Technická univerzita
Fakulta elektrotechniky a informatiky
Košice

Základy kódovania

Vysokoškolská učebnica

Košice 2012

Technická univerzita v Košiciach
Fakulta elektrotechniky a informatiky
Katedra matematiky a teoretickej informatiky

Základy kódovania

Vysokoškolská učebnica

Daniela Kravecová

Košice 2012

Základy kódovania

Prvé vydanie

Autor: © **RNDr. Daniela KRAVECOVÁ, PhD., 2012**

Recenzovali: Mgr. Jana Petrillová
Prof. RNDr. Ján Plavka, CSc.
Vydavateľ: Katedra matematiky a teoretickej informatiky
Fakulta elektrotechniky a informatiky
Technická univerzita v Košiciach

ISBN: 978-80-553-1178-4

Obsah

Predhovor

Táto učebnica je určená predovšetkým pre študentov druhého ročníka bakalárskej formy štúdiá na Fakulte elektrotechniky a informatiky Technickej univerzity v Košiciach.

Predkladaný text je založený na prednáškach z predmetu "Teória kódovania". Hlavným motívom na napísanie tejto učebnice bolo to, že na tejto fakulte nebola k dispozícií učebnica, ktorá by pokrývala obsah a rozsah týchto prednášok a bola zároveň v širšom meradle prístupná študentom. Cieľom tejto učebnice nie je vyčerpávajúcim spôsobom obsiahnuť problematiku teórie kódovania, ale prístupným spôsobom ponúknuť základné poznatky a ich aplikácie študentom, prípadne aj iným záujemcom.

Obsah učebnice je rozdelený do piatich kapitol, z ktorých prvá sa venuje nerovnomerným kódom. Obsahom ostatných kapitol sú blokové kódy, respektíve lineárne kódy. Z lineárnych kódov sú v tejto učebnici konkrétne zahrnuté Hammingove kódy, Reed–Mullerove kódy a cyklické kódy.

Snahou bolo napísať učebnicu čo najjednoduchšie, prispôsobiac úroveň učebnice úrovni matematickým základom, ktoré daní študenti majú. V mnohých kapitolách je k problematike pristupované intuitívne, bez striktných definícií, viet a ich dôkazov. Najvýraznejšie sa to prejavuje v kapitole venovanej cyklickým kódom. Teoretické poznatky sú ilustrované na vzorových príkladoch.

Učebnica pokrýva len úplné základy a vybrané témy z teórie kódovania. V budúcnosti by sa mohla rozšíriť o ďalšie témy, no predpokladom by bolo zaradenie kapitoly venujúcej sa algebraickým princípom, na ktorých sú tieto témy založené.

Na tomto mieste vyjadrujem vďaku prof. RNDr. Jánovi Plavkovi, CSc a Mgr. Janke Petrillovej za podrobné prečítanie, cenné pripomienky a recenziu tejto učebnice.

Košice 2012 Autorka

Kapitola 1

Kódovanie bez šumu

1.1 Matematické základy

V tejto časti sú stručne uvedené matematické základy, ktoré sú potrebné pre prácu s kódmi. Keďže základom, na ktorom je postavená teória kódovania, je použitie algebraických štruktúr a operácií, je tu prehľad najdôležitejších potrebných pojmov z tejto oblasti. Pojmy nie sú striktne v tvare definícií, skôr ako stručný prehľad a podobne aj vety, ktoré sa ich týkajú.

1.1.1 Modulárne operácie

Nech $a \in \mathbb{Z}, n \in \mathbb{N}$. Potom $a \ mod \ n$ je definované ako zvyšok po delení čísla a číslom n.

- modulárne sčítanie: $(a \ mod \ n) + (b \ mod \ n) = (a + b) \ mod \ n$,
- modulárne násobenie: $(a \ mod \ n) \cdot (b \ mod \ n) = (a \cdot b) \ mod \ n$
- $\forall a \in \{0, 1, 2, \ldots, n-1\}$ definujeme $-a$ tak, že $-a \in \{0, 1, 2, \ldots, n-1\}$ a zároveň $(a + (-a) \ mod \ n) = 0$.
- $\forall a \in \{0, 1, 2, \ldots, n-1\}$ definujeme a^{-1} tak, že $a^{-1} \in \{0, 1, 2, \ldots, n-1\}$ a zároveň $(a \cdot (a^{-1}) \ mod \ n) = 1$.
- modulárne odčítanie: $(a \ mod \ n) - (b \ mod \ n) = (a + (-b)) \ mod \ n$,
- modulárne delenie: $(a \ mod \ n) : (b \ mod \ n) = (a \cdot (b^{-1})) \ mod \ n$

Modulárne sčítanie podľa modulu 2 je známe ako operácia XOR a modulárne násobenie podľa modulu 2 ako AND.

1.1.2 Algebraické štruktúry

Nech je definovaná neprázdna množina G a binárna operácia $\square$. Štruktúru $(G, \square)$ nazývame grupou, ak:

- $\forall a, b \in G$ platí $a \square b \in G$,
- $\forall a, b, c \in G$ platí $a \square (b \square c) = (a \square b) \square c$,
- $\exists e \in G$ také, že $\forall a \in G$ je $a \square e = e \square a = a$,
- $\forall a \in G, \exists a' \in G$ také, že $a \square a' = a' \square a = e$.

Prvok e nazývame neutrálnym prvkom grupy a prvok a' nazývame inverzným prvkom grupy.

Ak navyše $\forall a, b \in G$ platí $a \square b = b \square a$, tak grupu nazývame komutatívnou grupou.

Nech je definovaná neprázdna množina T a dve binárne operácie $\square$ a $\triangle$. Štruktúru $(T, \square, \triangle)$ nazývame poľom, ak:

- $(T, \square)$ je komutatívna grupa,
- $(T - \{e\}, \triangle)$ je komutatívna grupa,,
- $\forall a, b, c \in T$ platí $a \triangle (b \square c) = (a \triangle b) \square (a \triangle c)$ a zároveň platí $(a \square b) \triangle c = (a \triangle c) \square (b \triangle c)$.

Neutrálny prvok vzhľadom na operáciu $\square$ (prvú operáciu) sa zvyčajne označuje 0 a neutrálny prvok vzhľadom na operáciu $\triangle$ (druhú operáciu) sa zvyčajne označuje 1.

Dá sa dokázať, že množina $T = \{0, 1, 2, \ldots, m-1\}$ s operáciou modulárneho sčítania podľa modulu m je komutatívnou grupou – $(T, +)$. Tiež sa dá ľahko ukázať, že $T - \{0\}$ s operáciou modulárneho násobenia podľa modulu m je komutatívnou grupou – $(T - \{0\}, \cdot)$, ak m je prvočíslo. Algebraická štruktúra $(T, +, \cdot)$ je poľom pre každé prvočíslo m.

Nech je definované pole $(T, \square, \triangle)$. Neprázdnu množinu L s dvomi binárnymi operáciami $\blacksquare, \blacktriangle$ nazývame lineárnym priestorom nad poľom $(T, \square, \triangle)$, ak:

- $(L, \blacksquare)$ je komutatívna grupa,
- $\forall x \in L, \forall t \in T$ platí $m \blacktriangle x \in L$,
- $\forall x, y \in L, \forall t, s \in T$ platí:
  - $m \blacktriangle (x \blacksquare y) = (m \blacktriangle x) \blacksquare (m \blacktriangle y)$,
  - $(m \triangle n) \blacktriangle x = m \blacktriangle (n \blacktriangle x)$,
  - $(m \square n) \blacktriangle x = (m \blacktriangle x) \blacksquare (n \blacktriangle x)$,
  - $1 \blacktriangle x = x$.

1.2 Základné pojmy

Definícia 1.2.1 *Nech je daná konečná neprázdna množina M, ktorú budeme nazývať abecedou. Slovom $\boldsymbol{m}$ nad abecedou M budeme nazývať ľubovoľnú konečnú neprázdnu postupnosť $m_1 m_2 \dots m_k$, kde $m_i \in M$, pre $\forall i = 1, 2, \dots, k$. Číslo k nazývame dĺžkou slova $\boldsymbol{m}$.*

Množinu všetkých slov spĺňajúcich dané kritériá, ktoré je možné zostrojiť nad abecedou M, budeme označovať M^*. Kritériom môže byť napríklad dĺžka slova.

Príklad 1.2.1 *Nech M je množina všetkých písmen slovenskej abecedy. Je poslaná správa:*

„Teória kódovania sa zaoberá konštrukciou kódov zameraných hlavne na schopnosť opravovať chyby, sú to takzvané samoopravné kódy, prípadne na zrýchlenie prenosu dát. Existuje aj taká časť teórie kódovania, ktorá sa zaoberá konštrukciou kódov slúžiacich hlavne na utajovanie dát. Táto vedecká disciplína sa volá kryptológia.“

Nájdite množinu M^* všetkých slov tejto správy, ktoré sa začínajú na samohlásku.

Riešenie: $M^* = \{\text{opravovať, existuje, aj, utajovanie}\}$

Príklad 1.2.2 *Nech $U = \{0, 1, 2\}$. Napíšte množinu U^* všetkých dvojznakových slov nad touto abecedou.*

Riešenie: $U^* = \{00, 01, 02, 10, 11, 12, 20, 21, 22\}$.

Definícia 1.2.2 *Kódovaním nazývame zobrazenie $K : A \to T^*$, ktoré každému prvku z konečnej množiny A priradí práve jedno slovo z konečnej množiny T^*.*

Vzhľadom na takto definované kódovanie zavádzame označenia a pomenovania:

- množina A – zdrojová abeceda,
- ľubovoľný prvok $a_i \in A$ – zdrojový znak,
- slovo $\boldsymbol{a} = a_1 a_2 \dots a_n$ nad množinou A – zdrojové slovo,
- množina A^* – množina zdrojových slov nad abecedou A.
- množina T – kódová abeceda,
- ľubovoľný prvok $t_i \in T$ – kódový znak,
- slovo $\boldsymbol{t} = t_1 t_2 \dots t_m$ nad množinou T – kódové slovo,
- množina T^* – množina možných kódových slov nad abecedou T.

Definícia 1.2.3 *Množinu všetkých kódových slov $\mathcal{K}$ priradených všetkým zdrojovým znakom zo zdrojovej abecedy A nazývame kód.*

Je zjavné, že $\mathcal{K} \subseteq T^*$.

Pomenovanie: Kód, v ktorom kódová abeceda obsahuje 2 prvky, sa nazýva binárny, ak obsahuje 3 prvky, sa nazýva ternárny, podobne je to pre quartérny, pentárny,.... Kód, v ktorom kódová abeceda obsahuje n prvkov, nazývame n–árny kód.

Definícia 1.2.4 *Nech je dané kódovanie $K : A \to T^*$. Ak pre ľubovoľné $a_1, a_2 \in A$, $a_1 \neq a_2$ platí, že $K(a_1) \neq K(a_2)$, tak dané kódovanie nazývame prostým kódovaním (prostým kódom).*

Teda kódovanie nazývame prostým, ak rôznym znakom zdrojovej abecedy sú vždy priradené rôzne kódové slová. V praxi nemá význam používať neprosté kódy, preto v ďalšom budeme vždy brať do úvahy len prosté kódovanie.

Príklad 1.2.3 *Je daná zdrojová abeceda $A = \{\alpha, \beta, \gamma, \delta\}$ a kódová abeceda $T = \{0, 1\}$. Navrhnite kódovanie všetkých zdrojových znakov kódovými slovami nad množinou T.*

Riešenie: Keďže máme len 4 zdrojové znaky, tak potrebujeme len 4 kódové slová. Stačia nám na to kódové slová dĺžky 2: $\alpha \to 00, \beta \to 01, \gamma \to 10, \delta \to 11$ a teda $\mathcal{K}= \{00, 01, 10, 11\}$.

Príklad 1.2.4 *Navrhnite kódovanie zdrojových znakov z predchádzajúceho príkladu tak, aby α bola zakódovaná jednoznakovým kódovým slovom.*

Riešenie: Možné kódovanie spĺňajúce danú podmienku môže vyzerať napríklad takto: $\alpha \to 0, \beta \to 11, \gamma \to 10, \delta \to 110$ a $K(a) = \{0, 11, 10, 110\}$.

Definícia 1.2.5 *Nech je dané kódovanie $K : A \to T^*$. Kódovaním zdrojových správ budeme nazývať zobrazenie $K^* : A^* \to T^*$ definované nasledovne:*

$$K^*(\boldsymbol{a}) = K^*(a_1 a_2 \dots a_n) = K(a_1)K(a_2)\dots K(a_n).$$

Kódovanie zdrojových správ je rozšírením kódovania zdrojových slov tak, že každé zdrojové slovo kódujeme znak po znaku, čím každej zdrojovej správe priradíme kódovú správu.

Definícia 1.2.6 *Nech je dané kódovanie $K : A \to T^*$. Dekódovaním kódových správ budeme nazývať proces, ktorým z prijatej kódovej správy $K^*(\boldsymbol{a})$ určíme prislúchajúcu zdrojovú správu $\boldsymbol{a}$.*

Príklad 1.2.5 *Majme kódovanie dané nasledujúcou tabuľkou:*

zdrojový znak	0	2	4	6	8
kódové slovo	000	010	100	101	111

Pomocou neho zakódujte zdrojové slová 248 *a* 2204.

Riešenie: $K^*(248) = 010100111$ a $K^*(2204) = 010010000100$.

Príklad 1.2.6 *Dekódujte prijatú správu* 101010111010000101010 *pomocou kódovania zadaného v predchádzajúcom príklade.*

Riešenie: Keďže všetky kódové slová majú dĺžku 3, tak prijatú kódovú správu najprv porozdeľujeme na trojznakové časti – kódové slová: 101|010|111|010|000|101|010. Každému kódovému slovu priradíme príslušný zdrojový znak, teda:

$$101|010|111|010|000|101|010 \rightarrow 6282062.$$

Príklad 1.2.7 *Dekódujte kódovú správu* 1001011010 *pomocou kódovania vytvoreného v príklade 1.2.4.*

Riešenie: Danú postupnosť kódových znakov nevieme jednoznačne rozdeliť na kódové slová a teda ju ani nevieme jednoznačne dekódovať. Ak by sme to skúsili od začiatku znak po znaku, tak jednoznačne dostaneme: 10|0|10|11010, no ďalej existujú dve možnosti: 10|0|10|11|0|10 alebo 10|0|10|110|10. Z takejto kódovej správy nevieme jednoznačne určiť zdrojovú správu.

Definícia 1.2.7 *Hovoríme, že kódovanie $K : A \rightarrow T^*$ je jednoznačne dekódovateľné, ak je jemu prislúchajúce kódovanie správ $K^* : A^* \rightarrow T^*$ prostým zobrazením.*

Teda, kódovanie $K : A \rightarrow T^*$ nazývame jednoznačne dekódovateľné, ak zo znalosti zakódovanej správy $K^*(a_1a_2 \dots a_n)$ vieme vždy jednoznačne určiť zdrojovú správu $a_1a_2 \dots a_n$.

Jedným zo základných delení kódov je na rovnomerné a nerovnomerné kódy.

Definícia 1.2.8 *Nech je dané kódovanie $K : A \rightarrow T^*$. Nech zdrojovému znaku $a_i \in A$ v tomto kódovaní prislúcha kódové slovo $\boldsymbol{t_i} = t_{i_1}t_{i_2} \dots t_{i_m}$. Číslo m nazývame dĺžkou slova a označujeme ju d_i.*

Definícia 1.2.9 *Rovnomerným alebo blokovým kódovaním (dĺžky n) nazývame také kódovanie, v ktorom všetky kódové slová majú rovnakú dĺžku (n). V opačnom prípade sa kódovanie nazýva nerovnomerným.*

Poznámka: Každé rovnomerné kódovanie je jednoznačne dekódovateľné.

Príklad 1.2.8 *Nech je daná kódová abeceda $T = \{0, 1, 2\}$. Čísla $0, 1, \dots, 9$ zakódujte blokovým kódom.*

Riešenie: Keďže zdrojových znakov je 10, tak dĺžka kódových slov musí byť aspoň 3 (všetkých slov dĺžky 2 je len $3^2 = 9$). Jedným z možných kódovaní je:

zdrojový znak	0	1	2	3	4	5	6	7	8	9
kódové slovo	000	010	020	100	110	120	200	210	220	221

1.3 Nerovnomerné kódy

1.3.1 Prefixové kódy

Medzi nerovnomernými kódmi majú významné miesto tzv. prefixové kódy.

Definícia 1.3.1 *Nech je daná kódová abeceda T. Prefixom (predponou) kódového slova $\boldsymbol{t} = t_1t_2\ldots t_m$ nad abecedou T nazývame hociktoré z kódových slov $t_1; t_1t_2; t_1t_2t_3; \ldots; t_1t_2\ldots t_m$.*

Definícia 1.3.2 *Kódovanie $K : A \to T^*$ sa nazýva prefixovým kódovaním, ak pre všetky kódové slová z T^* platí, že žiadne kódové slovo nie je prefixom iného kódového slova. Kód $\mathcal{K}$ v tomto prípade nazývame prefixový kód.*

Každé prefixové kódovanie je jednoznačne dekódovateľné, to je aj dôvod, prečo sú prefixové kódy také významné. To, čo ich však odlišuje od ostatných jednoznačne dekódovateľných kódov, je fakt, že sú to jediné kódy, ktoré je možné dekódovať znak po znaku od začiatku. Takže hocijakú správu môžeme dekódovať už počas jej prijímania, nie je nutné poznať celú správu.

Analógiou prefixových kódov sú napríklad sufixové – príponové kódy, no ich dekódovanie sa začína od posledného znaku, čo je pri dlhších správach veľkou nevýhodou. (Definíciu sufixových kódov je veľmi ľahké dostať malou obmenou v definícii prefixových kódov.)

Pozorovanie: Keďže v blokovom kódovaní majú všetky kódové slová rovnakú dĺžku, tak žiadne nie je prefixom iného a teda každé blokové kódovanie je zároveň aj prefixovým kódovaním.

Príklad 1.3.1 *Je dané nasledujúce kódovanie. Pomocou neho dekódujte správu 1100100110.*

zdrojový znak	*A*	*B*	*C*	*D*	*E*
kódové slovo	01	10	0100	001	110

Riešenie: Na rozdelenie správy na jednotlivé kódové slová máme dve možnosti:

110|0100|110	110|01|001|10
E C E	E A D B

Toto kódovanie nie je jednoznačne dekódovateľné a z toho vyplýva, že nemôže byť ani prefixové. Stačí, ak sa pozrieme do tabuľky a vidíme, že kódové slovo priradené zdrojovému znaku A je prefixom kódového slova priradeného zdrojovému znaku C.

Príklad 1.3.2 *Nech je dané nasledujúce kódovanie. Pomocou neho dekódujte správu 011011110.*

zdrojový znak	△	□	◯	◊
kódové slovo	0	01	011	111

Riešenie: Hneď na prvý pohľad je jasné, že to nie je prefixové kódovanie a už pri hľadaní prvého kódového slova nie je jasné, či to bude kódové slovo prislúchajúce △, □ alebo ◯. Pokúsime sa však oddeľovať jednotlivé kódové slová od konca:

$$\begin{array}{c} 011|01|111|0 \\ \hline \bigcirc \quad \triangle \quad \square \quad \Diamond \end{array}$$

Toto kódovanie je jednoznačne dekódovateľné, aj keď nie je prefixové. Je to príklad sufixového kódovania.

Pozorovanie: Prefixové kódovanie je jediné jednoznačne dekódovateľné kódovanie, ktoré je dekódovateľné znak po znaku od začiatku správy. No nie je to jediné jednoznačne dekódovateľné kódovanie.

1.3.2 Konštrukcia prefixových kódov

Príklad 1.3.3 *Zostrojte vhodný binárny prefixový kód pre cifry:* $0, 1, \ldots, 9$.

Riešenie: Pretože máme 10 zdrojových znakov, postačia nám kódové slová s dĺžkou maximálne 4 ($2^3 = 8, 2^4 = 16$). Keďže mamé binárny kód, tak všetky cifry rozdelíme do dvoch skupin s počtom prvkov líšiacim sa najviac o 1 a každej z nich priradíme jeden z kódových znakov 0 alebo 1. Tým zabezpečíme, že kódové slová z jednej skupiny nebudú prefixami kódových slov z druhej skupiny. Každú z týchto skupín rozdelíme na ďalšie dve podskupiny a pridáme im zase 0, respektíve 1, čím zabezpečíme, že kódové slová z jednej podskupiny nebudú prefixami kódových slov z druhej podskupiny. Pokračujeme dovtedy, kým každá malá skupinka neobsahuje maximálne jeden zdrojový znak. Takto dostaneme prefixový binárny kód pre danú zdrojovú abecedu.

0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9
00	00	00	01	01	10	10	10	11	11
0	1	2	3	4	5	6	7	8	9
000	000	001	010	011	100	100	101	110	111
0	1	2	3	4	5	6	7	8	9
0000	0001	001	010	011	1000	1001	101	110	111

Podobný algoritmus na nájdenie prefixového kódu je možné použiť aj pre ľubovoľný n-árny kód. No zdrojové znaky nebudeme postupne rozdeľovať do dvoch skupín, ale do n skupín.

Často sú pri konštrukciách prefixových kódov kladené ďalšie požiadavky na dĺžky slov, respektíve na efektivitu kódov. Ak sa niektorý zdrojový znak vyskytuje v správe častejšie ako iné, je vhodné ho kódovať kratším kódovým slovom ako ostatné a naopak pri zdrojovom znaku vyskytujúcom sa v správach veľmi zriedka nevadí, ak jemu prislúchajúce kódové slovo bude dlhšie.

Príklad 1.3.4 *Zostrojte vhodný binárny prefixový kód pre cifry:* $0, 1, \ldots, 9$ *tak, aby* 0 *zodpovedalo kódové slovo* 0000 *a zároveň vieme, že cifry* 2 *a* 4 *sa nachádzajú v správe najčastejšie.*

Riešenie: Nule teda priradíme kódové slovo 0000, ostalo nám nezakódovaných 9 znakov.

0	1	2	3	4	5	6	7	8	9
0000									

Ak by sme 2 a 4 priradili jednoznakové kódové slová, nutne dostaneme prefixový kód. Ak by sme 2 a 4 priradili dvojznakové kódové slová, ostane nám nezakódovaných 7 zdrojových znakov. Trojznakových binárnych slov máme 8, no 2 a 4 sú prefixami štyroch z nich, takže nepostačujú nám, preto ostaným musíme priradiť aspoň 4-znakové kódové slová. Tých je 16, 2 a 4 sú prefixami ôsmych z nich, nule je priradené jedno, takže ostáva ešte 7 4-znakových binárnych slov, čo nám postačuje pre kódovanie.

0	1	2	3	4	5	6	7	8	9
0000	0001	10	0010	11	0011	0100	0101	0110	0111

Všimnime si, že pri zostrojení daného kódu nás ani veľmi nezaujúmalo to, aké znaky budú mať jednotlivé slová, skôr sme sa zaoberali dĺžkou jednotlivých kódových slov.

Príklad 1.3.5 *Zostrojte vhodný ternárny prefixový kód pre nasledujúce zdrojové znaky tak, aby boli zachované predpísané dĺžky kódových slov.*

zdrojový znak	α	β	γ	δ	λ	κ
dĺžka kódového slova	2	1	2	1	2	2

Riešenie: Keďže ide o ternárny kód, tak nech kódová abeceda je $T = \{0, 1, 2\}$. Pokúsime sa podobnou úvahou ako v predchádzajúcom príklade zakódovať dané zdrojové znaky. Začneme od jednoznakových a pokračujeme s dvojznakovými.

zdrojový znak	α	β	γ	δ	λ	κ
dĺžka kódového slova	2	1	2	1	2	2
kódové slovo	20	0	21	1	22	?

K zakódovaniu 4 znakov nám ostali len tri dvojznakové kódové slová, takže požadované kódovanie sa nedá zostrojiť.
Pozorovanie: Pri konštrukcii prefixového kódu nezáleží ani tak na tom, ako volíme kódové slová, záleží na dĺžke kódových slov.

Kraftova nerovnosť

V príklade 1.3.4 sme vo svojej úvahe už zvažovali to, aké dĺžky slov by sme mali použiť, aby ten kód existoval. V príklade 1.3.5 sme sa presvedčili, že nie pre hocijakú voľbu dĺžok kódových slov sa požadovaný kód dá zostrojiť. V ďalšom sa teda pokúsime vo všeobecnosti odvodiť, za akých podmienok sa dá zostrojiť prefixový binárny kód so stanovenými

dĺžkami kódových slov. Potom to zovšeobecníme pre kód s kódovou abecedou obsahujúcou n kódových znakov.

Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$, dvojprvková kódová abeceda $T = \{t_1, t_2\}$ a prefixové kódovanie $K : A \to T^*$. Kódové slovo, ktoré je v kódovaní K priradené zdrojovému znaku a_i budeme označovať $K(a_i)$ a jeho dĺžku budeme označovať d_i. Kvôli prehľadnosti preusporiadame zdrojovú abecedu tak, aby platilo $d_1 \leqq d_2 \leqq \cdots \leqq d_r$.

Zvolíme ľubovoľné kódové slovo $K(a_1)$ dĺžky d_1. Ďalej zvolíme kódové slovo $K(a_2)$ dĺžky d_2 tak, aby $K(a_1)$ nebolo jeho prefixom, kódové slovo $K(a_3)$ dĺžky d_3 zvolíme tak, aby $K(a_1)$ a $K(a_2)$ neboli jeho prefixami,... a kódové slovo $K(a_r)$ dĺžky d_r zvolíme tak, aby slová $K(a_1), K(a_2), \ldots, K(a_{r-1})$ neboli jeho prefixami.

Teraz sa ideme zaoberať počtom takých kódových slov, ktoré splňajú predchádzajúce kritériá:

- Keďže ide o binárny kód, počet možných $K(a_1)$ je 2^{d_1}.
- Počet možných $K(a_2)$, ktoré nemajú prefix $K(a_1)$ je $\frac{2^{d_2}}{2^{d_1}} = 2^{d_2-d_1}$ a platí, že $2^{d_2-d_1} < 2^{d_2}$.
- Počet možných $K(a_3)$, ktoré nemajú prefix $K(a_1), K(a_2)$ je $2^{d_3-d_2} + 2^{d_3-d_1}$ a samozrejme platí, že $(2^{d_3-d_2} + 2^{d_3-d_1}) < 2^{d_3}$.
- ...
- Počet možných $K(a_r)$ takých, že žiadne z kódových slov $K(a_1), K(a_2), \ldots, K(a_{r-1})$ nie je ich prefixom je $(2^{d_r-d_{r-1}} + 2^{d_r-d_{r-2}} + \cdots + 2^{d_r-d_2} + 2^{d_r-d_1})$ a platí, že $(2^{d_r-d_{r-1}} + 2^{d_r-d_{r-2}} + \cdots + 2^{d_r-d_2} + 2^{d_r-d_1}) < 2^{d_r}$.

Poslednú nerovnosť upravujeme:

$$\begin{aligned} 2^{d_r-d_{r-1}} + 2^{d_r-d_{r-2}} + \cdots + 2^{d_r-d_2} + 2^{d_r-d_1} &< 2^{d_r} \quad / + 1 \\ 2^{d_r-d_{r-1}} + 2^{d_r-d_{r-2}} + \cdots + 2^{d_r-d_2} + 2^{d_r-d_1} + 1 &\leqq 2^{d_r} \quad / \cdot 2^{-d_r} \\ 2^{-d_{r-1}} + 2^{-d_{r-2}} + \cdots + 2^{-d_2} + 2^{-d_1} + 2^{-d_r} &\leqq 1 \end{aligned}$$

Výsledok toho odvodenia sformulujeme vo forme nasledujúcej vety:

Veta 1.3.1 *Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$. Prefixový binárny kód na zakódovanie tejto abecedy s dĺžkami kódových slov $d_1 \leqq d_2 \leqq \cdots \leqq d_r$ sa dá zostrojiť práve vtedy, ak platí nerovnosť*

$$2^{-d_r} + 2^{-d_{r-1}} + 2^{-d_{r-2}} + \cdots + 2^{-d_2} + 2^{-d_1} \leqq 1.$$

Poznámka: Predchádzajúce odvodenie možno ľahko modifikovať ako dôkaz tejto vety.

Predchádzajúcu vetu môžeme zovšeobecniť pre kódovanie s kódovou abecedou s n kódovými znakmi.

Veta 1.3.2 *(o Kraftovej nerovnosti) Nech je daná zdrojová abeceda* $A = \{a_1, a_2, \ldots, a_r\}$ *a kódová abeceda* $T = \{t_1, t_2, \ldots, t_n\}$. *Prefixové kódovanie* $K : A \to T^*$ *na zakódovanie tejto zdrojovej abecedy s dĺžkami kódových slov* $d_1, d_2, \ldots, d_r$ *sa dá zostrojiť práve vtedy, ak platí nerovnosť*

$$n^{-d_r} + n^{-d_{r-1}} + n^{-d_{r-2}} + \cdots + n^{-d_2} + n^{-d_1} \leqq 1.$$

Túto nerovnosť nazývame Kraftovou nerovnosťou.

Príklad 1.3.6 *Zostrojte binárný prefixový kód pre cifry:* $0, 1, \ldots, 9$ *tak, aby bol vhodný pre správy, v ktorých sa často opakuje* 0 *a* 2, *ale zriedka sa vyskytuje* 5 *a* 7.

Riešenie: Keďže 0 a 2 sa opakujú často, zvolíme pre nich, čo najkratšie kódové slová. Jednoznakové slová to byť nemôžu, tak nech sú to 2–znakové a ostatné nech sú 3–znakové. Potom Kraftová nerovnosť bude vyzerať takto: $2 \cdot 2^{-2} + 8 \cdot 2^{-3} \leqq 1$. Po úprave dostaneme $1/2 + 1 \leqq 1$, čo neplatí.
Teda nech všetky ostatné slová sú 4–znakové. Potom Kraftova nerovnosť bude mať tvar: $2 \cdot 2^{-2} + 8 \cdot 2^{-4} \leqq 1$ a po úprave dostaneme $1/2 + 1/2 \leqq 1$, a teda Kraftova nerovnosť platí. Vieme tiež, že v správach sa zriedka vyskytujú 5 a 7, tak pre ne pripusťme 5–znakové kódové slová a snažme sa o maximálny počet 3–znakových slov. Máme teda 10 zdrojových znakov, 2 budú zakódované 2–znakovými kódovými slovami, 2 budú zakódované 5–znakovými kódovými slovami, k bude zakódovaných 3–znakovými kódovými slovami a $6 - k$ budú zakódované 4–znakovými kódovými slovami. Kraftova nerovnosť bude v tvare:

$$\begin{aligned} 2 \cdot 2^{-2} + 2 \cdot 2^{-5} + k \cdot 2^{-3} + (6-k) \cdot 2^{-4} &\leqq 1 \\ 1/2 + 1/16 + k/8 + (6-k)/16 &\leqq 1 \quad / \cdot 16 \\ 8 + 1 + 2k + 6 - k &\leqq 16 \\ k &\leqq 1 \end{aligned}$$

Okrem cifier $0, 2, 5, 7$ budú ostatné zakódované jedným 3–znakovým kódovým slovom a piatimi 4–znakovými kódovými slovami. Vhodný kód môže byť napríklad takýto:

zdrojový znak	0	1	2	3	4	5	6	7	8	9
dĺžka kódového slova	2	4	2	4	4	5	4	5	4	3
kódové slovo	00	1000	01	1001	1010	11110	1011	11111	1110	110

Príklad 1.3.7 *Vráťme sa k príkladu 1.3.5 a analyzujme možnosť zostrojenia vhodného ternárneho prefixového kódu pre nasledujúce zdrojové znaky a dĺžky kódových slov.*

zdrojový znak	α	β	γ	δ	λ	κ
dĺžka kódového slova	2	1	2	1	2	2

Riešenie: Požadované dĺžky kódových slov dosadíme do Kraftovej nerovnosti a dostaneme:

$$2 \cdot 3^{-1} + 4 \cdot 3^{-2} \leqq 1.$$

Po úprave dostaneme $10 \leqq 9$, čo neplatí, takže požadovaný kód sa zostrojiť nedá.

Príklad 1.3.8 *Koľko znakov kódovej abecedy potrebujeme, aby sme cifry* $0, 1, \ldots, 9$ *zakódovali najviac dvojznakovými kódovými slovami?*

Riešenie: Neznámou v Kraftovej nerovnosti tentokrát bude n a za všetky dĺžky slov dosadíme 2:

$$\begin{aligned} 10 \cdot n^{-2} &\leqq 1 \quad / \cdot n^2 \\ 10 &\leqq n^2 \\ n &\geqq \sqrt{10} \\ n &\geqq 3,17 \end{aligned}$$

Na zakódovanie cifier $0, 1, \ldots, 9$ najviac dvojznakovými kódovými slovami potrebujeme kódovú abecedu s aspoň 4 kódovými znakmi.

Z vety 1.3.2 vyplýva, že ak máme prefixový kód s určitými dĺžkami kódových slov, tak určite pre neho platí Kraftova nerovnosť. Naopak, ak pre nejaký kód s určitými dĺžkami kódových slov platí Kraftova nerovnosť, tak sa dá zostrojiť prefixový kód s takými istými dĺžkami kódových slov. Takže daný kód nemusí byť nutne prefixový ba ani jednoznačne dekódovateľný. Vzťah medzi jednoznačnou dekódovateľnosťou kódu a Kraftovou nerovnosťou vyjadruje nasledujúca veta.

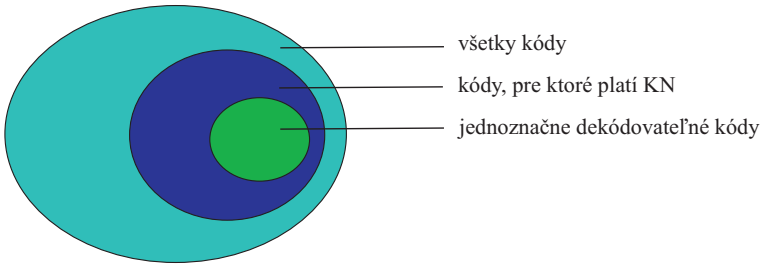
Veta 1.3.3 *(McMillanova) Pre každý jednoznačne dekódovateľný kód platí Kraftova nerovnosť.*

To znamená, že ak je kód jednoznačne dekódovateľný, tak pre neho platí Kraftova nerovnosť.

Obmenená veta k vete 1.3.3 je: "Ak pre kód neplatí Kraftova nerovnosť, tak kód nie je jednoznačne dekódovateľný."

Nič sa však nehovorí o tom, ak pre kód platí Kraftova nerovnosť. V tom prípade kód môže, ale tiež nemusí byť jednoznačne dekódovateľný.

Schématicky to možno znázorniť Venovými diagramami.



Príklad 1.3.9 *Pomocou McMillanovej vety analyzujte jednoznačnosť dekódovania nasledujúcich kódov:*

zdrojová abeceda	α	β	γ	δ
kód $\mathcal{K}_1$	00	01	10	011
kód $\mathcal{K}_2$	0	1	11	10
kód $\mathcal{K}_3$	1	10	101	111
kód $\mathcal{K}_4$	0	10	111	1100

Riešenie: Pre každý z daných kódov určíme platnosť Kraftovej nerovnosti:

$$\begin{aligned}
\mathcal{K}_1: &\quad 2^{-2}+2^{-2}+2^{-2}+2^{-3} = 7/8 \leqq 1\\
\mathcal{K}_2: &\quad 2^{-1}+2^{-1}+2^{-2}+2^{-2} = 3/2 \not\leqq 1\\
\mathcal{K}_3: &\quad 2^{-1}+2^{-2}+2^{-3}+2^{-3} = 1 \leqq 1\\
\mathcal{K}_4: &\quad 2^{-1}+2^{-2}+2^{-3}+2^{-4} = 15/16 \leqq 1
\end{aligned}$$

Z výpočtu vyplýva, že pre kód $\mathcal{K}_2$ Kraftova nerovnosť nie je splnená a kód nie je jednoznačne dekódovateľný. Pre ostatné kódy Kraftova nerovnosť je splnená, to znamená, že tieto kódy **môžu** byť jednoznačne dekódovateľné. Skúsme sa pozrieť na to, či skutočne sú, alebo nie sú jednoznačne dekódovateľné.

Kód $\mathcal{K}_1$ evidentne nie je prefixový, pretože kódové slovo $K_1(\beta)$ je prefixom kódového slova $K_1(\delta)$. No je to sufixový kód, keďže žiadne kódové slovo nie je sufixom iného kódového slova a teda je to jednoznačne dekódovateľný kód.

Kód $\mathcal{K}_3$ nie je prefixový, pretože kódové slovo $K_3(\alpha)$ je prefixom všetkých ostatných kódových slov, a podobne nie je ani sufixový. Ak by sme sa pokúsili dekódovať napríklad túto jednoduchú správu 101101, dostaneme dva rôzne výsledky: $\gamma\gamma$, $\beta\alpha\gamma$. Teda kód nie je jednoznačne dekódovateľný.

Keďže kód $\mathcal{K}_4$ je prefixový, tak je aj jednoznačne dekódovateľný.

Príklad 1.3.10 *Koľko zdrojových znakov možno zakódovať jednoznačne dekódovateľným binárnym kódom s dĺžkami kódových slov maximálne* 3*?*

Riešenie: Všetkých binárnych slov dĺžky 3 je $2^3 = 8$, teda toľko rôznych kódových slov môžeme zostrojiť a nimi zakódovať zdrojové znaky.

Definícia 1.3.3 *Nech je dané kódovanie* $K : A \to T^*$*. Ním definovaný* n*–árny jednoznačne dekódovateľný kód* $\mathcal{K}$ *sa nazýva úplnym kódom práve vtedy, ak pre ľubovoľnú* n*–árnu postupnosť* $\boldsymbol{t} \in T^*$ *existuje také kódové slovo* $\boldsymbol{k}_i \in \mathcal{K}$*, že buď postupnosť* $\boldsymbol{t}$ *je prefixom slova* $\boldsymbol{k}_i$ *alebo slovo* $\boldsymbol{k}_i$ *je prefixom postupnosti* $\boldsymbol{t}$*.*

Ak je daný kód s veľkým počtom kódových slov, je ťažké overovať jeho úplnosť pomocou definície. No úplnosť kódu súvisí s Kraftovou nerovnosťou, ako to vyjadruje nasledujúca veta.

Veta 1.3.4 *N–árny jednoznačne dekódovateľný kód* $\mathcal{K} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \ldots, \boldsymbol{k}_r\}$ *s dĺžkami kódových slov* $d_1, d_2, \ldots, d_r$ *je úplny práve vtedy, ak je prefixový a platí*

$$\sum_{i=1}^{r} n^{-d_i} = 1.$$

1.3.3 Najkratší kód

V príklade 1.3.6 sme hľadali kód, ktorý bol určený pre správy, kde sa niektoré znaky vyskytujú častejšie, niektoré menej často. Podľa toho sme volili dĺžku kódových slov. Nebolo však nijako kvantitatívne určené, ako často sa jednotlivé znaky v správach vyskytujú. Teraz sa budeme zaoberať kódovaním takých správ, v ktorých budeme poznať pravdepodobnosť výskytu jednotlivých zdrojových znakov a budeme sa snažiť zostrojiť čo najefektívnejší kód. Mierou efektivity kódu sa bude považovať priemerná dĺžka kódového slova a do úvahy budeme brať len jednoznačne dekódovateľné kódy.

Definícia 1.3.4 *Nech je daný n–árny jednoznačne dekódovateľný kód $\mathcal{K} = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \ldots, \boldsymbol{k}_r\}$ s dĺžkami kódových slov $d_1, d_2, \ldots, d_r$ a nech sa zdrojové znaky $a_1, a_2, \ldots, a_r$ vyskytujú s pravdepodobnosťami $p_1, p_2, \ldots, p_r$. Priemernou dĺžkou kódového slova kódu $\mathcal{K}$ budeme nazývať číslo*

$$\bar{d}(\mathcal{K}) = d_1 \cdot p_1 + d_2 \cdot p_2 + \cdots + d_r \cdot p_r.$$

Ak bude zjavné, o aký kód ide, symbol $\bar{d}(\mathcal{K})$ nahradíme symbolom $\bar{d}$.

Poznámka: Z teórie pravdepodobnosti je jasné, že súčet pravdepodobností všetkých zdrojových znakov je 1.

Príklad 1.3.11 *Porovnajte nasledujúce kódy vzhľadom na priemernú dĺžku kódového slova.*

zdrojová abeceda	0	1	2	3	4	5	6	7	8	9
pravdepod. výskytu	0,18	0,15	0,2	0,07	0,05	0,038	0,11	0,02	0,07	0,012
kód $\mathcal{K}_1$	00	1000	01	1001	1010	1100	1011	1101	1110	1111
kód $\mathcal{K}_2$	00	1000	01	1001	1010	11110	1011	11111	1110	110

Riešenie:
$\bar{d}(\mathcal{K}_1) = 0,18 \cdot 2 + 0,15 \cdot 4 + 0,2 \cdot 2 + 0,07 \cdot 4 + 0,05 \cdot 4 + 0,038 \cdot 4 + 0,11 \cdot 4 + 0,02 \cdot 4 + 0,07 \cdot 4 + 0,012 \cdot 4 = 2,84$
$\bar{d}(\mathcal{K}_1) = 0,18 \cdot 2 + 0,15 \cdot 4 + 0,2 \cdot 2 + 0,07 \cdot 4 + 0,05 \cdot 4 + 0,038 \cdot 5 + 0,11 \cdot 4 + 0,02 \cdot 5 + 0,07 \cdot 4 + 0,012 \cdot 3 = 2,886$
Kód $\mathcal{K}_1$ má priemernú dĺžku kódového slova menšiu ako kód $\mathcal{K}_2$, čo znamená, že je efektívnejší.

Definícia 1.3.5 *Najkratším n–znakovým kódovaním (kódom $\mathcal{K}$) zdrojovej abecedy $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu v správach $p_1, p_2, \ldots, p_r$ nazývame prefixové kódovanie tejto abecedy pomocou n kódových znakov tak, že priemerná dĺžka kódového slova $\bar{d}(\mathcal{K})$ je najmenšia možná.*

Medzi najčastejšie uvádzané metódy hľadania optimálnych prefixových kódov patria Shannon–Fanova metóda a Huffmanova metóda konštrukcie kódu. Popíšeme tieto metódy hľadania pre binárne kódy.

Shannon–Fanova konštrukcia prefixového kódu

Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $p_1, p_2, \ldots, p_r$. Konštrukciu popíšeme v niekoľkých krokoch:

1. Zdrojové znaky usporiadame tak, aby platilo $p_1 \geqq p_2 \geqq \cdots \geqq p_r$.
2. Množinu takto usporiadaných znakov rozdelíme na dve časti (pri zachovaní usporiadania) tak, aby rozdiel súčtov pravdepodobností týchto skupín bol čo najmenší.
3. Prvej skupine pridelíme kódový znak 0 a druhej 1 alebo naopak (je to vec dohody).
4. S každou z takto získaných podmnožín opakujeme proces popísaný v krokoch 2. a 3. dovtedy, kým nedostaneme len jednoprvkové množiny.
5. Kódové slová dostaneme tak, že čítame kódové znaky v poradí, ako boli pre daný znak prideľované.

Príklad 1.3.12 *Pomocou Shannon–Fanovej konštrukcie nájdite binárny prefixový kód pre zakódovanie danej zdrojovej abecedy s pravdepodnosťami výskytu a určte jeho priemernú dĺžku kódového slova.*

zdrojová abeceda	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
pravdepod. výskytu	$0,22$	$0,19$	$0,18$	$0,15$	$0,10$	$0,09$	$0,04$	$0,03$

Riešenie:

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
$0,22$	$0,19$	$0,18$	$0,15$	$0,10$	$0,09$	$0,04$	$0,03$
0	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1
		0	1	0	1	1	1
					0	1	1
						0	1

Kódové slová čítame zhora dole, tak ako boli jednotlivé znaky pridávané, takže kód $\mathcal{K}_1$ je takýto:

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
00	01	100	101	110	1110	11110	11111

Keďže to, ako rozdelíme jednotlivé množiny, nie je vždy jednoznačné, uvedieme aj ďalšiu možnosť Shannon–Fanovej konštrukcie.

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
$0,22$	$0,19$	$0,18$	$0,15$	$0,10$	$0,09$	$0,04$	$0,03$
0	0	0	1	1	1	1	1
0	1	1	0	0	1	1	1
	0	1	0	1	0	1	1
						0	1

Tento kód $\mathcal{K}_2$ je zase takýto:

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
00	010	011	100	101	110	1110	1111

Priemerná dĺžka kódového slova pre kód $\mathcal{K}_1$ je $2,82$ a pre kód $\mathcal{K}_2$ je $2,85$. Ako možno vidieť, rôzne Shannon–Fanove konštrukcie dávajú kódy s rôznymi priemernými dĺžkami kódových slov, čo znamená, že táto metóda neposkytuje vždy najkratší kód. Jej výhodou je však to, že je prehľadná a dosť rýchla.

Huffmanova konštrukcia binárneho prefixového kódu

Podstata Huffmanovej konštrukcie spočíva v tom, že sa konštrukcia kódu pre r zdrojových znakov postupne redukuje na konštrukciu kódu pre $r-1$ zdrojových znakov.

Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $p_1, p_2, \ldots, p_r$. Konštrukciu podobne ako pri predchádzajúcej metóde popíšeme v niekoľkých krokoch:

1. Zdrojové znaky usporiadame tak, aby platilo $p_1 \geqq p_2 \geqq \cdots \geqq p_r$.

2. Dvom zdrojovým znakom s najmenšími pravdepodobnosťami priradíme prvému kódový znak 0 a druhému 1 alebo naopak (podľa dohody). Potom ich zredukujeme do jedného spoločného zdrojového znaku s pravdepodobnosťou rovnou súčtu tých dvoch pravdepodobností. Takto sme dostali redukovanú abecedu s $r-1$ zdrojovými znakmi.

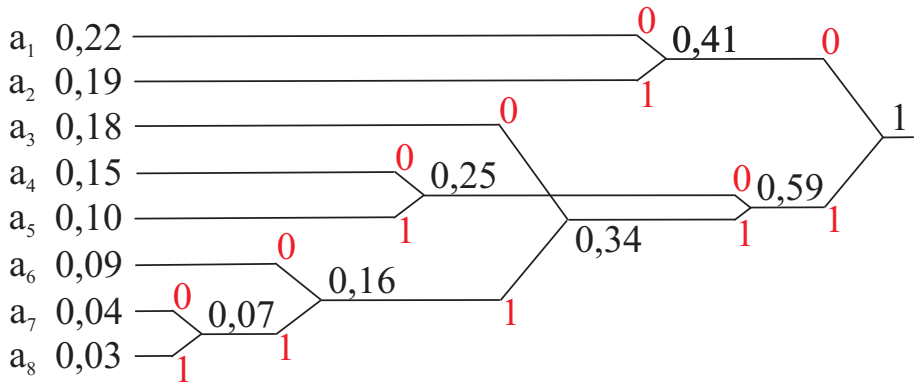
3. Znova postupujeme ako v kroku 2., len kódový znak 0 resp. 1 priradíme všetkým pôvodným zdrojovým znakom, ktoré boli do aktuálneho redukovaného znaku pospájané. Opakujeme dovtedy, kým neostane len jeden znak s pravdepodobnosťou 1.

4. Kódové slová zapíšeme tak, že čítame kódové znaky v opačnom poradí, ako boli danému znaku prideľované.

Príklad 1.3.13 *Pomocou Huffmanovej konštrukcie nájdite binárny prefixový kód pre zakódovanie danej zdrojovej abecedy s pravdepodnosťami výskytu a určte jeho priemernú dĺžku kódového slova.*

zdrojová abeceda	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
pravdepod. výskytu	$0,22$	$0,19$	$0,18$	$0,15$	$0,10$	$0,09$	$0,04$	$0,03$

Riešenie: Predchádzajúci algoritmus aplikujeme na tento príklad:



Kód $\mathcal{K}$ získaný Huffmanovou konštrukciou je takýto:

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
00	01	110	100	101	1110	11110	11111

a priemerná dĺžka kódového slova je $\bar{d}(\mathcal{K}) = 2,82$.

Huffmanovou konštrukciou dostaneme vždy najkratší kód, čo v nasledujúcich vetách dokážeme.

Veta 1.3.5 *Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $p_1, p_2, \ldots, p_r$ a platí $p_1 \geqq p_2 \geqq \cdots \geqq p_r$. Potom pre túto abecedu existuje najkratší prefixový kód $\mathcal{K} = \{K(a_1), K(a_2), \ldots, K(a_r)\}$ taký, že kódové slová $K(a_{r-1})$ a $K(a_r)$ majú rovnakú dĺžku a líšia sa len v poslednom znaku.*

Dôkaz: Zvoľme nejaký najkratší prefixový kód $\mathcal{K}$ abecedy A. Ak má byť kód najkratší, tak zdrojovým znakom s najmenšou pravdepodobnosťou musia zodpovedať kódové slová s najväčsou dĺžkou. (Ak by slovo $K(a_r)$ malo menšiu dĺžku ako nejaké $K(a_i), i < r$ a zároveň $p_r < p_i$, tak výmenou kódových slov $K(a_i)$ a $K(a_r)$ by sme dostali kód s menšou priemernou dĺžkou slova ako kód $\mathcal{K}$, čo je spor s tým, že $\mathcal{K}$ je najkratší.)

Predpokladajme, že kódové slová $K(a_{r-1})$ a $K(a_r)$ nemajú spoločný prefix dĺžky $d_r - 1$. Potom však musí existovať nejaké kódové slovo $K(a_i)$ dĺžky takej ako $K(a_r)$, ktoré má s $K(a_r)$ spoločný prefix. (Ináč by sme slovo $K(a_r)$ mohli nahradiť jeho prefixom a dostali by sme kód s menšou priemernou dĺžkou slova ako kód $\mathcal{K}$, čo je spor.) Podobne musí existovať nejaké kódové slovo $K(a_j)$ dĺžky $d_{r-1} = d_r$, ktoré má s $K(a_{r-1})$ spoločný prefix. Výmenou kódových slov $K(a_i)$ a $K(a_{r-1})$ dostávame kód s takou istou priemernou dĺžkou slova ako

kód $\mathcal{K}$, teda najkratší, kódové slová zodpovedajúce najmenším pravdepodobnostiam majú rovnaku dĺžku a líšia sa len v poslednom znaku. □

Definícia 1.3.6 *Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $P = \{p_1, p_2, \ldots, p_r\}$ a platí $p_1 \geqq p_2 \geqq \cdots \geqq p_r$. Zostrojíme abecedu $A' = \{a'_1, a'_2, \ldots, a'_{r-1}\}$ s pravdepodobnosťami $Q = \{q_1, q_2, \ldots, q_{r-1}\}$ tak, že zdrojové znaky a_{r-1} a a_r nahradíme jedným znakom a'_j a priradíme mu pravdepodobnosť $q_j = p_{r-1} + p_r$. Ostatným znakom ponecháme pôvodné pravdepodobnosti a všetky znaky zoradíme tak, aby platilo $q_1 \geqq q_2 \geqq \cdots \geqq q_{r-1}$. Takto zostrojenú abecedu A' nazývame redukovanou abecedou k abecede A.*

Veta 1.3.6 *(o Huffmanovom kóde) Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu znakov $P = \{p_1, p_2, \ldots, p_r\}$ takých, že platí $p_1 \geqq p_2 \geqq \cdots \geqq p_r$ a k nej redukovaná abeceda $A' = \{a'_1, a'_2, \ldots, a'_{r-1}\}$ s pravdepodobnosťami $Q = \{q_1, q_2, \ldots, q_{r-1}\}$, pre ktoré platí $q_1 \geqq q_2 \geqq \cdots \geqq q_{r-1}$. Ak kód $\mathcal{K} = \{K(a'_1), K(a'_2), \ldots, K(a'_{r-1})\}$ je najkratší prefixový binárny kód pre abecedu A' s pravdepodobnosťami Q, tak kód $\bar{\mathcal{K}} = \{K(a_1), K(a_2), \ldots, K(a_{j-1}), K(a_{j+1}), \ldots, K(a_j)0, K(a_j)1\}$, kde $K(a'_i) = K(a_i)$ pre $\forall i = 1, \ldots, r-1$, je najkratším kódom pre kódovanie abecedy A s pravdepodobnosťami P.*

Dôkaz: Keďže kód $\mathcal{K}$ je prefixový, tak aj kód $\bar{\mathcal{K}}$ je prefixový a jeho priemerná dĺžka $\bar{d}(\bar{\mathcal{K}}) = \bar{d}(\mathcal{K}) + q_j$. Aby sme ukázali, že $\bar{\mathcal{K}}$ je najkratší, musíme ukázať, že pre hocijaký iný kód $\bar{\mathcal{K}}^*$ platí $\bar{d}(\bar{\mathcal{K}}^*) \geqq \bar{d}(\bar{\mathcal{K}})$.

Podľa vety 1.3.5 existuje nejaký najkratší prefixový kód na kódovanie abecedy $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami $P = \{p_1, p_2, \ldots, p_r\}$, že platí $p_1 \geqq p_2 \geqq \cdots \geqq p_r$, označme ho $\bar{\mathcal{K}}^*$. V tomto kóde znakom a_{r-1} a a_r s minimálnymi pravdepodobnosťami p_{r-1} a p_r zodpovedajú kódové slová $\bar{K}^*(a_{r-1})$ a $\bar{K}^*(a_r)$ dĺžky $d^*_{r-1} = d^*_r$, ktoré sa líšia len posledným znakom a majú spoločný prefix $\bar{K}^*(a_{r-1,r})$. Priemerná dĺžka kódového slova pre tento kód je

$$\bar{d}(\bar{\mathcal{K}}^*) = d^*_1 p_1 + d^*_2 p_2 + \cdots + d^*_{r-1} p_{r-1} + d^*_r p_r = d^*_1 p_1 + d^*_2 p_2 + \cdots + d^*_r (p_{r-1} + p_r).$$

Zostrojíme kód $\mathcal{K}^*$ redukovanej abecedy takto:

zdrojový znak	a_1	a_2	$\ldots$	a_{r-2}	a_j
kódové slovo	$\bar{K}^*(a_1)$	$\bar{K}^*(a_2)$	$\ldots$	$\bar{K}^*(a_{r-2})$	$\bar{K}^*(a_{r-1,r})$

Priemerná dĺžka kódového slova pre redukovanú abecedu je

$$\bar{d}(\mathcal{K}^*) = d^*_1 p_1 + d^*_2 p_2 + \cdots + d^*_{r-2} p_{r-2} + (d^*_r - 1)(p_{r-1} + p_r).$$

Podobne zapíšeme, že pre priemernú dĺžku kódového slova pre kód $\bar{\mathcal{K}}$ platí

$$\bar{d}(\bar{\mathcal{K}}) = d_1 p_1 + d_2 p_2 + \cdots + d_{r-1} p_{r-1} + d_r p_r = d_1 p_1 + d_2 p_2 + \cdots + d_r (p_{r-1} + p_r)$$

a priemerná dĺžka kódového slova kódu $\mathcal{K}$ pre redukovanú abecedu je

$$\bar{d}(\mathcal{K}) = d_1 p_1 + d_2 p_2 + \cdots + d_{r-2} p_{r-2} + (d_r - 1)(p_{r-1} + p_r).$$

Odtiaľ dostávame, že platí

$$\bar{d}(\bar{\mathcal{K}}^*) = \bar{d}(\mathcal{K}^*) + p_{r-1} + p_r$$

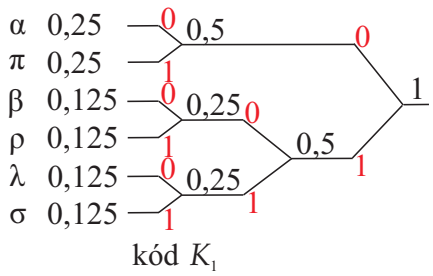
a tiež, že

$$\bar{d}(\bar{\mathcal{K}}) = \bar{d}(\mathcal{K}) + p_{r-1} + p_r.$$

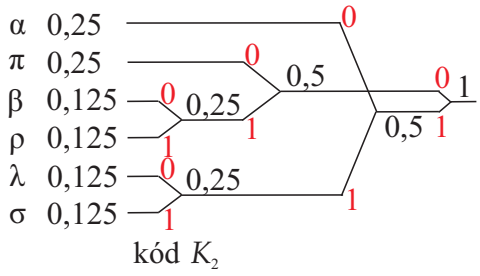
Pretože kód $\mathcal{K}$ je najkratším kódom redukovanej abecedy, tak platí $\bar{d}(\mathcal{K}) \leqq \bar{d}(\mathcal{K}^*)$. Dosadením dostaneme $\bar{d}(\bar{\mathcal{K}}) = \bar{d}(\mathcal{K}) + p_{r-1} + p_r \leqq \bar{d}(\mathcal{K}^*) + p_{r-1} + p_r = \bar{d}(\bar{\mathcal{K}}^*)$, teda platí $\bar{d}(\bar{\mathcal{K}}) \leqq \bar{d}(\bar{\mathcal{K}}^*)$, čo sme chceli dokázať. □

Príklad 1.3.14 *Nájdite najkratší binárny prefixový kód na zakódovanie zdrojovej abecedy* $A = \{\alpha, \beta, \lambda, \pi, \rho, \sigma\}$, *ak vieme, že znaky* α *a* π *sa nachádzajú v správach dvakrát častejšie ako ostatné znaky.*

Riešenie: Najprv si určíme pravdepodobnosti. Znaky $\beta, \lambda, \rho, \sigma$ majú rovnakú pravdepodobnosť, označíme ju p. Potom znaky α a π majú dvojnásobnu pravdepodobnosť, teda $2p$. Vieme, že súčet pravdepodobností je jedna, takže $2 \cdot (2p) + 4 \cdot p = 8p = 1 \Rightarrow p = 0,125$. Použijeme Huffmanovu konštrukciu pre binárne kódy.



kód K_1



kód K_2

zdrojový znak	α	β	λ	π	ρ	σ
pravdepodobnosť	$0,25$	$0,125$	$0,125$	$0,25$	$0,125$	$0,125$
kód $\mathcal{K}_1$	00	100	110	01	101	111
kód $\mathcal{K}_2$	10	010	110	00	011	111

Uviedli sme dve rôzne konštrukcie, no obe sú Huffmanove a výsledkami sú dva rôzne kódy, ale s rovnakými priemernými dĺžkami kódových slov $\bar{d}(\mathcal{K}_1) = \bar{d}(\mathcal{K}_2) = 2,5$.

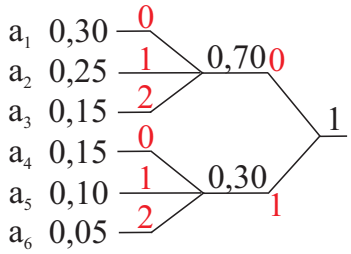
Huffmanova konštrukcia nie je jednoznačná v tom zmysle, že jej výsledkom môže byť aj viac rôznych kódov, dokonca ani dĺžky jednotlivých kódových slov sa nemusia zhodovať. Dochádza k tomu vtedy, ak pri redukcii najmenších pravdepodobností máme na výber z viacerých rovnakých hodnôt. No všetky kódy získané Huffmanovou konštrukciou majú rovnakú priemernú dĺžku kódového slova a sú najkratšie.

Podobne ako pre binárne kódy sa dajú skonštruovať najkratšie kódy aj pre rôzne n–árne kódy.

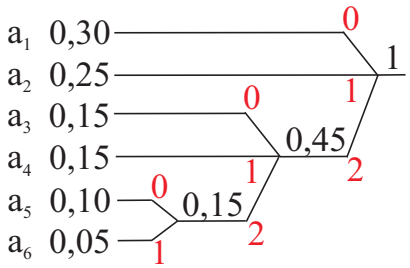
Príklad 1.3.15 *Nájdite najkratší ternárny prefixový kód pre zakódovanie danej zdrojovej abecedy s pravdepodnosťami výskytu a určte jeho priemernú dĺžku kódového slova.*

zdrojový znak	a_1	a_2	a_3	a_4	a_5	a_6
pravdepodobnosť	$0,3$	$0,25$	$0,15$	$0,15$	$0,1$	$0,05$

Riešenie: Použijeme podobnú konštrukciu ako bola Huffmanova pre binárne kódy, no redukovať budeme posledné 3 znaky s najmenšími pravdepodobnosťami.



Na konci nám však ostali len 2 znaky, ktoré sme redukovali. Teraz sa pokúsime použiť podobnú konštrukciu, no nezlučíme dva znaky v poslednej redukcii, ale v prvej.



zdrojový znak	a_1	a_2	a_3	a_4	a_5	a_6
pravdepodobnosť	$0,3$	$0,25$	$0,15$	$0,15$	$0,1$	$0,05$
kód $\mathcal{K}_1$	00	01	02	10	11	12
kód $\mathcal{K}_2$	0	1	20	21	220	221

Dostávame tak dva kódy s rôznymi dĺžkami kódových slov a ich priemerné dĺžky kódových slov sú $\bar{d}(\mathcal{K}_1) = 2$ a $\bar{d}(\mathcal{K}_2) = 1,6$. Je zrejmé, že kratším je kód $\mathcal{K}_2$ a záleží na tom, ako redukcie robíme.

Huffmanova konštrukcia n–árneho prefixového kódu

Huffmanova konštrukcia n–árneho prefixového kódu je založená na podobnom princípe ako pre binárny kód, no postupne sa redukujú nie dva znaky, ale n znakov s výnimkou prvej redukcie.

Nech je daná zdrojová abeceda $A = \{a_1, a_2, \ldots, a_r\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $P\{p_1, p_2, \ldots, p_r\}$ a kódová abeceda $T = \{t_1, t_2, \ldots, t_n\}$.

1. Zdrojové znaky usporiadame tak, aby platilo $p_1 \geqq p_2 \geqq \cdots \geqq p_r$.

2. Ak $r \leqq n$, tak zvolíme kód $\mathcal{K}$ tak, že $K(a_i) = t_i,\ \forall i = 1, 2, \ldots, r$ a priemerná dĺžka kódového slova $\bar{d}(\mathcal{K}) = p_1 + p_2 + \cdots + p_r$.

3. Ak $r > n$, tak s zdrojovým znakom s najmenšími pravdepodobnosťami priradíme s rôznych kódových znakov v ľubovoľnom poradí a zlučíme ich do jedného znaku a^*, ktorému priradíme pravdepodobnosť $p^* = p_{r-s+1}+p_{r-s+2}+\cdots+p_r$, čím vznikne redukovaná zdrojová abeceda $A' = \{a_1, a_2, \ldots, a_{r-s}, a^*\}$ s pravdepodobnosťami výskytu jednotlivých zdrojových znakov $P\{p_1, p_2, \ldots, p_{r-s}, p^*\}$.

4. Zdrojové znaky usporiadame podľa nerastúcich pravdepodobností. Ak je ich počet menší alebo rovný ako n, postupujeme ako v 2. Ak je ich počet väčší ako n, postupujeme ako v 3. s tým, že pri druhej a nasledujúcich redukciách zlučujeme n znakov s najmenšími pravdepodobnosťami.

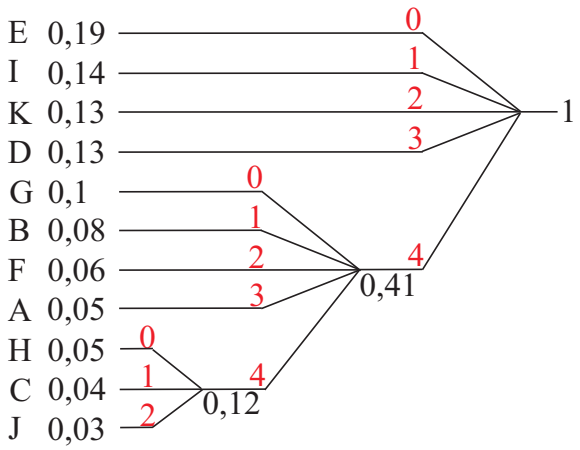
5. Kódové slová dostaneme tak, že čítame kódové znaky v opačnom poradí, ako boli danému znaku prideľované.

Číslo s určujúce počet redukovaných znakov pri prvej redukcii závisí od počtu zdrojových a kódových znakov. Pri prvej redukcii sčítavame s sčítancov, ostane nám $r - s$ pôvodných a jeden nový znak. Pri druhej sčítavame n sčítancov, $n - 1$ pôvodných a jeden z predchádzajúcej redukcie (ak by sme ho nesčítali teraz, vymení sa len poradie, kedy ho sčítame, nie to, či ho sčítame). Ak vykonáme prvú redukciu a k ďalších, tak dostaneme $r - s = k(n - 1) \Rightarrow s = r - k(n - 1)$. Keďže všetky čísla r, s, k, n sú celé nezáporné čísla, tak s je zvyšok po delení čísla r číslom $n - 1$. Ak by nám s vyšlo 0 resp. 1, tak v prvej redukcii sčítame $0 + n - 1$ resp. $1 + n - 1$ znakov.

Príklad 1.3.16 *Nájdite najkratší pentárny prefixový kód pre zakódovanie danej zdrojovej abecedy s pravdepodnosťami výskytu a určte jeho priemernú dĺžku kódového slova.*

zdrojový znak	A	B	C	D	E	F	G	H	I	J	K
pravdepodobnosť	$0,05$	$0,08$	$0,04$	$0,13$	$0,19$	$0,06$	$0,1$	$0,05$	$0,14$	$0,03$	$0,13$

Riešenie: Zdrojové znaky usporiadame podľa nerastúcich pravdepodobností a použijeme Huffmanovu konštrukciu. Číslo $r = 11$ dáva po delení číslom $n - 1 = 4$ zvyšok 3, preto pri prvej redukcii sčítame 3 zdrojové znaky a pri ďalších budeme sčítavať po 5 znakov.



Kód ktorý takto dostaneme je uvedený v tabuľke a priemerná dĺžka kódového slova je $\bar{d}(\mathcal{K}) = 2{\cdot}0,05{+}2{\cdot}0,08{+}3{\cdot}0,04{+}0,13{+}0,19{+}2{\cdot}0,06{+}2{\cdot}0,1{+}3{\cdot}0,05{+}0,14{+}3{\cdot}0,03{+}0,13 = 1,41$.

zdrojový znak	A	B	C	D	E	F	G	H	I	J	K
pravdepodobnosť	$0,05$	$0,08$	$0,04$	$0,13$	$0,19$	$0,06$	$0,1$	$0,05$	$0,14$	$0,03$	$0,13$
pravdepodobnosť	43	41	441	3	0	42	40	440	1	442	2

Kapitola 2

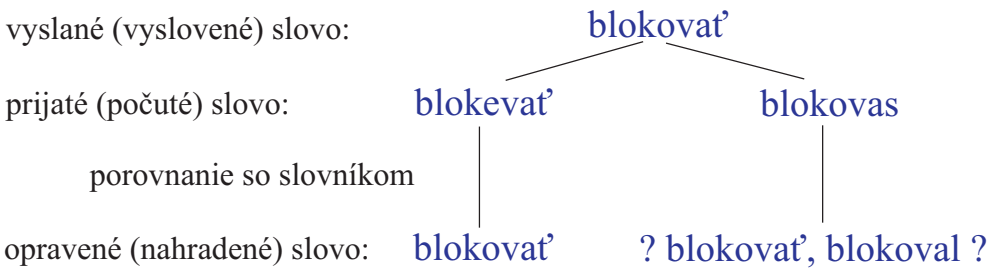
Bezpečnostné kódy

V predchádzajúcej kapitole sme sa zaoberali kódovaním zdrojových znakov a správ. Predpokladali sme, že vyslaná správa sa príjme v takej istej podobe, ako bola poslaná, resp. nebrali sme do úvahy možnosť vzniku chýb pri prenose informácií. Takže jediným cieľom bolo nájsť čo najkratšie kódy, aby sa čo najviac znížilo množstvo prenášaných informácií a tým zvýšila efektivita kódu.

Počas prenosu v reálnom prostredí však dochádza z rôznych príčin k tzv. šumu. To znamená, že počas prenosu správy dôjde k nejakej chybe a prijatá správa nie je úplne totožná s vyslanou. Môže dôjsť k "vzniku" znaku, aj keď nebol vyslaný, alebo k "pohlteniu" vyslaného znaku. Tieto chyby sa však objavujú dosť zriedka, preto sa nimi nebudeme zaoberať. Najčastejšou chybou, ktorá sa môže vyskytnúť, je zámena vyslaného kódového znaku za iný znak. Preto cieľom bude konštruovať kódy, ktoré budú slúžiť nie len na kódovanie zdroja, ale aj na detekciu a následne aj korekciu chýb vzniknutých počas prenosu, teda na zabezpečenie informácie. Všeobecne sa takéto kódy nazývajú bezpečnostné kódy alebo tiež samoopravné kódy. Takéto kódy už nie sú najkratšie, ale obsahujú aj určité množstvo nadbytočnej informácie a princíp činnosti týchto kódov spočíva v tom, že prijaté slovo sa porovnáva, či je kódovým slovom daného kódu, resp. ku ktorému kódovému slovu je najbližšie. Ako bezpečnostné kódy sa prevažne používajú blokové kódy, takže ďalej sa budeme zaoberať iba blokovými kódmi.

2.1 Detekcia a korekcia chýb

Opravovanie určitého "kódovania" vykonávame denne, mnohokrát úplne automaticky. Predstavme si množinu všetkých spisovných slov v slovenčine. Vyslovíme niektoré slovo, napríklad slovo "blokovať". Keďže je okolo hluk, poslucháč zachytí slovo "blokevať". No takmer každý poslucháč odhalí, že bolo povedané slovo "blokovať", pretože je to spisovné slovo, ktoré sa najviac podobá na počuté slovo. V podstate došlo k porovnaniu prijatého slova so slovníkom slovenských slov a bolo vybraté najbližsie slovo. Čo ak poslucháč zachytí slovo "blokovas"? Nevie, ktoré slovo bolo povedané, lebo existujú minimálne dve slová, ktoré sa líšia od počutého jednou hláskou: "blokoval", "blokovať".



Príklad 2.1.1 *Máme kód daný tabuľkou:*

zdrojový znak	△	□	◯
kódové slovo	001	011	111

Bolo vyslané slovo 001 *(△). Vznikla chyba a bolo prijaté slovo* 000*. Odhalí kód túto chybu, resp. ju aj opraví? A čo ak by nastala chyba na inom mieste a namiesto vyslaného slova* 001 *(△) by sa prijalo slovo* 101 *alebo* 011*?*

Riešenie: V každom z prípadov bolo vyslané kódové slovo 001 (△). Našim "slovníkom", s ktorým budeme prijaté slová porovnávať, je množina kódových slov $\{001, 011, 111\}$. Vo všetkých prípadoch došlo počas prenosu k zámene jediného znaku vo vyslanom slove.

- Chyba nastala na 3. mieste a prijalo sa slovo 000. Porovnáme s množinou kódových slov a zistíme, že slovo, ktoré sa líši len jedným znakom od prijatého je 001 (△), takže je najpravdepodobnejšie, že bolo aj vyslaným. Chybu sme teda odhalili a aj opravili.

- Chyba nastala na 1. mieste a prijalo sa slovo 101. Porovnáme ho s množinou kódových slov a zistíme, že také slovo sa v nej nenachádza, no sú tam dve slová, ktoré sa od prijatého líšia len jedným znakom: 001 (△) a 111 (◯). Chybu sme teda odhalili, no nevieme ju opraviť.

- Chyba nastala na 2. mieste a prijalo sa slovo 011 (□). Porovnáme s množinou kódových slov a zistíme, že takéto slovo sa v nej nachádza a teda zrejme ho budeme považovať za vyslané slovo. Chybu sme v tomto prípade ani neodhalili.

Princíp pri umelých kódoch je teda podobný ako v bežnom jazyku.

Ako sme už vyššie uviedli, budeme sa zaoberať len blokovými (rovnomernými kódmi). Pri prenose budeme predpokladať, že:

- chyba vzniká iba zámenou jedného znaku kódovej abecedy za iný znak,

- žiaden znak nie je odolnejší voči chybe ako iný,

- výsledok prenosu nejakého znaku neovplyvňuje, či bude ďalší znak prenesený správne, či nie.

Definícia 2.1.1 *Nech je daná zdrojová abeceda A, kódová abeceda T a nech je dané blokové kódovanie $K : A \to T^n$. Kód $\mathcal{K}$ prislúchajúci tomuto kódovaniu nazývame úplným kódom, ak platí $\mathcal{K} = T^n$.*

Je zrejmé, že pre zabezpečenie informácií nemá zmysel používať úplné kódy, pretože každé prijaté slovo bude kódovým bez ohľadu na to, či chyba vznikla, alebo nie. A tiež je zrejme, že na bezpečnosť kódu bude mať významný vplyv to, aký je veľký rozdiel medzi mohutnosťou (počtom prvkov) množín $\mathcal{K}$ a T^n. Znamená to, že správa nebude kódovaná s maximálnou informačnou hodnotou, ale bude obsahovať nadbytočnosť – redundanciu. Cieľom bezpečnostného kódu síce bude umelo zvýšiť redundanciu, no na druhej strane pri zachovaní primeranej úspornosti prenosu.
Označenie: Je dané blokové kódovanie $K : A \to T^n$ dĺžky n.

A – zdrojová abeceda

T – kódová abeceda

n – dĺžka každého kódového slova

T^n – množina všetkých usporiadaných n–tíc prvkov (n znakových slov) z T

$\mathcal{K}$ – množina všetkých kódových slov

$T^n - \mathcal{K}$ – množina všetkých nekódových slov dĺžky n (nekódových n znakových slov).

Definícia 2.1.2 *Hammingovou vzdialenosťou dvoch n znakových slov $\boldsymbol{v}, \boldsymbol{w}$ z množiny T^n nazývame počet odlišných znakov v týchto slovách. Označujeme ju $d(\boldsymbol{v}, \boldsymbol{w})$.*

Poznámka: Hammingova vzdialenosť je binárnou operáciou, ktorá dvojici slov priradí nezáporné celé číslo. Dá sa dokázať, že Hammingova vzdialenosť je metrikou na T^n a teda pre Hammingovu vzdialenosť dvoch slov $\boldsymbol{v}, \boldsymbol{w} \in T^n$ platí:

$$d(\boldsymbol{v}, \boldsymbol{w}) \geqq 0,$$
$$d(\boldsymbol{v}, \boldsymbol{w}) = d(\boldsymbol{w}, \boldsymbol{v}),$$
$$d(\boldsymbol{v}, \boldsymbol{w}) \leqq d(\boldsymbol{v}, \boldsymbol{x}) + d(\boldsymbol{x}, \boldsymbol{w}).$$

Definícia 2.1.3 *Minimálnou vzdialenosťou blokového kódu $\mathcal{K}$ nazývame najmenšiu nenulovú Hammingovu vzdialenosť dvoch kódových slov. Označujeme ju d.*

Príklad 2.1.2 *Je daný kód $\mathcal{K}$ tabuľkou. Určte Hammingovu vzdialenosť každých dvoch kódových slov a tiež minimálnu vzdialenosť kódu.*

A	B	C	D
0000	0110	1100	1111

Riešenie: Hammingove vzdialenosti sú zapísané v nasledujúcej tabuľke a minimálna vzdialenosť kódu je $d = 2$.

$d(\boldsymbol{v}, \boldsymbol{w})$	0000	0110	1100	1111
0000	0	2	2	4
0110	2	0	2	2
1100	2	2	0	2
1111	4	2	2	0

Definícia 2.1.4 *Nech sú vysielané kódové slová z $\mathcal{K}$ a prijímané slová z T^n. Ak sme prijali nekódové slovo, hovoríme, že sme odhalili chybu. (Ak sme prijali kódové slovo, tak buď k chybe nedošlo, alebo sme ju neodhalili.)*

Definícia 2.1.5 *Nech je vyslané kódové slovo. Hovoríme, že došlo k t–násobnej chybe, ak Hammingova vzdialenosť vyslaného a prijatého slova je najviac t.*

Definícia 2.1.6 *Hovoríme, že kód $\mathcal{K}$ odhaľuje t–násobné chyby, ak pri vyslaní kódového slova a vzniku t–násobnej chyby je prijaté slovo vždy nekódové.*

V príkladoch a ilustračných úlohách budeme používať niektoré špeciálne kódy, preto uvádzame ich stručný popis.

Binárny kód "dva z päť" je kód pozostávajúci zo všetkých pätíc núl a jednotiek, ktoré obsahujú práve dve jednotky. $\mathcal{K} = \{00011, 00101, 01001, 10001, 00110, 01010, 10010, 01100, 10100, 11000\}$.

Opakovací kód dĺžky n je q–árny kód tvorený všetkými n–ticami, ktoré majú všetky znaky rovnaké. Napríklad opakovací kód dĺžky 3 je kód $\mathcal{K} = \{000, 111, 222, \ldots, (q-1)(q-1)(q-1)\}$

Koktavý kód dĺžky n, kde n je párne, je q–árny kód tvorený všetkými n–ticami tak, že každý kódový znak sa opakuje dvakrát po sebe. Napríklad binárny koktavý kód dĺžky 6 je $\mathcal{K} = \{000000, 000011, 001100, 001111, 110000, 110011, 111100, 111111\}$.

Kód celkovej kontroly parity dĺžky n je binárny kód tvorený n–ticami tak, že posledný znak je súčtom predchádzajúcich znakov. Ako príklad uvádzame $\mathcal{K} = \{000, 011, 101, 110\}$.

Príklad 2.1.3 *Je daný binárny kód "dva z päť". Určte minimálnu vzdialenosť daného kódu a zistite, aké chyby je schopný jednoznačne odhaliť.*

Riešenie: Najprv zistíme, aký je počet kódových slov $|\mathcal{K}|$, teda počet pätíc núl a jednotiek takých, že jednotky sú práve dve. Je to vlastne počet dvojprvkových kombinácií z piatich prvkov:

$$|\mathcal{K}| = \binom{5}{2} = \frac{5!}{2! \cdot 3!} = 10$$

Vypíšeme všetky kódové slová.

00011	00101	01001	10001	00110
01010	10010	01100	10100	11000

Hammingove vzdialenosti zapíšeme do prehľadnej tabuľky:

$d(\boldsymbol{v},\boldsymbol{w})$	00011	00101	01001	10001	00110	01010	10010	01100	10100	11000
00011	0	2	2	2	2	2	2	4	4	4
00101	2	0	2	2	2	4	4	2	2	4
01001	2	2	0	2	4	2	4	2	4	2
10001	2	2	2	0	4	4	2	4	2	2
00110	2	2	4	4	0	2	2	2	2	4
01010	2	4	2	4	2	0	2	2	4	2
10010	2	4	4	2	2	2	0	4	2	2
01100	4	2	2	4	2	2	4	0	2	2
10100	4	2	4	2	2	4	2	2	0	2
11000	4	4	2	2	4	2	2	2	2	0

Minimálna vzdialenosť kódu "dva z päť" je $d = 2$.

Ak vznikne jednoduchá (jednonásobná) chyba:

- vyslané je 00011 $\rightarrow$ prijaté je $\underline{1}$0011 $\rightarrow$ kód chybu odhalí
- vyslané je 00011 $\rightarrow$ prijaté je 0001$\underline{0}$ $\rightarrow$ kód chybu odhalí

Pri jednoduchej chybe vzniknú tri jednotky, alebo jedna jednotka, takže chyba je odhalená vždy.

Ak vznikne dvojnásobná chyba:

- vyslané je 00011 $\rightarrow$ prijaté je 0$\underline{11}$11 $\rightarrow$ kód chybu odhalí
- vyslané je 00011 $\rightarrow$ prijaté je 00$\underline{10}$1 $\rightarrow$ kód chybu neodhalí, lebo vzniklo iné kódové slovo

Dvojnásobná chyba nemusí byť odhalená vždy.

Príklad 2.1.4 *Určte minimálnu vzdialenosť ternárneho opakovacieho kódu dĺžky* 4. *(Je to kód, ktorý pozostáva len zo slov vytvorených tak, že sa n–krát zopakuje kódový znak.) Určte minimálnu vzdialenosť daného kódu a zistite, aké chyby je schopný jednoznačne odhaliť.*

Riešenie: Kód je tvorený tromi kódovými slovami $\mathcal{K} = \{0000, 1111, 2222\}$. Hammingove vzdialenosti sú v nasledujúcej tabuľke:

$d(\boldsymbol{v},\boldsymbol{w})$	0000	1111	2222
0000	0	4	4
1111	4	0	4
2222	4	4	0

Minimálna vzdialenosť kódu je $d = 4$.

Kód je schopný odhaliť aj trojnásobné chyby, keďže v tomto prípade sa určite nebudú všetky znaky zhodovať, no niektoré štvornásobné chyby už neodhalí. Napr.: ak je vyslané slovo 0000 → prijaté je slovo $\underline{1}111$ → kód chybu neodhalí.

Pozorovaním v predchádzajúcich príkladoch sme si mohli všimnúť vzťah medzi minimálnou dĺžkou kódu a tým, aké chyby odhaľuje. Vplyv na to, aké veľké chyby kód odhalí, má to, ako veľmi sa od seba kódové slová líšia.

Pozorovanie: Blokový kód minimálnej vzialenosti d odhalí t–násobné chyby pre $t < d$. Ale nie je už schopný odhaliť všetky d–násobné chyby.

Definícia 2.1.7 *Hovoríme, že kód $\mathcal{K}$ opravuje t–násobné chyby, ak pri vyslaní kódového slova $\boldsymbol{v}$ a vzniku t–násobnej chyby má prijaté slovo $\boldsymbol{w}$ Hammingovu vzdialenosť $d(\boldsymbol{v}, \boldsymbol{w})$ menšiu, ako je jeho Hammingova vzdialenosť od ľubovoľného iného kódového slova $\boldsymbol{x}$. Teda $d(\boldsymbol{v}, \boldsymbol{x}) > d(\boldsymbol{v}, \boldsymbol{w})$ pre všetky $\boldsymbol{x} \neq \boldsymbol{v}$.*

Príklad 2.1.5 *Máme binárny kód "dva z päť". Aké chyby je schopný tento kód opraviť?*

Riešenie: Ak vznikne jednoduchá (jednonásobná) chyba:

- vyslané je 00011 → prijaté je $\underline{1}0011$ → kód chybu odhalí
- opraviť ju však nedokáže, pretože existujú až tri kódové slová (00011,10001,10010), ktoré sa od prijatého slova líšia len jedným znakom. Ináč povedané, najmenšiu Hammingovu vzdialenosť má prijaté slovo hneď od troch kódových slov.

Príklad 2.1.6 *Máme ternárny opakovací kód dĺžky 4. Aké chyby je schopný jednoznačne opraviť?*

Riešenie: Ak vznikne jednoduchá (jednonásobná) chyba:

- vyslané je 0000 → prijaté je $\underline{1}000$ → kód chybu odhalí
- kódové slovo s najmenšiou Hammingovou vzdialenosťou 1 je len jedno – 0000, teda vie túto chybu aj opraviť.

Ak vznikne dvojnásobná chyba:

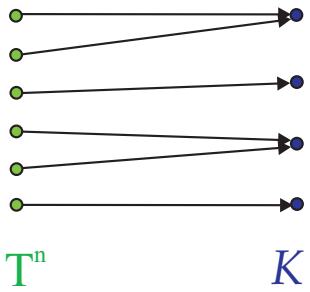
- vyslané je 0000 → prijaté je $\underline{1}0\underline{2}0$ → kód chybu odhalí
- kódové slovo s najmenšiou Hammingovou vzdialenosťou 2 je len jedno – 0000, teda vie túto chybu aj opraviť.
- ak je však vyslané 0000 → a prijaté je $\underline{1}0\underline{1}0$ → kód chybu odhalí
- kódové slová s najmenšiou Hammingovou vzdialenosťou 2 sú však dva – 0000 a 1111 a kód túto chybu opraviť nevie.

Opakovací kód dĺžky 4 jednoznačne opravuje len jednonásobné chyby.

Pozorovanie: Blokový kód minimálnej vzialenosti d opraví všetky t–násobné chyby pre $t < d/2$, resp. $d \geqq 2t + 1$.

Definícia 2.1.8 *Dekódovaním nazývame ľubovoľné zobrazenie* $\kappa : T^n \to \mathcal{K}$ *také, že* $\kappa(\boldsymbol{v}) = \kappa(v_1 v_2 \dots v_n) = v_1 v_2 \dots v_n$, *ak* $v_1 v_2 \dots v_n \in \mathcal{K}$.

Dekódovaním sa nemyslí dekódovanie kódovej správy ani kódového slova v zmysle, že sa jej priradí zodpovedajúca zdrojová správa, či zdrojový znak. Do prenosového kanála sa vysielajú kódové slová z $\mathcal{K}$, ktoré môžu byť vplyvom šumu zmenené, takže sa prijímajú slová z množiny T^n. Dekódovaním sa teda rozumie také zobrazenie, ktoré na výstupe koriguje vzniknuté chyby, takže prijatým slovám priradzuje kódové slová, no prijaté kódové slová musia byť zachované. Takéto dekódovanie nemusí byť prostým (injektívnym) zobrazením, no každopádne je "na" (surjektívnym) zobrazením.



Definícia 2.1.9 *Úplným dekódovaním nazývame zobrazenie množiny* T^n *na množinu* $\mathcal{K}$, *teda také dekódovanie, ktoré každému slovu z* T^n *priradí nejaké kódové slovo z* $\mathcal{K}$. *Čiastočným dekódovaním nazývame zobrazenie z množiny* T^n *na množina* $\mathcal{K}$.

Príklad 2.1.7 *Je daný binárny opakovací kód dĺžky* 6. *Sú dané dekódovania* κ_1 *a* κ_2. *Určte, či sú úplné, či čiastočné.* $\forall w_1 w_2 \dots w_6$ *je:*

$$\kappa_1(w_1 w_2 \dots w_6) = \begin{cases} 000000; & \text{ak } w_i = 0 \text{ pre aspoň 4 indexy,} \\ & \text{alebo ak } w_i = 0 \text{ pre 3 indexy a zároveň } w_1 = 0, \\ 111111; & \text{inak.} \end{cases}$$

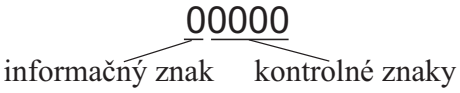
$$\kappa_2(w_1 w_2 \dots w_6) = \begin{cases} 000000; & \text{ak } w_i = 0 \text{ pre aspoň 4 indexy,} \\ 111111; & \text{ak } w_i = 1 \text{ pre aspoň 4 indexy.} \end{cases}$$

Riešenie: κ_1 je úplné dekódovanie, κ_2 je len čiastočné dekódovanie, keďže nemáme určené, čo sa má priradiť slovám, ktoré obsahujú tri nuly a tri jednotky.

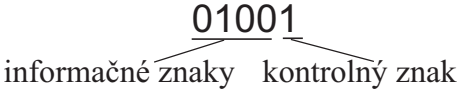
2.2 Informačné znaky

Jednou z možností zabezpečenia kódu je umelé zvýšenie redundancie (nadbytočnosti) pridaním kódových znakov ku kódovým slovám. Potom v každom kódovom slove budú znaky, ktoré nesú informáciu – informačné znaky a znaky, ktoré sú pridané pre zabezpečenie kódu – kontrolné znaky. Uvádzame dva príklady:

- Opakovací kód dĺžky 5:
  Ako to môžeme vidieť na nasledujúcej schéme, každé kódové slovo obsahuje jeden informačný a 4 kontrolné znaky.



- Kód celkovej kontroly parity dĺžky 5:
  Každé kódové slovo obsahuje 4 informačné a jeden kontrolný znak, ako to uvádza nasledujúca schéma.



Definícia 2.2.1 *Nech $\mathcal{K} \subset T^n$ je blokový kód dĺžky n. Ak existuje bijektívne zobrazenie ϕ všetkých slov dĺžky $k; k < n$ na množinu všetkých kódových slov (teda $\phi : T^k \to \mathcal{K}$), hovoríme, že kód $\mathcal{K}$ má k informačných znakov a $n - k$ kontrolných znakov. Zobrazenie ϕ nazývame kódovaním informačných znakov.*

Definícia 2.2.2 *Počet kontrolných znakov v kóde $\mathcal{K}$ nazývame absolútnou redundanciou kódu $\mathcal{K}$.*

Počet kontrolných znakov však nemusí mať veľkú výpovednú hodnotu o zabezpečovacej schopnosti kódu. Kódy rôznych dĺžok aj pri rovnakej absolútnej redundancii môžu mať veľmi odlišné zabezpečovacie schopnosti. Preto na porovnanie výhodnosti kódov zavádzame nový pojem.

Definícia 2.2.3 *Relatívnou redundanciou $R(\mathcal{K})$ kódu $\mathcal{K}$ nazývame pomer počtu kontrolných znakov v kóde ku počtu všetkých znakov v kóde.*

$$R(\mathcal{K}) = \frac{n-k}{n} = 1 - \frac{k}{n}.$$

Ak máme kód, kde sa nedajú jednoznačne odlíšiť informačné a zabezpečovacie znaky (napr. "dva z piatich"), tak sa pojem relatívnej redundancie definuje takto:

Definícia 2.2.4 *Nech je daný blokový kód $\mathcal{K}$ dĺžky n s kódovou abecedou, ktorá má t prvkov. Relatívnu redundanciu $R(\mathcal{K})$ kódu $\mathcal{K}$ definujeme nasledovne:*

$$R(\mathcal{K}) = 1 - \frac{log_t|\mathcal{K}|}{n}.$$

Často sa pri výbere kódov udávajú požiadavky na tzv. informačný pomer.

Definícia 2.2.5 *Informačný pomer $I(\mathcal{K})$ kódu $\mathcal{K}$ je definovaný takto:*

$$I(\mathcal{K}) = \frac{k}{n},$$

respektíve

$$I(\mathcal{K}) = \frac{log_t|\mathcal{K}|}{n}.$$

Informačný pomer teda možno ľahko vyjadriť ako $I(\mathcal{K}) = 1 - R(\mathcal{K})$.

Príklad 2.2.1 *Je daný binárny koktavý kód dĺžky 6. Koľko má informačných a koľko kontrolných znakov? Vypočítajte aj jeho relatívnu redundanciu.*

Riešenie: Vysielajú sa tri znaky, každý dvakrát po sebe, napr.: $010 \to 001100$ alebo $110 \to 111100$. Ide tu teda o kódovanie informačných znakov $\phi : T^3 \to \mathcal{K}$ a počet všetkých kódových slov je $|\mathcal{K}| = 2^3 = 8$. Z toho dostávame:

- počet všetkých znakov $n = 6$,
- počet informačných znakov $k = 3$,
- počet kontrolných znakov $n - k = 3$,
- relatívna redundancia $R(\mathcal{K}) = \frac{n-k}{n} = \frac{3}{6} = \frac{1}{2}$.

Definícia 2.2.6 *Blokový kód dĺžky n sa nazýva systematický, ak existuje celé číslo $k < n$ také, že $\forall v_1v_2 \dots v_k \in T^k$ existuje práve jedno kódové slovo $v_1v_2 \dots v_kv_{k+1} \dots v_n \in \mathcal{K}$.*

To znamená, že zabezpečovacia časť kódového slova je pripojená za informačnú časť.

Veta 2.2.1 *Minimálna vzdialenosť d systematického kódu dĺžky n nemôže prekročiť počet $n - k$ kontrolných znakov o viac ako 1. Teda $d \leqq n - k + 1$.*

Dôkaz: Máme blokový kód $\mathcal{K}$ dĺžky n. V kódovom slove si zvolíme ľubovoľných $k - 1$ znakov $v_1v_2 \dots v_{k-1}$ a označíme $\mathcal{K}_0 \subseteq \mathcal{K}$ množinu takých kódových slov z $\mathcal{K}$, ktoré majú prefix $v_1v_2 \dots v_{k-1}$.
Potom každé dve slová z $\mathcal{K}_0$ majú spoločných $k - 1$ znakov a teda ich Hammingova vzdialenosť je najviac $n-(k-1)$ a pre minimálnu vzdialenosť kódu $\mathcal{K}_0$ platí, že $d_0 \leqq n-(k-1)$. Keďže $\mathcal{K}_0 \subseteq \mathcal{K}$, tak platí $d \leqq d_0$ a odtiaľ $d \leqq n - (k - 1) = n - k + 1$. □

O čom vlastne táto veta hovorí?
Je tu problém pri výbere kódu – na jednej strane chceme mať v kódovom slove čo najviac informačných znakov vzhľadom na dĺžku (kvôli kapacitám prenosu), no na druhej strane potrebujeme aj čo najväčšie d, lebo chceme daným kódom opraviť čo najviac chýb vzniknutých pri prenose. Je nutné zvoliť vhodný kompromis medzi opravnou schopnosťou kódu (vyjadrenou minimálnou vzdialenosťou kódu d) a kapacitou prenosu (vyjadrenou počtom informačných znakov k) pri danej mohutnosti kódu (vyjadrenou počtom kódových znakov n). Hľadáme kódy, ktoré pri "rozumnom" počte informačných znakov majú relatívne jednoduchý algoritmus opravy "rozumného" počtu chýb.

Kapitola 3

Lineárne kódy

Základnou myšlienkou, na ktorej je založená teória bezpečnostných kódov, je použitie algebraických operácií na nejakej kódovej abecede. Keďže kódová abeceda je vždy konečná, aj štruktúry, s ktorými budeme pracovať, budú konečné štruktúry a operácie, ktoré budeme používať, budú založené na modulárnom sčítaní a modulárnom násobení. Keďže sa používajú blokové kódy dĺžky n, tak kódové slová sú n–tice vytvárané z prvkov kódovej abecedy a princíp počítania s kódovými slovami je taký, ako počítanie s algebraickými vektormi.

Základné pravidlá počítania s kódovými slovami

Nech je daná kódová abeceda $T = \{0, 1, 2, \ldots, m-1\}$ s m prvkami a nad ňou zostrojená množina T^n všetkých usporiadaných n–tíc vytvorených z prvkov kódovej abecedy, jej prvky budeme nazývať slová. Ľubovoľnú podmnožinu $\mathcal{K} \in T^n$ nazývame kódom a jej prvky nazývame kódové slová.

- Slovo je usporiadaná n–tica a budeme ho chápať ako algebraický vektor s n súradnicami.
- Súčtom dvoch slov $\boldsymbol{u}, \boldsymbol{v} \in T^n$ rozumieme slovo $\boldsymbol{u} + \boldsymbol{v} = (u_1 u_2 \ldots\ u_n) + (v_1 v_2 \ldots\ v_n) = ((u_1 + v_1)(u_2 + v_2) \ldots\ (u_n + v_n))$, kde jednotlivé súradnice sčítavame modulárne podľa modulu m.
- Skalárnym násobkom (skrátene len násobkom) slova $\boldsymbol{u} \in T^n$ číslom $a \in T$ rozumieme slovo $a \cdot \boldsymbol{u} = a \cdot (u_1 u_2 \ldots\ u_n) = ((a \cdot u_1)(a \cdot u_2) \ldots\ (a \cdot u_n))$, kde jednotlivé súradnice násobíme modulárne podľa modulu m.
- Lineárnou kombináciou slov $\boldsymbol{u}, \boldsymbol{v} \in T^n$ nazývame n–ticu $\boldsymbol{w}$, že existujú čísla $a, b \in T$ také, že $\boldsymbol{w} = a \cdot \boldsymbol{u} + b \cdot \boldsymbol{v}$.
- Skalárnym súčinom dvoch slov $\boldsymbol{u}, \boldsymbol{v} \in T^n$ rozumieme slovo $\boldsymbol{u} \cdot \boldsymbol{v} = (u_1 u_2 \ldots\ u_n) \cdot (v_1 v_2 \ldots\ v_n) = (u_1 \cdot v_1 + u_2 \cdot v_2 + \ldots\ + u_n \cdot v_n)$, kde jednotlivé súradnice sčítavame, resp. násobíme modulárne podľa modulu m.

Kódová abeceda $T = \{0, 1, 2, \ldots, m-1\}$, kde m je prvočíslo spolu s operáciami modulárne sčítanie a modulárne násobenie tvorí pole (čitateľ si to môže overiť požijúc Kapitolu 1.1.2).
Dá sa tiež dokázať, že množina T^n spolu s vyššie definovanými operáciami "súčet dvoch slov" a "násobenie slova číslom" je lineárnym priestorom nad poľom $T = \{0, 1, 2, \ldots, m-1\}$, ak m je prvočíslo.

Najbežnejšie používanými kódmi sú binárne kódy a keďže 2 je prvočíslo, tak $(\{0, 1\}, +, \cdot)$ je pole. Množina všetkých binárnych n znakových slov $\{0, 1\}^n$ s vyššie definovaným súčtom slov a násobením číslom je lineárnym priestorom nad poľom $(\{0, 1\}, +, \cdot)$. Platia na nej teda všetky tvrdenia a vzťahy, ktoré platia pre lineárne priestory.

Odteraz, ak budeme pracovať nad nejakou kódovou abecedou, vždy budeme používať modulárne sčítanie a násobenie, aj keď ich budeme označovať $+, \cdot$. Ak budeme uvažovať o slovách, resp. kódových slovách, budeme používať operácie súčet slov a násobenie číslom a budeme ich označovať tiež $+, \cdot$.

3.1 Základné vlastnosti lineárnych kódov

Definícia 3.1.1 *Nech je daná kódová abeceda $T = \{0, 1, 2, \ldots, m-1\}$, kde m je prvočíslo. Blokový kód $\mathcal{K} \subseteq T^n$ dĺžky n sa nazýva lineárny kód, ak pre ľubovoľné kódové slová $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{K}$ a pre ľubovoľné $a, b \in T$ je lineárna kombinácia $a \cdot \boldsymbol{u} + b \cdot \boldsymbol{v}$ tiež kódovým slovom, teda aj $(a \cdot \boldsymbol{u} + b \cdot \boldsymbol{v}) \in \mathcal{K}$.*

Poznámka: Takto definovaný lineárny kód tvorí tiež lineárny priestor, presnejšie podpriestor lineárneho priestoru T^n a platia pre neho všetky vlastnosti lineárnych priestorov.

Triviálnym nazývame:

- kód obsahujúci len kódové slovo s nulovými zložkami – $\mathcal{K} = \{000\ldots0\}$,
- kód obsahujúci všetky možné n–tice – $\mathcal{K} = T^n$.

Triviálne kódy sú prakticky nepoužiteľné.

Veta 3.1.1 *Každé kódové slovo lineárneho kódu $\mathcal{K}$ je možné vytvoriť ako lineárnu kombináciu iných kódových slov.*

Definícia 3.1.2 *Hammingovou váhou slova $\boldsymbol{u}$ nazývame počet nenulových znakov v slove. Označujeme ju $||\boldsymbol{u}||$.*

Z vlastností lineárnych priestorov vyplýva, že v množine všetkých kódových slov lineárneho kódu vieme nájsť takú minimálnu množinu kódových slov, že hocijaké kódové slovo toho kódu vieme zostrojiť ich lineárnou kombináciou.

Definícia 3.1.3 *Nech je daná kódová abeceda $T = \{0, 1, 2, \ldots, m-1\}$, kde m je prvočíslo a lineárny kód $\mathcal{K} \subset T^n$. Minimálnu podmnožinu $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ kódových slov daného kódu takú, že všetky kódové slová vieme zostrojiť ako ich lineárnu kombináciu, nazývame bázou lineárneho kódu $\mathcal{K}$.*

Hovoríme tiež, že báza generuje (vytvára) kód. Kód môže generovať aj iná, nie minimálna množina, no z nej sa potom vždy dá vybrať báza.

Príklad 3.1.1 *Nájdite nejakú bázu binárneho koktavého kódu dĺžky* 6.

Riešenie: Množinu kódových slov tvorí osem slov $\{000000, 110000, 001100, 001111, 111100, 110011, 000011, 111111\}$. Dá sa zostrojiť niekoľko báz, jednou z nich je napríklad táto $\{110000, 001100, 000011\}$. Hocijaké iné kódové slovo vieme zostrojiť ako lineárnu kombináciu (v tomto prípade ako súčet) bázových kódových slov.

Dôsledok 3.1.1 *Slovo tvorené len nulovými zložkami* $(00\ldots0)$ *musí byť kódovým slovom každého lineárneho kódu.*

Dôsledok 3.1.2 *V lineárnom kóde je minimálna vzdialenosť* d *rovná minimálnej váhe nenulového kódového slova, teda*

$$d = min_{v\in\mathcal{K}}||\boldsymbol{v}||.$$

Definícia 3.1.4 *Majme lineárny kód* $\mathcal{K}$*. Počet kódových slov v ľubovoľnej báze toho kódu nazývame dimenziou kódu a označujeme* k*.*

Veta 3.1.2 *Počet kódových slov v báze* B *lineárneho kódu* $\mathcal{K}$ *je rovný počtu informačných znakov v jednotlivých kódových slovách.*

Dôkaz: Zvolíme si ľubovoľnú bázu $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ lineárneho kódu dĺžky n, čo znamená, že dimenzia je rovná k.
Každé kódové slovo je možné zapísať v tvare $\boldsymbol{w} = u_1\boldsymbol{b}_1 + u_2\boldsymbol{b}_2 + \cdots + u_k\boldsymbol{b}_k$ pre práve jednu k–ticu $u_1u_2\ldots u_k$.
Dostali sme tak zobrazenie $\phi : T^k \to \mathcal{K}$ (je to kódovanie informačných znakov) dané predpisom $\phi(u_1u_2\ldots u_k) = u_1\boldsymbol{b}_1 + u_2\boldsymbol{b}_2 + \cdots + u_k\boldsymbol{b}_k$, čo podľa definície 2.2.1 znamená, že daný kód má počet informačných znakov rovný k, rovnako ako dimenziu. □

Definícia 3.1.5 *Nech je daný lineárny kód* $\mathcal{K}$*. Ak je dĺžka kódových slov* n *a dimenzia kódu je rovná* k*, tak kód* $\mathcal{K}$ *nazývame lineárny* (n, k)*–kód.*

3.2 Kontrolná a generujúca matica

3.2.1 Kontrolná matica

Z lineárnej algebry a vlastností lineárnych priestorov vieme, že všetky riešenia homogennej sústavy rovníc tvoria lineárny priestor. Kvôli dekódovaniu a samoopravným schopnostiam kódov sa javí ako výhodné vyjadriť kódy ako množiny riešení sústavy lineárnych rovníc. Pokúsime sa o to pri niektorých známych kódoch:

Binárny kód celkovej kontroly parity dĺžky 5:
Keďže tento kód je tvorený všetkými 5 znakovými binárnymi slovami, v ktorých je párny počet jednotiek, je zrejme, že súčet všetkých kódových znakov v ľubovoľnom kódovom slove je 0. Naopak, súčet všetkých kódových znakov v ľubovoľnom nekódovom slove je 1. Pre každé kódové slovo $v_1v_2v_3v_4v_5$ platí,

$$v_1 + v_2 + v_3 + v_4 + v_5 = 0.$$

Vo všeobecnosti v binárnom kóde celkovej kontroly parity dĺžky n pre každé kódové slovo $v_1v_2 \dots v_n$ platí,

$$v_1 + v_2 + \cdots + v_n = 0.$$

Binárny opakovací kód dĺžky n:
Tento kód je tvorený všetkými n znakovými binárnymi slovami, v ktorých sú všetky znaky rovnaké – jednotky alebo nuly. Každé kódové slovo teda musí vyhovovať sústave rovníc:

$$\begin{aligned} v_1 + v_2 &= 0 \\ v_1 + v_3 &= 0 \\ &\vdots \\ v_1 + v_n &= 0 \end{aligned}$$

Túto sústavu možno zapísať v tvare matice:

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & \dots & 0 & 0 \\ 1 & 0 & 1 & \dots & 0 & 0 \\ & \vdots & & & & \vdots \\ 1 & 0 & 0 & \dots & 1 & 0 \end{array}\right)$$

Respektíve:

$$\left(\begin{array}{ccccc|c} 1 & 1 & 0 & \dots & 0 & 0 \\ 1 & 0 & 1 & \dots & 0 & 0 \\ & \vdots & & & & \vdots \\ 1 & 0 & 0 & \dots & 1 & 0 \end{array}\right) \rightarrow \begin{pmatrix} 1 & 1 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 \\ & \vdots & & & \\ 1 & 0 & 0 & \dots & 1 \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Teda symbolicky zapíšeme:

$$H \cdot \boldsymbol{v}^T = \mathbf{0}^T,$$

kde H je matica danej sústavy rovníc (nie rozšírená matica), $\boldsymbol{v}^T$ je kódové slovo zapísané do stĺpca a $\mathbf{0}^T$ je nulový stĺpcový vektor.

Definícia 3.2.1 *Nech je daný lineárny kód $\mathcal{K}$ dĺžky n popísaný sústavou homogénnych lineárnych rovníc. Potom matica H, zostrojená tak, že riadky tvoria koeficienty rovníc daného kódu, sa nazýva kontrolnou maticou lineárneho kódu $\mathcal{K}$.*

Teda ľubovoľné slovo $\boldsymbol{v} = v_1 v_2 \dots v_n$ dĺžky n je kódovým slovom práve vtedy, ak spĺňa rovnicu $H \cdot \boldsymbol{v}^T = \mathbf{0}^T$.

Príklad 3.2.1 *Je daný binárny koktavý kód dĺžky* 6. *Popíšte ho rovnicami a zapíšte jeho kontrolnú maticu.*

Riešenie: Ide o kód, kde je vždy každý kódový znak vysielaný dvakrát po sebe, teda vždy dva po sebe idúce znaky sú rovnaké a potom ich súčet je 0. Takže sústava toho kódu je:

$$\begin{aligned} v_1 + v_2 &= 0 \\ v_3 + v_4 &= 0 \\ v_5 + v_6 &= 0 \end{aligned}$$

a kontrolná matica je

$$H = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Príklad 3.2.2 *Je daný ternárny kód pomocou kontrolnej matice*

$$H = \begin{pmatrix} 0 & 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 0 & 1 \end{pmatrix}.$$

Zistite, či slovo $\boldsymbol{v}_1 = 12201$ *a slovo* $\boldsymbol{v}_2 = 01022$ *sú kódovým slovom daného kódu.*

Riešenie: Bez toho, aby sme museli vypísať množinu všetkých kódových slov a potom dané slová s ňou porovnať, vieme jednoducho určiť, či sú alebo nie sú kódové. Dosadením do riadkov matice dostávame:

- pre $\boldsymbol{v}_1$:

$$\begin{aligned} 0 \cdot 1 + 1 \cdot 2 + 2 \cdot 2 + 1 \cdot 0 + 0 \cdot 1 &= 0 \\ 0 \cdot 1 + 1 \cdot 2 + 2 \cdot 2 + 0 \cdot 0 + 1 \cdot 1 &= 1 \end{aligned}$$

- pre $\boldsymbol{v}_2$:

$$\begin{aligned} 0 \cdot 0 + 1 \cdot 1 + 2 \cdot 0 + 1 \cdot 2 + 0 \cdot 2 &= 0 \\ 0 \cdot 0 + 1 \cdot 1 + 2 \cdot 0 + 0 \cdot 2 + 1 \cdot 2 &= 0 \end{aligned}$$

Z toho vyplýva, že $\boldsymbol{v}_1$ nie je kódovým slovom, zatiaľ čo $\boldsymbol{v}_2$ je kódovým slovom.

3.2.2 Generujúca matica

V definícii 2.2.1 je zadefinované kódovanie informačných znakov. Teraz sa na chvíľu zastavíme pri tom, čo znamená kódovanie informačných znakov lineárnym kódom.

Uvažujme binárny lineárny (n, k)–kód a majme nejakú jeho bázu $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$. Hocijaké kódové slovo $\boldsymbol{v}$ sa dá zapísať ako lineárna kombinácia bázových slov: $\boldsymbol{v} = a_1\boldsymbol{b}_1 + a_2\boldsymbol{b}_2 + \cdots + a_k\boldsymbol{b}_k$, kde $a_1, a_2, \ldots, a_k \in \{0, 1\}$. Táto rovnica lineárnej kombinácie sa dá prepísať v maticovom tvare:

$$\begin{pmatrix} a_1 & a_2 & \ldots & a_k \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_k \end{pmatrix} = \begin{pmatrix} v_1 & v_2 & \ldots & v_n \end{pmatrix} = \boldsymbol{v}.$$

Usporiadanú k–ticu $(a_1 a_2 \ldots a_k)$ možno považovať za informačné slovo prislúchajúce kódovému slovu $\boldsymbol{v}$.

Definícia 3.2.2 *Matica zostrojená z bázy lineárneho kódu tak, že prvky bázy tvoria riadky matice, sa nazýva generujúca matica kódu a označujeme ju G.*

Poznámka: Pre jeden kód môže existovať viac báz, a teda aj viac generujúcich matíc. Rovnica $\boldsymbol{a} \cdot G = \boldsymbol{v}$ popisuje samotný algoritmus kódovania informačných znakov, kde:

- $\boldsymbol{a} = (a_1 a_2 \ldots a_k)$ je vstup (informačné slovo),
- G je generujúca matica kódu,
- $\boldsymbol{v} = (v_1 v_2 \ldots v_n)$ je výstup (kódové slovo).

Kód je teda jednoznačne daný svojou generujúcou maticou.

Veta 3.2.1 *Vzájomná výmena riadkov generujúcej matice kódu, pripočítanie riadku matice k inému riadku a výmena stĺpcov matice nemá vplyv na zabezpečovaciu schopnosť kódu.*

Takýmito úpravami vieme získavať ďalšie kódy ekvivalentné s pôvodným kódom. Ekvivalentný v tomto zmysle znamená s rovnakou dimenziou a s takou istou zabezpečovacou schopnosťou.

Veta 3.2.2 *Lineárny (n, k)–kód je systematický práve vtedy, ak má generujúcu maticu v tvare $G = (E|P)$, kde E je jednotková matica rozmeru $k \times k$ a P je ľubovoľná matica rozmeru $k \times (n - k)$.*

Veta 3.2.3 *Každý lineárny kód je ekvivalentný s nejakým systematickým kódom.*

Príklad 3.2.3 *Je daný binárny koktavý kód pomocou generujúcej matice. Nájdite k nemu ekvivalentný systematický kód.*

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Riešenie: V generujúcej matici vymeníme druhý stĺpec so štvrtým a tretí s piatym a dostávame maticu v tvare $G = (E|P)$, čo je matica ekvivalentného systematického kódu.

$$G_S = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

3.2.3 Vzťah medzi kontrolnou a generujúcou maticou

Nech je daný lineárny (n, k)–kód.
Vieme, že pre kontrolnú maticu platí: $H \cdot \boldsymbol{v}^T = \mathbf{0}^T \Rightarrow \boldsymbol{v} \cdot H^T = \mathbf{0}$.
Tiež vieme, že pre generujúcu maticu platí: $\boldsymbol{a} \cdot G = \boldsymbol{v}$.

$$\begin{aligned} \boldsymbol{a} \cdot G &= \boldsymbol{v} \quad / \cdot H^T \\ \boldsymbol{a} \cdot G \cdot H^T &= \boldsymbol{v} \cdot H^T \\ \boldsymbol{a} \cdot G \cdot H^T &= \mathbf{0} \end{aligned}$$

Keďže to má platiť pre všetky kódové slová, tak $\boldsymbol{a} \neq \mathbf{0}$ a teda $G \cdot H^T = \mathbf{0}$. Rozpíšeme to:

$$G \cdot H^T = \begin{pmatrix} \boldsymbol{g}_1 \\ \boldsymbol{g}_2 \\ \vdots \\ \boldsymbol{g}_k \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{h}_1 \boldsymbol{h}_2 \ldots \boldsymbol{h}_{n-k} \end{pmatrix} = \begin{pmatrix} \boldsymbol{g}_1 \cdot \boldsymbol{h}_1 & \boldsymbol{g}_1 \cdot \boldsymbol{h}_2 & \ldots & \boldsymbol{g}_1 \cdot \boldsymbol{h}_{n-k} \\ \boldsymbol{g}_2 \cdot \boldsymbol{h}_1 & \boldsymbol{g}_2 \cdot \boldsymbol{h}_2 & \ldots & \boldsymbol{g}_2 \cdot \boldsymbol{h}_{n-k} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{g}_k \cdot \boldsymbol{h}_1 & \boldsymbol{g}_k \cdot \boldsymbol{h}_2 & \ldots & \boldsymbol{g}_k \cdot \boldsymbol{h}_{n-k} \end{pmatrix} = \mathbf{0},$$

a odtiaľ dostávame, že pre skalárne súčiny platí:

$$\boldsymbol{g}_i \cdot \boldsymbol{h}_j = g_{i1} \cdot h_{j1} + g_{i2} \cdot h_{j2} + \cdots + g_{in} \cdot h_{jn} = 0, \tag{3.1}$$

pre všetky $i \in \{1, 2, \ldots, k\}$ a $j \in \{1, 2, \ldots, (n-k)\}$.

Príklad 3.2.4 *Je daný ternárny lineárny kód pomocou generujúcej matice. Nájdite jeho kontrolnú maticu.*

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Riešenie: Rozmer generujúcej matice je 3×5, teda je to $(5,3)$–kód a kontrolná matica bude mať rozmer 2×5. Je to ternárny kód, z toho vyplýva, že všetky operácie vykonávame podľa modulu 3. Z rovnice 3.1 dostávame, že ľubovoľný riadok $\boldsymbol{h}_j = h_{j1}h_{j2}h_{j3}h_{j4}h_{j5}$ kontrolnej matice vyhovuje rovniciam:

$$\begin{aligned} h_{j1} + h_{j2} + h_{j3} + h_{j4} &= 0 \\ h_{j2} + h_{j4} + h_{j5} &= 0 \\ h_{j1} + h_{j2} &= 0 \quad / + 2R_1 \end{aligned}$$

Danú sústavu upravíme na stupňovitý (lichobežníkový) tvar.

$$\begin{aligned} h_{j1} + h_{j2} + h_{j3} + h_{j4} &= 0 \\ h_{j2} + h_{j4} + h_{j5} &= 0 \\ 2h_{j3} + 2h_{j4} &= 0 \end{aligned}$$

Dostávame sústavu troch rovníc s piatimi neznámymi, a teda riešenie bude obsahovať dva voliteľné parametre. Keďže vieme, že kontrolná matica má mať dva riadky $\boldsymbol{h}_1, \boldsymbol{h}_2$, tak parametre h_{j4} a h_{j5} volíme dvakrát.

1. Zvolíme, nech $h_{14} = 1$ a $h_{15} = 0$. Postupným dosadením do všetkých troch riadkov sústavy dostávame, že $h_{11} = 2, h_{12} = 2$ a $h_{13} = 1$. Dostávame prvý riadok kontrolnej matice (22110).

2. Zvolíme, nech $h_{24} = 0$ a $h_{25} = 1$. Postupným dosadením do všetkých troch riadkov sústavy dostávame, že $h_{21} = 1, h_{22} = 2$ a $h_{23} = 0$. Takto máme aj druhý riadok kontrolnej matice (12001).

Kontrolná matica daného kódu je:

$$H = \begin{pmatrix} 2 & 2 & 1 & 1 & 0 \\ 1 & 2 & 0 & 0 & 1 \end{pmatrix}.$$

Pre systematické kódy sa dá odvodiť jednoduchší spôsob určenia kontrolnej matice, pretože majú generujúcu maticu v tvare $G = (E|P)$.

Veta 3.2.4 *Lineárny kód s generujúcou maticou v tvare $G = (E|P)$ má kontrolnú maticu $H = (-P^T|E')$, kde E a E' sú jednotkové matice príslušného rozmeru a P^T je transponovanou maticou k matici P.*

Príklad 3.2.5 *Nasledujúcou generujúcou maticou je daný systematický binárny lineárny kód. Nájdite jeho kontrolnú maticu.*

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Riešenie: Maticu rozdelíme na časti E a P.

$$G = \left(\begin{array}{ccc|cc} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{array}\right).$$

Kontrolná matica bude mať rozmer 2×5. Príslušná E' bude rozmeru 2×2 a $-P^T$ bude rozmeru 2×3.

$$H = \left(\begin{array}{ccc|cc} 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{array}\right).$$

Príklad 3.2.6 *Nasledujúcou generujúcou maticou je daný binárny lineárny kód. Nájdite jeho kontrolnú maticu.*

$$G = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Riešenie: Keďže matica G nie je v tvare $G = (E|P)$, nie je na prvý pohľad zrejmé, či ide o systematický kód. Kontrolnú maticu by sme teda mali hľadať ako v príklade 3.2.4. Vieme však, že generujúca matica je tvorená prvkami bázy. Pre jeden kód však môžu existovať viaceré bázy. Teda jeden a ten istý kód môže byť určený viacerými generujúcimi maticami. Ďalšie generujúce matice dostaneme z už známej matice pomocou ekvivalentných riadkových maticových úprav.
Takto danú maticu vieme riadkovými úpravami previesť na tvar $G = (E|P)$.

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Môžeme vidieť, že daný kód je predsa len systematický a z takto získanej generujúcej matice ľahko vytvoríme kontrolnú maticu:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Definícia 3.2.3 *Nech je daný lineárny (n, k)–kód $\mathcal{K} \subseteq T^n$. Kód $\mathcal{K}^\perp \subseteq T^n$ nazývame duálnym kódom ku kódu $\mathcal{K}$, ak obsahuje všetky $\boldsymbol{v} \in T^n$ také, že $\forall \boldsymbol{u} \in \mathcal{K}$ a $\forall \boldsymbol{v} \in \mathcal{K}^\perp$ platí $\boldsymbol{u} \cdot \boldsymbol{v} = 0$.*

Označme generujúcu a kontrolnú maticu kódu $\mathcal{K}$ symbolmi $G_\mathcal{K}$, $H_\mathcal{K}$ a generujúcu a kontrolnú maticu kódu $\mathcal{K}^\perp$ symbolmi $G_{\mathcal{K}^\perp}$, $H_{\mathcal{K}^\perp}$. Ak vynásobíme $G_{\mathcal{K}^\perp} \cdot G_\mathcal{K}^T$, dostávame maticu, ktorej prvkami budú skalárne súčiny kódových slov z kódu $\mathcal{K}$ a kódových slov z kódu $\mathcal{K}^\perp$. Keďže z definície 3.2.3 pre $\forall \boldsymbol{u} \in \mathcal{K}$ a pre $\forall \boldsymbol{v} \in \mathcal{K}^\perp$ platí $\boldsymbol{u} \cdot \boldsymbol{v} = 0$, tak dostávame nulovú maticu. Z toho je však zrejmé, že generujúca matica kódu $\mathcal{K}$ je kontrolnou maticou kódu $\mathcal{K}^\perp$. Podobne môžeme ukázať, že naopak kontrolná matica kódu $\mathcal{K}$ je generujúcou maticou kódu $\mathcal{K}^\perp$. Tieto tvrdenia zhrnieme v nasledujúcej vete.

Veta 3.2.5 *Nech je daný lineárny (n,k)–kód $\mathcal{K} \subseteq T^n$. K nemu duálny kód $\mathcal{K}^\perp \subseteq T^n$ je $(n, n-k)$–kódom a platí:*

$$G_{\mathcal{K}} = H_{\mathcal{K}^\perp} \wedge H_{\mathcal{K}} = G_{\mathcal{K}^\perp}.$$

Príklad 3.2.7 *Je daný binárny kód pomocou generujúcej matice. Nájdite k nemu duálny kód (jeho generujúcu maticu).*

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Riešenie: Keďže kontrolná matica H daného kódu je zároveň generujúcou maticou $G^\perp$ duálneho kódu, tak stačí nájsť ju. Keďže daná generujúca matica nie je v tvare $G = (E|P)$, skúsime, či ju vieme na taký tvar upraviť riadkovými úpravami, teda či je to systematický kód.

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Je to teda systematický kód s generujúcou maticou

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

a jeho kontrolnú maticu zapíšeme ľahko v tvare $H = (-P^T|E')$, takže

$$H = G^\perp = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

V predchádzajúcom príklade si môžeme všimnúť, že generujúca matica kódu je taká istá ako kontrolná, teda taká istá, ako generujúca matica duálneho kódu.

Definícia 3.2.4 *Lineárny kód $\mathcal{K}$ sa nazýva samoduálny, ak $\mathcal{K}^\perp = \mathcal{K}$.*

Jeho generujúca matica je zároveň kontrolnou maticou.

3.3 Dekódovanie lineárnych kódov

K dekódovaniu lineárnych kódov sa používajú dva hlavné spôsoby:

- štandardné dekódovanie,
- urýchlené dekódovanie pomocou syndrómov.

Definícia 3.3.1 *Nech je informačným kanálom vyslané kódové slovo* $\boldsymbol{v} = v_1v_2 \dots v_n \in \mathcal{K}$ *a prijaté slovo* $\boldsymbol{w} = w_1w_2 \dots w_n \in T^n$. *Chybovým slovom nazývame slovo* $\boldsymbol{e} = e_1e_2 \dots e_n \in T^n$ *také, že* $\boldsymbol{e} = \boldsymbol{w} - \boldsymbol{v}$, *resp.* $\boldsymbol{e} + \boldsymbol{v} = \boldsymbol{w}$

Je zjavné, že v chybovom slove sa nenulový znak objaví na tých miestach, kde nastala chyba pri prenose. Ak chyba nenastala, chybovým slovom bude nulové slovo $(00 \dots 0)$.

Ak by chybové slovo $\boldsymbol{e}$ bolo kódovým slovom, tak $\boldsymbol{e} + \boldsymbol{v}$ by tiež bolo kódovým slovom a lineárny kód by takúto chybu neobjavil. Lineárny kód objavuje len tie chyby, pri ktorých chybové slovo nie je kódovým slovom.

Označenie: Nech máme slovo $\boldsymbol{a} \in T^n$ a množinu $U \subset T^n$. Symbolom $\boldsymbol{a} + U$ budeme označovať množinu $\{\boldsymbol{a} + \boldsymbol{u} | \boldsymbol{u} \in U\}$.
Teda množina $\{\boldsymbol{a} + \boldsymbol{u} | \boldsymbol{u} \in U\}$ je taká množina, ktorú dostaneme z množiny U pripočítaním slova $\boldsymbol{a}$ ku každému prvku množiny U.

Definícia 3.3.2 *Nech je daný lineárny kód* $\mathcal{K} \subseteq T^n$ *a slovo* $\boldsymbol{w} \in T^n$. *Triedou slova* $\boldsymbol{w}$ *podľa kódu* $\mathcal{K}$ *nazývame množinu* $\boldsymbol{w} + \mathcal{K}$.

Príklad 3.3.1 *Je daný binárny lineárny* $(5, 3)$*– kód pomocou nasledujúcej generujúcej matice. Nájdite triedy slov:* $\boldsymbol{w}_1 = 10001$, $\boldsymbol{w}_2 = 11001$, $\boldsymbol{w}_3 = 01101$, $\boldsymbol{w}_4 = 01000$.

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Riešenie: Keďže ide o binárny $(5, 3)$– kód, počet kódových slov je $2^3 = 8$ a všetky kódové slová dostaneme ako všetky rôzne lineárne kombinácie bázových slov: $\mathcal{K} = \{00000, 10000, 01001, 00111, 11001, 10111, 01110, 11110\}$. Triedami sú:

- pre $\boldsymbol{w}_1 = 10001$: $\boldsymbol{w}_1 + \mathcal{K} = \{10001, 00001, 11000, 10110,\ 01000, 00110, 11111, 01111\}$,
- pre $\boldsymbol{w}_2 = 11001$: $\boldsymbol{w}_2 + \mathcal{K} = \{11001, 01001, 10000, 11110,\ 00000, 01110, 10111, 00111\}$,
- pre $\boldsymbol{w}_3 = 01101$: $\boldsymbol{w}_3 + \mathcal{K} = \{01101, 11101, 00100, 01010,\ 10100, 11010, 00011, 10011\}$,
- pre $\boldsymbol{w}_4 = 01000$: $\boldsymbol{w}_4 + \mathcal{K} = \{01000, 11000, 00001, 01111,\ 10001, 11111, 00110, 10110\}$.

Všimnime si, že $\boldsymbol{w}_2+\mathcal{K}=\mathcal{K}$ a $\boldsymbol{w}_2\in\mathcal{K}$. Ďalej $\boldsymbol{w}_1+\mathcal{K}=\boldsymbol{w}_4+\mathcal{K}$. Ak urobíme rozdiel $\boldsymbol{w}_1-\boldsymbol{w}_4$, dostaneme kódové slovo. Ak urobíme rozdiel ľubovoľného prvku z $\boldsymbol{w}_1+\mathcal{K}$ a ľubovoľného prvku z $\boldsymbol{w}_4+\mathcal{K}$, tiež vždy dostaneme kódové slovo. Naopak, ak urobíme rozdiel ľubovoľného prvku z $\boldsymbol{w}_1+\mathcal{K}$ a ľubovoľného prvku napríklad z $\boldsymbol{w}_3+\mathcal{K}$, nikdy nedostaneme kódové slovo.

Pozorovaním dôjdeme k niekoľkým vlastnostiam tried slov podľa daného kódu, ktoré uvádzame bez dôkazov:

- Trieda kódového slova tvorí samotný kód.
- Trieda nekódového slova neobsahuje žiadne kódové slovo.
- Môžu existovať zhodné triedy rôznych nekódových slov.
- Počet prvkov každej triedy podľa daného kódu je zhodný s počtom kódových slov kódu.
- Dve rôzne triedy sú navzájom disjunktné, teda nemôžu existovať dve rôzne triedy, ktoré by mali spoločné len niektoré prvky.

Veta 3.3.1 *Pre q–árny lineárny (n,k)–kód je počet tried podľa daného kódu rovný q^{n-k}.*

Definícia 3.3.3 *Reprezentantom triedy $\boldsymbol{w}+\mathcal{K}$ nazývame také slovo danej triedy, ktoré má najmenšiu Hammingovu váhu. Ak je slov s najmenšou Hammingovou váhou v triede viac, vyberieme ľubovoľné z nich. Budeme ho označovať $\boldsymbol{r}_w$.*

Dôsledok 3.3.1 *Reprezentantom kódu je nulové slovo.*

Príklad 3.3.2 *Nájdite reprezentantov tried $\boldsymbol{w}_1+\mathcal{K}$, $\boldsymbol{w}_2+\mathcal{K}$ a $\boldsymbol{w}_3+\mathcal{K}$ z príkladu 3.3.1.*

Riešenie:

- Reprezentantom $\boldsymbol{w}_1+\mathcal{K}$ je 00001. (No môže byť aj 01000.)
- Reprezentantom $\boldsymbol{w}_2+\mathcal{K}$ je 00000, lebo je to kód.
- Reprezentantom $\boldsymbol{w}_3+\mathcal{K}$ je 00100.

Štandardné dekódovanie – algoritmus:

Nech je vyslané slovo $\boldsymbol{v}\in\mathcal{K}$.

1. Prijme sa slovo $\boldsymbol{w}\in T^n$. Ak $\boldsymbol{w}\in\mathcal{K}$, tak je prehlásené za vyslané slovo a $\boldsymbol{v}=\boldsymbol{w}$.
2. Ak $\boldsymbol{w}\notin\mathcal{K}$, tak vyhľadáme triedu, do ktorej patrí a nájdeme jej reprezentanta $\boldsymbol{r}_w$.
3. Odpočítame $\boldsymbol{w}-\boldsymbol{r}_w$, získame tak kódové slovo, ktoré prehlásime za vyslané slovo.

Dekódovanie δ môžeme zjednodušene zapísať rovnicou:

$$\boldsymbol{v} = \delta(\boldsymbol{w}) = \boldsymbol{w} - \boldsymbol{r}_w.$$

Tabuľka Slepianovho štandardného rozmiestnenia:

Postup dekódovania sa dá prehľadne znázorniť tabuľkou Slepianovho štandardného rozmiestnenia. Najprv popíšeme niektoré jej parametre a zákonitosti:

Nech je daný q–árny ($|T| = q$) lineárny (n, k)–kód.

- Existuje tu q^{n-k} rôznych tried, teda tabuľka bude mať q^{n-k} riadkov, v prvom riadku bude samotný kód.
- Keďže v každom ďalšom riadku sú uvedené slová príslušných tried, tabuľka bude mať q^k stĺpcov.
- Poradie v riadkoch je volené tak, že prvý v riadku bude vždy reprezentant a za ním ďalšie slová budú umiestnené podľa poradia slov v prvom riadku, ku ktorým sa pripočítaval reprezentant.
- Dekódovanie prebieha tak, že prijaté slovo vyhľadáme v tabuľke a priradíme mu ako vyslané to slovo, ktoré stojí v tom istom stĺpci, ale v prvom riadku.

Výsledná tabuľka Slepianovho štandardného rozmiestnenia pre q–árny lineárny (n, k)–kód vyzerá nasledovne:

	reprezentant				
kód $\mathcal{K}$	$\boldsymbol{r}_0 = 00\ldots0$	$\boldsymbol{v}_1$	$\boldsymbol{v}_2$	$\ldots$	$\boldsymbol{v}_{q^k}$
trieda K_1	$\boldsymbol{r}_1$	$\boldsymbol{r}_1 + \boldsymbol{v}_1$	$\boldsymbol{r}_1 + \boldsymbol{v}_2$	$\ldots$	$\boldsymbol{r}_1 + \boldsymbol{v}_{q^k}$
trieda K_2	$\boldsymbol{r}_2$	$\boldsymbol{r}_2 + \boldsymbol{v}_1$	$\boldsymbol{r}_2 + \boldsymbol{v}_2$	$\ldots$	$\boldsymbol{r}_2 + \boldsymbol{v}_{q^k}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
trieda $K_{q^{n-k}}$	$\boldsymbol{r}_{q^{n-k}}$	$\boldsymbol{r}_{q^{n-k}} + \boldsymbol{v}_1$	$\boldsymbol{r}_{q^{n-k}} + \boldsymbol{v}_2$	$\ldots$	$\boldsymbol{r}_{q^{n-k}} + \boldsymbol{v}_{q^k}$

Veta 3.3.2 *Štandardné dekódovanie (kód) spoľahlivo opravuje práve tie chybové slová, ktoré sú reprezentantmi tried.*

Veta 3.3.3 *Každé štandardné dekódovanie δ je optimálne v tom zmysle, že žiadne iné dekódovanie neopravuje väčšiu množinu chybových slov.*

Pre bežne požívané kódy je použitie štandardného dekódovania dosť zložité. Napríklad pri binárnych kódoch dĺžky $n = 64$ by sa muselo prehľadávať 2^{64} slov dĺžky 64, čo je značne zdĺhavé. Toto dekódovanie sa značne urýchli použitím kontrolnej matice kódu a zavedením tzv. syndrómov.

Definícia 3.3.4 *Nech je daný q–árny lineárny (n, k)–kód $\mathcal{K}$ a jeho kontrolná matica H. Pre každé slovo $\boldsymbol{w} \in T^n$ definujeme syndróm slova ako slovo $\boldsymbol{s}_w = s_1 s_2 \ldots s_k \in T^k$ získané nasledujúcim predpisom $\boldsymbol{s}_w = \boldsymbol{w} \cdot H^T$.*

Je zrejmé, že pre všetky kódové slová je syndróm nulový. Princíp urýchleného dekódovania pomocou syndrómov je založený na nasledujúcej vete.

Veta 3.3.4 *Všetky slová patriace do rovnakej triedy majú rovnaký syndróm.*

Dôkaz: Inak povedané, všetky slová v danej triede majú taký istý syndróm ako reprezentant triedy, teda ako príslušné chybové slovo.

Nech vyšleme kódové slovo $\boldsymbol{v}$ a prijme sa slovo $\boldsymbol{w} = \boldsymbol{v} + \boldsymbol{r}_w$. Vieme, že $\boldsymbol{v} \cdot H^T = \boldsymbol{0}$. Z toho dostávame:

$$\boldsymbol{s}_w = \boldsymbol{w} \cdot H^T = (\boldsymbol{v} + \boldsymbol{r}_w) \cdot H^T = \boldsymbol{v} \cdot H^T + \boldsymbol{r}_w \cdot H^T = \boldsymbol{0} + \boldsymbol{r}_w \cdot H^T = \boldsymbol{r}_w \cdot H^T = \boldsymbol{s}_{r_w}.$$

□

Urýchlené dekódovanie pomocou syndrómov – algoritmus:

Najprv si nájdeme reprezentantov všetkých tried a vypočítame ich syndrómy.

trieda	K_1	K_2	...	$K_{q^{n-k}}$
reprezentant	$\boldsymbol{r}_1$	$\boldsymbol{r}_2$	...	$\boldsymbol{r}_{q^{n-k}}$
syndróm	$\boldsymbol{s}_1$	$\boldsymbol{s}_2$	...	$\boldsymbol{s}_{q^{n-k}}$

1. Prijme sa slovo $\boldsymbol{w} \in T^n$. Ak $\boldsymbol{w} \in \mathcal{K}$, tak je prehlásené za vyslané slovo a $\boldsymbol{v} = \boldsymbol{w}$.
2. Ak $\boldsymbol{w} \notin \mathcal{K}$, tak vypočítame syndróm $\boldsymbol{s}_w = \boldsymbol{w} \cdot H^T$.
3. Podľa syndrómu vyhľadáme v tabuľke príslušného reprezentanta $\boldsymbol{r}_w$.
4. Dekódujeme podľa predpisu $\boldsymbol{v} = \delta(\boldsymbol{w}) = \boldsymbol{w} - \boldsymbol{r}_w$.

Príklad 3.3.3 *Je daný binárny lineárny kód nasledujúcou generujúcou maticou. Pomocou syndrómov dekódujte slová* $\boldsymbol{w}_1 = 10111$ *a* $\boldsymbol{w}_2 = 00101$.

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Riešenie: Je zrejmé, že je to $(5, 3)$–kód a tak počet tried bude 4. Z generujúcej matice vypočítame kontrolnú maticu

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \Rightarrow H = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Vyberieme reprezentantov tried tak, že postupne vyberáme slová s čo najnižšou Hammingovou váhou a vypočítame ich syndrómy podľa vzorca $\boldsymbol{s}_w = \boldsymbol{w} \cdot H^T$. Ak nám vznikne syndróm, aký sme už predtým dostali, slovo nebude reprezentantom ďalšej triedy. Pokračujeme dovtedy, kým nedostaneme všetky možné syndrómy. V tomto príklade ich bude $2^{5-3} = 4$.

trieda	K_1	K_2	K_3	K_4
reprezentant	00000	00001	00010	00100
syndróm	00	01	10	11

Pre slová $\boldsymbol{w}_1$ a $\boldsymbol{w}_2$ vypočítame syndrómy:

$$\boldsymbol{s}_1 = \boldsymbol{w}_1 \cdot H^T = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \end{pmatrix}.$$

$$\boldsymbol{s}_2 = \boldsymbol{w}_2 \cdot H^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix}.$$

Syndróm slova $\boldsymbol{w}_1$ je nulový, a to znamená, že je to kódové slovo. Syndróm slova $\boldsymbol{w}_2$ je (1 0) a teda toto slovo nie je kódovým slovom a patrí do triedy K_3. Dekódujeme ho nasledovne: $\delta(\boldsymbol{w}_2) = \boldsymbol{w}_2 - \boldsymbol{r}_3 = 00101 - 00010 = 00111$. Toto slovo bolo vyslaným kódovým slovom.

3.4 Perfektné kódy

Doteraz sme sa zaoberali kódmi, ktoré boli lineárne a opravovali t–násobné chyby pre nejaké t. Nebrali sme do úvahy redundanciu daných kódov. Teraz sa zamerame na kódy opravujúce t–násobné chyby s najnižšou redundanciou – perfektné kódy.

Definícia 3.4.1 *Lineárny kód nazývame perfektným kódom pre t–násobné opravy, ak množina všetkých slov, ktoré majú Hammingovu váhu najviac t, tvorí systém reprezentantov jeho tried.*

Poznámka: Takýto kód je schopný opraviť každú t–násobnú chybu v prijatom slove a navyše má najmenšiu možnú redundanciu zo všetkých kódov schopných opravovať t–násobné chyby.

Veta 3.4.1 *Binárny lineárny kód opravuje jednoduché chyby práve vtedy, ak stĺpce jeho kontrolnej matice sú nenulové a navzájom rôzne.*

Veta 3.4.2 *Jedinými netriviálnymi linárnymi binárnymi perfektnými kódmi sú nasledujúce kódy:*

1. *Hammingov kód pre opravu jednoduchých chýb,*
2. *Golyaov kód pre opravu trojnásobných chýb,*

3. opakovací kód dĺžky $2t-1$ *pre* t*–násobné chyby, kde* $t \in \mathbb{N}$.

Samozrejme, myslia sa tým aj kódy s nimi ekvivalentné.

V literatúre sa dá nájsť podobná kategorizácia aj pre rôzne q–árne kódy. Opakovací kód sme už popisovali, Golayov kód je cyklickým kódom a budeme sa mu venovať pri cyklických kódoch. Teraz si zadefinujeme Hammingov kód a popíšeme jeho vlastnosti.

3.4.1 Hammingove binárne kódy

Definícia 3.4.2 *Binárny lineárny* (n,k)*–kód sa nazýva Hammingovým kódom, ak má kontrolnú maticu, ktorej stĺpce tvoria všetky nenulové binárne slová dĺžky* $n-k$ *a neopakujú sa.*

Príklad 3.4.1 *Zapíšte kontrolnú maticu Hammingovho* $(7,4)$*–kódu, určte počet tried, vypíšte reprezentantov a syndrómy pre každú triedu.*

Poznámka: Kontrolná matica má mať rozmer $(n-k)\times n$, teda 3×7 a všetky stĺpce majú byť nenulové a navzájom rôzne.

$$H=\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Všetkých tried bude $2^{7-4}=8$. Podľa vety 3.3.2 kód opravuje práve tie chybové slová, ktoré sú reprezentantmi tried, navyše tento kód je perfektný pre jednoduché opravy, takže reprezentatmi budú všetky binárne 7 znakové slová s Hammingovou váhou 1. Reprezententom samotného kódu je nulové slovo. Syndrómy príslušných tried vypočítame vynásobením reprezentanta triedy kontrolnou maticou podľa $\boldsymbol{s}_i=\boldsymbol{r}_i\cdot H^T$.

trieda	$\mathcal{K}=K_0$	K_1	K_2	K_3	K_4	K_5	K_6	K_7
reprezentant	0000000	1000000	0100000	0010000	0001000	0000100	0000010	00000001
syndróm	000	001	010	011	100	101	110	111

Cieľom ďalšej analýzy bude nájsť vzťah medzi počtom všetkých znakov a počtom informačných (resp. kontrolných) znakov v Hammingovom binárnom kóde. Inak povedané, chceme zistiť, pre aké n a k sa Hammingov kód dá zostrojiť.

Nech je daný binárny (n,k)–kód. Jeho kontrolná matica je rozmeru $(n-k)\times n$, teda má $(n-k)$ riadkov a n stĺpcov. Aby to bol Hammingov kód, stĺpcami musia byť všetky nenulové binárne slová dĺžky $n-k$ a žiadne z nich sa nemôže opakovať. Všetkých nenulových binárnych slov dĺžky $n-k$ je $2^{n-k}-1$, a teda toľko stĺpcov má kontrolná matica. Z toho vyplýva vzťah $n=2^{n-k}-1$. Ak navyše položíme $n-k=l$, tak dostávame $n=2^l-1$ a $k=2^l-l-1$. Uvedieme tabuľku niekoľkých prvých Hammingových binárnych kódov. Pre porovnanie uvádzame aj ich redundancie.

n	3	7	15	31	63	...
k	1	4	11	26	57	...
R	0,67	0,43	0,27	0,16	0,1	...

Poznámka: Každý lineárny $(2^l - 1, 2^l - l - 1)$–kód s minimálnou vzdialenosťou aspoň 3 je Hammingov.

Dekódovanie Hammingovym binárnym kódom

Hammingov kód je z hľadiska dekódovania veľmi efektívny, dekóduje veľmi rýchlo. Z definície 3.4.4 vieme, že stĺpce kontrolnej matice tvoria všetky nenulové binárne slová dĺžky $n-k$ a neopakujú sa. Usporiadame ich tak, aby tvorili binárny rozvoj čísel $1, 2, \ldots, 2^{n-k}-1$ v tomto poradí. Dekódujeme takto:

1. Je vyslané kódové slovo $\boldsymbol{v}$, prijme sa slovo $\boldsymbol{w}$.
2. Pomocou kontrolnej matice vypočítame jeho syndróm $\boldsymbol{s}_w$, ktorý je binárnym rozvojom niektorého z čísel $0, 1, 2, \ldots, 2^{n-k} - 1$, označme ho i.
3. V prijatom slove $\boldsymbol{w}$ zameníme i–tý znak a dostaneme vyslané slovo $\boldsymbol{v} = \delta(\boldsymbol{w})$.
4. Ak syndróm bol nulový, tak $\delta(\boldsymbol{w}) = \boldsymbol{w} = \boldsymbol{v}$

Veta 3.4.3 *Uvedené dekódovanie je správne v prípade jednoduchej chyby.*

Príklad 3.4.2 *Použitím Hammingovho* $(15, 11)$*–kódu dekódujte nasledujúce prijaté slová:* $\boldsymbol{w}_1 = 110011001100110$*,* $\boldsymbol{w}_2 = 101101000011010$*,* $\boldsymbol{w}_3 = 001001000000001$.

Riešenie: Zapíšeme kontrolnú maticu daného kódu tak, aby stĺpce postupne tvorili binárny rozvoj čísel $1, 2, \ldots, 15$.

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

- $\boldsymbol{s}_1 = \boldsymbol{w}_1 \cdot H^T = 0110 \sim 2^2 + 2^1 = 6 \Rightarrow \delta(\boldsymbol{w}_1) = 11001\underline{0}001100110$,
- $\boldsymbol{s}_2 = \boldsymbol{w}_2 \cdot H^T = 1001 \sim 2^2 + 2^1 = 9 \Rightarrow \delta(\boldsymbol{w}_2) = 10110100\underline{1}011010$,
- $\boldsymbol{s}_3 = \boldsymbol{w}_3 \cdot H^T = 1010 \sim 2^2 + 2^1 = 10 \Rightarrow \delta(\boldsymbol{w}_3) = 00100100\underline{1}00001$.

Veta 3.4.4 *Hammingove binárne kódy sú perfektné kódy pre opravy jednoduchých chýb. Navyše každý perfektný binárny kód pre jednoduché opravy je Hammingov.*

Definícia 3.4.3 *Rozšíreným Hammingovým kódom nazývame binárny kód, ktorý vznikne z Hammingovho kódu jeho rozšírením o znak celkovej kontroly parity.*

Ak je $v_1 v_2 \ldots v_n$ kódovým slovom Hammingovho kódu, tak slovo $v_1 v_2 \ldots v_n v_{n+1}$ je kódovým slovom rozšíreného Hammingovho kódu, ak $v_1 + v_2 + \cdots + v_n = v_{n+1}$

Veta 3.4.5 *Rozšírený Hammingov kód má minimálnu váhu* 4.

3.4.2 Hammingove q–árne kódy

Veta 3.4.6 *Lineárny q–árny kód opravuje jednoduché chyby práve vtedy, ak žiaden stĺpec kontrolnej matice nie je (skalárnym) násobkom iného stĺpca.*

Znamená to, že ak $\boldsymbol{h}$ je stĺpcom kontrolnej matice takého kódu, tak $i \cdot \boldsymbol{h}$, pre každé $i = 0, 1, \ldots, q-1$, nie je stĺpcom tej istej kontrolnej matice. Táto veta nám poskytuje spôsob, ako zadefinovať Hammingov q–árny kód.

Definícia 3.4.4 *Lineárny q–árny (n, k)–kód sa nazýva Hammingovým q–árnym (n, k)–kódom, ak má kontrolnú maticu s nasledujúcimi vlastnosťami:*

- *žiaden stĺpec matice nie je skalárnym násobkom iného stĺpca (lineárnou kombináciou môže byť),*
- *každé nenulové q–árne slovo dĺžky $(n - k)$ je skalárnym násobkom nejakého stĺpca kontrolnej matice.*

Maticu H Hammingovho q–árneho kódu môžeme zostrojiť zo všetkých q–árnych slov danej dĺžky, ktoré majú v svojom zápise prvý nenulový znak jednotku.

Príklad 3.4.3 *Zostrojte kontrolnú maticu Hammingovho kvartérneho kódu pre $n - k = 2$.*

Riešenie: Zapíšeme do stĺpcov všetky kvartérne dvojznakové slová, ktorých prvý nenulový znak je 1 a usporiadame ich vzostupne.

$$H = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 2 & 3 \end{pmatrix}$$

Je to teda $(5, 3)$–kód.

Dekódovanie Hammingovym q–árnym kódom

Majme danú kontrolnú maticu Hammingovho q–árneho kódu H. Dekódujeme nasledovne:

1. Je vyslané kódové slovo $\boldsymbol{v}$, prijme sa slovo $\boldsymbol{w}$.
2. Pomocou vzťahu $\boldsymbol{s}_w = \boldsymbol{w} \cdot H^T$ vypočítame syndróm prijatého slova $\boldsymbol{w}$. Syndróm bude nejakým m–násobkom, $m \in \{1, 2, \ldots, q-1\}$, i–tého stĺpca matice H.
3. Od prijatého slova odpočítame slovo tvorené nulami, iba na i–tom mieste bude m.
4. Ak je syndróm nulový, tak $\boldsymbol{w} = \boldsymbol{v}$.

Príklad 3.4.4 *Pomocou Hammingovho ternárneho $(13, 10)$–kódu dekódujte slová: $\boldsymbol{w}_1 =$ 1201011201120, $\boldsymbol{w}_2 =$ 2221210210210.*

Riešenie: Zostrojíme kontrolnú maticu pre daný kód:

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \end{pmatrix}.$$

Vypočítame syndrómy prijatých slov $\boldsymbol{w}_1$ a $\boldsymbol{w}_2$.

- $\boldsymbol{s}_1 = \boldsymbol{w}_1 \cdot H^T = 201 = 2 \cdot 102$. Syndróm $\boldsymbol{s}_1$ je dvojnásobkom siedmeho stĺpca kontrolnej matice a teda $\delta(\boldsymbol{w}_1) = 1201011201120 - 0000002000000 = 120101\underline{2}201120$.
- $\boldsymbol{s}_2 = \boldsymbol{w}_2 \cdot H^T = 020 = 2 \cdot 010$. Syndróm $\boldsymbol{s}_2$ je teda dvojnásobkom druhého stĺpca kontrolnej matice (010), preto $\delta(\boldsymbol{w}_2) = 2221210210210 - 0200000000000 = 2\underline{2}21210210210$.

Podobne ako pri binárnych, tak aj pri q–árnych kódoch sa budeme snažiť nájsť vzťah medzi počtom všetkých znakov a počtom informačných (resp. kontrolných) znakov v Hammingovom binárnom kóde. Teda chceme zistiť, pre aké n a k sa Hammingov q–árny kód dá zostrojiť.

Nech je daný q–árny (n, k)–kód. Jeho kontrolná matica má $(n-k)$ riadkov a n stĺpcov. Stĺpcami sú všetky q–árne slová danej dĺžky, ktoré majú vo svojom zápise ako prvý nenulový znak jednotku.

- Počet takých $(n-k)$–tíc, že na $(n-k)$–tom mieste je jednotka, je 1.
- Počet takých $(n-k)$–tíc, že na $(n-k-1)$–tom mieste je jednotka, je q.
- Počet takých $(n-k)$–tíc, že na $(n-k-2)$–tom mieste je jednotka, je q^2.

  $\vdots$

- Počet takých $(n-k)$–tíc, že na prvom mieste je jednotka, je q^{n-k-1}.

Počet všetkých stĺpcov kontrolnej matice je $1+q+q^1+q^2+\cdots+q^{n-k-1}-n$. Použitím vzorca pre súčet $n-k$ členom geometrickej postupnosti dostaneme, že počet stĺpcov kontrolnej matice je

$$\frac{q^{n-k}-1}{q-1} = n.$$

Uvádzame tabuľku prvých troch ternárnych Hammingových kódov:

n	4	13	41	...
k	2	10	36	...
R	0,5	0,23	0,1	...

3.5 Rozšírenie a zúženie lineárnych kódov

V praxi sa často používajú kódy s presne danými parametrami, ako sú dĺžka slova, redundancia, ... Preto sa niektoré kódy upravujú použitím vhodných transformácií na požadované kódy. Uvádzame len dva najčastejšie sa vyskytujúce transformácie pre binárne kódy.

Definícia 3.5.1 *Nech je daný binárny lineárny (n,k)–kód $\mathcal{K}$. Rozšírením kódu $\mathcal{K}$ nazývame $(n+1,k)$–kód $\mathcal{K}^*$, ktorého kódové slová vzniknú z kódových slov kódu $\mathcal{K}$ pridaním znaku kontroly parity.*

Teda ak $v_1v_2\ldots v_n \in \mathcal{K}$, tak $v_1v_2\ldots v_nv_{n+1} \in \mathcal{K}^*$ práve vtedy, ak $v_1+v_2+\cdots+v_n+v_{n+1}=0$. V generujúcej matici zostávajú tie isté riadky, len na koniec pridáme znak kontroly parity. Počet riadkov generujúcej matice sa teda nezmení a dimenzia rozšíreného kódu je taká istá, ako dimenzia pôvodného kódu.

Z predchádzajúcej definície a z definície 3.2.1 môžeme veľmi ľahko odvodiť kontrolnú maticu rozšíreného kódu. Všetky homogénne rovnice kódu ostávajú také isté, čo znamená že v kontrolnej matici na konci každého riadku pridáme len nulu a pribudne jedna rovnica vyjadrujúca paritu:

$$v_1 + v_2 + \cdots + v_n + v_{n+1} = 0,$$

ktorá sa v kontrolnej matici prejaví ako nový jednotkový riadok.

$$H_{\mathcal{K}^*} = \left(\begin{array}{ccc|c} & & & 0 \\ & H_{\mathcal{K}} & & 0 \\ & & & \vdots \\ \hline 1 & 1 & \ldots & 1 \end{array}\right).$$

Príklad 3.5.1 *Zostrojte kontrolnú a generujúcu maticu rozšíreného Hammingovho $(8,4)$–kódu.*

Riešenie: Kontrolnou maticou je nasledujúca matica:

$$H^* = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Ekvivalentnými riadkovými úpravami sa ju pokúsime upraviť na tvar $H^* = (-P^T|E')$, takže

$$H^* = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Generujúca matica bude v tvare $G^* = (E|P)$, teda

$$G^* = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

Použitím riadkových ekvivalentných úprav sa čitateľ môže presvedčiť, že $G^* \sim H^*$ a teda rozšírený Hammingov $(8, 4)$–kód je samoduálny.

Veta 3.5.1 *Ak má binárny lineárny (n, k)–kód $\mathcal{K}$ dimenziu d, tak pre dimenziu rozšíreného kódu $\mathcal{K}^*$ platí:*

$$d^* = \begin{cases} d; & \text{ak } d \text{ je párne,} \\ d+1; & \text{ak } d \text{ je nepárne.} \end{cases}$$

Definícia 3.5.2 *Nech je daný binárny lineárny (n, k)–kód $\mathcal{K}$. Zúžením kódu $\mathcal{K}$ nazývame kód $\hat{\mathcal{K}}$, ktorého kódové slová vzniknú z kódových slov kódu $\mathcal{K}$ vynechaním i–tého znaku.*

Teda ak $v_1 v_2 \dots v_n \in \mathcal{K}$, tak $v_1 v_2 \dots v_{i-1} v_{i+1} \dots v_n v_{n+1} \in \hat{\mathcal{K}}$ pre hocijaké $i = 1, 2, \dots, n$. Je to buď $(n-1, k)$–kód, alebo $(n-1, k-1)$–kód.

Generujúcu maticu zúženého kódu $G_{\hat{\mathcal{K}}}$ dostaneme vynechaním i–tého stĺpca v generujúcej matici $G_{\mathcal{K}}$.

Kapitola 4

Reed-Mullerove kódy

Reed–Mullerove kódy boli prvýkrát popísané v roku 1954 a takmer hneď sa začali využívať. Ich výhodou je to, že sú to kódy, ktoré sú použiteľné pre opravu voliteľného počtu chýb. Ich význam a rozšírenosť spočíva v tom, že majú jednoduchý popis a jednoduchý algoritmus dekódovania. Sú založené na boolovských funkciách, preto prvá časť kapitoly je venovaná tejto problematike.

4.1 Boolovské funkcie a boolovské polynómy

Definícia 4.1.1 *Nech je daná množina $B = \{0, 1\}$. Boolovskou funkciou m premenných nazývame funkciu $f : B^m \to B$, ktorá každej m–tici núl a jednotiek priradí nulu alebo jednotku.*

Túto funkciu $f(x_1, x_2, \ldots, x_m)$ zvyčajne zapisujeme buď do pravdivostnej tabuľky pomocou slov dĺžky 2^m (keďže počet všetkých binárnych m–tíc je 2^m), alebo ako boolovský polynóm.

Pravdivostná tabuľka:
Pre boolovskú funkciu m premenných je pravdivostná tabuľka tvorená stĺpcami, ktoré sú binárnym rozvojom čísel $0, 1, 2, \ldots, 2^m - 1$ v tomto poradí, no binárny rozvoj je zapisovaný odspodu. V podstate ide o zápis všetkých možných binárnych m–tíc. Počet riadkov je teda m plus jeden riadok pre hodnoty samotnej funkcie.

Príklad 4.1.1 *Zapíšte príklady nejakej boolovskej funkcie:*

a) $f_1 : B^2 \to B$

b) $f_2 : B^3 \to B$

Riešenie:

$f_1 : B^2 \to B$

x_1	0	1	0	1
x_2	0	0	1	1
f_1	0	1	0	0

$f_2 : B^3 \to B$

x_1	0	1	0	1	0	1	0	1
x_2	0	0	1	1	0	0	1	1
x_3	0	0	0	0	1	1	1	1
f_2	1	1	1	0	0	1	0	1

Boolovskú funkciu $f(x_1, x_2, \ldots, x_m)$ môžeme chápať ako binárne slovo $f_0 f_1 f_2 \ldots f_{2^m-1}$ dĺžky 2^m, kde pre každé číslo $j = 0, 1, 2, \ldots, m$ je $f_j = f(j_1, j_2, \ldots, j_m)$, pokiaľ má číslo j binárny rozvoj $j_m j_{m-1} \ldots j_1$. (Binárny rozvoj čísla berieme odzadu!)

Množinu všetkých boolovských funkcií m premenných potom berieme ako množinu všetkých binárnych slov $f_0 f_1 f_2 \ldots f_{2^m-1}$ dĺžky 2^m. Patria medzi nich aj konštantné funkcie $\mathbf{0} = 000 \ldots 0$ a $\mathbf{1} = 111 \ldots 1$.

Na tejto množine sú definované dve binárne a jedna unárna operácia:

- logický súčet $(f + g)$ – ide o sčítanie dvoch binárnych slov po zložkách podľa modulu 2,
- logický súčin $(f \cdot g)$ – ide o násobenie dvoch binárnych slov po zložkách podľa modulu 2,
- negácia f' – je definovaná ako súčet $1 + f$.

Príklad 4.1.2 *Sú dané dve boolovské funkcie:* $f = 0011, g = 1101$. *Zapíšte* $f + g, f \cdot g, f'$.

Riešenie: $f + g = 1110, f \cdot g = 0001, f' = 1100$.

Veta 4.1.1 *Nech je daná boolovská funkcia* $f(x_1, x_2, \ldots, x_m) = f_0 f_1 f_2 \ldots f_{2^m-1}$. *Potom prvá polovica toho slova je binárnym zápisom boolovskej funkcie* $f(x_1, x_2, \ldots, x_{m-1}, 0)$ *a druhá polovica toho slova je binárnym zápisom boolovskej funkcie* $f(x_1, x_2, \ldots, x_{m-1}, 1)$.

Teda boolovská funkcia $f(x_1, x_2, \ldots, x_m)$ m premenných určuje dve boolovské funkcie $m-1$ premenných a to takto:

- $f(x_1, x_2, \ldots, x_{m-1}, 0) = f_0 f_1 f_2 \ldots f_{2^{m-1}-1}$,
- $f(x_1, x_2, \ldots, x_{m-1}, 1) = f_{2^{m-1}} f_{2^{m-1}+1} \ldots f_{2^m-1}$.

Boolovský polynóm:

Boolovský polynóm je iný spôsob reprezentácie boolovských funkcií – pomocou (logických) súčtov a súčinov premenných x_i a $\mathbf{1}$. Uvedieme niektoré pravidlá:

- Keďže všetky operácie robíme podľa modulu 2, tak najvyššia mocnina bude 1 ($x_i \cdot x_i = x_i^2 = x_i^0 = 1$).
- Každý polynóm m premenných je nejakým súčtom členov $x_1^{i_1} x_2^{i_2} \ldots, x_m^{i_m}$, kde $i_1, i_2, \ldots, i_m$ sú buď 0, alebo 1.

- Vo všeobecnosti môžeme boolovské polynómy zapísať v tvare

$$f(x_1, x_2, \ldots, x_m) = \sum_{i=0}^{2^m-1} q_i \cdot x_1^{i_1} x_2^{i_2} \ldots, x_m^{i_m},$$

  kde q_i je 0 alebo 1 a číslo i má binárny rozvoj $i_m i_{m-1} \ldots i_1$.

- Stupňom boolovského polynómu budeme nazývať najväčší počet činiteľov v jednotlivých sčítancoch boolovského polynómu.

Napríklad polynóm $f(x_1, x_2) = x_2 + x_1 x_2$ je tvaru

$$f(x_1, x_2) = q_0 + q_1 x_1 + q_2 x_2 + q_3 x_1 x_2,$$

kde $q_0 = q_1 = 0$ a $q_2 = q_3 = 1$ a je stupňa 2.

Príklad 4.1.3 *Nájdite binárnu interpretáciu boolovskej funkcie dvoch premenných danej nasledujúcim boolovským polynómom.* $f = \mathbf{1} + x_2 + x_1 x_2$.

Riešenie: Keďže ide o funkciu dvoch premenných, z pravdivostnej tabuľky vieme:

x_1	0	1	0	1
x_2	0	0	1	1

Takže $f = \mathbf{1} + x_2 + x_1 x_2$=1111 + 0011 + 0101 · 0011=1100 + 0001 = 1101.

Je zrejme, že previesť boolovský polynóm na binárny zápis je možné jednoduchou aplikáciou súčinov a súčtov.

Veta 4.1.2 *Pre každú boolovskú funkciu $m+1$ premenných platí:*

$f(x_1, x_2, \ldots, x_m, x_{m+1}) = f(x_1, x_2, \ldots, x_m, 0) + [f(x_1, x_2, \ldots, x_m, 0) + f(x_1, x_2, \ldots, x_m, 1)] x_{m+1}$.

Príklad 4.1.4 *Zapíšte funkciu $f = 1101$ ako boolovský polynóm.*

Riešenie: Vidíme, že ide o funkciu dvoch premenných. Budeme postupne používať predchádzajúcu vetu.
$f = 1101 = 11 + (11 + 01)x_2 = 11 + (10)x_2 = 1 + (1 + 1)x_1 + [1 + (1 + 0)x_1]x_2 = 1 + 0x_1 + 1x_2 + 1x_1 x_2 = 1 + x_2 + x_1 x_2$.

Definícia 4.1.2 *Každé číslo $i = 0, 1, 2, \ldots, 2^m - 1$ vieme zapísať v binárnom zápise $i = i_m i_{m-1} i_{m-2} \ldots i_2 i_1$. Pre každé takéto číslo i definujeme množinu $M(i)$ takých čísel $j \leqq i$, ktoré majú vo svojom binárnom zápise jednotku iba na tých miestach, kde ich má aj číslo i.*
Teda $M(i) = \{j_m j_{m-1} j_{m-2} \ldots j_2 j_1 |$ *ak* $j_k = 1$, *tak* $i_k = 1$, $\forall k = 1, 2, \ldots, m\}$.

Príklad 4.1.5 *Vypíšte všetkých členov množín $M(0) - M(15)$.*

Riešenie: Keďže $15 = 2^4 - 1$, tak budeme posudzovať binárny zápis čísel vo forme binárnych štvoríc. Napríklad číslo 5 má binárny zápis 0101. Do množiny $M(5)$ budú patriť čísla s binárnymi zápismi $0000, 0001, 0100, 0101$, teda $0, 1, 4, 5$.

$M(0)$	0	$M(8)$	$0, 8$
$M(1)$	$0, 1$	$M(9)$	$0, 1, 8, 9$
$M(2)$	$0, 2$	$M(10)$	$0, 2, 8, 10$
$M(3)$	$0, 1, 2, 3$	$M(11)$	$0, 1, 2, 3, 8, 9, 10, 11$
$M(4)$	$0, 4$	$M(12)$	$0, 4, 8, 12$
$M(5)$	$0, 1, 4, 5$	$M(13)$	$0, 1, 4, 5, 8, 9, 12, 13$
$M(6)$	$0, 2, 4, 6$	$M(14)$	$0, 2, 4, 6, 8, 10, 12, 14$
$M(7)$	$0, 1, 2, 3, 4, 5, 6, 7$	$M(15)$	$0, 1, 2, \ldots, 14, 15$

Veta 4.1.3 *Ak boolovská funkcia* $f(x_1, x_2, \ldots, x_m) = f_0 f_1 f_2 \ldots f_{2^m-1}$ *je reprezentovaná polynómom*

$$\sum_{i=0}^{2^m-1} q_i \cdot x_1^{i_1} x_2^{i_2} \ldots, x_m^{i_m},$$

kde $i = i_m \cdot 2^{m-1} + i_{m-1} \cdot 2^{m-2} + \cdots + i_2 \cdot 2^1 + i_1 \cdot 2^0$, *tak*

$$q_i = \sum_{j \in M(i)} f_j.$$

Príklad 4.1.6 *Nájdite koeficienty* q_i *pre boolovskú funkciu* $f(x_1, x_2) = 1101$.

Riešenie:

$$\begin{aligned} q_0 &= f_0 = 1 \\ q_1 &= f_0 + f_1 = 1 + 1 = 0 \\ q_2 &= f_0 + f_2 = 1 + 0 = 1 \\ q_3 &= f_0 + f_1 + f_2 + f_3 = 1 + 1 + 0 + 1 = 1 \end{aligned}$$

Potom $f(x_1, x_2) = 1 + 0 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_1 x_2 = 1 + x_2 + x_1 x_2$.

Príklad 4.1.7 *Napíšte boolovskú funkciu* $f(x_1, x_2, x_3) = 11010010$ *ako polynóm a určte, akého je stupňa.*

Riešenie:

$$
\begin{aligned}
q_0 &= f_0 = 1 \\
q_1 &= f_0 + f_1 = 1 + 1 = 0 \\
q_2 &= f_0 + f_2 = 1 + 0 = 1 \\
q_3 &= f_0 + f_1 + f_2 + f_3 = 1 + 1 + 0 + 1 = 1 \\
q_4 &= f_0 + f_4 = 1 + 0 = 1 \\
q_5 &= f_0 + f_1 + f_4 + f_5 = 1 + 1 + 0 + 0 = 0 \\
q_6 &= f_0 + f_2 + f_4 + f_6 = 1 + 0 + 0 + 1 = 0 \\
q_7 &= f_0 + f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 = 1 + 1 + 0 + 1 + 0 + 0 + 1 + 0 = 0
\end{aligned}
$$

$f(x_1, x_2, x_3) = 1 + x_2 + x_1x_2 + x_3$. Najväčší počet činiteľov je v sčítanci x_1x_2. Je 2, a preto stupeň polynómu je 2.

4.2 Generovanie a vlastnosti Reed–Mullerovych kódov

Definícia 4.2.1 *Reed–Mullerovym kódom $R(r, m)$ nazývame množinu všetkých boolovských funkcií m premenných, ktoré sú reprezentované boolovskými polynómami stupňa najviac r. Pre $r = -1$ definujeme kód predpisom $R(-1, m) = \{00\ldots0\}$.*

Poznámka: Dá sa dokázať, že $R(r, m)$ kód je lineárny.

Teraz sa budeme zaoberať počtom informačných znakov v $R(r, m)$ kóde, teda jeho dimenziou. Tá určuje aj počet riadkov generujúcej matice.

$R(-1, m)$: pozostáva len z nulového slova, a teda jeho dimenzia je 0.

$R(0, m)$: pozostáva z nulového a jednotkového slova. Jednotkové slovo je jeho generátorom, a teda jeho dimenzia je 1.

$R(1, m)$: je tvorený boolovskými polynómami stupňa najviac 1. Generovať ho bude polynóm stupňa 0 a polynómy obsahujúce práve jednu premennu, ktorých je m. Dimenzia je teda $\binom{m}{0} + \binom{m}{1}$.

$R(2, m)$: je tvorený boolovskými polynómami stupňa najviac 2. Generovať ho budú polynómy obsahujúce 0 premenných, práve jednu premennú a práve jeden súčin dvoch premenných. Dimenzia je teda $\binom{m}{0} + \binom{m}{1} + \binom{m}{2}$.

...

$R(r, m)$: dimenzia je $\binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{r}$.

Poznámka: Reed–Mullerovym kódom $R(0, m)$ je opakovací kód dĺžky m.

Základné charakteristiky:

- Dĺžka kódového slova $R(r,m)$ kódu je $n = 2^m$.
- Počet informačných znakov $R(r,m)$ kódu je $k = \binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{r}$.
- Počet kontrolných znakov $R(r,m)$ kódu je $n - k = \binom{m}{r+1} + \binom{m}{r+2} + \cdots + \binom{m}{m}$.
- Minimálna vzdialenosť $R(r,m)$ kódu $d = 2^{m-r}$.
- Kód $R(r,m)$ odhaľuje t–násobné chyby pre $t < 2^{m-r}$.
- Kód $R(r,m)$ opravuje t–násobné chyby pre $t < 2^{m-r-1}$.

Generujúca matica:

Generujúcu maticu $R(r,m)$ kódu budeme označovať $G_{r,m}$.

- $R(-1,m)$: pozostáva len z nulového slova, a teda nemá generujúcu maticu.
- $G_{0,m}$: pozostáva z nulového a jednotkového slova, generátorom bude jednotkové slovo.

$$G_{0,m} = \begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix}$$

- $G_{1,m}$: je tvorená slovami zodpovedajúcimi boolovským polynómom obsahujúcim žiadnu premennú a práve jednu premennú: $1, x_1, x_2, \dots, x_m$.
- $G_{2,m}$: je tvorená slovami zodpovedajúcimi boolovským polynómom obsahujúcim žiadnu premennú, práve jednu premennú a práve jednu dvojicu premenných: $1, x_1, x_2, \dots, x_m$, $x_1x_2, x_1x_3, \dots, x_{m-1}x_m$.

  ...

- $G_{r,m}$: jej riadky sú tvorené slovami zodpovedajúcimi postupne všetkým boolovským súčinom $1, x_1, x_2, \dots, x_m, x_1x_2, x_1x_3, \dots, x_1x_2 \dots x_{m-1}x_m$.

Ako príklad uvádzame generujúce matice kódov $R(1,4)$ a $R(2,3)$.

$$G_{1,4} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{matrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{matrix}$$

$$G_{2,3} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{matrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \end{matrix}$$

Veta 4.2.1 *Pre generujúcu maticu $R(r,m)$ kódu platí:*

$$G_{r,m} = \left(\begin{array}{c|c} G_{r,m-1} & G_{r,m-1} \\ \hline \mathbf{0} & G_{r-1,m-1} \end{array} \right)$$

Veta 4.2.2 *Duálnym kódom k $R(r,m)$ kódu je $R(m-r-1,m)$.*

V definícii 2.2.1 je zadefinované kódovanie informačných znakov. Ak hovoríme o kódovaní pomocou Reed-Mullerovych kódov, myslí sa tým práve toto kódovanie informačných znakov na kódové slová. Ako je vyššie uvedené, každé číslo $i = 0, 1, 2, \ldots, 2^m - 1$ vieme zapísať m znakovým binárnym zápisom $i = i_m i_{m-1} i_{m-2} \ldots i_2 i_1$, teda tvorí m–ticu núl a jednotiek, ktorú vieme zapísať boolovským polynómom $\sum_{i=0}^{2^m-1} q_i \cdot x_1^{i_1} x_2^{i_2} \ldots, x_m^{i_m}$, kde q_i je 0 alebo 1.

Pri kódovaní pomocou $R(r,m)$ kódu sú informačnými znakmi hodnoty q_i pre všetky čísla $i = 0, 1, 2 \ldots 2^m - 1$, ktorých binárny rozvoj $i = i_m i_{m-1} i_{m-2} \ldots i_2 i_1$ má najviac m jednotiek. Vyšle sa polynóm

$$\sum_{i=0}^{2^m-1} q_i \cdot x_1^{i_1} x_2^{i_2} \ldots, x_m^{i_m},$$

kde $q_i = 0$ vtedy, ak má i v binárnom rozvoji váhu viac ako m. Prakticky to prebieha tak, že sa informačné slovo dĺžky k vynásobí generujúcou maticou.

4.3 Dekódovanie Reed–Mullerovych kódov

Pod pojmom dekódovanie Reed–Mullerovych kódov budeme rozumieť dekódovanie v zmysle definície 2.1.8, teda ako opravu chýb v kódových slovách spôsobených prenosom. Algoritmus dekódovania Reed–Mullerovych kódov je jednoduchý a ľahko implementovateľný. Podstata spočíva v tom, že sa pomocou kontrolných rovníc zostavených zo znakov prijatého slova vypočítajú informačné znaky. Tie sa prijímajú na základe hlasovania tak, aby väčšina rovníc platila.

Najjednoduchšie sa princíp hlasovania ilustruje na kóde $R(0,m)$. Nech je vyslané slovo $\boldsymbol{v} = v_0 v_1 \ldots v_{n-1}$, kde $n = 2^m$, prijaté bolo slovo $\boldsymbol{w} = w_0 w_1 \ldots w_{n-1}$. Vieme, že v opakovacom kóde má platiť $v_0 = v_1 = \cdots = v_{n-1}$, stačí teda hľadať zložku v_0:

$$\begin{aligned} v_0 &= w_0 \\ v_0 &= w_1 \\ v_0 &= w_2 \\ &\ldots \\ v_0 &= w_{n-1} \end{aligned}$$

Ak pre väčšinu rovníc platí, že $v_0 = 0$, tak prijímame za v_0 nulu a kódové slovo je $\boldsymbol{v} = 00 \ldots 0$. Ak pre väčšinu rovníc platí, že $v_0 = 1$, tak prijímame za v_0 jednotku a $\boldsymbol{v} = 11 \ldots 1$. Ak je ich počet nerozhodný, tak sa najčastejšie prijíma $v_0 = w_0$.

Dekódovací algoritmus pre $R(r,m)$:

Nech je vyslané kódové slovo $\boldsymbol{v} = v_0 v_1 \dots v_{2^m-1}$ a prijaté bolo slovo $\boldsymbol{w} = w_0 w_1 \dots w_{2^m-1}$. Kódové slovo $\boldsymbol{v}$ reprezentujeme polynómom

$$\sum_{i=0}^{2^m-1} q_i \cdot x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m},$$

ktorý je stupňa najviac r.

V prvom kroku určíme tie koeficienty q_i, ktoré prislúchajú súčinom $x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m}$ stupňa práve r (teda $||i|| = r$). Sú to tie, pre ktoré má číslo i vo svojom binárnom rozvoji práve r jednotiek. Pre každé také q_i vygenerujeme rovnice

$$q_i = \sum_{j \in M(i)} w_{j+s},$$

kde $s \in M(2^m - 1 - i)$. Každá z týchto rovníc nám dá hodnotu 0 alebo 1. Hlasovaním sa potom rozhodneme prijať tú hodnotu, ktorá sa vyskytovala častejšie. Ak je počet núl a jednotiek rovnaký, tak počet chýb je aspoň $d/2$ a za q_i prijmeme

$$q_i = \sum_{j \in M(i)} w_j.$$

Z takto určených q_i vytvoríme polynóm, pripočítame ho k prijatému slovu $\boldsymbol{w}$ a dostaneme $\boldsymbol{w}^{(1)}$:

$$\boldsymbol{w}^{(1)} = \boldsymbol{w} + \sum_{||i||=r} q_i \cdot x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m}$$

V ďalšom kroku opakujeme to isté, ako v prvom, no vytvárame rovnice pre tie q_i, ktoré prislúchajú súčinom $x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m}$ stupňa práve $r-1$ (teda $||i|| = r-1$). Získame slovo

$$\boldsymbol{w}^{(2)} = \boldsymbol{w}^{(1)} + \sum_{||i||=r-1} q_i \cdot x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m}$$

Takto pokračujeme dovtedy, kým nezískame aj koeficienty q_i, ktoré prislúchajú súčinom $x_1^{i_1} x_2^{i_2} \dots, x_m^{i_m}$ prvého stupňa (teda $||i|| = 1$) a dostaneme slovo $\boldsymbol{w}^{(r)}$.

Ako posledný určujeme koeficient q_0 a to hlasovaním v slove $\boldsymbol{w}^{(r)}$. Ak je v ňom viac jednotiek, tak $q_0 = 1$ a ak viac núl, tak $q_0 = 0$.

Príklad 4.3.1 *Pomocou kódu* $R(1,4)$ *dekódujte prijaté slovo* $\boldsymbol{w} = 0101011110101100$.

Riešenie: Je to kód, ktorého kódové slová sú reprezentované boolovskými polynómami stupňa najviac 1 a dĺžka kódových slov je $2^4 = 16$. Každé kódové slovo sa dá zapísať ako polynóm

$$\boldsymbol{v} = q_0 + q_1 x_1 + q_2 x_2 + q_4 x_3 + q_8 x_4.$$

Najprv budeme určovať koeficienty pri súčinoch stupňa 1, teda q_1, q_2, q_4, q_8. Pre každý z týchto koeficientov vygenerujeme rovnice, dosadíme do nich hodnoty z prijatého $\boldsymbol{w}$ a vypočítame ich.

q_1: $j \in M(1) = \{0, 1\}$
$s \in M(2^4 - 1 - 1) = M(14) = \{0, 2, 4, 6, 8, 10, 12, 14\}$

$$\begin{aligned} q_1 &= w_0 + w_1 = 1 & \qquad q_1 &= w_8 + w_9 = 1 \\ q_1 &= w_2 + w_3 = 1 & \qquad q_1 &= w_{10} + w_{11} = 1 \\ q_1 &= w_4 + w_5 = 1 & \qquad q_1 &= w_{12} + w_{13} = 0 \\ q_1 &= w_6 + w_7 = 0 & \qquad q_1 &= w_{14} + w_{15} = 0 \end{aligned}$$
$$q_1 = 1$$

q_2: $j \in M(2) = \{0, 2\}$
$s \in M(2^4 - 2 - 1) = M(13) = \{0, 1, 4, 5, 8, 9, 12, 13\}$

$$\begin{aligned} q_2 &= w_0 + w_2 = 0 & \qquad q_2 &= w_8 + w_{10} = 0 \\ q_2 &= w_1 + w_3 = 0 & \qquad q_2 &= w_9 + w_{11} = 0 \\ q_2 &= w_4 + w_6 = 1 & \qquad q_2 &= w_{12} + w_{14} = 1 \\ q_2 &= w_5 + w_7 = 0 & \qquad q_2 &= w_{13} + w_{15} = 1 \end{aligned}$$
$$q_2 = 0$$

q_4: $j \in M(4) = \{0, 4\}$
$s \in M(2^4 - 4 - 1) = M(13) = \{0, 1, 2, 3, 8, 9, 10, 11\}$

$$\begin{aligned} q_4 &= w_0 + w_4 = 0 & \qquad q_4 &= w_8 + w_{12} = 0 \\ q_4 &= w_1 + w_5 = 0 & \qquad q_4 &= w_9 + w_{13} = 1 \\ q_4 &= w_2 + w_6 = 1 & \qquad q_4 &= w_{10} + w_{14} = 1 \\ q_4 &= w_3 + w_7 = 0 & \qquad q_4 &= w_{11} + w_{15} = 0 \end{aligned}$$
$$q_4 = 0$$

q_8: $j \in M(8) = \{0, 8\}$
$s \in M(2^4 - 8 - 1) = M(7) = \{0, 1, 2, 3, 4, 5, 6, 7\}$

$$\begin{aligned} q_8 &= w_0 + w_8 = 1 & \qquad q_8 &= w_4 + w_{12} = 1 \\ q_8 &= w_1 + w_9 = 1 & \qquad q_8 &= w_5 + w_{13} = 0 \\ q_8 &= w_2 + w_{10} = 1 & \qquad q_8 &= w_6 + w_{14} = 1 \\ q_8 &= w_3 + w_{11} = 0 & \qquad q_8 &= w_7 + w_{15} = 1 \end{aligned}$$
$$q_8 = 1$$

$\boldsymbol{w}^{(1)} = \boldsymbol{w} + x_1 + x_4 = 0101011110101100 + 0101010101010101 + 0000000011111111 = 0000001000000110$. Toto slovo obsahuje 3 jednotky a 13 núl a teda hlasovaním prijímame, že $q_0 = 0$.
Vyslané bolo kódové slovo $\boldsymbol{v} = x_1 + x_4 = 0101010101010101 + 0000000011111111 = 0101010110101010$. Všimnime si, že s prijatým slovom sa líši v 3 znakoch. Vieme, že kód opravuje všetky t–násobné chyby, ak $t < d/2$. Daný kód túto chybu opravil jednoznačne, pretože v kóde $R(1, 4)$ je minimálna vzdialenosť $d = 2^{4-1} = 8$.

Príklad 4.3.2 *Pomocou kódu* $R(2,3)$ *dekódujte prijaté slovo* $\boldsymbol{w} = 11100011$.

Riešenie: $R(2,3)$ je kód, ktorého kódové slová sú reprezentované boolovskými polynómami stupňa najviac 2 a dĺžka kódových slov je $2^3 = 8$. Každé kódové slovo sa dá zapísať ako polynóm

$$\boldsymbol{v} = q_0 + q_1x_1 + q_2x_2 + q_3x_1x_2 + q_4x_3 + q_5x_1x_3 + q_6x_2x_3.$$

Najprv budeme určovať koeficienty pri súčinoch stupňa 2, teda q_3, q_5, q_6. Pre každý z týchto koeficientov vygenerujeme rovnice a dosadíme do nich hodnoty z prijatého $\boldsymbol{w}$.

q_3: $j \in M(3) = \{0,1,2,3\}$
$s \in M(2^3 - 3 - 1) = M(4) = \{0,4\}$

$$\begin{aligned} q_3 &= w_0 + w_1 + w_2 + w_3 = 1 \\ q_3 &= w_4 + w_5 + w_6 + w_7 = 0 \\ q_3 &= 1 \end{aligned}$$

q_5: $j \in M(5) = \{0,1,4,5\}$
$s \in M(2^3 - 5 - 1) = M(2) = \{0,2\}$

$$\begin{aligned} q_5 &= w_0 + w_1 + w_4 + w_5 = 0 \\ q_5 &= w_2 + w_3 + w_6 + w_7 = 1 \\ q_5 &= 0 \end{aligned}$$

q_6: $j \in M(6) = \{0,2,4,6\}$
$s \in M(2^3 - 6 - 1) = M(1) = \{0,1\}$

$$\begin{aligned} q_6 &= w_0 + w_2 + w_4 + w_6 = 1 \\ q_6 &= w_1 + w_3 + w_5 + w_7 = 0 \\ q_6 &= 1 \end{aligned}$$

Keďže aj pri všetkých troch koeficientoch nastal nerozhodný stav, vieme povedať, že došlo k chybám na aspoň $d/2 = 1$ pozíciách, koeficienty sme vypočítali podľa $q_i = \sum_{j \in M(i)} w_j$ a $\boldsymbol{w}^{(1)} = \boldsymbol{w} + x_1x_3 = 01100011 + 00000101 = 01100110$. To však znamená, že nevieme zaručiť, že toto dekódovanie správne opraví prijaté slovo.

Ďalej budeme určovať koeficienty pri súčinoch stupňa 1, teda q_1, q_2, q_4 tak, že pre každý z týchto koeficientov vygenerujeme rovnice a dosadíme do nich hodnoty z vypočítaného slova $\boldsymbol{w}^{(1)}$.

q_1: $j \in M(1) = \{0,1\}$
$s \in M(2^3 - 1 - 1) = M(6) = \{0,2,4,6\}$

$$\begin{aligned} q_1 &= w_0^{(1)} + w_1^{(1)} = 1 \qquad & q_1 &= w_4^{(1)} + w_5^{(1)} = 1 \\ q_1 &= w_2^{(1)} + w_3^{(1)} = 1 \qquad & q_1 &= w_6^{(1)} + w_7^{(1)} = 1 \\ & q_1 = 1 \end{aligned}$$

q_2: $j \in M(2) = \{0, 2\}$
$s \in M(2^3 - 2 - 1) = M(5) = \{0, 1, 4, 5\}$

$$q_2 = w_0^{(1)} + w_2^{(1)} = 1 \qquad q_2 = w_4^{(1)} + w_6^{(1)} = 1$$
$$q_2 = w_1^{(1)} + w_3^{(1)} = 1 \qquad q_2 = w_5^{(1)} + w_7^{(1)} = 1$$
$$q_2 = 1$$

q_4: $j \in M(4) = \{0, 4\}$
$s \in M(2^3 - 4 - 1) = M(3) = \{0, 1, 2, 3\}$

$$q_4 = w_0^{(1)} + w_4^{(1)} = 0 \qquad q_4 = w_2^{(1)} + w_6^{(1)} = 0$$
$$q_4 = w_1^{(1)} + w_5^{(1)} = 0 \qquad q_4 = w_3^{(1)} + w_7^{(1)} = 0$$
$$q_4 = 0$$

$\boldsymbol{w}^{(2)} = \boldsymbol{w}^{(1)} + x_1 + x_2 = 01100110 + 01010101 + 00110011 = 00000000$. Toto slovo obsahuje len 8 núl, a teda hlasovaním prijímame, že $q_0 = 0$. Kódom $R(2, 3)$ bolo stanovené, že vyslané bolo kódové slovo $\boldsymbol{v} = x_1 + x_2 + x_1x_3 = 01100011$.

Kapitola 5

Cyklické kódy

V praxi sa používajú tie triedy lineárnych bezpečnostných kódov, ktoré spĺňajú požadované parametre, majú jednoduchý popis a jednoduchý algoritmus dekódovania. No dekódovanie niektorých kódov je pri dlhších kódoch z výpočtového hľadiska náročné. Preto sa hľadajú také podtriedy triedy lineárnych kódov, ktorých dekódovanie by bolo výpočtovo efektívnejšie. Cyklické kódy sú podskupinou lineárnych kódov, ktoré sú založené na silnejších algebraických štruktúrach ako lineárne kódy vo všeobecnosti, a to im dáva vyššiu výpočtovú efektivitu. Pre prácu s týmito kódmi sa používa reprezentácia slov pomocou polynómov.

5.1 Polynomická reprezentácia slov

Definícia 5.1.1 *Nech je daná konečná q–prvková množina* $T = \{0, 1, \ldots, q-1\}$ *a nech* $n \in \mathbb{N}$. *Výraz v tvare*

$$a(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n = \sum_{i=0}^{n} a_i x^i$$

budeme nazývať polynómom premennej x *s koeficientami* $a_i \in T,\ i = 0, 1, \ldots, n$. *Ak* $a_n \neq 0$, *tak číslo* n *nazývame stupňom polynómu* $a(x)$ *a označujeme ho* $st(a(x))$.

Definícia 5.1.2 *Koreňom polynómu* $a(x)$ *nad* T *nazývame každé také číslo* $t \in T$, *pre ktoré* $a(t) = 0$.

Slová $a_0 a_1 \ldots a_{n-1}$ nad kódovou abecedou T potom môžeme formálne popísať takýmto polynómom

$$a(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}.$$

V tomto zápise však nemusí byť $a_n \neq 0$ ako je tomu pri bežných polynómoch. Pre prácu s polynómami reprezentujúcimi kódové slová lineárnych kódov platia určité pravidlá.

Základné pravidlá počítania s polynómami

Nech je daná kódová abeceda $T = \{0, 1, \ldots, q-1\}$ s q prvkami a nad ňou zostrojená množina $T_{n-1}(x)$ všetkých polynómov $a(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}$ s koeficientami z kódovej abecedy.

- Súčtom dvoch polynómov $a(x), b(x) \in T_n(x)$ rozumieme polynóm $a(x) + b(x) = (a_0 + b_0) + (a_1 + b_1)x + (a_2 + b_2)x^2 + \cdots + (a_{n-1} + b_{n-1})x^{n-1}$, kde koeficienty sčítavame modulárne podľa modulu q.

- Skalárnym násobkom (skrátene len násobkom) polynómu $a(x) \in T_{n-1}(x)$ a čísla $\alpha \in T$ rozumieme polynóm $\alpha \cdot a(x) = \alpha a_0 + \alpha a_1 x + \alpha a_2 x^2 + \cdots + \alpha a_{n-1} x^{n-1}$, kde koeficienty násobíme číslom α modulárne podľa modulu q.

- Súčinom dvoch polynómov $a(x), b(x) \in T_{n-1}(x)$ rozumieme polynóm $a(x) \cdot b(x)$, ktorý dostaneme vynásobením každého člena $a_i x^i$ každým členom $b_j x^j$, teda

$$a(x) \cdot b(x) = \sum_{i=0}^{n-1} \Big(\sum_{j=0}^{n-1} a_i x^i \cdot b_j x^j \Big)$$

  pre $\forall i, j \in \{0, 1, \ldots, n-1\}$ kde $a_i x^i \cdot b_j x^j = (a_i b_j) x^{i+j}$, kde koeficienty a_i, b_j násobíme modulárne podľa modulu q a exponenty i, j násobíme modulárne podľa modulu $n+1$.

Ako príklad uvádzame násobenie binárnych polynómov prislúchajúcich slovám dĺžky 5.

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

Delenie a deliteľnosť polynómov

Z lineárnej algebry vieme, že pre každé dva polynómy $a(x)$ a $b(x)$, $b(x) \neq 0$, nad T existujú jediné polynómy $p(x)$ a $r(x)$ nad T také, že platí

$$a(x) = p(x)b(x) + r(x) \tag{5.1}$$

a $st(r(x)) < st(b(x))$.

Delenie polynómu $a(x)$ polynómom $b(x)$ nad T teda rozumieme určenie polynómov $p(x)$ a $r(x)$ z (5.1).

Definícia 5.1.3 *Ak pre dva polynómy $a(x)$ a $b(x)$ nad T je polynóm $r(x)$ z (5.1) nulový, tak hovoríme, že polynóm $b(x)$ delí polynóm $a(x)$, resp. je jeho deliteľom. Každý polynóm nad T je deliteľný sebou samým a hocijakým polynómom nultého stupňa. Tieto delitele nazývame triviálnymi deliteľmi.*

Definícia 5.1.4 *Ak polynóm $a(x)$ nemá nad T iné ako triviálne delitele, nazýva sa ireducibilný polynóm. V opačnom prípade ho nazývame reducibilný polynóm.*

Platí, že každý reducibilný polynóm stupňa n sa dá jednoznačne rozložiť na súčin ireducibilných polynómov stupňa menej ako n.

Príklad 5.1.1 *Vydeľte polynóm $x^4 + x^3 + 1$ polynómom $x^2 + 1$ nad $T = \{0, 1\}$.*

Riešenie:

$$\begin{array}{l} (x^4 + x^3 + 1) : (x^2 + 1) = x^2 + x + 1 \\ \underline{-(x^4 + x^2)} \\ (x^3 + x^2 + 1) \\ \underline{-(x^3 + x)} \\ (x^2 + x + 1) \\ \underline{-(x^2 + 1)} \\ x \end{array}$$

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

5.2 Vytváranie cyklických kódov

Keďže v praxi sú používané prevažne binárne kódy, ďalej sa budeme zaoberať už len cyklickými binárnymi kódmi, teda kódovou abecedou bude množina $B = \{0, 1\}$.

Definícia 5.2.1 *Lineárny (n, k)–kód $\mathcal{C} \subset B^n$ sa nazýva cyklickým, ak s každým kódovým slovom $v_0 v_1 \dots v_{n-1} \in \mathcal{C}$ je kódové aj slovo $v_{n-1} v_0 v_1 \dots v_{n-2}$. Týmto je definovaný cyklický posun znakov v slovách.*

Definícia 5.2.2 *Každé kódové slovo $v_0 v_1 \dots v_{n-1}$ dĺžky n vieme reprezentovať binárnym polynómom $v(x) = v_0 + v_1 x + v_2 x^2 + \dots + v_{n-1} x^{n-1} \in B_{n-1}(x)$ stupňa $n - 1$, ktorý budeme nazývať kódový polynóm.*

Množinu kódových polynómov kódu $\mathcal{C}$ budeme označovať $\mathcal{C}(x)$ a je zjavné, že $\mathcal{C}(x) \subset B_{n-1}(x)$.

Cyklický posun v lineárnom kóde daný definíciou 5.2.1 môžeme polynómom vyjadriť takto: "Ak polynóm $v(x) \in \mathcal{C}(x)$, tak aj polynóm $x \cdot v(x) \in \mathcal{C}(x)$." Postupným cyklickým posunom dostaneme, že každý $x, x^2, x^3 \dots$ násobok kódového polynómu $v(x)$ je tiež kódovým polynómom.

Vzhľadom na to, akou algebraickou štruktúrou cyklický kód je, sa dá dokázať, že platí nasledujúca veta.

Veta 5.2.1 *Nech je daný cyklický kód $\mathcal{C} \subset B^n$ reprezentovaný kódovými polynómami $v(x) \in \mathcal{C}(x)$. Potom pre každý kódový polynóm $v(x)$ a pre každý polynóm $p(x) \in B_{n-1}(x)$ je polynóm $v(x)p(x)$ kódovým polynómom.*

Z toho sa ďalej dá dokázať, že pre každý kód dokonca existuje jediný polynóm $g(x) \in \mathcal{C}(x)$ taký, že všetky kódové slová sú jeho nejakým násobkom $v(x)p(x)$, kde $p(x) \in B_{n-1}(x)$.

Definícia 5.2.3 *Polynóm $g(x) \in \mathcal{C}(x)$ spĺňajúci predchádzajúcu vlastnosť nazývame generujúcim polynómom daného kódu.*

Cyklické kódy dĺžky n sú vytvárané polynómami stupňa najviac $n-1$ takým spôsobom, že generujúcim polynómom kódu môžu byť všetky ireducibilné polynómy z $B_{n-1}(x)$, ktoré sú deliteľmi polynómu $x^n + 1 \in B_n(x)$ alebo ich vzájomné súčiny.

Veta 5.2.2 *Nech je daný lineárny cyklický (n, k)–kód $\mathcal{C}$ s jeho generujúcim polynómom $g(x)$. Potom pre stupeň generujúceho polynómu platí*

$$st(g(x)) = n - k.$$

Takže dimenziu cyklického kódu daného generujúcim polynómom môžeme určiť ako rozdiel dĺžky kódu a stupňa generujúceho polynómu.

Príklad 5.2.1 *Nájdite generujúce polynómy všetkých možných cyklických kódov dĺžky 7.*

Riešenie: Polynóm $x^7 + 1$ sa dá rozložiť na súčin troch ireducibilných polynómov:

$$x^7 + 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

Počet všetkých možných generujúcich polynómov je teda 2^3. V nasledujúcej tabuľke je uvedený ich prehľad.

kód	generujúci polynóm $g(x)$	$st(g(x))$
$\mathcal{C}_1$	1	0
$\mathcal{C}_2$	$x + 1$	1
$\mathcal{C}_3$	$x^3 + x + 1$	3
$\mathcal{C}_4$	$x^3 + x^2 + 1$	3
$\mathcal{C}_5$	$(x + 1)(x^3 + x + 1)$	4
$\mathcal{C}_6$	$(x + 1)(x^3 + x^2 + 1)$	4
$\mathcal{C}_7$	$(x^3 + x + 1)(x^3 + x^2 + 1)$	6
$\mathcal{C}_8$	$x^7 + 1$	7

Ak poznáme generujúci polynóm kódu $\mathcal{C}$ vieme jednoducho zapísať generujúcu maticu daného kódu. Vznikne postupnými cyklickými posunmi koeficientov generujúceho polynómu. Teda generujúci polynóm tu prevzal úlohu generujúcej matice.

$$G_{\mathcal{C}} = \begin{pmatrix} g_0 & g_1 & g_2 & \dots & g_{n-k} & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & g_2 & \dots & g_{n-k} & 0 & \dots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \dots & g_{n-k} & \dots & 0 \\ & & & \vdots & & & & \vdots & \\ 0 & 0 & \dots & 0 & g_0 & g_1 & g_2 & \dots & g_{n-k} \end{pmatrix}$$

Podobne úlohu kontrolnej matice pri polynomickej reprezentácií kódu preberá kontrolný polynóm.

Definícia 5.2.4 *Nech je daný cyklický kód $\mathcal{C}$ dĺžky n a nech polynóm $g(x) \in \mathcal{C}(x)$ je jeho generujúcim polynómom. Potom polynóm $h(x) \in B_{n-1}(x)$ taký, že platí*

$$x^n + 1 = h(x) \cdot g(x)$$

nazývame kontrolným polynómom kódu $\mathcal{C}$.

Príklad 5.2.2 *Nájdite kontrolné polynómy všetkých cyklických kódov z príkladu 5.2.1.*

Riešenie: Keďže pre kontrolný polynóm má platiť $x^7 + 1 = h(x) \cdot g(x)$, tak kontrolnými polynómami jednotlivých kódov budú súčiny tých ireducibilných polynómov z rozkladu $x^7 + 1$, ktoré netvoria generujúce polynómy.

kód	generujúci polynóm $g(x)$	kontrolný polynóm $h(x)$
$\mathcal{C}_1$	1	$x^7 + 1$
$\mathcal{C}_2$	$x + 1$	$(x^3 + x + 1)(x^3 + x^2 + 1)$
$\mathcal{C}_3$	$x^3 + x + 1$	$(x + 1)(x^3 + x^2 + 1)$
$\mathcal{C}_4$	$x^3 + x^2 + 1$	$(x + 1)(x^3 + x + 1)$
$\mathcal{C}_5$	$(x + 1)(x^3 + x + 1)$	$x^3 + x^2 + 1$
$\mathcal{C}_6$	$(x + 1)(x^3 + x^2 + 1)$	$x^3 + x + 1$
$\mathcal{C}_7$	$(x^3 + x + 1)(x^3 + x^2 + 1)$	$x + 1$
$\mathcal{C}_8$	$x^7 + 1$	1

Veta 5.2.3 *Nech je daný cyklický kód $\mathcal{C}$ dĺžky n a polynóm $h(x) \in B_{n-1}(x)$ je jeho kontrolným polynómom. Ak pre ľubovoľný polynóm $v(x) \in B_{n-1}(x)$ platí, že $v(x)p(x) = 0$, tak $v(x) \in \mathcal{C}(x)$.*

Teda cyklický kód s kontrolným polynómom $h(x) \in B_{n-1}(x)$ pozostáva práve z tých polynómov $v(x) \in B_{n-1}(x)$, pre ktoré platí $v(x)p(x) = 0$.

Ak poznáme kontrolný polynóm kódu $\mathcal{C}$, vieme zapísať kontrolnú maticu daného kódu. Vznikne postupnými cyklickými posunmi koeficientov kontrolného polynómu zapísaných od konca. Teda posun bude v opačnom smere ako pri generujúcej matici.

$$H_{\mathcal{C}} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & h_k & \dots & h_2 & h_1 & h_0 \\ 0 & 0 & \dots & 0 & h_k & \dots & h_2 & h_1 & h_0 & 0 \\ 0 & \dots & 0 & h_k & \dots & h_2 & h_1 & h_0 & 0 & 0 \\ & & & \vdots & & & & \vdots & & \\ h_k & \dots & h_2 & h_1 & h_0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

Príklad 5.2.3 *Zapíšte generujúcu aj kontrolnú maticu kódu $\mathcal{C}_5$ z príkladu 5.2.1.*

Riešenie: Generujúcim polynómom je $g(x) = (x+1)(x^3+x+1) = 1+x^2+x^3+x^4$. Prvý riadok generujúcej matice budú tvoriť koeficienty generujúceho polynómu doplnené nulami na počet 7.

$$G_{\mathcal{C}_5} = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Kontrolným polynómom je $h(x) = x^3 + x^2 + 1$. Prvý riadok kontrolnej matice budú tvoriť koeficienty kontrolného polynómu zapísané od konca riadku a dopredu doplnené nulami na počet 7. Cyklicky budeme riadky posúvať doľava.

$$H_{\mathcal{C}_5} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Pozorovanie: Generujúci polynóm kódu $\mathcal{C}$ je kontrolným polynómom kódu $\mathcal{C}^\perp$ a naopak.

5.3 Kódovanie pomocou cyklických kódov

Pod pojmom kódovanie je myslené kódovanie v zmysle definície 2.2.1, teda ide o kódovanie informačných znakov. Je to zobrazenie, ktoré informačnému slovu dĺžky k priradí kódové slovo dĺžky n, ktoré sa potom vysiela.

Nech je daný cyklický (n,k)–kód $\mathcal{C}$ s jeho generujúcim polynómom $g(x)$. Každé informačné slovo, ktoré budeme chcieť týmto kódom zakódovať, je dĺžky k.

- Informačné slovo budeme reprezentovať informačným polynómom stupňa najviac $k-1$. Označíme ho $i(x) = i_0 + i_1x + i_2x^2 + \cdots + i_{k-1}x^{k-1}$.
- Kódový polynóm $v(x)$ prislúchajúci informačnému polynómu $i(x)$ vypočítame takto: $v(x) = i(x) \cdot g(x)$.

Keďže informačný polynóm je stupňa najviac $k-1$ a generujúci polynóm $g(x)$ je stupňa najviac $n-k$, tak kódové slovo je stupňa najviac $n-1$. Toto kódovanie informačných znakov je správne, no nie je systematické. Teda z kódového polynómu sa nedá bezprostredne určiť informačný polynóm, a tým aj zdrojová správa. Existuje však aj systematické kódovanie informačných znakov, v ktorom z kódového polynómu sa dá ľahko a bez výpočtu určiť informačný polynóm.

Systematické kódovanie pomocou cyklických kódov

- Informačné slovo budeme reprezentovať informačným polynómom vytvoreným tak, že koeficienty priradzujeme najvyšším mocninám od konca: $i(x) = i_0x^{n-1} + i_1x^{n-2} + i_2x^{n-3} + \cdots + i_{k-1}x^{n-k}$.

- Vypočítame zvyšok $r(x)$ po delení polynómu $i(x)$ generujúcim polynómom $g(x)$, čo znamená, že platí $i(x) = p(x)g(x) + r(x)$.
- Kódový polynóm $v(x)$ prislúchajúci informačnému polynómu $i(x)$ vypočítame takto: $v(x) = i(x) - r(x)$.

Kódové slovo je v tvare $v(x) = i_0 i_1 \dots i_{k-1} r_{n-k-1} \dots r_1 r_0$. Z uvedeného tvaru je zjavné, že prvých k znakov kódového slova určuje informačné slovo.

Ľahké kódovanie a dekódovanie informačných znakov je tiež jednou z výhod cyklických kódov a aj dôvodom ich širokého využitia.

5.4 Dekódovanie cyklických kódov

Dekódovanie cyklických kódov je založené na podobnom princípe ako dekódovanie lineárnych kódov pomocou syndrómov. Pojem syndróm tu však bude chápaný v trošku pozmenenej forme.

Definícia 5.4.1 *Nech je daný cyklický kód $\mathcal{C}$ dĺžky n a nech polynóm $g(x) \in \mathcal{C}(x)$ je jeho generujúcim polynómom. Vieme, že pre každý polynóm $w(x) \in T_{n-1}(x)$ a $g(x)$ jednoznačne existujú jediné polynómy $p(x)$ a $s(x)$ nad T také, že platí*

$$w(x) = p(x)g(x) + s(x)$$

a $st(s(x)) < st(g(x))$. Polynóm $s(x)$ z tohto zápisu nazývame syndrómom (syndrómovým polynómom) polynómu $w(x)$ podľa kódu $\mathcal{C}$.

Tým, že kódové slovo v systematickom kódovaní vzniká z informačného slova podľa predpisu $v(x) = i(x) - r(x) = p(x)g(x) + r(x) - r(x) = p(x)g(x)$, teda kódové slovo je deliteľné generujúcim polynómom bez zvyšku, a teda jeho syndrómom je nulový polynóm.

Príklad 5.4.1 *Nájdite všetky syndrómové polynómy kódu $\mathcal{C}_3$ z príkladu 5.2.1.*

Riešenie: Je to cyklický kód s generujúcim polynómom $g(x) = x^3 + x + 1$ a kontrolným polynómom $h(x) = (x+1)(x^3 + x^2 + 1) = x^4 + x^2 + x + 1$. Jeho kontrolná matica má tvar

$$H_{\mathcal{C}_3} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Je to vlastne kontrolná matica Hammingovho $(7, 4)$–kódu s poprehadzovanými stĺpcami. Tento kód má minimálnu vzdialenosť 3 a teda opravuje všetky jednoduché chyby. Chybové polynómy budú váhy najviac 1. Vypíšeme ich a vypočítame k nim syndrómy.

chybový polynóm	syndrómový polynóm
0	0
1	1
x	x
x^2	x^2
x^3	$x+1$
x^4	x^2+x
x^5	x^2+x+1
x^6	x^2+1

Veta 5.4.1 *Nech je daný cyklický kód $\mathcal{C}$ dĺžky n s generujúcim polynómom $g(x) \in \mathcal{C}(x)$. Nech je vyslaný polynóm $v(x) \in \mathcal{C}(x)$ a nech je prijatý polynóm $w(x) \in T_{n-1}(x)$, $w(x) = v(x) + e(x)$, kde $e(x)$ je polynóm prislúchajúci chybovému slovu. Potom syndróm polynómu $w(x)$ je taký istý ako syndróm chybového polynómu $e(x)$.*

Dekódovací algoritmus možno zapísať v krokoch:

Nech je daný cyklický (n,k)–kód $\mathcal{C}$ s generujúcim polynómom $g(x) \in \mathcal{C}(x)$ a nech je vyslaný polymóm $v(x) \in \mathcal{C}(x)$.

1. Prijme sa polymóm $w(x) \in T_{n-1}(x)$.
2. Vypočítame syndrómový polynóm $s(x)$ ako zvyšok po delení polynómu $w(x)$ generujúcim polynómom $g(x)$, teda podľa predpisu
$$w(x) = p(x)g(x) + s(x).$$
3. Podľa syndrómu určíme príslušný chybový polynóm $e(x)$.
4. Dekódujeme podľa predpisu $v(x) = w(x) + e(x)$.

Použitie polynomickej reprezentácie pri cyklických kódoch umožnilo nahradiť operácie s vektormi a maticami jednoduchšie imlementovateľnými operáciami s polynómami.

Algoritmus dekódovania sa v praxi rôzne modifikuje využitím ďalších vlastností cyklických kódov na výpočtovo menej náročnejšie algoritmy. Tieto modifikácie sú prispôspobené rôznym faktorom. Napríklad Meggitov algoritmus dekódovania binárnych cyklických kódov je výhodný pri použití tzv. LFSR (linear feedback shift register) dekódera. Inou modifikáciou dekódovacieho algoritmu je " Error trapping " dekódovací algoritmus. Ten je zase výhodný pre cyklické kódy opravujúce jednonásobné chyby, alebo dvojnásobné chyby, ak sa ich výskyt očakáva na blízkych pozíciách.

Golayov kód

Veta 3.4.2 uvádza, že jediným perfektným kódom okrem Hammingových a opakovacích kódov je Golayov kód pre trojnásobné opravy. Tento kód je cyklický a uvedieme jeho základné parametre.

Je to kód dĺžky 23. Rozklad polynómu $x^{23}+1$ na ireducibilné polynómy je takýto:

$$x^{23}+1=(x+1)(1+x^2+x^4+x^5+x^6+x^{10}+x^{11})(1+x+x^5+x^6+x^7+x^8+x^{11}).$$

Generujúcim polynóm Golayovho kódu je polynóm $g(x)=1+x^2+x^4+x^5+x^6+x^{10}+x^{11}$, takže kontrolným polynómom tohto kódu je $h(x)=(x+1)(1+x+x^5+x^6+x^7+x^8+x^{11})$. Je to teda binárny cyklický $(23,12)$–kód. Jeho minimálna vzdialenosť je 7, a teda skutočne opravuje trojnásobné chyby.

V literatúre [9] môže čitateľ nájsť ďalšie vlastnosti a charakteristiky Golayovho kódu.

Literatúra

[1] Adámek, J.: *Kódování*. Praha : SNTL, 1989.

[2] Adámek, J.: *Kódování a teórie informace*. Praha : ČVUT, 1991.

[3] Biolek, D.: *Datová komunikace*. Brno, 2002, [cit.25.06.2012]. Dostupné na: http://user.unob.cz/biolek/vyukaVUT/skripta/DKO.pdf .

[4] Čipková, K., Satko, L.: *Základy kódovania*. Bratislava : Nakladateľstvo STU, 2009, ISBN 978-80-227-3016-7.

[5] Koblitz, N.: *A course in number theory and Cryptography – 2nd edition*. New York : Springer-Verlag, 1994, ISBN0-387-94293-9.

[6] Legéň, A.: *Grupy, okruhy a zväzy*. Bratislava : ALFA, 1980.

[7] Levický, D.: *Kryptografia v informačnej bezpečnosti*. Košice : Elfa, s. r. o., 2005, ISBN 80-8086-002-X.

[8] Mac Lane, S., Birkhoff, G.: *Algebra*. Bratislava : ALFA, 1974.

[9] Olejár, D., Stanek, M.: *Úvod do teórie kódovania*. 2007, [cit.27.06.2012]. Dostupné na: http://www.dcs.fmph.uniba.sk/texty/codebook.pdf .

.

Názov:	ZÁKLADY KÓDOVANIA
Autor:	© RNDr. Daniela KRAVECOVÁ, PhD., 2012
Recenzovali:	Mgr. Jana Petrillová Prof. RNDr. Ján Plavka, CSc.
Vydavateľ:	Katedra matematiky a teoretickej informatiky Fakulta elektrotechniky a informatiky Technická univerzita v Košiciach
Miesto vydania:	Košice
Rok vydania:	2012
Vydanie:	Prvé
Rozsah:	77 strán
ISBN:	978-80-553-1178-4