elektrorevue
ISSN 1213 - 1539

Rok / Year:	Svazek / Volume:	Číslo / Number:	Jazyk / Language
2022	22	6	CZ

Program pro výuku a simulaci kanálového kódování

Software for education and simulation of channel coding

Radek Závorka

xzavor03@stud.feec.vutbr.cz

Fakulta elektrotechniky a komunikačních technologií VUT v Brně

DOI: -

Abstract: This article deals with the topic of channel coding and comparison of channel codes. Description of the design of the software, which demonstrates principles of channel codes and simulation of channel coding, is included. Article focuses on the chosen types of codes from the easiest to the more complex ones, which are used in real practise, e.g. Hamming code, cyclic code, convolutional code and LDPC code. These codes are compared in terms of bit error rate depending on the signal to noise ratio. Described educational/simulation software should help with easier understanding of the encoding/decoding process and principles and functions of chosen channel codes.

elektrorevue
ISSN 1213 - 1539

Program pro výuku a simulaci kanálového kódování

Radek Závorka

Fakulta elektrotechniky a komunikačních technologií VUT v Brně
Email: xzavor03@stud.feec.vutbr.cz

Abstrakt *– Tento článek se zabývá problematikou kanálového kódování a porovnáním jednotlivých kódů. V článku je rovněž popsán program, který je zaměřen na demonstraci principu funkce jednotlivých kódů a na simulaci kanálového kódování. Pozornost je zaměřena na kanálové kódy běžně využívané v praxi, ty jsou postupně popsány od jednodušších po komplexní: Hammingův kód, cyklický kód, konvoluční kód a LDPC kód. Uvedené kódy jsou srovnány z hlediska bitové chybovosti v závislosti na poměru signálu k šumu. Vytvořený program pro demonstraci kanálových kódů slouží k seznámení s problematikou kódování/dekódování a umožňuje snáze pochopit principy a funkce jednotlivých kódů.*

1 Úvod

V reálném přenosovém řetězci působí na přenášená data mnoho nežádoucích vlivů, které způsobují chyby v přenosu. Jedná se o přídavný šum, mnohacestná šíření, přeslechy a další. Cílem kanálového kódování je zabezpečit data proti chybám při přenosu. Existuje mnoho metod a kódů, jež umí chybu po průchodu přenosovým kanálem nejen detekovat, ale dokonce i opravit. Podle toho se kódy nazývají detekční nebo korekční. [1]

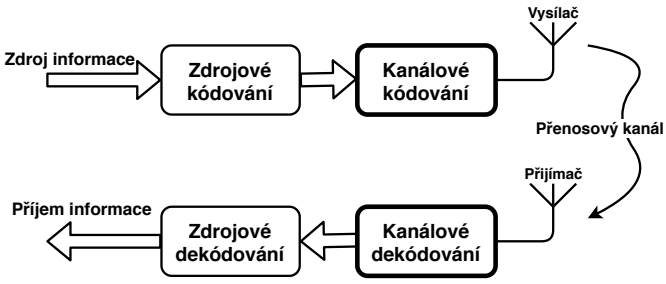
Jednotlivým kódům jsou věnovány samostatné kapitoly, kde je stručně popsaná teorie vysvětlující jejich funkci.

Kapitola 3 je věnována porovnání diskutovaných kódů s ohledem na bitovou chybovost (BER) v závislosti na poměru signálu k šumu (SNR).

Závěrečná kapitola se zabývá popisem vytvořeného programu v programovacím prostření Matlab, jehož účel je zamýšlen jako výukový program. Uživatelské prostředí je navrženo tak, aby jeho používání bylo snadné a intuitivní. Po zadání vstupního vektoru dat je možno informační slovo kódovat, následně provést simulaci průchodu přenosovým kanálem a na závěr stiskem příslušného tlačítka provést dekódování s vyhodnocením správnosti přenosu.

2 Teorie kanálových kódů

V této části jsou popsány jednotlivé kanálové kódy uvedeny výše, které jsou také obsaženy ve vytvořeném programu. Na obrázku 1 je zjednodušené blokové schéma komunikačního kanálu obsahujícího vysílač s kanálovým kodérem, přenosový kanál, kde může dojít k chybě v přenášené informaci a přijímač s kanálovým dekodérem. Řada kódů



Obrázek 1: Blokové schéma zjednodušeného komunikačního kanálu.

pracuje v principu na kódování a dekódování pomocí generující a kontrolní matice, jejíž odvození se pro jednotlivé kódy liší a obecný tvar lze nalézt např. v [2]. Potom se označují jako kódy (n, k), kde k značí počet informačních bitů a n počet bitů zakódované zprávy. Obecný tvar systematické generující matice uvádí rovnice (1).

$$G = [I_k P] = \begin{bmatrix} 1 & 0 & \dots & 0 & p_{11} & p_{12} & \dots & p_{1,(n-k)} \\ 0 & 1 & \dots & 0 & p_{21} & p_{22} & \dots & p_{2,(n-k)} \\ \vdots & & & \vdots & \vdots & & & \vdots \\ 0 & 0 & \dots & 1 & p_{k1} & p_{k2} & \dots & p_{k,(n-k)} \end{bmatrix}, \tag{1}$$

kde I_k je jednotková matice velikosti $k \times k$ a P je část matice zabezpečující kontrolu parity velikosti $k \times (n-k)$ [3]. Z toho vyplývá, že při vytváření kódových slov dochází k určité redundaci, která je nutná pro detekci chyby a její opravu. Počet opravitelných chyb se liší v závislosti na typu použitého kódu a jeho složitosti.

2.1 Hammingův kód

Jedná se o nesystematický blokový kód se schopností opravy jednonásobných chyb. V modifikované podobě umožňuje detekci až dvojnásobných chyb [4]. Kódování a dekódování je založeno na využití generující a kontrolní matice, kdy postup odvození těchto matic lze nalézt např. v [5]. Kódování zprávy je poté relativně jednoduché a provede se vynásobením informačního slova s generující (G) maticí, vycházející z množiny konečných těles GF(2) (Galois Field), podrobný popis GF lze nalézt např. v [6] :

$$c = m.G, \tag{2}$$

elektrorevue
ISSN 1213 - 1539

kde c je výsledná zakódovaná zpráva, m informační slovo a G generující matice.

Dekódování probíhá obdobně, vynásobením kódového přijatého slova s kontrolní maticí:

$$s = c.H^T, \tag{3}$$

kde s značí hodnotu syndromu a H kontrolní matici. Kontrolní matice (4) pro rozměr kódu (7, 4) je poskládána způsobem, aby pořadí chyby odpovídalo pozici v zakódované zprávě.

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{4}$$

Hodnota syndromu určuje, zda při přenosu došlo k chybě a v případě, že není nulový, určuje přímo pozici chyby (jedná-li se o chybu jednonásobnou). Chybu je tak možno přímo opravit a z přijaté zprávy vyčíst informační bity vyskytující se na pozicích vyjma mocniny dvojky.

2.2 Cyklický kód

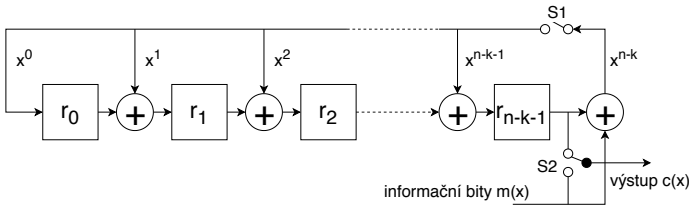
Tento kód se nejčastěji používá jako CRC (cyclic redundancy check) doplněk pro kontrolu správnosti přenosu. Výhoda je v jednoduchém sestavení kódovacího/dekódovacího obvodu a dobré efektivnosti kódování. Rozměr kódu se opět označuje (n, k). Informační slovo se obvykle zadává ve formě koeficientů polynomu $(m_{k-1}, m_{k-2}, m_{k-3}, \ldots, m_0)$, se kterými se nám později lépe pracuje. Stejným způsobem se značí kódové slovo a platí, že cyklický posuv doprava nebo doleva vytvoří opět platné kódové slovo z podprostoru kódových slov. Jednotlivé prvky kódového slova jsou brány z množiny konečných těles GF(2). [1]

Pro kódování/dekódování se využívá generující polynom $g(x)$, který musí mít následující vlastnosti:

- Stupeň mnohočlenu je $n - k$,
- Dělí beze zbytku dvojčlen $x^n - 1$,
- Pro opravu jednonásobných chyb musí být $g(x)$ alespoň trojčlenem a z množiny násobků $g(x)$ je nutno vyloučit všechny dvojčleny, aby minimální kódová vzdálenost byla $d_{min} > 2$ [7].

Systematické kódování je možné provádět několika způsoby, a to za pomocí generující a kontrolní matice, kdy je však pro rozměrné kódy nutno počítat velké množství prvků Galoisova pole. Další možností je přímo pomocí generujícího mnohočlenu a dělení mnohočlenu mnohočlenem, kdy na závěr k informačnímu slovu připojíme zbytek po dělení. Po přijetí slova provedeme dělení stejným mnohočlenem a je-li zbytek nulový, nedošlo při přenosu k chybě. V opačném případě provedeme opravu. Je-li Hammingova váha zbytku rovna jedné, doplníme ho nulami a přičteme k přijatému slovu, takto lze opravit chyby na pozicích kontrolních bitů. Pokud je Hammingova váha větší než jedna, budeme postupně provádět cyklické posuvy prvků v přijatém slově a znovu počítat syndromy tak dlouho, dokud nebude Hammingova váha zbytku rovna jedné. Poté sečteme kódové slovo se syndromem a provedeme stejný počet posuvů zpět. Nezáleží, kterým směrem provádíme cyklické posuvy, ale musíme se držet jednoho směru a na konci posunout slovo správným směrem zpět. [1]

Další možností je realizace cyklických kódů pomocí zpětnovazebních posuvných registrů. Na obrázku 2 je obecné blokové schéma kodéru.



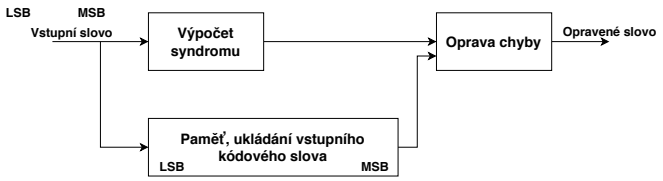
Obrázek 2: Obecné blokové schéma kodéru CK s posuvnými registry.

Postup vytvoření kodéru a kódování je shrnut v následujících krocích [1]:

- Počet posuvných registrů odpovídá stupni generujícího mnohočlenu $g(x)$, přičemž platí, že členy x^0 a x^{n-k} jsou v $g(x)$ vždy obsaženy.
- Zda je mezi posuvnými registry sčítací člen, záleží na uspořádání $g(x)$, konkrétně na tom, jestli je daný člen v polynomu obsažen, viz obrázek 2, kde jsou pozice jednotlivých členů vyznačeny.
- Informační bity jsou pomocí zpětné vazby přiváděny na začátek pole posuvných registrů, přičemž spínač S1 je sepnut, dokud nejsou všechny informační bity nasunuty.
- Po nasunutí posledního informačního bitu dojde k rozpojení spínače S1 a k přepnutí spínače S2.
- Spínač S2 slouží k oddělení informačních a kontrolních bitů. V první fázi jdou informační bity zároveň do posuvných registrů, a také se ukládají v paměti.
- Po nasunutí posledního informačního bitu je v registrech uložen zbytek po dělení, který má doplnit informační bity. Rozpojením spínače S1 s přepnutím S2 dochází v hodinovém taktu k jejich postupnému vyčítání, kdy na konci posledního cyklu dostaneme celé kódové slovo.
- Všechny registry mají v tomto okamžiku hodnotu 0 a může se začít kódovat nové slovo.

Po průchodu komunikačním kanálem, kdy může dojít k chybě v přenášeném slově, je třeba provést kontrolu přenosu, dekódování a případnou opravu. K tomu slouží obvod dekodéru, jehož obecné blokové schéma je na obrázku 3 a postup dekódování a případné opravy jsou opět shrnuty do přehledných bodů [1]:

elektrorevue
ISSN 1213 - 1539



Obrázek 3: Blokové schéma obecného dekodéru CK s posuvnými registry.

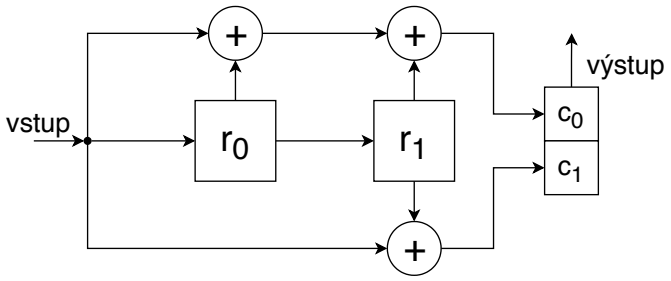
- Nejdříve se stejně jako při kódování vypočte syndrom. Ten je kompletní po nasunutí celého kódového slova, které se opět nasouvá od nejvíce významného bitu.
- Zároveň se přijaté kódové slovo ukládá do paměti posuvných registrů.
- Pokud je po nasunutí všech bitů syndrom nulový, znamená to, že slovo nebylo při přenosu poškozeno a není třeba jej opravovat.
- Není-li syndrom nulový, slovo je poškozeno a bude potřeba jej opravit. K tomu slouží blok oprava chyby, ve kterém se modulo(2) sčítají bity přijatého kódového slova a nově vypočteného korekčního bitu.

2.3 Konvoluční kód

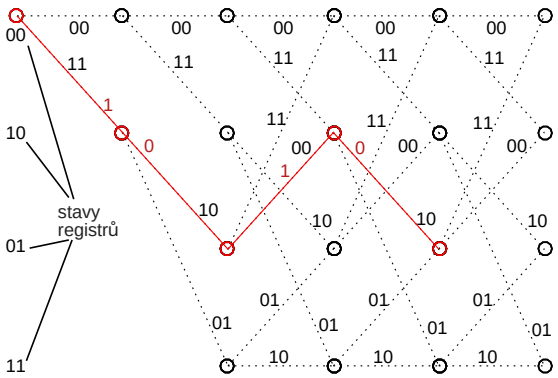
Předchozí dva kódy byly blokové, tzn. že se vždy kódoval blok bitů přesně dané velikosti. Oproti tomu u konvolučního kódu dochází k průběžnému kódování toku vstupních dat, přičemž hodnota na výstupu kodéru je také závislá na vstupních bitech, díky tomu je tento kód označován jako kód s pamětí. Symbol R vyjadřuje *rychlost kódu*, což je poměr mezi vstupními k a výstupními n bity (5). [1]

$$R = \frac{k}{n} \tag{5}$$

Kódování je možné pomocí schématu blokového kodéru [8], jehož možné zapojení je na obrázku 4, odvozeného z generujícího polynomu. Jinou možností je použití kódovací tabulky nebo kódového stromu [9]. Nejčastější a pro názornou ukázku nejlepší je však použití mřížového (trellis) diagramu, viz obrázek 5 s naznačenou cestou pro kódování informačního slova $m = [1\ 0\ 1\ 0]$.



Obrázek 4: Blokové schéma kodéru konvolučního kódu.



Obrázek 5: Mřížový (trellis) diagram.

Postup kódování je následující: v prvním kroku se vychází ze stavů registrů "00"a podle hodnoty vstupního bitu se postupuje v mřížovém diagramu. Při nulovém vstupním bitu dojde k pohybu horní větví a při hodnotě vstupního bitu "1" jdeme spodní větví, jak je zobrazeno pro příklad $m = [1\ 0\ 1\ 0]$ na obrázku 5. Další postup je analogický. Aktuální stavy paměťových registrů, ze kterých se vychází, jsou zapsány na začátku každého řádku. Hodnoty výstupních bitů jsou zapsány k jednotlivým propojům mezi uzly a výsledné kódové slovo je poté dáno vyčtením hodnot náležících jednotlivým cestám, tedy $c = [11\ 10\ 00\ 10]$.

Dekódování probíhá nejčastěji tak, že se porovná přijatá zpráva, která může být zasažena chybou se všemi posloupnostmi, jež mohly být vyslány. Hledá se tzv. věrohodnostní poměr a snažíme se o jeho maximalizaci. Pokud je přijetí všech kódových slov stejně pravděpodobné, pak dekodér, který dosahuje nejlepšího věrohodnostního poměru, je ten, který porovnává podmíněné pravděpodobnosti. Také je možné vyjádřit pravděpodobnostní funkci viz (6). Kde Z značí přijaté slovo a U^m některou z možných přijatých sekvencí. Dekodér vybere $U^{m\prime}$, jestliže [12]

$$P(Z|U^{m'}) = maxP(Z|U^m) — \textit{přes všechny } U^m. \tag{6}$$

Předpokládáme-li kanál s aditivním Gaussovským šumem s nulovou střední hodnotou a také kanál bez paměti, tzn. že šum ovlivní každý symbol nezávisle na ostatních symbolech. Pak pro konvoluční kód s rychlostí $1/n$ můžeme napsat věrohodnost jako [12]:

$$P(Z|U^m) = \prod_{i=1}^{\infty} P(Z_i|U_i^m), \tag{7}$$

kde Z_i je i-tá větev přijaté sekvence Z, U_i^m je i-tá větev konkrétní kódové sekvence U^m, z_{ji} je j-tý kódový symbol Z_i a u_{ji}^m je j-tý kódový symbol U_i^m. Každá větev se skládá z n kódových symbolů [12].

Dekodér tedy musí najít cestu kódovým stromem nebo mřížovým diagramem tak, aby pravděpodobnost (7) byla maximalizována. Je-li přijatá binární sekvence Z vytvořena z L slov definovaných větvemi kódu, pak počet

elektrorevue ISSN 1213 - 1539

posloupností U^m , se kterými musí být přijatá posloupnost porovnána, je 2^L [3].

Metoda dekódování je založena na maximalizaci věrohodnostního poměru, kterou v roce 1967 vyvinul *A. J. Viterbi* [10]. V principu se přijatá zpráva porovná se všemi posloupnostmi, které mohly být vyslány. To lze snadno provést pomocí mřížového diagramu, kdy se pro jednotlivé cesty počítá Hammingova vzdálenost, tj. rozdíl mezi přijatým a možným kódovým slovem. Jako nejpravděpodobnější kódové slovo se nakonec určí to, jehož cesta mřížovým diagramem má nejmenší celkovou Hammingovu vzdálenost od přijatého slova. [1]

2.4 LDPC kód

Tzv. kód s řídkou paritní maticí (Low Density Parity Check) je název, který odpovídá principu kódu. Ke kódování se opět využívá generující matice, která vychází z matice kontrolní, jejíž výraznou vlastností je, že obsahuje převážně nuly a pouze nízký počet prvků "1". Generování samotných matic je zásadní a obsáhlý proces, který lze podrobně nalézt např. v [5]. Obecně lze definovat paritní H matici jako matici s nízkým počtem jedniček jak v řádcích, tak ve sloupcích a dle [11] se dále uvádí:

- Každý řádek obsahuje ρ jedniček,
- Každý sloupec obsahuje γ jedniček,
- Počet společných jedniček mezi libovolnými dvěma sloupci není větší než jedna,
- ρ a γ jsou malé v porovnání s délkou kódu a počtem řádků v paritní H matici.

Samotné kódování je relativně jednoduché a odpovídá (2). Proces dekódování je oproti tomu značně náročnější a byla vyvinuta řada algoritmů lišících se jak účinností, tak složitostí. [1] Nejjednodušším neiterativním algoritmem je metoda s tvrdým rozhodováním, která spočívá v předávání bitů mezi bitovými a součtovými uzly. Jelikož se nejedná o iterativní algoritmus, je jeho výhoda v rychlosti, avšak zaostává za ostatními v počtu opravitelných chyb. Další algoritmy, které zde budou zmíněny jsou iterativní. Jedná se například o algoritmus Bit-flipping, kdy je postup následující:

- Vypočteme hodnotu syndromu mezi přijatým slovem a kontrolní H_{EG} maticí.
- Pokud není syndrom na všech pozicích nulový, vypočítáme pomocný vektor S podle (8)

$$S = s.H. \tag{8}$$

- Porovnáme hodnoty vektoru S a pokud hodnota na pozici S_i je větší, než předchozí největší hodnota, uložíme pozici i (hodnotám i odpovídá pro první prvek vektoru S hodnota "1", až poslednímu prvku vektoru hodnota "2^{2s}").
- Negujeme hodnoty v přijatém slově na pozicích určených v předchozím kroku.
- Vypočteme nový syndrom s a pokud není na všech pozicích nulový, opakujeme algoritmus až do maximálního zadaného počtu iterací.

Jiným iterativním algoritmem je Sum-Product (SP), jehož podrobný popis lze nalézt např. v [5]. Cílem je najít vektor d, který bude odpovídat vyslanému slovu c, a jehož syndrom bude nulový. Zde bude ilustrován zjednodušený algoritmus Sum-Product, který vychází z plné verze, ale je snazší pro pochopení. Popis obou verzí lze nalézt například v [11]. Jedná se o algoritmus s měkkým rozhodováním a využívá simulaci modelu kanálu pomocí přídavného bílého Gaussovského šumu. Na začátku je potřeba udělat odhad pravděpodobností f_j^x dle přijatého symbolu podle (9) resp. (10)

$$f_j^1 = \frac{1}{1 + e^{-\frac{2Ay_j}{\sigma^2}}}, \tag{9}$$

potom můžeme jednoduše odvodit

$$f_j^0 = 1 - f_j^1, \tag{10}$$

kde y_j je hodnota přijatého bitu na pozici j. Počítá se s tím, že zpráva je modulovaná, proto je potřeba pro simulaci přijaté slovo nejdříve převést do polárního formátu s amplitudou A, která byla zvolena s hodnotou 1. Dále se počítá se standardním rozptylem σ^2, který vyjadřuje střední kvadratickou odchylku šumu s normálním rozdělením v přenosovém kanále. V navrženém programu se počítá s hodnotou $\sigma^2 = 0{,}8$ podle [11].

Pro výpočet koeficientů R_{ij}^x je nutné stanovit funkce Q_{ij}^x. Bylo však dokázáno [13], že ve zjednodušené verzi algoritmu můžeme určit tyto funkce jako

$$Q_{ij}^0 = f_j^0, \tag{11}$$

$$Q_{ij}^1 = f_j^1. \tag{12}$$

Tato verze provádí výpočet iterativním provedením kroků, a to horizontálním a vertikálním způsobem, kdy se jednotlivé koeficienty určují podle paritní H matice. Hodnota δQ_{ij} se určí podle

$$\delta Q_{ij} = Q_{ij}^0 - Q_{ij}^1, \tag{13}$$

a hodnota δR_{ij} je dána rovnicí (14)

$$\delta R_{ij} = \prod_{j' \in N(i) \setminus j} \delta Q_{ij'}. \tag{14}$$

V dalším kroku se určí koeficienty R_{ij}^0 a R_{ij}^1, které jsou nutné pro odhad přijatých bitů.

$$R_{ij}^0 = 1/2.(1 + \delta R_{ij}) \tag{15}$$

$$R_{ij}^1 = 1/2.(1 - \delta R_{ij}) \tag{16}$$

elektrorevue
ISSN 1213 - 1539

Koeficienty Q^0_{ij} a Q^1_{ij} jsou aktualizovány ve vertikálním kroku. Jsou určeny pro každý pár uzlu i, j a pro všechny možné hodnoty proměnné x, které jsou v binárním případě $x = 0$ nebo $x = 1$. Vypočteme

$$Q^x_j = \alpha_j f^x_j \prod_{i \in M(j)} R^x_{ij}, \tag{17}$$

kde $M(j)$ odkazuje na bitové uzly, koeficient f^x_j značí pravděpodobnosti určené výše a α_j se vypočítá dle rovnice

$$\alpha_j = \frac{1}{f^0_j \prod_{i \in M(j)} R^0_{ij} + f^1_j \prod_{i \in M(j)} R^1_{ij}}. \tag{18}$$

Výsledná hodnota vektoru d je určena podle (19), tzn. vyslaný bit je určen podle velikosti Q^x_j.

$$d_j = max(Q^x_j) \tag{19}$$

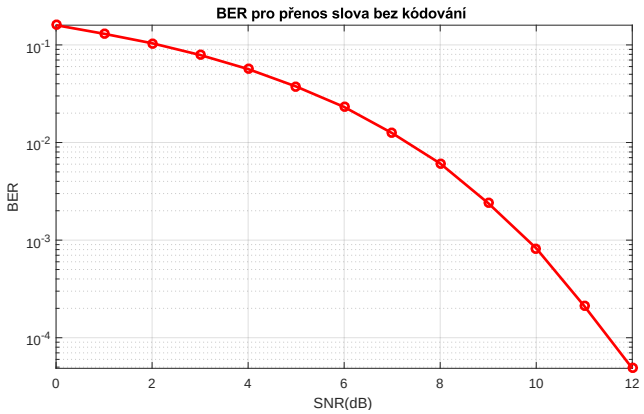
Pro další iteraci je nutné aktualizovat koeficienty $f^0_j = Q^0_j$ a $f^1_j = Q^1_j$. Další iterace se bude počítat pouze v případě, že nově vypočítaný syndrom nebude nulový a zároveň nebyl překročen maximální nastavený počet iterací.

3 Porovnání a vyhodnocení chybovosti

Tato kapitola porovnává a vyhodnocuje jednotlivé kódy a dekódovací algoritmy především z hlediska bitové chybovosti, což je poměr mezi počtem chybných a počtem všech přijatých bitů, viz (20). Všechny simulace byly provedeny s počtem 100 000 iterací a byl použit ideální kanál s aditivním bílým Gaussovským šumem. Počet vysílaných bitů odpovídá nastavení kódu a je uveden v každém grafu.

$$BER = \frac{počet\ chybných\ bitů}{počet\ všech\ bitů} \tag{20}$$

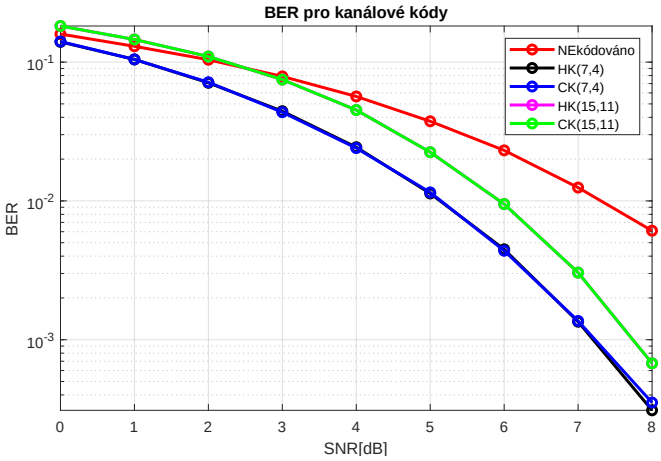
Na obrázku 6 je zobrazena bitová chybovost bez použití kanálového kódování s BPSK modulací v závislosti na SNR. Použitím kódovacích metod a opravných kódů je možné chybovost snížit, což bude ukázáno dále.



Obrázek 6: Závislost chybovosti na SNR pro přenos nekódovaného slova.

3.1 BER Hammingův a cyklický kód

Hammingův a cyklický kód patří k základním a nejjednodušším kódům. Jejich schopností je oprava pouze jedné chyby, tudíž dle teorie je výsledná chybovost obou kódů pro stejný počet informačních bitů totožná. To znamená, že při nízké hodnotě SNR, kdy je chybou zasaženo více bitů, nemůže dojít ke korektní opravě. Pro dlouhé kódové slovo dojde k ovlivnění více bitů a celková chybovost se tak zvýší. Vzájemné srovnání pro různé kódové délky nalezneme na obrázku 7.



Obrázek 7: Závislost chybovosti na SNR pro Hammingův kód.

3.2 BER konvoluční kód

Bitová chybovost konvolučních kódů v závislosti na SNR dále závisí na nastavení kódu, a to na rychlosti a počtu bitů informačního slova m. Výsledná závislost je na obrázku 8. Zde je vidět, že nižší rychlost, tzn. větší počet bitů kódového slova, zajišťuje lepší BER, což je však vykoupeno větší náročností na zpracování. Jak je v grafu patrno, pro rychlost kódu $R = 1/3$ je dosaženo nižší chybovosti při delším informačním slově. [1]

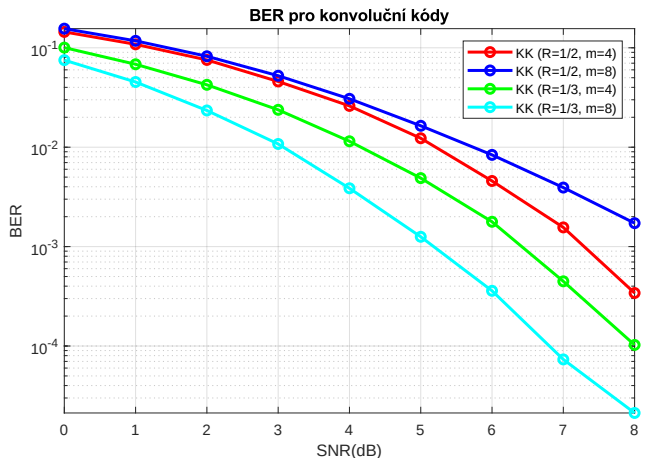
3.3 BER LDPC kódy

LDPC kódy dosahují z uvedených kódů nejnižší bitové chybovosti. Nejlépe je na tom podle předpokladů zjednodušený algoritmus Sum-product, jehož závislost chybovosti je vynesena v grafu na obrázku 9, kde parametr s značí řád kódu.

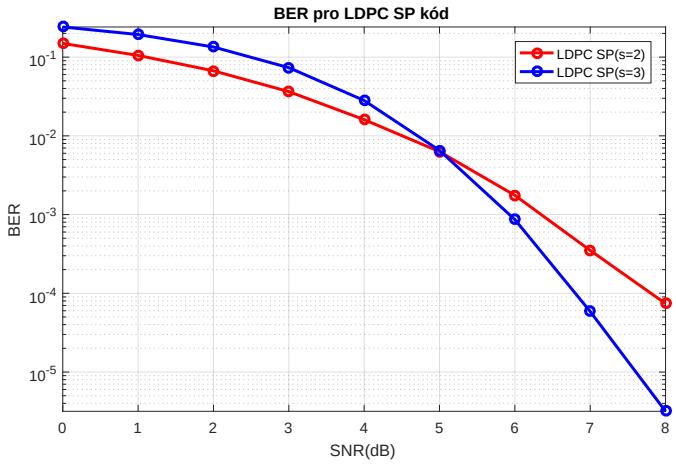
3.4 BER vzájemné srovnání

Jak Hammingovy, tak cyklické kódy jsou na tom s chybovostí totožně, jelikož oba jsou schopné detekovat dvě a opravit pouze jednu chybu. Chybovost konvolučního kódu dosahuje při čtyřech informačních bitech téměř shodných výsledků s kódem Hammingovým a cyklickým, avšak

elektrorevue
ISSN 1213 - 1539



Obrázek 8: Závislost chybovosti na SNR pro konvoluční kód.

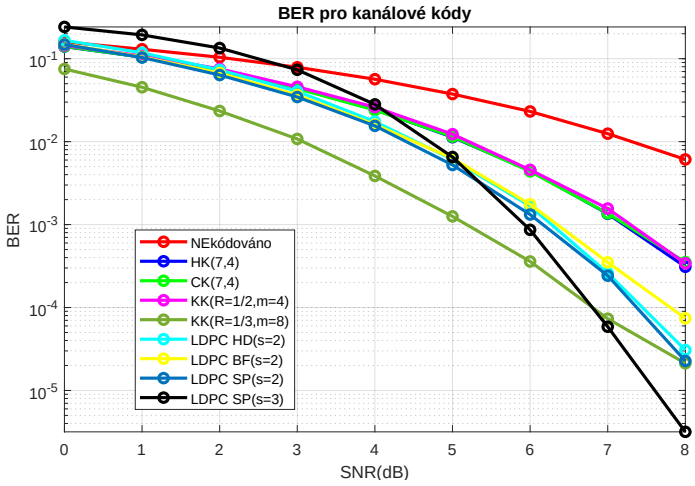


Obrázek 9: Závislost chybovosti na SNR pro LDPC SP kód.

výsledný počet bitů je roven osmi oproti sedmi u Hammingova a cyklického kódu, proto se jeví jako méně výhodný ve srovnání s těmito kódy při daném počtu informačních bitů. Při vyšší redundanci, tj. rychlosti $R = 1/3$ a více, dosahuje nižší chybovosti oproti ostatním. Nejlépe je na tom LDPC kód s algoritmem Sum-product při vyšším řádu kódu, jak je vidět také v grafu na obrázku 10.

4 Program pro demonstraci kanálového kódování

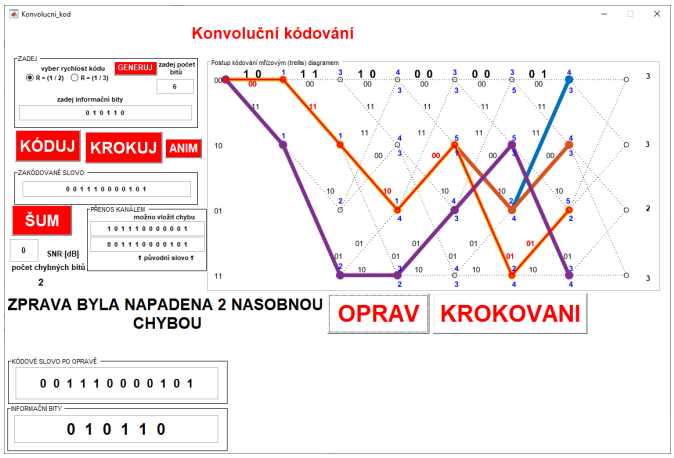
Vytvořený program má sloužit jako učební pomůcka, pomoci kterého si může uživatel ověřit funkce jednotlivých kódovacích algoritmů, následně simulovat průchod přenosovým kanálem, kdy může dojít ke změně přenášené informace a na závěr provést dekódování a pozorovat, kolik chyb je kód schopen korektně opravit. V programu jsou implementovány výše popsané algoritmy. Náhled hlavního okna



Obrázek 10: Závislost chybovosti na SNR pro vybrané kanálové kódy.

programu je na obrázku 11. Z hlavního okna programu si uživatel zvolí požadovaný kód, čímž dojde k otevření nového okna. Náhled pro konvoluční kód je na obrázku 12.

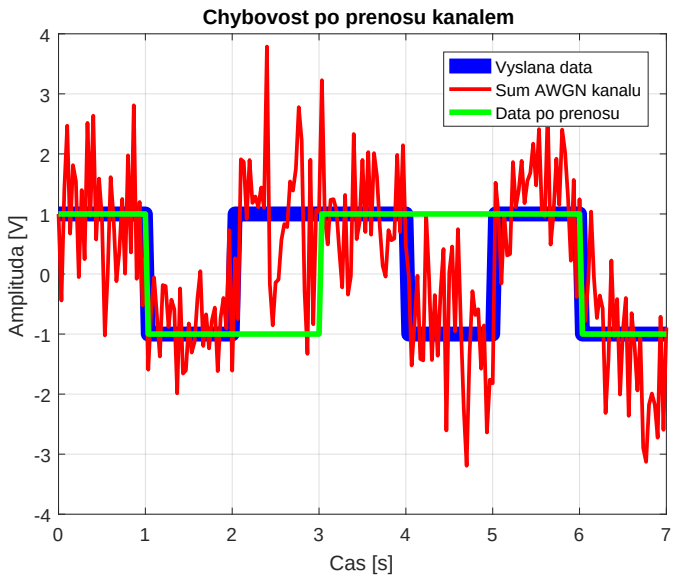
Obrázek 11: Náhled na hlavní okno programu.



Obrázek 12: Okno programu konvolučního kódu v režimu dekódování.

elektrorevue
ISSN 1213 - 1539

V úvodu je pro každý kód potřeba zadat několik parametrů, které se liší s typem kódu. Může se jednat o počet paritních bitů, rychlost kódu nebo generující polynom. Navíc je vždy nutno zadat, nebo nechat náhodně vygenerovat informační bity. Stiskem tlačítka "KÓDUJ"dojde k aplikaci daného algoritmu a zakódování informačního slova s názorným zobrazením principu kódování. Poté je možnost simulace průchodu ideálním přenosovým kanálem zasaženým AWGN šumem, viz obrázek 13. Na závěr je možné



Obrázek 13: Graf simulace přenosového AWGN kanálu.

provést dekódování přeneseného slova a tento proces včetně principu opravy sledovat. Jestliže je chyb příliš, může nastat případ, že dojde k opravě na jiné kódové slovo, jež by bylo po příjmu vyhodnoceno jako vyslané. Na všechny tyto eventuality program upozorní.

5 Závěr

V článku byl zmíněn důvod zavedení kanálového kódování, kterým je detekce a oprava chyby po přenosu komunikačním kanálem. Byly popsány jednotlivé metody kódování a dekódování pro vybrané kanálové kódy, jako je Hammingův, cyklický, konvoluční a LDPC kód.

Předmětem diskuse je rovněž porovnání jednotlivých kódů z hlediska bitové chybovosti v závislosti na kvalitě přijímaného signálu. Zde vychází nejlépe LDPC kód, který disponuje několika dekódovacími algoritmy, jejichž výsledky se mírně liší. Chybovost konvolučního kódu vychází lépe při větší redundanci a vyšším počtu informačních bitů. Pokud srovnáme Hammingův a cyklický kód s konvolučním (R = 1/2) pro čtyři informační bity, vychází všechny podobně, avšak zakódované slovo konvolučního kódu se skládá z vyššího počtu bitů.

Poslední kapitola se věnuje popisu vytvořeného programu pro demonstraci kanálového kódování, který si klade za cíl provést uživatele interaktivně celým procesem a usnadnit mu pochopení principu jednotlivých kódů.

Literatura

[1] ZÁVORKA, Radek. *Program pro demonstraci kanálového kódování.* Brno, 2020, 85 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky.

[2] CANCELLIERI, Giovanni. *Polynomial Theory of Error Correcting Codes.* Italy: Springer International Publishing, 2015. ISBN 978-3-319-01727-3.

[3] PROKEŠ, Aleš. *Rádiové komunikační systémy.* Purkyňova 118, 612 00 Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2013. ISBN 978-xxxxxx.

[4] HUFFMAN, W. Cary a Vera PLESS. *Fundamentals of Error Correcting Codes.* Cambridge: Cambridge University Press, 2003. ISBN 978-0-511-078280-7.

[5] LIN, Shu a Daniel J. COSTELLO. *Error control coding: fundamentals and applications.* 2nd ed. Upper Saddle River, N.J.: Pearson-Prentice Hall, c2004. ISBN 01-304-2672-5.

[6] ADÁMEK, Jiří. *Kódování.* 1. vyd. Praha: SNTL - Nakladatelství technické literatury, 1989. 191 s.

[7] VLČEK, Karel. *Komprese a kódová zabezpečení v multimediálních komunikacích.* Praha: BEN - technická literatura, 2000. ISBN 80-860-5668-6.

[8] SWEENEY, Peter. *Error control coding: from theory to practice.* 2002. New York: Wiley, c2002. ISBN 04-708-4356-X.

[9] BENVENUTO, Nevio a Giovanni CHERUBINI. *Algorithms for communications systems and their applications.* 2002. New York: J. Wiley, 2002. ISBN 04-708-4389-6.

[10] PETERSON, W. W. a E. J. WELDON. *Error-Correction Codes.* Second edition. Cambridge: MIT Press, 1972. ISBN 0 262 16 039 0.

[11] MOREIRA, Jorge Castiñeira a Patrick Guy FARRELL. *Essentials of Error-Control Coding.* Chichester: Wiley, 2006. ISBN 978-0-470-02920-6.

[12] SKLAR, Bernard. *Digital communications: fundamentals and applications.* 2nd ed. Upper Saddle River, N.J.: Prentice-Hall PTR, c2001. ISBN 01-308-4788-7.

[13] D. J. C. MacKay and R. M. Neal *Near Shannon limit performance of low density parity check codes.* in Electronics Letters, vol. 33, no. 6, pp. 457-458, 13 March 1997.