.

- Kodna drevesa so večnivojska drevesa s stopnjami vozlišč, enakimi številu znakov abecede koda b.
- Veje, ki izstopajo iz vozlišč, označimo z znaki iz množice $B = \{z_1, z_2, ..., z_b\}$.
- Kodne zamenjave določajo nizi oznak (zaporedij) vej, ki se začnejo v korenu drevesa in končujejo v kateremkoli vozlišču v notranjosti drevesa.

Kodna drevesa - primer

Primer 5.4

Kodni drevesi kodov $\mathcal{C}$ in $\mathcal{D}$ iz primera 5.3 sta ponazorjeni na sliki.



Opazimo, da se kodne zamenjave koda $\mathcal{D}$ nahajajo tudi v notranjih vozliščih kodnega drevesa, kar ne velja za kod $\mathcal{C}$, ki je trenutni.

Za trenutne kode namreč velja, da se kodna drevesa ne vejijo od izbrane kodne zamenjave naprej, ker v trenutnem kodu nobena kodna zamenjava ni predpona kakšni drugi kodni zamenjavi. □

Gospodarno kodiranje

Če si pri kodiranju vira zastavimo še nalogo, da bomo informacijo, ki jo odda vir informacije, zakodirali z ***najmanjšim*** možnim številom znakov iz abecede koda, pravimo takšnemu kodiranju vira ***gospodarno kodiranje***.

Vzemimo, da diskreten vir odda znak x_i z verjetnostjo

$$P(x_i) = p_i, \qquad i = 1, ..., a, \qquad \sum_{i=1}^{a} p_i = 1,$$

ter da je dolžina kodne zamenjave $f(x_i)$ v kodu (A, B, f) enaka n_i (n_i je število znakov abecede B v kodni zamenjavi $f(x_i) \in E$).

Mera gospodarnosti koda

DEFINICIJA 5.7 *Mera gospodarnosti koda vira (A, B, f) je povprečna dolžina njegovih kodnih zamenjav*

$$\overline{n} = \sum_{i=1}^{a} p_i n_i .$$

Oglejmo si, kateri izmed štirih kodov iz primera 5.3 je najgospodarnejši, to je, kateri izmed štirih kodov ima najmanjšo povprečno dolžino kodnih zamenjav $\overline{n}$.

Mera gospodarnosti koda

Primer 5.5

Vzemimo, da vir informacije iz primera 5.3 oddaja znake iz abecede $A = \{x_1, x_2, x_3\}$ v skladu s porazdelitvijo verjetnosti $P = (0{,}5,\ 0{,}3,\ 0{,}2)$.

V tem primeru je povprečna dolžina kodnih zamenjav koda $\mathcal{A}$ enaka $\overline{n}_{\mathcal{A}} = p_1 n_1 + p_2 n_2 + p_3 n_3 = 0{,}5 \cdot 2 + 0{,}3 \cdot 2 + 0{,}2 \cdot 2 = 2{,}0$, povprečna dolžina kodnih zamenjav koda $\mathcal{B}_1$ je $\overline{n}_{\mathcal{B}_1} = 1{,}2$, povprečne dolžine kodnih zamenjav kodov $\mathcal{B}_2, \mathcal{C}$ in $\mathcal{D}$ pa: $\overline{n}_{\mathcal{B}_2} = \overline{n}_{\mathcal{C}} = \overline{n}_{\mathcal{D}} = 1{,}5$. □

S stališča gospodarnosti lahko trdimo, da je enakomerni kod $\mathcal{A}$ najslabši, neenakomerni kod $\mathcal{B}_1$ pa najboljši, ker omogoča kodiranje sporočil z najmanjšo povprečno dolžino kodnih zamenjav.

Vendar moramo nalogo iskanja gospodarnih kodov omejiti na tiste neenakomerne kode, ki jih je možno enolično dekodirati.

Enakomerni kodi

Enakomerni kodi so praviloma negospodarni kodi vira (glej primer 5.5).

Vendar se zaradi preprostih postopkov kodiranja in dekodiranja ter njihove neodvisnosti od statističnih lastnosti vira v računalniških sistemih za obdelavo in prenos informacije za prilagoditev vira tehničnim zahtevam kanala uporabljajo skoraj izključno enakomerni kodi.

Gospodarno kodiranje vira, ki ga omogočajo neenakomerni kodi, se nato udejanji v procesu obdelave informacije nad enakomernimi kodi danega vira informacije.

Kod ASCII

Kod ASCII je enakomerni kod (A, B, f), kjer so:

- A množica alfanumeričnih znakov z močjo 128,
- B abeceda koda $\{0, 1\}$ in
- f preslikava, ki je standardizirana in podana tabelarično. (Glej tabelo 9.2.1 v Dodatku učbenika *Informacija in kodi*.)

V ameriških stadardih ANSI (American National Standard Institution) je kod dobil ime ASCII (American Standard Code for Information Interchange).

Vse kodne zamenjave koda ASCII so dolge $m = 7$ (dvojiških) znakov. Navadno se kodnim zamenjavam doda še en (osmi) znak, ki služi za preverjanje sodosti, to je za odkrivanje lihega števila napak na kodnih zamenjavah pri njihovem prenosu po kanalu.

Drugi standardni enakomerni kodi

Danes obstaja vrsta standardnih enakomernih in neenakomernih kodov (kodnih tabel), ki so v glavnem izšli iz koda ASCII in so bili razviti za kodiranje znakov različnih jezikov in jezikovnih skupin.

- Mednarodna organizacija ISO je določila vrsto kodov ISO-8859-1, ISO-8859-2, ...
- Podjetje IBM je v sodelovanju z ANSI določilo kode CP850, CP852, CP855, ...
- Podjetje Microsoft je določilo svoje kode Windows-1250, Windows-1251, ...
- Mednarodni konzorcij UNICODE je v sodelovanju z IEEE določil v zadnjem času najbolj pogosto uporabljene univerzalne mednarodne kode UTF-8, UTF-16, UTF-32.

Za kodiranje znakov abecede slovenščine pridejo v poštev kodne tabele CP852, Windows-1250, ISO-8859-2, UTF-8, UTF-16, UTF-32.

Neenakomerni kodi

Gospodarne kode iščemo med neenakomernimi kodi, ki jih lahko enolično dekodiramo, zato najprej raziščimo pogoje, pod katerimi lahko sestavimo enolične neenakomerne kode.

Temeljni rezultat povzemata Kraftov in McMillanov izrek.

IZREK 5.1 (Kraftov izrek) *Če neenakomerni kod (A, B, f) z dolžinami kodnih zamenjav $n_1, n_2, \ldots, n_a$ zadošča neenačbi*

$$\sum_{i=1}^{a} b^{-n_i} \leq 1, \qquad \textit{(neenačba Krafta in McMillana)} \qquad (1)$$

obstaja trenutni kod z dolžinami kodnih zamenjav $n_1, n_2, \ldots, n_a$.

Dokaz: Glej učbenik *Informacija in kodi*, str. 71.

Neenakomerni kodi - neenačba Krafta in McMillana

IZREK 5.2 (McMillanov izrek) *Za vsak kod* (A, B, f) *z dolžinami kodnih zamenjav* $n_1, \ldots, n_a$*, ki omogoča enolično dekodiranje, velja*

$$\sum_{i=1}^{a} b^{-n_i} \leq 1. \qquad \textit{(neenačba Krafta in McMillana)}$$

Dokaz: Glej učbenik *Informacija in kodi*, str. 72.

Oglejmo si, ali kodi iz primera 5.3 zadoščajo neenačbi Krafta in McMillana.

Neenakomerni kodi - primer

Primer 5.6

Kodi iz primera 5.3:

A	kod $\mathcal{A}$	kod $\mathcal{B}_1$	kod $\mathcal{B}_2$	kod $\mathcal{C}$	kod $\mathcal{D}$
x_1	00	0	0	0	0
x_2	01	1	01	10	01
x_3	10	01	10	11	11

Kod $\mathcal{B}_1$ ne zadošča neenačbi Krafta in McMillana, ker

$$\sum_{i=1}^{a} b^{-n_i} = 2^{-1} + 2^{-1} + 2^{-2} = \frac{5}{4} > 1,$$

zato ne moremo sestaviti enoličnega neenakomernega (netrenutnega ali trenutnega) koda z dolžinami kodnih zamenjav: $n_1 = 1$, $n_2 = 1$ in $n_3 = 2$.

Neenakomerni kodi

Kod $\mathcal{B}_2$ zadošča neenačbi Krafta in McMillana, ker

$$\sum_{i=1}^{a} b^{-n_i} = 2^{-1} + 2^{-2} + 2^{-2} = 1 \ \leq 1,$$

zato lahko sestavimo enoličen neenakomeren kod z dolžinami kodnih zamenjav: $n_1 = 1$, $n_2 = 2$ in $n_3 = 2$ (glej koda $\mathcal{D}$ in $\mathcal{C}$ iz primera 5.3).

Kod $\mathcal{C}$ je trenuten kod, zato po pričakovanju zadošča neenačbi Krafta in McMillana, ki je enaka gornji neenačbi

Kod $\mathcal{D}$ je neenakomeren kod, ki ga je možno enolično dekodirati, vendar ni trenuten. Ker zadošča neenačbi Krafta in McMillana (enaka kot zgoraj) lahko sestavimo trenuten kod z istimi dolžinami kodnih zamenjav (glej kod $\mathcal{C}$ iz primera 5.3). $\square$

Gospodarni kodi

Raziščimo pogoje, pod katerimi je možno sestaviti enolične neenakomerne kode z najmanjšo povprečno dolžino kodnih zamenjav $\overline{n}$.

Najpomembnejši rezultat je vsebovan v izrekih 5.3 – 5.5, ki jih pripisujemo Shannonu in Fanoju.

Gospodarni kodi

IZREK 5.3 *Naj bo dan diskreten stacionaren vir informacije brez spomina, v katerem znak* $x_i \in A$ *oddamo z verjetnostjo* $p_i \geq 0$ $(i = 1, 2, ..., a;\ \sum_{i=1}^{a} p_i = 1)$*. Tedaj obstaja neenakomerni kod* (A, B, f) *s povprečno dolžino kodnih zamenjav* $\overline{n}$*, ki ga je možno enolično dekodirati, in velja*

$$\frac{H}{K \log_d b} \leq \overline{n} < \frac{H}{K \log_d b} + 1, \tag{2}$$

kjer je

$$H = -K \sum_{i=1}^{a} p_i \log_d p_i$$

entropija diskretnega stacionarnega vira brez spomina.

Dokaz: Glej učbenik *Informacija in kodi*, str. 74.

Gospodarni kodi

Za stacionarne vire brez spomina bi v lahko v dokazu izreka 5.3 pokazali, da za vsak kod (A, B, f), ki omogoča enolično dekodiranje, velja

$$\overline{n} \geq \frac{H}{K \log_d b}.$$

$H/(K \log_d b)$ je torej najmanjša povprečna dolžina kodnih zamenjav, ki jo dosežemo le, če lahko zapišemo verjetnosti znakov abecede vira p_i $(i = 1, 2, ..., a)$ kot celoštevilčne potence osnove b.

Vendar pa se lahko spodnji meji poljubno približamo tudi, če namesto posameznih znakov kodiramo r-terice (*bloke*) znakov, ki jih obravnavamo kot posamezne znake hiperabecede A^r.

Gospodarni kodi

IZREK 5.4 *Naj bo dan diskreten stacionaren vir informacije brez spomina, v katerem znak* $x_i \in A$ *oddamo z verjetnostjo* $p_i \geq 0$ $(i = 1, 2, ..., a;\ \sum_{i=1}^{a} p_i = 1)$. *Tedaj obstaja trenutni kod* (A^r, B, f), *ki omogoča, da se s povprečno dolžino kodnih zamenjav* $\overline{n}$ *poljubno približamo spodnji meji*

$$\frac{H}{K \log_d b}, \tag{3}$$

kjer je H *entropija diskretnega stacionarnega vira informacije brez spomina.*

Dokaz: Glej učbenik *Informacija in kodi*, str. 75.

Iz dokaza izreka 5.4 sledi, da se gospodarnost koda s podaljševanjem blokov znakov, ki jim prirejamo kodne zamenjave[4], povečuje.

[4] Z večanjem r se eksponentno podaljšujeta tudi časa, ki sta potrebna za kodiranje in dekodiranje sporočil, kar je seveda slabo.

Gospodarni kodi

Rezultat izreka 5.4 lahko posplošimo, da velja za vsak diskreten stacionaren vir informacije.

IZREK 5.5 (Shannonov izrek o gospodarnem kodiranju) *Naj bo dan diskreten stacionaren vir informacije, v katerem znak* $x_i \in A$ *oddamo z verjetnostjo* $p_i \geq 0$ $(i = 1, 2, ..., a;\ \sum_{i=1}^{a} p_i = 1)$. *Tedaj obstaja trenutni kod* (A^r, B, f), *ki omogoča, da se povprečna dolžina kodnih zamenjav* $\overline{n}$ *poljubno približa meji*

$$\frac{H}{K \log_d b}, \tag{4}$$

kjer je H *entropija diskretnega stacionarnega vira informacije.*

Dokaz: Glej učbenik *Informacija in kodi*, str. 76.

Gospodarni kodi

Za trenutni kod s povprečno dolžino kodnih zamenjav, vsebovano v intervalu:

$$
\begin{aligned}
\frac{H}{K \log_d b} &\leq \overline{n} < \frac{H}{K \log_d b} + 1, \\
\frac{H}{K \log_d b} &\leq \frac{\overline{n}_r}{r} < \frac{H}{K \log_d b} + \frac{1}{r}
\end{aligned}
$$

oziroma

$$
\frac{H(X_1, ..., X_r)}{r} \frac{1}{K \log_d b} \leq \overline{n} < \frac{H(X_1, ..., X_r)}{r} \frac{1}{K \log_d b} + \frac{1}{r},
$$

pravimo, da je ***gospodaren***, za gospodarni kod s povprečno dolžino kodnih zamenjav, ki je najbližja vrednosti $H/(K \log_d b)$ pa pravimo, da je ***(globalno) optimalen***.

Gospodarni kodi

Za iskanje gospodarnih kodov je pomembna naslednja posledica izrekov 5.1-5.5, kakor tudi trditvi 5.1 in 5.2, ki govorita o lastnostih gospodarnih kodov.

POSLEDICA 5.1 *Če je kod (A, B, f) gospodaren kod v razredu enoličnih neenakomernih kodov s fiksiranima a in b, to je, če je $\overline{n}(f) \leq \overline{n}(g)$, kjer je $\overline{n}(g)$ povprečna dolžina kodnih zamenjav v kateremkoli kodu (A, B, g) iz istega razreda, potem v istem razredu obstaja trenutni kod (A, B, h), za katerega velja*

$$\overline{n}(h) = \overline{n}(f). \tag{5}$$

Dokaz: Glej učbenik *Informacija in kodi*, str. 77.

Gospodarni kodi

Vidimo, da se lahko pri iskanju gospodarnih kodov omejimo le na množico ***trenutnih*** kodov, to je kodov z dolžinami kodnih zamenjav, ki zadoščajo neenačbi Krafta in McMillana.

TRDITEV 5.1 *Če je kod (A, B, f) gospodaren kod, potem velja :*

$$p_j > p_k \Longrightarrow n_j \le n_k\,, \qquad j \neq k, \quad j, k \in \{1, 2, ..., a\}.$$

Ta trditev pravi, da v gospodarnem kodu znakom z večjo verjetnostjo pripisujemo krajše kodne zamenjave.

Dokaz: Glej učbenik *Informacija in kodi*, str. 77.

Gospodarni kodi

TRDITEV 5.2 *Obstaja takšen gospodaren trenutni kod (A, B, f), da imata kodni zamenjavi $f(x_{a-1})$ in $f(x_a)$ enaki dolžini $n_{a-1} = n_a$, razlikujeta pa se le v zadnjem znaku.*

Dokaz: Glej učbenik *Informacija in kodi*, str. 78.

V nadaljevanju si oglejmo tri postopke gradnje gospodarnih kodov $(A, \{0, 1\}, f)$.

Huffmanov kod

Huffmanov algoritem omogoča sestavljanje gospodarnega trenutnega koda za podani množici $A = \{x_1, x_2, ..., x_a\}$ in $B = \{0, 1\}$ $(a > 2)$ in podano porazdelitev verjetnosti $p_1, p_2, ..., p_a$ $(p_i \geq 0,\ \sum_{i=1}^{a} p_i = 1)$.

Začetek postopka

1. **korak** Naredi:
   a) postavi $A_0 \leftarrow A$,
   b) znake v A_0 uredi tako, da je $p_1 \geq p_2 \geq \ldots \geq p_a$,
   c) znaku $x_{a-1} \in A_0$ pripiši znak $0 \in B$, znaku $x_a \in A_0$ pa znak $1 \in B$.

Huffmanov kod

2. korak

za $j \leftarrow 1$ **do** $a-2$ **naredi**:

a) sestavi množico A_j tako, da združiš zadnja znaka množice A_{j-1} v en znak in mu pripiši verjetnost, ki jo dobimo kot vsoto verjetnosti združenih znakov. Množica A_j ima $a_j = a - j$ znakov,
b) množico A_j uredi tako, da je $p_1 \geq p_2 \geq \ldots \geq p_{a_j}$,
c) znaku $x_{a_j-1} \in A_j$ pripiši znak $0 \in B$, znaku $x_{a_j} \in A_j$ pa znak $1 \in B$.

konec-za

3. korak Kodno zamenjavo za znak abecede vira x_i sestavimo tako, da vse znake abecede B, ki so se pojavljali z indeksom i, jemljemo po vrsti od konca proti začetku postopka.

Konec postopka

Huffmanov kod - primer

Primer 5.7

Vzemimo, da je dan vir brez spomina, ki oddaja znake iz abecede $A = \{x_1, x_2, x_3\}$; $a = 3$.

Vzemimo naprej, da so verjetnosti oddaje znakov iz A enake $P(x_1) = 0{,}60$, $P(x_2) = 0{,}30$ in $P(x_3) = 0{,}10$ ter abeceda koda $B = \{0, 1\}$; $b = 2$. Poišči Huffmanov kod znakov iz A.

Huffmanov kod - primer

Spodnja tabela vsebuje a) ponazoritev poteka Huffmanovega algoritma in b) tabelo Huffmanovega koda (znaki iz abecede vira in njihove kodne zamenjave).

<table>
<tr><td>x_1</td><td>x_2</td><td>x_3</td></tr>
<tr><td>0,6</td><td>0,3</td><td>0,1</td></tr>
<tr><td></td><td>0</td><td>1</td></tr>
<tr><td>x_1</td><td colspan="2">x_{23}</td></tr>
<tr><td>0,6</td><td colspan="2">0,4</td></tr>
<tr><td>0</td><td colspan="2">1</td></tr>
</table>

a)

x_i	$f(x_i)$
x_1	0
x_2	10
x_3	11

b)

Huffmanov kod - primer

Povprečna dolžina kodnih zamenjav $\overline{n}$, entropija vira brez spomina H in *uspešnost* koda η, ki jo definiramo kot razmerje med entropijo vira in povprečno dolžino kodnih zamenjav, so:

$$\overline{n} = \sum_{i=1}^{3} p_i n_i = 1{,}40 \text{ znaka},$$
$$H = -\sum_{i=1}^{a} p_i \log_2 p_i = 1{,}295 \text{ bitov in}$$
$$\eta = H/\overline{n} \cdot 100\% = 92{,}5\ \%.$$

□

Huffmanov kod dekodiramo s pomočjo kodne tabele, ki jo (obvezno) *pripnemo* nizu kodnih zamenjav danega sporočila.

Huffmanov kod - primer

Primer 5.8

Oglejmo si, kako dekodiramo niz kodnih zamenjav 11100010011 s pomočjo tabele Huffmanovega koda iz primera 5.7.

- Dekodiranje začnemo takoj po sprejetju prvega znaka kodiranega sporočila.
- Preverimo, če vsebuje kodna tabela (b) zamenjavo 1.
- Ker je ne vsebuje, počakamo na sprejem drugega znaka.
- Preverimo, če vsebuje kodna tabela zamenjavo 11.
- Ugotovimo, da je niz 11 kodna zamenjava znaka abecede vira x_3.
- Počakamo na sprejem tretjega znaka. Ker kodna tabela ne vsebuje zamenjave 1, počakamo na sprejem četrtega znaka. Preverimo, če vsebuje kodna tabela zamenjavo 10.
- Ugotovimo, da je niz 10 kodna zamenjava znaka abecede vira x_2, itn. dokler ne dekodiramo tudi zadnje kodne zamenjave sprejetega sporočila.

Rezultat dekodiranja niza kodnih zamenjav 11100010011 je niz znakov abecede vira $x_3 x_2 x_1 x_1 x_2 x_1 x_3$. □

Huffmanov kod

Gospodarnost koda povečamo, če namesto posameznih znakov s Huffmanovim algoritmom kodiramo r-terice znakov v sporočilu.

Pri tem, če dolžina sporočila, ki ga kodiramo, ni celi večkratnik števila r, na koncu sporočila pripnemo med 1 in $r-1$ 'lažnih' znakov[5].

[5] Znakov z verjetnostjo oddaje 0. V angleški literaturi: *dummy*.

Huffmanov kod - primer

Primer 5.9

Vzemimo, da sta dana vir brez spomina iz primera 5.7 ter abeceda koda $B = \{0, 1\}$. Poišči Huffmanov kod znakov iz A^2.

Ker bomo kodirali pare znakov iz A, tvorimo množici

$$\begin{aligned} A^2 = \{x_1^* &= x_1x_1,\ x_2^* = x_1x_2,\ x_3^* = x_2x_1,\ x_4^* = x_2x_2,\ x_5^* = x_1x_3, \\ x_6^* &= x_3x_1,\ x_7^* = x_2x_3,\ x_8^* = x_3x_2,\ x_9^* = x_3x_3\} \quad \text{ter} \end{aligned}$$

$$\begin{aligned} P_2 = \{P(x_1^*) &= P(x_1x_1) = P(x_1)P(x_1) = 0{,}36,\ P(x_2^*) = 0{,}18, \\ P(x_3^*) &= 0{,}18,\ P(x_4^*) = 0{,}09,\ P(x_5^*) = 0{,}06,\ P(x_6^*) = 0{,}06, \\ P(x_7^*) &= 0{,}03,\ P(x_8^*) = 0{,}03,\ P(x_9^*) = 0{,}01\}. \end{aligned}$$

Huffmanov kod - primer

<table>
<tr><td>x_1^*</td><td>x_2^*</td><td>x_3^*</td><td>x_4^*</td><td>x_5^*</td><td>x_6^*</td><td>x_7^*</td><td>x_8^*</td><td>x_9^*</td></tr>
<tr><td>0,36</td><td>0,18</td><td>0,18</td><td>0,09</td><td>0,06</td><td>0,06</td><td>0,03</td><td>0,03</td><td>0,01</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td><td>1</td></tr>
<tr><td>x_1^*</td><td>x_2^*</td><td>x_3^*</td><td>x_4^*</td><td>x_5^*</td><td>x_6^*</td><td colspan="2">x_{89}^*</td><td>x_7^*</td></tr>
<tr><td>0,36</td><td>0,18</td><td>0,18</td><td>0,09</td><td>0,06</td><td>0,06</td><td colspan="2">0,04</td><td>0,03</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">0</td><td>1</td></tr>
<tr><td>x_1^*</td><td>x_2^*</td><td>x_3^*</td><td>x_4^*</td><td colspan="3">x_{789}^*</td><td>x_5^*</td><td>x_6^*</td></tr>
<tr><td>0,36</td><td>0,18</td><td>0,18</td><td>0,09</td><td colspan="3">0,07</td><td>0,06</td><td>0,06</td></tr>
<tr><td></td><td></td><td></td><td></td><td colspan="3"></td><td>0</td><td>1</td></tr>
<tr><td>x_1^*</td><td>x_2^*</td><td>x_3^*</td><td colspan="2">x_{56}^*</td><td>x_4^*</td><td colspan="3">x_{789}^*</td></tr>
<tr><td>0,36</td><td>0,18</td><td>0,18</td><td colspan="2">0,12</td><td>0,09</td><td colspan="3">0,07</td></tr>
<tr><td></td><td></td><td></td><td colspan="2"></td><td>0</td><td colspan="3">1</td></tr>
<tr><td>x_1^*</td><td>x_2^*</td><td>x_3^*</td><td colspan="4">x_{4789}^*</td><td colspan="2">x_{56}^*</td></tr>
<tr><td>0,36</td><td>0,18</td><td>0,18</td><td colspan="4">0,16</td><td colspan="2">0,12</td></tr>
<tr><td></td><td></td><td></td><td colspan="4">0</td><td colspan="2">1</td></tr>
<tr><td>x_1^*</td><td colspan="6">x_{456789}^*</td><td>x_2^*</td><td>x_3^*</td></tr>
<tr><td>0,36</td><td colspan="6">0,28</td><td>0,18</td><td>0,18</td></tr>
<tr><td></td><td colspan="6"></td><td>0</td><td>1</td></tr>
<tr><td>x_1^*</td><td colspan="2">x_{23}^*</td><td colspan="6">x_{456789}^*</td></tr>
<tr><td>0,36</td><td colspan="2">0,36</td><td colspan="6">0,28</td></tr>
<tr><td></td><td colspan="2">0</td><td colspan="6">1</td></tr>
<tr><td colspan="8">$x_{23456789}^*$</td><td>x_1^*</td></tr>
<tr><td colspan="8">0,64</td><td>0,36</td></tr>
<tr><td colspan="8">0</td><td>1</td></tr>
</table>

a)

x_i^*	$f(x_i^*)$
x_1^*	1
x_2^*	000
x_3^*	001
x_4^*	0100
x_5^*	0110
x_6^*	0111
x_7^*	01011
x_8^*	010100
x_9^*	010101

b)

Huffmanov kod - primer

Povprečna dolžina kodnih zamenjav iz tabele b) $\overline{n}_2$ in uspešnost koda η sta:

$$\overline{n}_2 = \sum_{i=1}^{9} P(x_i^*) \cdot n_i = 2{,}67 \text{ znaka in}$$
$$\eta = H/\overline{n} \cdot 100\% = 2H/\overline{n}_2 \cdot 100\% = 97{,}0 \text{ \%}.$$

Ker je uspešnost Huffmanovega koda iz primera 5.7 bila 92,5 %, sledi, da je kodiranje parov znakov iz abecede vira povečalo uspešnost Huffmanovega koda za 4.5 %.

Opazimo, da je razsežnost tabele Huffmanovega koda parov znakov iz abecede A enaka $a^r = 3^2$, torej, da eksponentno narašča s podaljševanjem blokov sporočila, ki jih kodiramo. □

Aritmetični kod

Tako kot Huffmanov algoritem, tudi algoritem aritmetičnega kodiranja temelji na poznavanju verjetnosti r-teric (blokov), ki jih kodiramo, vendar omogoča dekodiranje koda danega sporočila brez kodne tabele.

Zadošča, da poznamo abecedo vira informacije in verjetnosti oddaje posameznih znakov.

Tudi če dolžina sporočila, ki ga kodiramo, ni celi večkratnik dolžine bloka, sporočilu ni potrebno pripenjati *lažnih znakov*, ker dolžina blokov, med kodiranjem sporočila, ni nujno stalna.

Aritmetični kod

Zasnovo aritmetičnega kodiranja lahko strnemo v naslednje korake:

a) Interval realnih števil $[0, 1)$ razdelimo na a (pod)intervalov tako, da priredimo vsakemu znaku x_i iz abecede vira A interval $\mathcal{R}(x_i) = [s_i, z_i)$, ki je sorazmeren z njegovo verjetnostjo oddaje. Pri tem velja: $s_1 = 0$, $z_a = 1$, $s_i = z_{i-1}$ za $1 < i \leq a$ ter $\mathcal{R}(x_i) \cap \mathcal{R}(x_j) = \emptyset$, če $x_i \neq x_j$. Torej:

$$\begin{aligned} \mathcal{R}(x_1) &= [\,0,\, z_1\,), \\ \mathcal{R}(x_2) &= [\,s_2 = z_1,\, z_2\,), \end{aligned}$$

in tako naprej do

$$\mathcal{R}(x_a) = [\,s_a = z_{a-1},\, 1\,).$$

Aritmetični kod

b) Kodiranje r-terice (bloka) znakov $\mathrm{x} = x_1 x_2 \cdots x_r$ temelji na r-kratni preslikavi intervala

$$[\, s, z\,) \longrightarrow [\, s', z'\,), \tag{6}$$

kjer sta:

$$s' = s + (z - s) \cdot s_j,$$

in

$$z' = s + (z - s) \cdot z_j.$$

c) Aritmetični kod bloka znakov x je poljubno realno število, ki leži na r-tem intervalu $[s', z')$.

Aritmetični kod - algoritem

VHOD:

1. abeceda vira informacije $A = \{x_1, ..., x_i, ..., x_a\}$,
2. porazdelitev verjetnosti $P = (p_1, ..., p_i, ..., p_a)$, kjer je $p_i = P(x_i)$,
3. r-terica (blok) znakov $\mathrm{x} = x_1 \cdots x_j \cdots x_r$ iz abecede vira A, ki ga želimo gospodarno kodirati. Znak $x_r \in A$ je poseben znak (PZ), ki označuje konec bloka x, na primer pika na koncu stavka[6].

IZHOD:

Kodna zamenjava $f(\mathrm{x})$ bloka x, to je niz znakov iz abecede koda $B = \{0, 1\}$.

---

[6] Če kodiramo bloke fiksne dolžine, znak PZ, ki označuje konec bloka, ni potreben.

Aritmetični kod - algoritem

Začetek postopka

1. **korak** Priredi vsakemu znaku $x_i \in A$ interval realnih števil $\mathcal{R}(x_i) = [s_i, z_i)$, ki je sorazmeren verjetnosti znaka p_i tako, da velja: $s_1 = 0$, $z_a = 1$, $s_i = z_{i-1}$ za $1 < i \leq a$ ter $\mathcal{R}(x_i) \cap \mathcal{R}(x_j) = \emptyset$, če $x_i \neq x_j$.
2. **korak** Postavi: $s \leftarrow 0$, $z \leftarrow 1$ in $j \leftarrow 1$
3. **korak** Dokler $j \neq r$, ponavljaj:
   - a) za znak x_j iz bloka znakov x izračunaj:
     - a1) $s \leftarrow s + (z - s) \cdot s_j$
     - a2) $z \leftarrow s + (z - s) \cdot z_j$
   - b) $j \leftarrow j + 1$
4. **korak** Zapiši s in z v dvojiškem številskem sistemu
5. **korak** Če je $(s)_2 = 0, a_1 a_2 ... a_{t-1} 0...$ in $(z)_2 = 0, a_1 a_2 ... a_{t-1} 1...$, potem je niz dvojiških znakov $a_1 a_2 ... a_{t-1} 1$ kodna zamenjava $f(\mathrm{x})$ r-terice znakov $\mathrm{x} \in A^r$.

Konec postopka

Aritmetični kod - primer

Primer 5.10

Dani sta abeceda vira

$$A = \{x_1, x_2, x_3, x_4 = \mathrm{PZ}\}, \quad a = 4,$$

in verjetnosti znakov v sporočilu:

$$P(x_1) = 0{,}5,\ P(x_2) = 0{,}1,\ P(x_3) = 0{,}3,\ P(x_4) = 0{,}1.$$

Vzemimo, da je r-terica znakov, ki jo želimo gospodarno zakodirati z aritmetičnim kodom, enaka

$$x_1 x_2 x_1 x_3 \mathrm{PZ}.$$

Interval realnih števil $[0, 1)$ najprej razdelimo na $a = 4$ podintervale:

$$\mathcal{R}(x_1) = [0,\ 0{,}5),\ \mathcal{R}(x_2) = [0{,}5,\ 0{,}6),\ \mathcal{R}(x_3) = [0{,}6,\ 0{,}9),\ \mathcal{R}(x_4) = [0{,}9,\ 1),$$

Aritmetični kod - primer

nato zakodiramo dani blok znakov $x_1 x_2 x_1 x_3 \mathrm{PZ}$ z $r = 5$-imi preslikavami (6):

$$[0,1) \xrightarrow{x_1} [s' = 0 + (1-0)\cdot 0 = 0,\ z' = 0 + (1-0)\cdot 0{,}5 = 0{,}5) \xrightarrow{x_2}$$

$$\xrightarrow{x_2} [0{,}25,\ 0{,}3) \xrightarrow{x_1} [0{,}25,\ 0{,}275) \xrightarrow{x_3} [0{,}265,\ 0{,}2725) \xrightarrow{\mathrm{PZ}} [0{,}27175,\ 0{,}2725).$$

Ker sta $(0{,}27175)_{10} = (0{,}0100010110\ldots)_2$[7] in
$(0{,}2725)_{10} = (0{,}0100010111\ldots)_2$,
je aritmetični kod danega sporočila 0100010111. □

[7] Pretvorbo necelega desetiškega števila 0,27175 v dvojiško število ponazarja naslednja razpredelnica:

$$\begin{array}{lllll}
0{,}27175 \times 2 = & 0{,}54350 \times 2 = & 1{,}08700 & & \\
0{,}08700 \times 2 = & 0{,}174 \times 2 = & 0{,}348 \times 2 = & 0{,}696 \times 2 = & 1{,}392 \\
0{,}392 \times 2 = & 0{,}784 \times 2 = & 1{,}568 & & \\
0{,}568 \times 2 = & 1{,}136 & & & \\
0{,}136 \times 2 = & 0{,}272 \times 2 = & \ldots & &
\end{array}$$

Aritmetični kod - dekodiranje

Dekodiranje aritmetičnega koda temelji na večkratni preslikavi dvojiškega števila

$$v = 0, a_1 a_2 ... a_{t-1} 1 \longrightarrow v',$$

kjer je

$$v' = \frac{v - s_j}{z_j - s_j}. \tag{7}$$

Za dan v in za vsak novi v' iščemo interval $\mathcal{R}(x_j)$, v katerem se ta nahaja, ter ta del sporočila dekodiramo z znakom $x_j \in A$.

Aritmetični kod - algoritem dekodiranja

VHOD:

1. abeceda vira informacije $A = \{x_1, ..., x_i, ..., x_a\}$,
2. porazdelitev verjetnosti $P = (p_1, ..., p_i, ..., p_a)$, kjer je $p_i = P(x_i)$,
3. kodna zamenjava $f(\mathrm{x})$ bloka znakov x, to je niz znakov $a_1 a_2 ... a_{t-1} 1$ iz abecede koda $B = \{0, 1\}$.

IZHOD: r-terica (blok) znakov iz abecede vira $\mathrm{x} = x_1 \cdots x_j \cdots x_r$, kjer sta $x_j \in A$ $(j = 1, 2, ..., r)$ in $x_r = \mathrm{PZ}$.

Aritmetični kod - algoritem dekodiranja

Začetek postopka

1. **korak** Priredi vsakemu znaku $x_i \in A$ interval realnih števil $\mathcal{R}(x_i) = [s_i, z_i)$, ki je sorazmeren verjetnosti znaka p_i tako, da velja: $s_1 = 0$, $z_a = 1$, $s_i = z_{i-1}$ za $1 < i \leq a$ ter $\mathcal{R}(x_i) \cap \mathcal{R}(x_j) = \emptyset$, če $x_i \neq x_j$.
2. **korak** Niz dvojiških znakov $a_1 a_2 ... a_{t-1} 1$ obravnavaj kot dvojiško število $v = 0, a_1 a_2 ... a_{t-1} 10 \cdots 0$ in ga zapiši v desetiškem številskem sistemu.
3. **korak** Postavi $j \leftarrow 1$.
4. **korak** Dokler $x_j \neq \mathrm{PZ}$, ponavljaj:
   a) najdi znak $x_j \in A$ tako, da bo $v \in \mathcal{R}(x_j)$,
   b) vpiši znak x_j v niz x,
   c) izračunaj $v \leftarrow (v - s_j)/(z_j - s_j)$,
   d) $j \leftarrow j + 1$

Konec postopka

Aritmetični kod - primer dekodiranja

Primer 5.11

Dani sta abeceda vira

$$A = \{x_1, x_2, x_3, x_4 = \mathrm{PZ}\}, \quad a = 4,$$

in verjetnosti znakov v bloku znakov:

$$P(x_1) = 0{,}5,\ P(x_2) = 0{,}1,\ P(x_3) = 0{,}3,\ P(x_4) = 0{,}1.$$

Interval realnih števil $[0, 1)$ razdelimo na $a = 4$ podintervale:

$\mathcal{R}(x_1) = [0,\ 0{,}5)$, $\mathcal{R}(x_2) = [0{,}5,\ 0{,}6)$, $\mathcal{R}(x_3) = [0{,}6,\ 0{,}9)$, $\mathcal{R}(x_4) = [0{,}9,\ 1)$.

Vzemimo, da je kod bloka znakov x, ki ga želimo dekodirati, enak 0100010111. Tega zapišemo kot

$$v = (0{,}01000101110 \cdots 0)_2 = 2^{-2} + 2^{-6} + 2^{-8} + 2^{-9} + 2^{-10} = (0{,}2724609375)_{10}.$$

Aritmetični kod - primer dekodiranja

Vrednost $v = 0{,}2724609375$ dekodiramo kot x_1, ker je $0,2724609375 \in \mathcal{R}(x_1) = [0,\ 0{,}5)$.

Iz izraza (7) izračunamo

$$v' = \frac{0{,}2724609375 - 0}{0{,}5 - 0} = 0{,}544921875.$$

Vrednost $v' = 0{,}544921875$ dekodiramo kot x_2, ker je $0{,}544921875 \in \mathcal{R}(x_2) = [0{,}5,\ 0{,}6)$.

Postavimo $v = 0{,}544921875$ in iz izraza (7) izračunamo $v' = 0{,}44921875$. Vrednost $v' = 0{,}44921875$ dekodiramo kot x_1, ker je $0{,}44921875 \in \mathcal{R}(x_1) = [0,\ 0{,}5)$.

Aritmetični kod - primer dekodiranja

Postavimo $v = 0{,}44921875$ in iz izraza (7) izračunamo $v' = 0{,}8984375$. Vrednost $v' = 0{,}8984375$ dekodiramo kot x_3, ker je $0{,}8984375 \in \mathcal{R}(x_3) = [0{,}6,\ 0{,}9)$.

Postavimo $v = 0{,}8984375$ in iz izraza (7) izračunamo $v' = 0{,}9947916667$. Vrednost $v' = 0{,}9947916667$ dekodiramo kot PZ, ker je $0{,}9947916667 \in \mathcal{R}(x_4 = \mathrm{PZ}) = [0{,}9,\ 1)$.

Ko dekodiramo znak PZ, ustavimo postopek dekodiranja. Torej, kod 1011101001 dekodiramo kot $x_1 x_2 x_1 x_3 \mathrm{PZ}$. $\square$

Kod Lempela, Ziva in Welcha (LZW)

Videli smo, da zahtevata Huffmanov in aritmetični kod vnaprejšnje poznavanje porazdelitve verjetnosti znakov v sporočilu, ki ga kodiramo.

Če te ne poznamo, lahko gospodarno kodiramo sporočila tako, da sproti ugotavljamo krajše nize znakov, ki se v sporočilu ponavljajo, ter jih nato učinkovito zakodiramo.

Primer 5.12

Vzemimo niz dvojiških znakov

$$1011010100010\cdots$$

Od začetka proti koncu niza iščemo najkrajše podnize, ki se v nizu prvič pojavijo. Dobimo

$$1, 0, 11, 01, 010, 00, 10\cdots \qquad \square$$

Kod LZW - algoritem kodiranja

Algoritem gospodarnega kodiranja LZW temelji na ugotavljanju podnizov danega niza (na primer besedila, zakodiranega s kodom ASCII).

TABELE:

ST	Tabela, ki hrani sporočilo (besedilo), ki ga želimo kodirati,
KT	tabela s kodnimi zamenjavami sporočila iz tabele ST,
PT	pomožna tabela – *slovar*, ki hrani na mestih med 0 in 255 znake abecede vira, od mesta 256 naprej pa različne nize znakov abecede vira.

VHOD: Tabela ST.

IZHOD: Tabela KT.

Kod LZW

1. korak Preberi prvi znak s iz tabele ST.

2. korak Dokler ne prideš do zadnjega znaka v tabeli ST, ponavljaj:

a) t je naslednji znak v tabeli ST
b) spni oba znaka (niza) v niz $u \leftarrow s \| t$
c) **če** je u v slovarju PT,
   potem
   c1) postavi $s \leftarrow u$,
   sicer:
   c2) prenesi naslov, ki ga ima s v slovarju PT, v tabelo KT,
   c3) vpiši u v slovar PT na prvo prosto mesto po naslovu 255,
   c4) postavi $s \leftarrow t$.
konec-če

3. korak Prenesi naslov, ki ga ima s v slovarju PT, v tabelo KT.

Kod LZW - primer

Primer 5.13

Vzemimo, da imamo v tabeli ST sporočilo $TRALALALALA$.

Tabela kodnih zamenjav KT je na začetku prazna.

Vzemimo, da so v slovarju PT na mestih med 0 in 255 zapisani znaki abecede vira tako, da jim mesto določa njihov kod ASCII.

Na primer, črka A je zapisana na 65-em mestu, ker je kod ASCII črke A enak $(65)_{10} = (01000001)_2$ (glej tabelo ASCII znakov).

Postopek kodiranja:

Preberemo prvi znak sporočila iz tabele ST in postavimo $s = T$. Nato preberemo še drugi znak ter postavimo $t = R$. Spnemo znaka s in t. Dobimo $u = TR$, česar ni v slovarju PT.

Kod LZW - primer

Zato prenesemo naslov znaka $T(=s)$ iz slovarja PT, to je $(84)_{10}$ oziroma $(01010100)_2$, v tabelo KT, njegovo kodno zamenjavo, to je niz znakov $TR(=u)$, pa vpišemo v slovar PT na mesto 256.

Postavimo $s=t=R$ in nadaljujemo s tretjim znakom sporočila $t=A$. Sledi $u=RA$, česar ni v slovarju PT. Zato vpišemo RA v slovar PT na mesto 257.
Naslov znaka R v slovarju PT je njegova kodna zamenjava itn.

Vsakokrat, ko se pojavi v tabeli znak PZ (=poseben znak), to pomeni, da se bo povečala razsežnost kodnih zamenjav za en dvojiški znak od znaka PZ naprej.

Znakom PZ navadno prirejamo desetiške vrednosti $255, 511, 1023, \ldots$.

Kod LZW - primer

Tabela ponazarja postopek kodiranja in dograjevanja slovarja PT.

Sporočilo ST $TRALALALALA$						
s	t	u	Kod KT		Slovar PT	
			$(\cdot)_{10}$	$(\cdot)_2$	Vpis	Mesto
T	R	TR	84	01010100	TR	256
R	A	RA	82	01010100	RA	257
A	L	AL	65	01000001	AL	258
L	A	LA	76	01001011	LA	259
A	L	AL				
			PZ	11111111		
AL	A	ALA	258	100000010	ALA	260
A	L	AL				
AL	A	ALA				
ALA	L	$ALAL$	260		$ALAL$	261
L	A	LA		100000100		
LA		LA	259	100000011		

Kod LZW - primer

Ugotovimo, da smo sporočilo, ki je bilo zakodirano z ASCII kodom z 88-imi dvojiškimi znaki, gospodarno zakodirali s kodom LZW s 67-imi dvojiškimi znaki:

$$01010100\ 01010100\ 01000001\ 01001011\ 11111111\ 100000010\ 100000100\ 100000011.$$

$\square$

Po končanem kodiranju slovarja PT ne potrebujemo več.

Zapomniti si moramo le naslove posameznih znakov iz abecede vira (prvih 256 mest).

Zato ponavadi znake abecede vira zakodiramo s katerim izmed standardiziranih kodov z $m \leq 8$ (v našem primeru smo uporabili kod ASCII), kod znaka pa uporabimo tudi kot njegov naslov v slovarju.

Kod LZW - algoritem dekodiranja

Algoritem za dekodiranje mora poleg dekodiranja dograjevati slovar PT, prav tako kot v primeru kodiranja.

VHOD: Tabela KT.

IZHOD: Tabela ST.

Začetek postopka

1. **korak** Preberi kod c iz vhodne tabele KT (tabela s kodiranim sporočilom).
2. **korak** V izhodno tabelo ST (tabela z dekodiranim sporočilom) vpiši niz znakov s, ki je v slovarju PT vpisan na mestu c.
3. **korak** V slovar PT vpiši niz $s\|t$, kjer je t prvi znak naslednjega niza s.

Konec postopka

Kod LZW - algoritem dekodiranja

Opisani algoritem ne more dekodirati niza znakov sporočila oblike $t \| z \| t$, kjer je t poljubni znak, z pa poljubni niz znakov.

Problem rešimo tako, da:

a) vzamemo zadnji niz, ki smo ga dekodirali (vpisali v tabelo ST), to je niz oblike $t \| z$,
b) na njegov konec pripnemo prvi znak niza t (dobimo $t \| z \| t$) in ga
c) vpišemo na naslednje prazno mesto v slovar PT.

Kod LZW - primer

Primer 5.14

Vzemimo, da želimo dekodirati niz kodnih zamenjav

$$84\ 82\ 65\ 76\ \mathrm{PZ}\ 258\ 260\ 259.$$

Mesta v slovarju med 0 in 255 so znaki, ki se lahko pojavijo v sporočilu. Znak PZ tolmačimo, kot je bilo opisano.

Kod	Sporočilo	Slovar *PT*	
KT	*ST*	Vpis	Mesto
84	*T*		
82	*R*	*TR*	256
65	*A*	*RA*	257
76	*L*	*AL*	258
258	*AL*	*LA*	259
260		*ALA*	260
	ALA		
259	*LA*	*ALAL*	261

Kod LZW - algoritem dekodiranja

Na sporno situacijo naletimo pri dekodiranju kodne zamenjave 260, ker je v trenutku dekodiranja mesto 260 v slovarju PT prazno (brez vpisa).

Desetiški kod LZW 84 82 65 76 PZ 258 260 259. smo dekodirali kot sporočilo *TRALALALALA*. $\square$

Algoritem LZW lahko na različne načine kombiniramo s Huffmanovim algoritmom in algoritmom aritmetičnega koda, na primer s kodiranjem kodnih zamenjav algoritma LZW z aritmetičnim kodom ipd.

Oglejmo si kako učinkoviti so nekateri programski paketi za gospodarno kodiranje besedil, ki jih lahko najdemo na svetovnem spletu.

Primerjava programov za gospodarno kodiranje

Primer 5.15

Za preizkus učinkovitosti desetih programskih paketov za gospodarno kodiranje podatkov, ki jih lahko najdemo na svetovnem spletu[8], uporabimo vzorec slovenskih leposlovnih besedil, ki obsega 2 721 416 besed oziroma 16 784 110 znakov iz nabora 133-ih znakov.

Podrobnosti o uporabljenih postopkih gospodarnega kodiranja niso znani; iz časov izvajanja programov pri gospodarnem kodiranju vzorca slovenskih leposlovnih besedil sklepamo, da najboljši trije, to so Boa Constrictor, RKIVE in Ufa, ter verjetno tudi UHARC, temeljijo na zaporedni uporabi več postopkov gospodarnega kodiranja.

[8] Programi so brezplačni ali jih je treba kupiti po določenem preizkusnem obdobju.

Primerjava programov za gospodarno kodiranje

Ime programa	Različica	Avtor	Čas 'stiskanja' [s]	$\overline{n}$
ARJ	2.60	ARJ Software	84	3,27
ESP	1.92	GyikSoft	60	3,22
PKZIP	2.04g	PKWARE	74	3,20
RAR	2.50b3	RarSoft	138	3,09
JAR	1.02	ARJ Software	195	3,00
UHARC	0.2	Uwe Harklotz	751	2,73
Arhangel	1.38	Jurij Ljapko	188	2,52
Ufa	0.04b1	Igor Pavlov	703	2,24
RKIVE	1.92b1	Malcolm Taylor	2 148	2,23
Boa	0.58b	Ian Sutton	756	2,16

Tabela: Časi 'stiskanja' in povprečne dolžine kodnih zamenjav vzorca slovenskih leposlovnih besedil z desetimi programskimi paketi za gospodarno kodiranje.

Vprašanja

- Kaj je kod vira informacije?
- Zakaj kodiramo sporočila vira informacije?
- Kateri kodi vira informacije so enakomerni?
- Kateri kodi vira informacije so neenakomerni?
- Kaj pomeni, da kod vira informacije omogoca enoznačno dekodiranje?
- Za katere kode vira informacije pravimo, da so trenutni?
- Kaj je kodno drevo?
- Kako merimo optimalnost koda vira informacije?
- O čem govori Kraft-McMillanova neenakost?
- Kakšna je povprečna dolžina kodnih zamenjav optimalnih kodov vira informacije?
- Kaj trdi Shannon-Fanojev izrek?