Diskrétní matematika – 8. týden
Lineární kódy

Lukáš Vokřínek

Masarykova univerzita
Fakulta informatiky

podzim 2020

Obsah přednášky

Doporučené zdroje

- Jan Slovák, Martin Panák, Michal Bulant
Matematika drsně a svižně, *e*-text na
`www.math.muni.cz/Matematika_drsne_svizne`.

Doporučené zdroje

- Jan Slovák, Martin Panák, Michal Bulant
Matematika drsně a svižně, *e*-text na
`www.math.muni.cz/Matematika_drsne_svizne`.
- W. J. Gilbert, W. K. Nicholson, Modern algebra with applications, 2nd ed. John Wiley and Sons (Pure and applied mathematics) ISBN 0-471-41451-4

Plán přednášky

1. (n, k)–kódy
2. Polynomiální kódy
3. Lineární kódy

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o k bitech.

$\mathbb{Z}_2 = \{0, 1\}$

{sudá čísla, lichá čísla}

(zbytkové třídy (mod 2))

zpráva $b_0 b_1 \ldots b_{n-1}$

pošleme $\underbrace{??}\ \underbrace{b_0 b_1 \ldots b_{n-1}}_{\text{lze extrahovat}}$

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o k bitech.

Přenosové chyby chceme

1. rozpoznávat
2. opravovat

a za tím účelem přidáváme dodatečných $n - k$ bitů informace pro pevně zvolené $n > k$. Mluvíme pak o (n, k)-*kódu*.

$$\underbrace{b_0 \cdots b_{k-1}}_{k}$$

$$\underbrace{c_0 \cdots c_{n-k} \; b_0 \cdots b_{k-1}}_{n}$$

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o k bitech.

Přenosové chyby chceme

$0 \qquad k-1$

$0/1 \; --- \; 0/1$

1. rozpoznávat
2. opravovat

a za tím účelem přidáváme dodatečných $n-k$ bitů informace pro pevně zvolené $n > k$. Mluvíme pak o (n, k)-*kódu*.

Všech slov o k bitech je 2^k a každé z nich má jednoznačně určovat jedno **kódové slovo** z 2^n možných. Máme tedy ještě

$2^k \qquad 2^n$

$$2^n - 2^k = 2^k(2^{n-k} - 1)$$

slov, které jsou chybové. Lze tedy tušit, že pro veliké k nám i malý počet přidaných bitů dává hodně redundantní informace.

$+1 \equiv -1 \pmod 2$

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.

kontrolní bity informační bity

$c\, b_0 b_1 \cdots b_{k-1}$

bit t.ž. jedniček je celkově sudý počet

$c + b_0 + b_1 + \cdots + b_{k-1} = 0 \pmod 2$

$c = \underbrace{b_0 + b_1 + \cdots + b_{k-1}} \pmod 2$

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.
Pokud při přenosu dojde k lichému počtu chyb, přijdeme na to.
Dvě různá kódová slova se při tomto kódu vždy liší alespoň ve dvou pozicích, chybové slovo se ale od dvou různých kódových slov liší pouze v pozici jedné. Nemůžeme proto umět chyby opravovat ani kdybychom věděli, že došlo k právě jedné.

00 11

nelze

01 došlo k chybě = není kódové slovo

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.
Pokud při přenosu dojde k lichému počtu chyb, přijdeme na to.
Dvě různá kódová slova se při tomto kódu vždy liší alespoň ve dvou pozicích, chybové slovo se ale od dvou různých kódových slov liší pouze v pozici jedné. Nemůžeme proto umět chyby opravovat ani kdybychom věděli, že došlo k právě jedné.
Navíc neumíme detekovat tak obvyklé chyby, jako je záměna dvou sousedních hodnot ve slově.

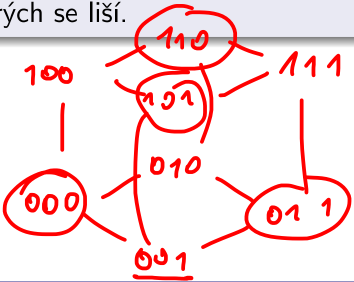
01 $\longrightarrow$ 10

Definice

Hammingova vzdálenost dvou slov je rovna počtu bitů, ve kterých se liší.

Definice

Hammingova vzdálenost dvou slov je rovna počtu bitů, ve kterých se liší.



žádné nekódové slovo nesousedí s ≥ 2 kódovými
$\Rightarrow$ umíme opravit jedn. chybu
$\Leftrightarrow$ vzd. kód. slov ≥ 3

Věta

1. *Kód odhaluje r a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov právě* $r + 1$.
2. *Kód opravuje r a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov ~~právě~~* $2r + 1$.

alespoň $2r+1$ / $2r+2$

kód r 2r+1 kód

Plán přednášky

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.

$0 \longrightarrow 000$

$1 \longrightarrow 111$

001, 010, 100

$000 \overset{}{\underset{3}{\text{———}}} 111$

umíme opravit jedn. chyby

Jak konstruovat kódová slova, abychom je snadno rozpoznali?
Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.

$(b_0 + b_1 + \dots + b_{k-1})\ b_0 b_1 \dots b_{k-1}$

Docela systematickou cestou je využití dělitelnosti polynomů.
Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom
$m(x) = b_0 + b_1 x + \dots + b_{k-1} x^{k-1} \in \mathbb{Z}_2[x]$.

koef. jsou zb. třídy mod 2

$(1+x) + (1+x^2) = x + x^2$

$110 + 101 = 011$

nepřenáší se
ne +
ale XOR

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3,1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.

Docela systematickou cestou je využití dělitelnosti polynomů. Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom (n,k)

$m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1} \in \mathbb{Z}_2[x]$.

Definice

Nechť $p(x) = a_0 + \cdots + a_{n-k} x^{n-k} \in \mathbb{Z}_2[x]$ je polynom s $a_0 = 1$, $a_{n-k} = 1$. **Polynomiální kód generovaný polynomem** $p(x)$ je (n,k)–kód jehož slova jsou polynomy stupně ~~menšího než~~ n $\leq n-1$ dělitelné $p(x)$.

$p(x) = 1 + x$ kód. sl. $(1+x)(1+x)$

$= 1 + x^2$

mod 2

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.
Docela systematickou cestou je využití dělitelnosti polynomů.
Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom $m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1} \in \mathbb{Z}_2[x]$.

Definice

Nechť $p(x) = a_0 + \cdots + a_{n-k} x^{n-k} \in \mathbb{Z}_2[x]$ je polynom s $a_0 = 1$, $a_{n-k} = 1$. **Polynomiální kód generovaný polynomem** $p(x)$ je (n, k)–kód jehož slova jsou polynomy stupně menšího než n dělitelné $p(x)$.
Zpráva $m(x)$ je zakódována jako $v(x) = r(x) + x^{n-k} m(x)$, kde $r(x)$ je zbytek po dělení polynomu $x^{n-k} m(x)$ polynomem $p(x)$.

x^0 $\quad$ x^{k-1}
$b_0 \; b_1 \; --- \; b_{k-1}$ $\qquad$ $m(x)$
x^{n-k} $\quad$ x^{n-1}
$(\quad) \; b_0 \; b_1 \cdots b_{k-1}$ $\qquad$ $? + x^{n-k} \, m(x)$

$m(x)$ kódování

$r(x)$ stupně $n-k-1$

$a_0 + a_1 x + \cdots + a_{n-k} x^{n-k}$

$r(x) + x^{n-k} m(x)$ dělitelné $p(x)$

zbytek po dělení $p(x)$ je

$r(x)$ + zbytek $x^{n-k} m(x) = 0$ (mod 2)

$r(x)$ = zbytek $x^{n-k} \cdot m(x)$ po dělení $p(x)$

$r(x) + x^{n-k} \cdot m(x)$

Z definice víme

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.

Z definice víme

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.
Původní zpráva je obsažena přímo v polynomu $v(x)$, takže dekódování správného slova je snadné.

$b_0 b_1 \cdots b_{k-1}$

$\_\,\_\,\_\,\_\,\_\,.\, b_0 b_1 \cdots b_{k-1}$

Z definice víme

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.
Původní zpráva je obsažena přímo v polynomu $v(x)$, takže dekódování správného slova je snadné.

Příklad

$(k+1, k)$

1. Polynom $p(x) = 1 + x$ generuje $(\cancel{n, n -} 1)$–kód kontroly parity pro všechna $n \geq 3$.
2. Polynom $p(x) = 1 + x + x^2$ generuje $(3, 1)$–kód opakování bitů.

$0 \rightarrow 0 \quad 000$

$1 \xrightarrow{x^2+x+\dots} 1+x+x^2 \quad 111$

První tvrzení plyne z toho, že $1 + x$ dělí polynom $v(x)$ tehdy a jen tehdy, když $v(1) = 0$ a to nastane tehdy, když je ve $v(x)$ sudý počet nenulových koeficientů. Druhé je zřejmé.

$b_0 b_1 \dots b_{k-1}$

$? \rightarrow b_0 \dots \rightarrow b_{k-1}$

kdy je $n(x)$ děl. $1+x$?

$n(x) = q(x) \cdot (x+1) + n(1)$

$=$ součet koef. polynomu $n(x)$

$\Leftrightarrow$ $n(x)$ má sudý počet jedniček

$b_0 b_1 \cdots b_{k-1}$

$c\ b_0 b_1 \cdots b_{k-1}$

↳ tak aby byl sudý počet 1

Přenos slova $v \in \mathbb{Z}_2[x]$ dopadne příjmem polynomu

kódové kódové tedy $e(x) = u(x) + v(x)$

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.

$p(x) \mid u(x)$

$\Updownarrow$ $p(x) \mid v(x)$

$p(x) \mid e(x)$

malá chyba

jednoduchá: $e(x) = x^l$

dvojitá: $e(x) = x^m + x^{m+l} = x^m(1 + x^l)$

$p(x) \nmid x^i \Rightarrow$ umíme rozpoznat jed. ch.

$p(x) \nmid x^i$ a $p(x) \nmid 1 + x^l$ pro $1 \leq l \leq n-1$

Přenos slova $v \in \mathbb{Z}_2[x]$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.
Chyba je rozpoznatelná pouze, když generátor kódu $p(x)$ nedělí $e(x)$. Máme proto zájem o polynomy, které nevystupují jako dělitelé zbytečně často.

Přenos slova $v \in \mathbb{Z}_2[x]$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.
Chyba je rozpoznatelná pouze, když generátor kódu $p(x)$ nedělí $e(x)$. Máme proto zájem o polynomy, které nevystupují jako dělitelé zbytečně často.

nelze rozložit na součin

Definice

Ireducibilní polynom $p(x) \in \mathbb{Z}_2[x]$ stupně m se nazývá **primitivní**, jestliže $p(x)$ dělí polynom $(1 + x^\ell)$ pro $\ell = 2^m - 1$ ale nedělí jej pro žádná menší ℓ.

pro $n = 2^m - 1$ funguje

nedělí $1+x$, $1+x^2$, ..., $1+x^{2^m-2}$ dělí až $1+x^{2^m-1}$

Věta

Je-li $p(x)$ primitivní polynom stupně m, pak pro všechna $n \leq 2^m - 1$ rozpoznává příslušný $(n, n-m)$–kód všechny jednoduché a dvojité chyby.

Věta

Je-li $p(x)$ primitivní polynom stupně m, pak pro všechna $n \leq 2^m - 1$ rozpoznává příslušný $(n, n-m)$–kód všechny jednoduché a dvojité chyby. ⇒ opravit jedn. ch.

Důsledek

Je-li $q(x)$ primitivní polynom stupně m, pak pro všechna $n \leq 2^m - 1$ rozpoznává $(n, n-m-1)$–kód generovaný polynomem $p(x) = q(x)(1+x)$ všechny dvojité chyby a všechna slova s lichým počtem chyb. 1,2,3 ale opravit stále pouze jedn.

Tabulka dává o informace o výsledcích předchozích dvou vět pro několik polynomů:

Tabulka dává o informace o výsledcích předchozích dvou vět pro několik polynomů:

primitivní polynom	kontrolní bity	délka slova
$1 + x + x^2$	2	3
$1 + x + x^3$	3	7
$1 + x + x^4$	4	15
$1 + x^2 + x^5$	5	31
$1 + x + x^6$	6	63
$1 + x^3 + x^7$ $m=7$	7	127 $n = 2^7 - 1$
$1 + x^2 + x^3 + x^4 + x^8$	8	255
$1 + x^4 + x^9$	9	511
$1 + x^3 + x^{10}$	10	1023

1013 $\xrightarrow{\text{Lod.}}$ 1023

Nástroje pro konstrukci primitivních polynomů dává teorie konečných polí. Souvisí s tzv. primitivními prvky v Galoisových polích $G(2^m)$.

$\underline{\underline{p=2}}$

$\to \mathbb{Z}_p$ zb. tř. mod p

$\to \mathbb{F}_{p^m} \ni g$ je kořen nej. polynomu

$\parallel$

$\mathbb{Z}_p^{\,m}$

$\underline{\underline{p(x)}} \mid \underbrace{x^{p^m-1}-1}_{g \text{ kořen}}$

st. m

Nástroje pro konstrukci primitivních polynomů dává teorie konečných polí. Souvisí s tzv. primitivními prvky v Galoisových polích $G(2^m)$.
Ze stejné teorie lze také dovodit příjemnou realizaci dělení se zbytkem (tj.) ověřování, zda je přijaté slovo kódové, pomocí zpožďovacích registrů. Jde o jednoduchý obvod s tolika prvky, kolik je stupeň polynomu.

Plán přednášky

1. (n, k)–kódy
2. Polynomiální kódy
3. Lineární kódy

$b_0 b_1 \dots b_{k-1}$ — $b_0 + b_1 x + \dots + b_{k-1} x^{k-1}$

— $(b_0, b_1, \dots, b_{k-1}) \in (\mathbb{Z}_2)^k$

$$\begin{array}{r} 01 \\ 11 \\ \hline 10 \end{array}$$

Definice

Lineární kód je injektivní lineární zobrazení $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$. Matice G typu n/k reprezentující toto zobrazení v standardních bazích se nazývá generující **matice kódu**.

kódová slova = součty sloupců matice G

$$g(x+y) = g(x) + g(y)$$

$$g(x) = G \cdot x \qquad \text{mod } 2$$

matice n/k

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

Definice

Lineární kód je injektivní lineární zobrazení $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$. Matice G typu n/k reprezentující toto zobrazení v standardních bazích se nazývá generující **matice kódu**.

Pro každé slovo u je

$$v = G \cdot u$$

příslušné kódové slovo.

Věta

Každý polynomiální (n, k)–kód je lineární kód.

$u(x) \longrightarrow r(x) + x^{n-k} u(x)$ děl. $p(x)$

$v(x) \longrightarrow s(x) + x^{n-k} v(x)$ děl. $p(x)$

$u(x) + v(x) \xrightarrow{?} r(x) + s(x) + x^{n-k}(u(x) + v(x))$ –||–

$\checkmark$

Věta

Každý polynomiální (n, k)–kód je lineární kód.

Matice příslušná k polynomu $p(x) = 1 + x + x^3$ a jím určenému $(7, 4)$–kódu je

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

$1000 \longleftrightarrow 1$

$\cdot x^3 + $ zl.

$1101000 \longleftrightarrow 1+x+x^3$

$1101000 \longleftrightarrow 1+x+x^3$

1. sl.

$1101000 \longleftrightarrow 1+x+x^3$

$0110100 \longleftrightarrow x+x^2+x^4$

2. sl.

$0011010 \longleftrightarrow x^2+x^3+x^5$

3. sl.

$1110010 \longleftrightarrow 1+x+x^2+x^5$

$0111001 \longleftrightarrow x+x^2+x^3+x^6$

4. sl.

$1010001 \longleftrightarrow 1+x^2+x^6$

Věta

Je-li $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$ lineární kód s (blokově zapsanou) maticí

$$G = \begin{pmatrix} P \\ \mathbb{I}_k \end{pmatrix},$$

$$\begin{array}{c} P \\ \\ \\ \mathbb{I}_4 \\ \\ \\ \end{array} \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ \hline 1 & & & 0 \\ & 1 & & \\ & & 1 & \\ 0 & & & 1 \end{pmatrix}$$

$$G \cdot u = \begin{pmatrix} P \cdot u \\ \hline u \end{pmatrix} \begin{array}{l} \text{kontr} \\ \text{info} \end{array}$$

potom zobrazení $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^{n-k}$ s maticí

$$H = \begin{pmatrix} \mathbb{I}_{n-k} & P \end{pmatrix}$$

Kdy je $v = \begin{pmatrix} w \\ \hline u \end{pmatrix} \begin{array}{l} \text{kontr.} \\ \text{info} \end{array}$ kódové slovo?

$\iff w = P \cdot u$

má následující vlastnosti

1. $\operatorname{Ker} h = \operatorname{Im} g$
2. *Přijaté slovo $v = G \cdot u$ je kódové slovo právě, když je $H \cdot v = 0$.*

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

$$\qquad \mathbb{I}_3 \qquad\qquad P$$

$$\iff (\mathbb{I} \ \ P) \begin{pmatrix} w \\ u \end{pmatrix} = w + P u = 0$$

Věta

Je-li $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$ *lineární kód s (blokově zapsanou) maticí*

$$G = \begin{pmatrix} P \\ \mathbb{I}_k \end{pmatrix},$$

potom zobrazení $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^{n-k}$ *s maticí*

$$H = \begin{pmatrix} \mathbb{I}_{n-k} & P \end{pmatrix}$$

má následující vlastnosti

1. $\operatorname{Ker} h = \operatorname{Im} g$
2. *Přijaté slovo* $v = G \cdot u$ *je kódové slovo právě, když je* $H \cdot v = 0$.

Matici H z věty se říká **matice kontroly parity** přílušného (n, k)–kódu.

Jak jsme viděli, přenos zprávy u dává výsledek

$$v = u + e,$$

kde ale neznáme u, e a hledáme takový "rozklad", kde e obsahuje co nejméně jedniček (oprava chyby za předpokladu co nejmenšího počtu chyb).

Jak jsme viděli, přenos zprávy u dává výsledek

$$v = u + e,$$

kde ale neznáme u, e a hledáme takový “rozklad”, kde e obsahuje co nejméně jedniček (oprava chyby za předpokladu co nejmenšího počtu chyb).

kontr

Je-li $v = \begin{pmatrix} x \\ \hdashline y \end{pmatrix}$, pak jednou z možností na odeslanou zprávu je

info

$Gy = \begin{pmatrix} Py \\ y \end{pmatrix}$ (ne nutně optimální), tedy

obdrženo posláno? chyba?

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} Py \\ y \end{pmatrix} + \begin{pmatrix} x + Py \\ 0 \end{pmatrix}$$

Jak jsme viděli, přenos zprávy u dává výsledek

$$v = u + e,$$

kde ale neznáme u, e a hledáme takový "rozklad", kde e obsahuje co nejméně jedniček (oprava chyby za předpokladu co nejmenšího počtu chyb).

Je-li $v = \begin{pmatrix} x \\ y \end{pmatrix}$, pak jednou z možností na odeslanou zprávu je $Gy = \begin{pmatrix} Py \\ y \end{pmatrix}$ (ne nutně optimální), tedy

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} Py \\ y \end{pmatrix} + \begin{pmatrix} x + Py \\ 0 \end{pmatrix}$$

$H \cdot v$

kde $s = x + Py$ je právě syndrom slova v a jedná se tedy o chybu za předpokladu, že k ní došlo pouze na kontrolních bitech (z informačních bitů lze proto přečíst původní slovo).

$$\begin{pmatrix} x \\ y \end{pmatrix} = \underline{\begin{pmatrix} Py \\ y \end{pmatrix}} + \begin{pmatrix} x+Py \\ 0 \end{pmatrix}$$

kódy

Protože kódová slova jsou právě součty sloupců matice G, lze všechny další možnosti obdržet přičítáním sloupců g_i ke kódovému slovu i chybě:

jeden

$$\begin{pmatrix} x \\ y \end{pmatrix} = \left(\begin{pmatrix} Py \\ y \end{pmatrix} + g_i \right) + \left(\begin{pmatrix} x + Py \\ 0 \end{pmatrix} + g_i \right)$$

atd.

Protože kódová slova jsou právě součty sloupců matice G, lze všechny další možnosti obdržet přičítáním sloupců g_i ke kódovému slovu i chybě:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \underbrace{\left(\begin{pmatrix} Py \\ y \end{pmatrix} + g_i \right)}_{} + \underbrace{\left(\begin{pmatrix} x + Py \\ 0 \end{pmatrix} + g_i \right)}_{}$$

$$\begin{array}{cc} & k \\ n & \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

atd.

Přitom přičtení každého sloupce vyrobí jednu 1 v informačních bitech chyby, snažíme se jejich počet kompenzovat snížením počtu 1 v kontrolních bitech. Toto budeme zkoušet pouze pro malý počet chyb, viz cvičení.

↳ jednoduché / dvojité

$$\frac{k(k-1)}{2}$$