Kódování

89. Definujte těleso Zn a uveďte jeho základní vlastnosti.

Těleso $\mathbf{Z}_2$

10.2. Poznámka. Definice lineárního prostoru 1.6 předpokládá, že skaláry (čísla, kterými násobíme vektory) jsou reálná čísla. V této kapitole budeme pracovat s modifikovanou definicí lineárního prostoru, kde reálná čísla nahradíme tělesem $\mathbf{Z}_2$.

Poznamenávám, že pojem tělesa jsem přesně zavedl v textu za poznámkou 1.29. Pokud čtenář tuto část textu přeskočil, může si pod pojmem těleso zhruba představit množinu s operacemi sčítání a násobení. Tyto operace mají podobné vlastnosti, jako sčítání a násobení reálných čísel (komutativita, asociativita, distributivita, atd.).

10.3. Definice. *Těleso $\mathbf{Z}_2$* je dvoubodová množina $\{0,1\}$, na které je definováno sčítání $+ : \mathbf{Z}_2 \times \mathbf{Z}_2 \to \mathbf{Z}_2$ a násobení $\cdot : \mathbf{Z}_2 \times \mathbf{Z}_2 \to \mathbf{Z}_2$ takto:

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Tedy: $0+0=0, \quad 0+1=1+0=1, \quad 1+1=0, \quad 0\cdot 0=0\cdot 1=1\cdot 0=0, \quad 1\cdot 1=1.$

10.4. Poznámka. Sčítání na $\mathbf{Z}_2$ je shodné s logickou operací XOR (vylučovací nebo) a násobení na $\mathbf{Z}_2$ je shodné s operací AND (logická konjunkce).

Nebo jinak: Násobení na $\mathbf{Z}_2$ je stejné, jako jsme zvyklí násobit celá čísla, a sčítání skoro taky, až na jedinou výjimku: $1+1=0$.

Nebo ještě jinak: Prvek 0 v $\mathbf{Z}_2$ si můžeme představit jako jakékoli sudé číslo a prvek 1 jako jakékoli číslo liché. Sčítáme a násobíme pak sudá čísla se sudými, s lichými atd. Tyto operace pak dávají jako výsledek čísla sudá nebo lichá přesně podle pravidel počítání v $\mathbf{Z}_2$.

Nebo ještě jinak: provedeme operaci sčítání a násobení jako v případě celých čísel, ale pokud výsledek padne mimo množinu $\{0,1\}$, použijeme zbytek při dělení výsledku číslem 2.

10.5. Příklad. Na množině uspořádaných n-tic prvků ze $\mathbf{Z}_2$, tj. na množině $\mathbf{Z}_2^n$, zavedeme operaci sčítání $+ : \mathbf{Z}_2^n \times \mathbf{Z}_2^n \to \mathbf{Z}_2^n$ analogicky, jako v případě sčítání uspořádaných n-tic reálných čísel, jen samozřejmě pracujeme se sčítáním podle definice 10.3. Pro $(a_1,\ldots,a_n) \in \mathbf{Z}_2^n$ a $(b_1,\ldots,b_n) \in \mathbf{Z}_2^n$ definujeme

$$(a_1,\ldots,a_n)+(b_1,\ldots,b_n) \stackrel{\text{df}}{=} (a_1+b_1,\ldots,a_n+b_n).$$

Například $(1,0,0,1,1)+(1,1,0,0,1)=(0,1,0,1,0)$.

Dále definujeme násobení těchto uspořádaných n-tic jedničkou a nulou přirozeným způsobem:

$$1\cdot(a_1,\ldots,a_n) \stackrel{\text{df}}{=} (a_1,\ldots,a_n), \qquad 0\cdot(a_1,\ldots,a_n) \stackrel{\text{df}}{=} (0,\ldots,0).$$

Povšimněte si analogie s příklady 1.11 a 1.69.

Pokud nahradíme v definici lineárního prostoru 1.6 množinu reálných čísel $\mathbf{R}$ nějakým tělesem, pak takovému lineárnímu prostoru říkáme, že je definován „nad tělesem“ (podrobněji viz definici 1.67). V tomto příkladě jsme zavedli na množině $\mathbf{Z}_2^n$ operace tak, že dostáváme *lineární prostor nad tělesem $\mathbf{Z}_2$*.

Nulový vektor tohoto lineárního prostoru je vektor $(0,\ldots,0)$. Tento lineární prostor má celkem 2^n vektorů, které mezi sebou umíme sčítat a samozřejmě tyto vektory umíme násobit jedničkou nebo nulou.

90. Vysvětlete pojmy kód, kódové slovo, Hammingova velikost, lineární kód.

Kód, kodové slovo: 151) 10.13 aneb:

"Kódování v obecném smyslu zahrnuje (1) algoritmus, kterým informace převádíme do posloupnosti slov (tzv. kodér) a (2) algoritmus, kterým zpětně z těchto slov získáváme původní informaci (dekodér). Slova, která vytváří kodér, se nazývají *kódová slova*. Množina všech kódových slov se nazývá *kód*. Je-li kód množinou slov stejné délky (každé kódové slovo má stejnýž počet znaků abecedy), mluvíme o tzv. *blokovém kódu*. Blokový kód *délky n* značí, že všechna kódová slova mají *n* znaků abecedy."

Hammingova velikost: 152) 10.18 aneb:

"*Hammingova velikost slova* ***u*** je počet jedniček v tomto slově a zanačíme ji $\|\boldsymbol{u}\|$"

Lineární kód: 153) 10.25 aneb:

"Binární blokový kód *K* délky *n* je *lineární*, pokud *K* tvoří lineární podprostor lineárního prostoru **Z2**. Jestli že dimenzi tohoto podprostoru označíme *k*, pak mluvíme o *lineárním (n,k) kódu*."

10.13. Definice. Nechť A je konečná množina (tzv. *abeceda*). Pak *slovo* je libovolná konečná posloupnost prvků z A. *Kód, kódové slovo*

Kódování v obecném smyslu zahrnuje (1) algoritmus, kterým informace převádíme do posloupnosti slov (tzv. *kodér*) a (2) algoritmus, kterým zpětně z těchto slov získáváme původní informaci (*dekodér*).

Slova, která vytváří kodér, se nazývají *kódová slova*. Množina všech kódových slov se nazývá *kód*.

Je-li kód množinou slov stejné délky (každé kódové slovo má stejný počet znaků abecedy), mluvíme o tzv. *blokovém kódu*. Blokový kód *délky n* značí, že všechna kódová slova mají n znaků abecedy.

10.14. Poznámka. Typicky $A = \{0,1\}$, tj. abeceda se skládá jen ze dvou znaků (tzv. bitů, anglicky bits, což je původně zkratka z BInary digiTS) a slova jsou posloupnosti těchto bitů.

10.15. Příklad. ASCII kód je množina 7bitových slov, která reprezentují jednotlivá písmena anglické abecedy a další běžné znaky (číslice, tečku, vykřičník, otazník, mezeru, # neboli vězení atd.). Tato množina obsahuje 91 slov, protože v době vzniku tohoto kódu byl požadavek na kódování 91 znaků. Kodér i dekodér pak pracují s tabulkou těchto znaků, u kterých jsou uvedena odpovídající kódová slova. Tato tabulka může být na straně kodéru technicky realizována třeba ovladačem klávesnice a na straně dekodéru fontem.

Jedná se o blokový kód. Od počátku existence počítačů byl tento blokový kód rozšířen o redundantní nulový bit na začátku, takže často je ASCII kód prezentován jako množina 8bitových slov. Později začal být tento bit využíván pro různá rozšíření ASCII kódu, která zahrnují i reprezentaci některých písmen s diakritickými znaménky.

10.16. Poznámka. Je potřeba si uvědomit, že slova jsou do paměťového média nebo do přenosové linky vkládána za sebou bez oddělovačů. Blokový kód má tu výhodu, že dekodér dokáže snadno rozdělit tento „tok znaků abecedy" na slova a těm pak přidělit význam například pomocí nějaké tabulky. Nevýhoda blokového kódu spočívá v tom, že plýtvá místem, neboť tušíme, že pokud navrhneme pro častěji se vyskytující slova kratší posloupnosti znaků, celkový počet znaků abecedy pro přenášené/ukládané informace může být menší. To ostatně je (alespoň zhruba) i vlastnost přirozeného jazyka. Tam máme ovšem abecedu rozsáhlejší (nebinární) a za prvek abecedy můžeme považovat i mezeru: oddělovač mezi slovy, který dekodéru pomůže. Nebo v Morseově abecedě máme také tři znaky: tečka, čárka a mezera. Bez mezery by bylo dekódování morseovky nemožné. Máme-li k dispozici jen binární abecedu $A = \{0,1\}$, pak je potřeba při návrhu kódu se slovy nestejné délky myslet na možnosti dekodéru. Je to technicky možné, ale není to obsahem tohoto textu. Příkladem neblokového kódu je UTF8, který kóduje znaky abeced všech jazyků světa. Písmena anglické abecedy a běžné znaky jsou reprezentovány 8bitovým slovem, ale písmena dalších jazyků jsou kódována 16bitovým slovem nebo i delším (24 bitů a 32 bitů).

Nadále budeme pracovat jen s blokovými kódy nad binární abecedou.

10.18. Definice. Nechť $A = \{0,1\}$. *Hammingova velikost slova* $\boldsymbol{u} \in A^n$ je počet jedniček v tomto slově a značíme ji $\|\boldsymbol{u}\|$. *Hammingova vzdálenost slov* $\boldsymbol{v} \in A^n$ *a* $\boldsymbol{w} \in A^n$ je počet bitů, ve kterých se tato dvě slova liší. Značíme ji $\mathrm{d}(\boldsymbol{v}, \boldsymbol{w})$.

10.19. Poznámka. Nechť $A = \{0,1\}$ a $\boldsymbol{v}, \boldsymbol{w} \in A^n$. Pak $\boldsymbol{v} + \boldsymbol{w}$ (v aritmetice $\mathbf{Z}_2$) je slovo, které má jedničky právě v místech, kde se $\boldsymbol{v}$ a $\boldsymbol{w}$ liší. Takže platí: $\mathrm{d}(\boldsymbol{v}, \boldsymbol{w}) = \|\boldsymbol{v} + \boldsymbol{w}\|$.

10.20. Poznámka. Předpokládejme, že $\boldsymbol{v}$ je slovo vyslané kodérem a $\boldsymbol{w}$ slovo přijaté dekodérem. Pak $\mathrm{d}(\boldsymbol{v}, \boldsymbol{w})$ udává počet chyb ve slově, které vznikly během přenosu.

Poznámka v poznámce: předpokládáme, že díky technickým parametrům zařízení nikdy nedojde k chybě, kdy se jednička nebo nula ze slova zcela vytratí nebo vznikne nová, tj. nikdy nehrozí riziko, že by na straně dekodéru byl přečten jiný počet jedniček a nul než byl vyslán kodérem.

10.21. Příklad. Je dán tento kód: $K = \{0000, 0011, 0101, 1001, 0110, 1010, 1100, 1111\}$. Jedná se o blokový binární kód délky 4. Pro potřeby tohoto příkladu nebudeme specifikovat druh informace, kterou potřebujeme přenášet. Protože kód obsahuje jen 8 slov, může být původní informace zapsána pomocí nějaké 8 znakové abecedy.

Zajímavé na tomto kódu je, že každé kódové slovo obsahuje sudý počet jedniček. Pokud dojde k jediné chybě ve slovu, máme jistotu, že dekodér přijme nekódové slovo (s lichým počtem jedniček) a ohlásí chybu. Je-li pravděpodobnost výskytu dvou nebo více chyb v jednom slově zanedbatelná a nám postačuje jen detekovat chyby (neopravovat je), je toto rozumný návrh kódu.

Povšimneme si, že minimální Hammingova vzdálenost mezi dvěma různými kódovými slovy tohoto kódu je 2, takže jedna chyba způsobí vytvoření nekódového slova.

10.24. Poznámka. Při návrhu dekodéru s detekcí nebo opravou chyb se s výhodou využijí nástroje lineární algebry, jako je násobení matic, vymezení podprostorů a bází, řešení homogenních soustav atd. Binární slova délky n budeme v tomto případě považovat za vektory z lineárního prostoru $\mathbf{Z}_2^n$, takže je můžeme sčítat. Ostatně, už v poznámce 10.19 jsem zmínil sčítání slov $\boldsymbol{v}$ a $\boldsymbol{w}$. V teorii kódování se binární slova zapisují jedničkami a nulami bez mezer (viz příklady 10.21 a 10.22), zatímco v lineární algebře jsme dosud zapisovali vektory do závorek a jejich složky oddělovali čárkami. Věřím, že nedojde k nedorozumění, pokud dále v textu o kódování budu zapisovat vektory způsobem, jako v příkladu 10.21. *Lineární kód*

10.25. Definice. Binární blokový kód K délky n je *lineární*, pokud K tvoří lineární podprostor lineárního prostoru $\mathbf{Z}_2^n$. Jestliže dimenzi tohoto podprostoru označíme k, pak mluvíme o *lineárním (n,k) kódu*.

10.26. Věta. Nejmenší Hammingova vzdálenost mezi slovy lineárního kódu K je rovna nejmenší Hammingově velikosti nenulového kódového slova.

Důkaz. Stačí si uvědomit, že pro $\boldsymbol{v}_1, \boldsymbol{v}_2 \in K$ je $\|\boldsymbol{v}_1 + \boldsymbol{v}_2\| = \mathrm{d}(\boldsymbol{v}_1, \boldsymbol{v}_2)$, a přitom $\boldsymbol{v}_1 + \boldsymbol{v}_2 \in K$, protože K je lineární kód. Navíc $\|\boldsymbol{v}\| = \mathrm{d}(\boldsymbol{v}, \boldsymbol{o})$.

10.27. Příklad. Kód z příkladu 10.21 je lineární, protože K tvoří podprostor lineárního prostoru $\mathbf{Z}_2^4$. Skutečně, sečteme-li dva vektory se sudým počtem jedniček, dostaneme vektor se sudým počtem jedniček. Výsledek násobení vektoru z K konstantou α zůstane v K, protože v $\mathbf{Z}_2$ číslo α může být jen 0 nebo 1.

Báze kódu z příkladu 10.21 je například $\{0011, 0101, 1100\}$, takže dimenze kódu je 3 a jedná se tedy o *lineární* $(4,3)$ *kód*.

91. Co je a k čemu slouží generující a kontrolní matice lineárního kódu?

Generující a kontrolní matice

10.32. Definice. *Generující matice lineárního kódu K* je po řádcích zapsaná báze tohoto kódu.

Kontrolní matice lineárního kódu K je taková matice $\mathbf{H}$ s lineárně nezávislými řádky, pro kterou platí: množina řešení homogenní soustavy $\mathbf{H}\boldsymbol{x} = \boldsymbol{o}$ je rovna kódu K.

10.33. Věta. Nechť $\mathbf{G}$ je generující matice a $\mathbf{H}$ kontrolní matice lineárního (n,k) kódu. Pak $\mathbf{G}$ má k řádků a $\mathbf{H}$ má $n-k$ řádků. Obě matice mají n sloupců. Jinými slovy, generující matice má tolik řádků, kolik je v kódu informačních bitů, kontrolní matice má tolik řádků, kolik má kód kontrolních bitů a počet sloupců obou matic je roven počtu přenášených bitů v jednom slově.

Důkaz. Matice $\mathbf{G}$ má k řádků, protože báze prostoru dimenze k obsahuje k vektorů. Počet řádků matice H plyne z věty 5.14. Konečně n sloupců obou matic plyne přímo z definice těchto matic.

10.34. Příklad. Kód z příkladu 10.21 může mít následující generující matici:

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

protože $\{1001, 0101, 0011\}$ je báze kódu K. Popíšu, jak se obvykle tato báze sestavuje. Vyjde se ze standardní báze vstupního kódu: $\{100, 010, 001\}$ a aplikuje se na ní zobrazení kodéru. Všechny tři prvky této báze mají lichý počet jedniček, takže poslední kontrolní bit kodér nastaví na jedničku.

Kontrolní matice našeho kódu je

$$\mathbf{H} = (1 \quad 1 \quad 1 \quad 1),$$

protože množina řešení rovnice $x_1 + x_2 + x_3 + x_4 = 0$ je shodná s množinou slov, které mají sudý počet jedniček (sčítáme jedničky modulo 2), a to jsou právě všechna kódová slova.

10.37. Věta. Nechť $\mathbf{G}$ je generující a $\mathbf{H}$ je kontrolní matice lineárního (n,k) kódu. Pak $\mathbf{H} \cdot \mathbf{G}^T = \mathbf{O}_1$ a také $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{O}_2$, kde $\mathbf{O}_1$ je nulová matice s $n-k$ řádky a k sloupci a $\mathbf{O}_2 = \mathbf{O}_1^T$.

Důkaz. Kód s uvedenými maticemi označíme písmenem K. Řádky matice $\mathbf{G}$ alias sloupce matice $\mathbf{G}^T$ jsou podle definice generující matice prvky kódu K. Podle definice kontrolní matice musí tyto sloupce matice $\mathbf{G}^T$ alias prvky kódu K být řešením soustavy $\mathbf{H}\boldsymbol{x} = \boldsymbol{o}$. Přesně to říká vztah $\mathbf{H} \cdot \mathbf{G}^T = \mathbf{O}_1$, pokud jej rozepíšeme po jednotlivých sloupcích matice $\mathbf{G}^T$.

Vztah $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{O}_2$ vzniká transponováním matic na levé i pravé straně vztahu $\mathbf{H} \cdot \mathbf{G}^T = \mathbf{O}_1$.

10.43. Věta. Kód je systematický právě tehdy, když existuje kontrolní matice tohoto kódu tvaru $(\mathbf{C}^T|\mathbf{E}')$, kde $\mathbf{E}'$ je jednotková matice.

Důkaz. Tvrzení věty je důsledkem skutečnosti, že kód má generující matici tvaru $\mathbf{G} = (\mathbf{E}|\mathbf{C})$ právě tehdy, když má kontrolní matici tvaru $\mathbf{H} = (\mathbf{C}^T|\mathbf{E}')$.

10.44. Poznámka. Je-li dána kontrolní matice v jiném tvaru než $(\mathbf{C}^T|\mathbf{E}')$, pak z toho ještě neplyne, že kód není systematický. Eliminací kontrolní matice můžeme získat jinou kontrolní matici, která ovšem přísluší stejnému kódu. Stačí si uvědomit, že eliminací matice soustavy dostáváme případně matici jiné soustavy, ale se stejnou množinou řešení. Pokud tedy po eliminaci kontrolní matice získáme matici tvaru $(\mathbf{C}^T|\mathbf{E}')$, pak je kód systematický.

92. Navrhněte kontrolní a generující matici Hamingova (7,4) kódu a popište proces kódování, dekódování a opravy chyby.

10.56. **Věta.** Slovo s jednou jedničkou je kódové právě tehdy, když kontrolní matice obsahuje nulový sloupec. Dvě různá slova s jednou jedničkou mají společný syndrom právě tehdy, když kontrolní matice obsahuje aspoň dva stejné sloupce.

Důkaz. Stačí si uvědomit, že syndrom slova s jednou jedničkou na i-tém místě je roven i-tému sloupci kontrolní matice. To vyplývá z maticového násobení $\mathbf{H} \cdot \boldsymbol{w}^T = \boldsymbol{s}^T$.

10.57. Poznámka. Aby lineární (n, k) kód opravoval všechny jednoduché chyby ve slově, je podle poznámky 10.55 nutné, aby $n = 2^c - 1$, kde $c = n - k$, a dále podle věty 10.56 je nutné, aby kontrolní matice neměla žádný sloupec nulový a všechny sloupce od sebe vzájemně různé. Těch sloupců musí být $n = 2^c - 1$, a přitom výška sloupce je c. Z toho nám vyplývá jediný možný tvar kontrolní matice (až na pořadí sloupců): v jednotlivých sloupcích kontrolní matice napíšeme ve dvojkové soustavě všechna čísla $1, 2, 3, \ldots, n$. Lineárnímu kódu s takovou kontrolní maticí říkáme *Hammingův kód*.

Hammingův kód

10.58. Příklad. Ukážeme si Hammingův $(7, 4)$ kód. Podle předchozí poznámky napíšeme ve dvojkové soustavě do sloupců kontrolní matice čísla 1,2,3,4,5,6,7:

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

Syndromy a první sloupec tabulky pro opravování chyb napíšeme (kvůli úspoře místa v tomto textu) místo do sloupců do řádků:

000	001	010	011	100	101	110	111
0000000	1000000	0100000	0010000	0001000	0000100	0000010	00000001

Vidíme, že při této volbě pořadí sloupců kontrolní matice má dekodér výrazně usnadněnou práci: nemusí prohledávat v tabulce syndromů, aby zjistil odpovídající chybové slovo. Stačí, aby interpretoval syndrom jako číslo zapsané ve dvojkové soustavě. Toto číslo udává pozici bitu chybového slova, kde se nalézá jednička.

Jak tedy bude dekodér postupovat při obdržení slova $\boldsymbol{w}$? Vypočte syndrom pomocí $\boldsymbol{s}^T = \mathbf{H} \cdot \boldsymbol{w}^T$ a interpretuje jej jako číslo i ve dvojkové soustavě. Je-li $i = 0$, je $\boldsymbol{w}$ kódové slovo a dekodér nic neopravuje. Je-li $i > 0$, pak dekodér opraví v obdrženém slově i-tý bit.

Tím je kompletně navržen dekodér Hammingova $(7, 4)$ kódu a zbývá ještě navrhnout kodér. Eliminací kontrolní matice přecházíme ke kontrolní matici stejného kódu (poznámka 10.44):

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

Podle věty 10.43 se tedy jedná o systematický kód, protože $\mathbf{H}' = (\mathbf{C}^T | \mathbf{E}')$. Podle poznámky 10.41 nyní přejdeme ke generující matici $\mathbf{G} = (\mathbf{E} | \mathbf{C})$:

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

Kodér necháme nejprve kopírovat první čtyři informační bity do výstupu a další tři kontrolní bity $\boldsymbol{v}'$ počítáme ze vstupního slova $\boldsymbol{u}$ podle poznámky 10.47:

$$\boldsymbol{v}' = \boldsymbol{u} \cdot \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

10.59. Poznámka. Analogicky se postupuje při návrhu $(15, 11)$, $(31, 26)$, $(63, 57)$ atd. Hammingových kódů. Všimněte si, že s rostoucím n se výrazně zlepšuje poměr informačních bitů ku celkovému počtu přenášených bitů. To je pro uživatele, kteří se zajímají jen o informační bity, dobrá zpráva. Ovšem prodlužováním délky přenášených slov se zase zvyšuje pravděpodobnost výskytu více než jedné chyby ve slově. Dekodér Hammingova kódu v takovém případě selže.

10.60. Poznámka. Ve výpočetní technice se pracuje s přenosy 8 bitů, 16 bitů, 32 bitů atd. Hammingův kód předpokládá přenos slov délky o jeden bit kratší. Co se zbylým bitem? Použijeme jej pro kontrolu parity. Tím dostáváme *rozšířený Hammingův kód*, který umožní spolehlivě opravit jednu chybu a detekovat chyby dvě.

Rozšířený Hammingův kód

10.61. Příklad. K Hammingovu kódu $(7,4)$ přidáme kontrolní bit parity a dostáváme lineární $(8,4)$ kód s následující kontrolní maticí:

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

Tento kód umí opravit jednu chybu ve slově a detekovat chyby dvě. Jak dekodér může postupovat? Přijme slovo $\boldsymbol{w}$ a vypočte syndrom $\boldsymbol{s}^T = \mathbf{H} \cdot \boldsymbol{w}^T$. Je-li syndrom nulový vektor, je slovo $\boldsymbol{w}$ kódové a dekodér nic neopravuje. Jsou-li první tři bity syndromu nulové a poslední nenulový, došlo při přenosu jen k chybě posledního kontrolního bitu parity. Je-li na prvních třech pozicích syndromu aspoň jeden bit nenulový a poslední bit syndromu je rovněž nenulový, došlo k lichému počtu chyb ve slově. Dekodér předpokládá, že došlo k jediné chybě a podle prvních třech bitů syndromu zjistí, který bit ve slově má opravit (stejně jako v příkladu 10.58). Je-li konečně poslední bit syndromu nulový, ale syndrom obsahuje aspoň jeden nenulový bit, pak došlo k sudému počtu chyb. Tento počet chyb neumí dekodér opravit, ale detekuje tento stav jako dvojnásobnou chybu.

10.62. Poznámka. Všimněte si, že nejmenší Hammingova vzdálenost mezi dvěma slovy rozšířeného Hammingova kódu je 4. To je v souladu s výsledky příkladu 10.23.

Dekodér lineárního kódu

10.48. Poznámka. Dekodér při kontrole, zda se jedná o kódové slovo, použije kontrolní matici. Nechť dekodér přijme slovo $\boldsymbol{w}$. Pak $\mathbf{H} \cdot \boldsymbol{w}^T$ je nulový vektor právě tehdy, když je slovo $\boldsymbol{w}$ kódové. V takovém případě dekodér předpokládá, že nedošlo při přenosu slova k chybě, odstraní kontrolní bity a tím získá původní informaci.

Pokud $\mathbf{H} \cdot \boldsymbol{w}^T$ není nulový vektor, dekodér má jistotu, že došlo k chybě a že $\boldsymbol{w}$ není kódové slovo. Má-li chybu opravit, pak údaj $\mathbf{H} \cdot \boldsymbol{w}^T$ bude při opravě potřebovat. Napíšeme-li výsledek násobení $\mathbf{H} \cdot \boldsymbol{w}^T$ do řádku, dostáváme tzv. *syndrom* vektoru $\boldsymbol{w}$.

10.49. Definice. Nechť $\mathbf{H}$ je kontrolní matice lineárního kódu. *Syndrom* slova $\boldsymbol{w}$ je vektor $\boldsymbol{s}$, pro který platí $\boldsymbol{s}^T = \mathbf{H} \cdot \boldsymbol{w}^T$.

Nechť $\boldsymbol{v}$ je slovo vyslané kodérem a $\boldsymbol{w}$ je slovo přijaté dekodérem. Pak $\boldsymbol{e} = \boldsymbol{w} - \boldsymbol{v}$ je *chybové slovo*. Protože v $\mathbf{Z}_2^n$ je $-\boldsymbol{v} = \boldsymbol{v}$, chybové slovo lze počítat jako $\boldsymbol{w} + \boldsymbol{v}$.

10.50. Poznámka. Jedničkové bity chybového slova označují místa, kde došlo k poškození slova $\boldsymbol{v}$. Úkolem dekodéru je na základě znalosti $\boldsymbol{w}$ zjistit chybové slovo $\boldsymbol{e}$. Pokud se mu to podaří, pak vypočte původní informaci jako $\boldsymbol{v} = \boldsymbol{w} - \boldsymbol{e}$.

Než se pustíme do sestavování tabulky, podle které bude dekodér opravovat chyby, je potřeba si uvědomit platnost dvou tvrzení. První z nich platí dokonce obecně na libovolném lineárním prostoru.

10.51. Věta. Nechť K je lineární podprostor lineárního prostoru L a nechť $\boldsymbol{e}_1 \in L$, $\boldsymbol{e}_2 \in L$. Pak množiny $M_1 = \{\boldsymbol{e}_1 + \boldsymbol{v};\ \boldsymbol{v} \in K\}$, $M_2 = \{\boldsymbol{e}_2 + \boldsymbol{v};\ \boldsymbol{v} \in K\}$ jsou buď disjunktní nebo totožné.

Důkaz. Sporem. Předpokládáme, že množiny M_1 a M_2 mají společný bod $\boldsymbol{a}$ a přitom nejsou totožné, tj. existuje vektor $\boldsymbol{b} \in M_1$, který neleží v M_2. Protože $\boldsymbol{a}$ i $\boldsymbol{b}$ leží v množině M_1, je $\boldsymbol{a} = \boldsymbol{e}_1 + \boldsymbol{u}$, $\boldsymbol{b} = \boldsymbol{e}_1 + \boldsymbol{v}$, kde $\boldsymbol{u}$ i $\boldsymbol{v}$ leží v K. Pak $\boldsymbol{w} = \boldsymbol{b} - \boldsymbol{a} = \boldsymbol{v} - \boldsymbol{u}$ leží v K, protože K je podprostor. Je tedy $\boldsymbol{b} = \boldsymbol{a} + \boldsymbol{w}$. Protože $\boldsymbol{a}$ leží i v množině M_2, je $\boldsymbol{a} = \boldsymbol{e}_2 + \boldsymbol{x}$, kde $\boldsymbol{x} \in K$. Dosadíme-li tento poznatek do vztahu pro $\boldsymbol{b}$, dostaneme $\boldsymbol{b} = \boldsymbol{e}_2 + \boldsymbol{x} + \boldsymbol{w}$. Protože K je podprostor, $\boldsymbol{x} + \boldsymbol{w}$ leží v K. Je tedy $\boldsymbol{b} = \boldsymbol{e}_2 + \boldsymbol{z}$, kde $\boldsymbol{z} \in K$. To ale znamená, že $\boldsymbol{b} \in M_2$, což je sporu s předpokladem.

10.52. Věta. Nechť $\boldsymbol{v}$ je kódové slovo a $\boldsymbol{e}$ je libovolné slovo. Pak slova $\boldsymbol{e}$ i $\boldsymbol{e} + \boldsymbol{v}$ mají stejný syndrom. Jinými slovy kódová slova modifikovaná stejnou chybou vytvářejí skupinu slov se společným syndromem.

Důkaz. $\mathbf{H} \cdot (\boldsymbol{e} + \boldsymbol{v})^T = \mathbf{H} \cdot \boldsymbol{e}^T + \mathbf{H} \cdot \boldsymbol{v}^T = \mathbf{H} \cdot \boldsymbol{e}^T + \boldsymbol{o}^T = \mathbf{H} \cdot \boldsymbol{e}^T$.