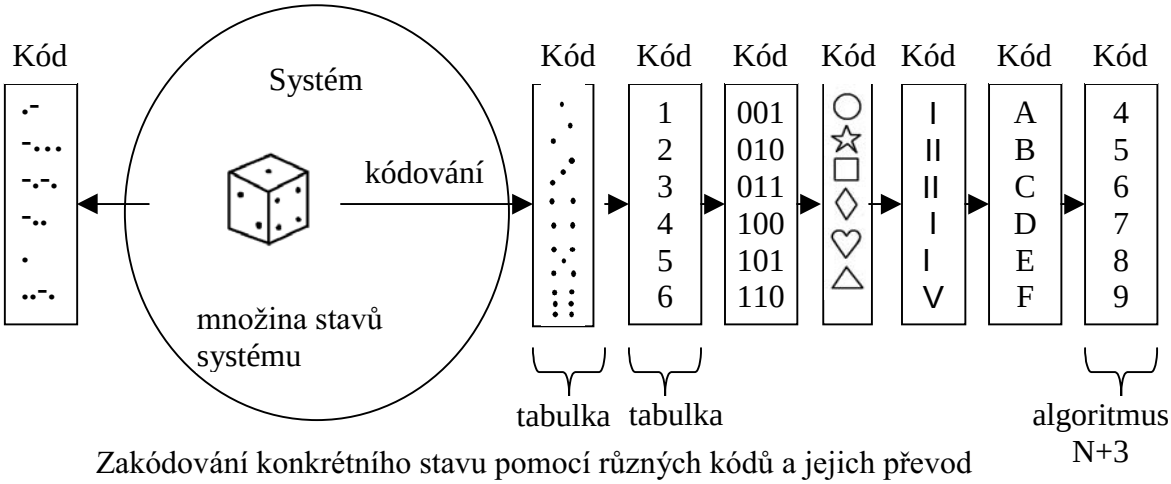
3. KÓDY A KÓDOVÁNÍ

Kód je množina symbolů, kterými vyjadřujeme jednotlivé stavy systému. Množina symbolů může být vyjádřena buď pomocí tabulky nebo dohodnutým systémem pravidel (algoritmem). Mějme např. systém, který je určen polohou hrací kostky (krychle). Po hodu kostkou se může kostka (systém) nacházet v jednom z šesti stavů (poloh kostky). Poloha kostky je nejčastěji vyjadřována pomocí množiny tečkových symbolů. Padne-li určitý počet teček, pak si jistě tento stav nepoznačíme v tečkovém kódu, ale pomocí arabských číslic 1 až 6, které tvoří jinou množinu symbolů pro vyjádření stavu systému (kostky). Jiným příkladem množiny symbolů jsou např. různé jednoduché obrázky (⬭ ☆ □ ◇ ♡ △), které umožní hrát dětem předškolního věku „Člověče, nezlob se“, aniž by uměli počítat, neboť mohou posouvat své figurky na odpovídající obrázky hrací cesty. Kódování je činnost, během které převádíme jeden kód na druhý, a to buď pomocí tabulky nebo vhodným algoritmem. Vztah mezi systémem, kódem a kódováním ilustruje obr.3.1.



Zakódování konkrétního stavu pomocí různých kódů a jejich převod pomocí kódování:

např. stav systému → ⁝⁝ → 6 → 110 → VI → F → ..-. → 9

Obr.3.1. Vztah mezi kódem, kódováním a systémem

Když k přenosu informace použijeme signál (fyzikální veličinu, která nese danou informaci) můžeme původní definici kódu zobecnit tak, že množinu symbolů „považujeme“ za stavy jiného systému. Např. hrací kostce bychom mohli přiřadit např. systém s fázovou modulací, který může generovat signál s šesti modulačními stavy (fázemi: 0°, 60°, 120°, 180°, 240°, 300°) nebo generátor, který může vytvářet šest různých kmitočtů. Z tohoto pohledu je pak možno kód definovat také tak, že je to dohodnutý systém pravidel pro jednoznačné přiřazení významu ke znakům nebo signálovým prvkům.

Kód lze libovolně přetvářet jeden na druhý, aniž by se změnila velikost informace. Při přechodu z jednoho kódu do druhého se mění množství znaků (nemusí být vždy stejné) a mění se i pravděpodobnost výskytu jednotlivých symbolů, přičemž dochází ke změně redundance. Lze najít takový kód, který bude mít největší entropii na symbol, tj. nejmenší počet symbolů na dané množství informace. Takový kód bude nejvýhodnější z hlediska ideálního přenosu (bez rušení), neboť bude vyžadovat nejmenší počet symbolů a nejkratší dobu přenosu. Protože reálné kanály jsou vždy s rušením, je nutné používat kódy, u kterých se zavádí záměrně redundance umožňující detekci případně korekce chyb.

Tabulka č.3.1 Mezinárodní kód MTA 2, MTA 3, inverzní kód a kód 3 z 5

Čísl složk	Písmen	Číslic	5-prvkov mezinárod kód č.2	7-prvkov mezinárod kód č.3	Inverzní kód		Kód 3 z 5
					prvn část	druh část	
1	A	-	11000	0011010	1100	0011	
2	B	?	10011	0011001	1001	1001	
3	C	:	01110	1001100	0111	0111	
4	D	kdo tam?	10010	0011100	1001	0110	
5	E	3	10000	0111000	1000	1000	0011
6	F	“	10110	0010011	1011	1011	
7	G	ˇ	01011	1100001	0101	0101	
8	H	´	00101	1010010	0010	1101	
9	I	8	01100	1110000	0110	1001	1101
10	J	zvone	11010	0100011	1101	1101	
11	K		11110	0001011	1111	0000	
12	L		01001	1100010	0100	1011	
13	M	.	00111	1010001	0011	0011	
14	N	,	00110	1010100	0011	1100	
15	O	9	00011	1000110	0001	1110	1110
16	P	0	01101	1001010	0110	0110	1001
17	Q	1	11101	0001101	1110	0001	1010
18	R	4	01010	1100100	0101	1010	0101
19	S	‚	10100	0101010	1010	0101	
20	T	5	00001	1000101	0000	0000	0110
21	U	7	11100	0110010	1110	1110	1011
22	V	ů	01111	1001001	0111	1000	
23	W	2	11001	0100101	1100	1100	1100
24	X	/	10111	0010110	1011	0100	
25	Y	6	10101	0010101	1010	1010	0111
26	Z	+	10001	0110001	1000	0111	
27	návrat vozíku (CR		00010	1000011	0001	0001	
28	posun o řádek (LF		01000	1011000	0100	0100	(0100
29	písmenová změna (LS)		11111	0001110	1111	1111	(1111
30	číslicová změna (F		11011	0100110	1101	0010	
31	mezera		00100	1101000	0010	0010	(0010
32	(nepoužito)		00000	0000111	0000	1111	
33	signál opakování		-	0110100	-	-	
34	signál		trvale „0“	0101001	-	-	
35	signál		trvale „0“	0101100	-	-	

3.1 Rozdělení kódů

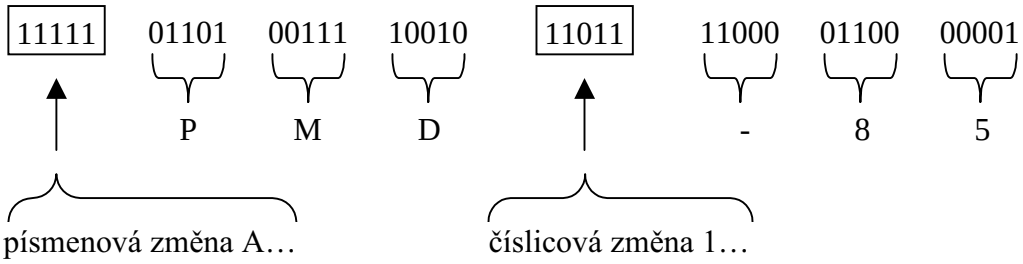
Kódy dělím podle různých hledisek. Např. kódy můžeme rozdělit na rovnoměrné a nerovnoměrné. Rovnoměrné kódy mají všechny znaky (symboly) tvořené stejným počtem prvků (viz. obr.3.1 dvojkový kód o třech prvcích). Nerovnoměrné mají znaky tvořené různým počtem prvků (např. Morseova abeceda).

Mezi základní používané kódy patří telegrafní kód, ISO-7, ASCII, kódy váhové (poziční) doplňkový kód, Brayův kód, atd. V poslední době nabývají na významu v souvislosti s přenosem

nebo uchováním informace kódy bezpečnostní, které se dělí na detekční a korekční (např. cyklický kód, Hammingův kód apod.).

Telegrafní kód

Telegrafní kód se používá v dálnopisných sítích. Schválen byl mezinárodní telegrafní unií v roce 1932 jako mezinárodní telegrafní abeceda č.2 (MTA 2). Česká verze kódu MTA 2 je uvedena v tabulce 3.1. Kód je pětiprvkový, tj. má kapacitu $N = 2^5 = 32$. Protože v anglické abecedě je 26 písmen a k tomu ještě potřebujeme 10 číslic, je kapacita kódu nedostačující. Tento rozpor se řeší tak, že se většina značek (kombinace pěti prvků) využívá dvakrát. Aby se poznalo, jaký význam značka má, předřazuje se ke skupině značek z levého sloupce tabulky 3.1 písmenová změna LS (A…) a skupině značek z prostředního sloupce číslicová změna FS (1…). Např. zpráva „PMD-85“ by byla zakódována takto:



Kód ISO-7

S nástupem počítačů a terminálových zařízení ztrácí dálnopisy postupně svůj význam a předpokládá se po roce 1990 jejich postupný zánik. Kód MTA 2 nevyhovuje již současným potřebám přenosu údajů. Proto byl organizacemi CCITT (Comité consultatif internacional télégraphique et téléphonique) a ISO (International Standards Organization) v sedmdesátých letech schválen nový kód ISO-7 (viz. tabulka 3.2).

Tak např. písmenu [A] je přiřazena značka 1000001 = 41H
(sloupec(4) řádek(1))

a číslici [1] značka 0110001 = 31H
(sloupec(3) řádek(1))

Kód ISO-7 byl odvozen z kódu ASCII, který vznikl v USA, kde se používá běžně u mikropočítačů (ASCII = American Standard Code for Information Interchange,viz tab. 3.2).

.

Poznámka: váhový kód 8421 neboli kód BCD (Binary Code Decimal) vyjadřuje desítkové číslice 0 až 9 pomocí čtyřprvkového dvojkového kódu (viz tab. 3.3).

Z kódu ISO – 7 (KOI – 7) se vyšlo při sestavení osmiprvkového kódu ISO – 8 (KOI 8). Osmý paritní bit ISO – 7 byl nahrazen významovým bitem umožňujícím vyjádřit malá i velká písmena azbuky a v české verzi písmena s háčky a čárkami (KOI-8-čs-2). Prvních sedm sloupců kódované tabulky ISO – 8 je shodných se sloupci kódovací tabulky ISO – 7.

Vytváření číselných (váhových) kódů, jejich vzájemné převody, inverzní a doplňkový kód jsou uvedeny ve skriptech “Mikroprocesorová technika“ [12].

Binárně (váhy 2^4)				2^6 2^5 2^4	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
2^3	2^2	2^1	2^0	sloup řádek	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
0	0	0	0	(0)	NUL	DLE	SP	0		P		p
0	0	0	1	(1)	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	(2)	STX	DC2	“	2	B	R	b	r
0	0	1	1	(3)	ETX	DC3	£	3	C	S	c	s
0	1	0	0	(4)	EOT	DC4	$	4	D	T	d	t
0	1	0	1	(5)	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	(6)	ACK	SYN	&	6	F	V	f	v
0	1	1	1	(7)	BEL	ETB	`	7	G	W	g	w
1	0	0	0	(8)	SS	CAN	(	8	H	X	h	x
1	0	0	1	(9)	HT	EM	)	9	I	Y	i	y
1	0	1	0	(10)	LF	SUB	*	:	J	Z	j	z
1	0	1	1	(11)	VT	ESC	+	;	K		k	
1	1	0	0	(12)	FF	FS	,		L		l	
1	1	0	1	(13)	CR	GS	-	=	M		m	
1	1	1	0	(14)	SO	RS	.		N		n	
1	1	1	1	(15)	SI	US	/	?	O		o	DEL

Tab 3.2 (ASCII = American Standard Code for Information Interchange).

Znak D(H)	Značka kódu přímého	Grayova
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10(A)	1 0 1 0	1 1 1 1
11(B)	1 0 1 1	1 1 1 0
12(C)	1 1 0 0	1 0 1 0
13(D)	1 1 0 1	1 0 1 1
14(E)	1 1 1 0	1 0 0 1
15(F)	1 1 1 1	1 0 0 0

kód BCD

Tab. 3.3 Grayův kó*d*

Grayův kód

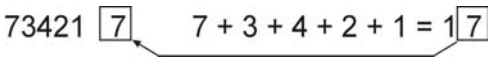
Grayův kód patří do skupiny kódů nazývaných se změnou v jednom místě. Tento kód se používá v analogově číslicových převodnících. Nevýhodou Grayova kódu je to, že jednotlivá místa značek nemají přiřazeny pevné váhy, což ztěžuje dokódování. Algoritmus překódování přímého dvojkového kódu do Grayova je následující :

- dvojková číslice přímého kódu s nejvyšší vahou (2^3) se ponechá beze změny,
- každá následující dvojková číslice přímého kódu se invertuje, když ji v přímém kódu předchází na vyšší váze jednička

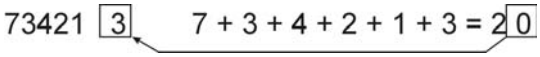
3.2 Principy snížení chybovosti přenosu dat

a) Bez zabezpečovacích zařízení

Nejjednodušší způsob zajišťující snížení chybovosti přenosu je u zprávy, která sama má již značnou nadbytečnost (redundanci). Např. psaný text (telegram) obsahuje velké množství nadbytečné informace, takže je možné většinou chybu opravit ihned podle smyslu zprávy. Má-li sama zpráva malou redundanci (např. číselné položky) je nutné ji účelně zvýšit. Nejjednodušší způsob zabezpečující přidání nadbytečné informace je opakování zprávy. Z následujícího vzájemného porovnání dvakrát přenesené zprávy je možné odhalit (detekovat) případné chyby, neboť je málo pravděpodobné, že by při opakovaném přenosu chyby vznikaly systematicky na stejných místech zprávy. Přeneseme-li zprávu třikrát, lze dokonce zjištěné chyby většinou přímo opravit (korigovat), protože správnému přenosu se značnou pravděpodobností odpovídá symbol (znak), který se opakuje dvakrát. Tento způsob snížení chybovosti přenosu je však velmi nehospodárný a pomalý.
U číselných položek zprávy můžeme použít zabezpečení úpravou zprávy kontrolním součtem (např. máme převést číslo 73421)



nebo jednomístným součtem na určitou hodnotu, např. “0”



Dále lze použít tzv. AN zabezpečení. Číslo N se násobí číslem A, např. A = 11

Se zabezpečenými čísly pomocí AN zabezpečení lze provádět základní početní operace (sčítání, odčítání, násobení a dělení). Přijatá číselná zpráva je bez chyby vyjde-li kontrolní dělení číslem A beze zbytku.

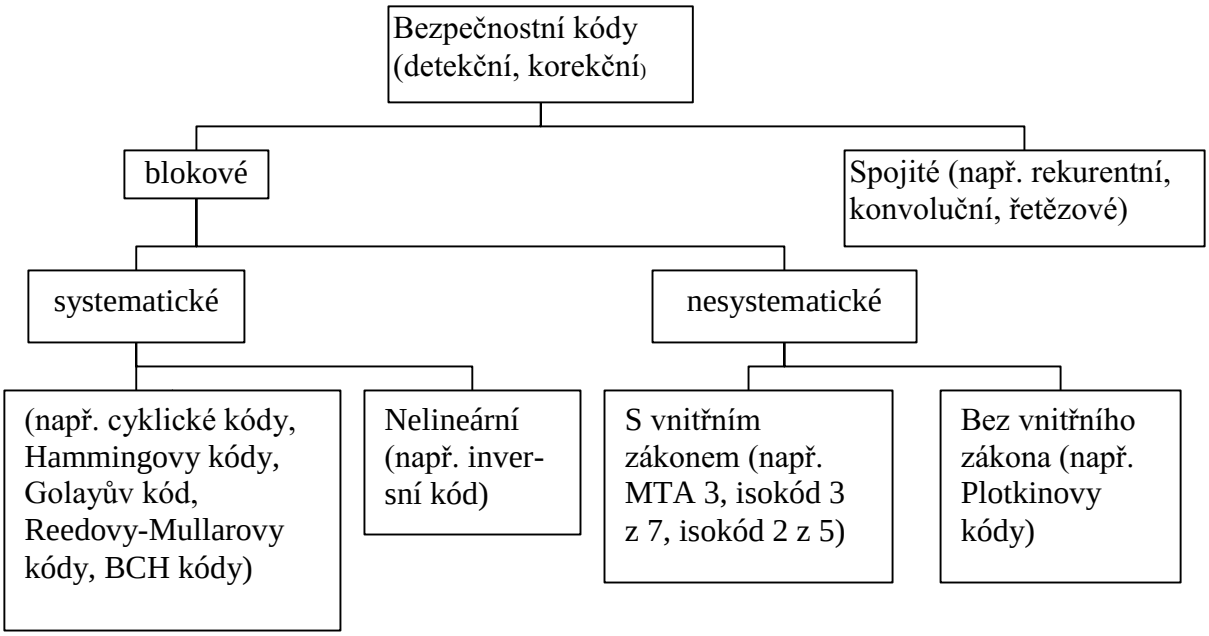
Zabezpečení úpravou zprávy je velmi účinné. Výhodou je, že nevyžaduje žádná přídavná zařízení, takže přenosové kanály (např. dálnopisné) je možno použít v existující podobě. Nevýhodou je pracnost a zdlouhavost vlastního zabezpečení a kontroly (detekce chyb).

b) Bezpečnostní kódy (pomocí zabezpečovacích zařízení)

V současné době existuje velký počet bezpečnostních kódů a na jejich výzkumu se dále pokračuje. V každé oblasti přenosu a zpracování diskrétních informací se postupně stabilizuje užívání určitých kódů, které se ukazují jako nejvýhodnější z hlediska kompromisu mezi stupněm zabezpečení proti chybám a jednoduchostí, případně cenou realizace příslušných zařízení. Bezpečnostní (redundantní) kódy dělíme na detekční a korekční. Klasifikace bezpečnostních kódů je na obr. 3.2.

U blokových kódů se přenášená posloupnost binárních prvků dělí na samostatné bloky, navzájem nezávislé. U spojitých kódů je redundance vkládána spojitě do posloupnosti prvků. U systematických kódů (separable codes) je rozložení informačních a redundantních prvků ve všech blocích shodné. Je známo, které prvky v bloku jsou informační a které redundantní. U nesystematických kódů je redundance jakoby rovnoměrně rozptýlena ve všech prvcích bloku. Kódy s vnitřním zákonem se vyznačují tím, že se při dekódování kontroluje splnění daného zákona (např. zda všechny kódové složky mají stejný počet jedniček). U kódů bez vnitřního zákona lze správný příjem kódové složky ověřit pouze porovnáním se soupisem (tabulkou) všech složek. Mají velkou kódovou vzdálenost a nelze je snadno technicky realizovat. Lineární kódy se vyznačují tím, že libovolná lineární kombinace kódových složek

je opět složkou kódu. Lineární dvojkové kódy se též všeobecně nazývají grupové, protože kód tvoří grupu vzhledem k operaci sčítání modulo 2 (tj. součet mod 2 libovolných dvou kódových složek je rovněž kódovou složkou). Kód nelineární výše uvedené vlastnosti lineárních kódů nemají.



Obr. 3.2 Rozdělení bezpečnostních kódů

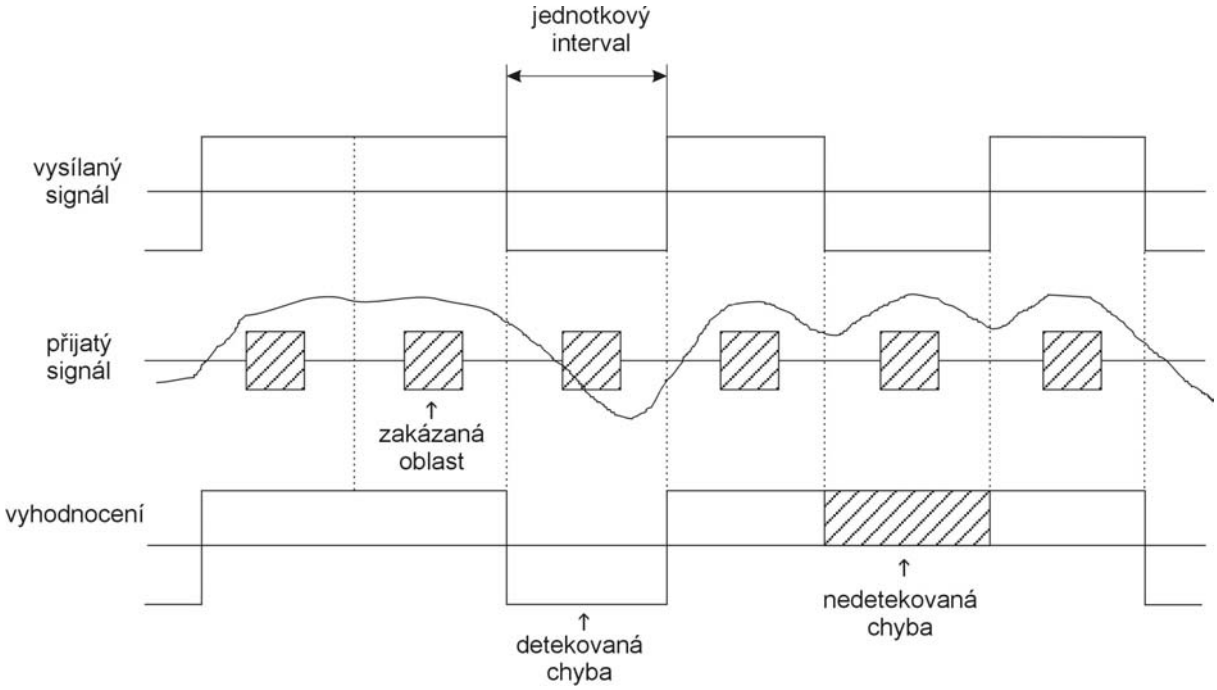
c) Kontrolou kvality signálu

Jestliže se kontrolovaný parametr přijímaného signálu odchýlí v důsledku poruchy od jmenovité hodnoty o více než připouští dovolená tolerance, vyšle přijímací stanice k vysílací stanici žádost o opakování poslední značky (nebo celého bloku), během které (kterého) porucha nastala. Uvedená metoda je založena na předpokladu, že vybočení daného parametru signálu z tolerančních mezí znemožní bezchybné vyhodnocení signálu v přijímači. Princip detekce chyb kontrolou kvality signálu je na obr. 3.3.
V důsledku časových ztrát, způsobených neužitečným opakováním, je použití tolerančních detektorů méně efektivní než bezpečnostní kódování. Proto se často zabezpečení kontrolou kvality signálu kombinuje se zabezpečením bezpečnostními kódy.

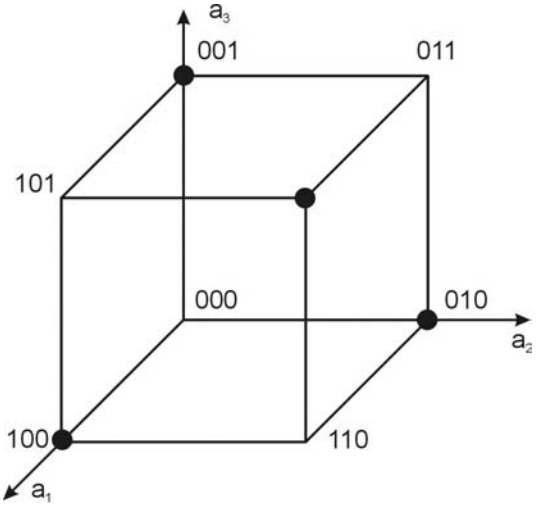
3.3 Geometrická reprezentace kódů

Uspořádanou posloupnost prvků kódové složky můžeme považovat za souřadnice bodu v n-rozměrném euklidovském prostoru. Danou kódovou složku můžeme tedy znázornit jako bod v prostoru. Např. tříprvkový binární kód lze znázornit pomocí trojrozměrné krychle o hraně jednotkové délky (viz obr. 3.4).
Vzdálenost d (A_i, A_j) dvou kódových složek je dána počtem míst, ve kterých se obě složky A_i, A_j liší. Z hlediska geometrického modelu je vzdálenost kódových složek A_i a A_j dána počtem hran krychle, kterými je třeba projít na cestě z vrcholu A_i do vrcholu A_j. Váha w kódové

složky je dána počtem jedniček v kódové složce. Pravidla pro sčítání a odčítání modulo 2 jsou:



Obr.3.3 Princip detekce chyb kontrolou kvality signálu.



$A = (a_1, a_2, a_3)$

	kódové složky	
A_i	platné	neplatné
		0 0 0
A_1	0 0 1	
A_2	0 1 0	
		0 1 1
A_3	1 0 0	
		1 0 1
A_4	1 1 1	

Obr 3.4 geometrické znázornění 3-prvkového dvojkového kódu

$0 \oplus 0 = 0$ $\quad 0 \ominus 0 = 0$
$0 \oplus 1 = 1$ $\quad 0 \ominus 1 = 1$
$1 \oplus 0 = 1$ $\quad 1 \ominus 0 = 1$

kde symbol $\oplus$ označuje operaci sčítání modulo 2
symbol $\ominus$ označuje operaci odčítání modulo 2.
Je vidět, že není rozdíl ve výsledku mezi sčítáním a odečítáním u operací modulo 2. Tady rozdíl dvou kódových složek dává stejný výsledek jako jejich součet. Vzdálenost dvou kódových složek zjistíme podle:

$$d(A_i, A_j) = w(A_i \ominus A_j) = w(A_i \oplus A_j) \qquad (3.1)$$

Např d (A_1,A_2) podle obrázku 3.4: d(A_1, A_2) = w(001 ⊕ 010) = w(011) = 2
Úplný obraz o vzdálenostech mezi kódovými složkami daného kódu poskytuje matice vzdáleností. Např. tříprvkový binární kód se složkami A_1 = 001, A_2 = 010, A_3 = 100, A_4 = 111 má matici vzdáleností:

d(A_i, A_j)	A_1	A_2	A_3	A_4
A_1	0	2	2	2
A_2		0	2	2
A_3			0	2
A_4				0

Nejmenší vzdálenost vyskytující se mezi kódovými složkami daného kódu se nazývá kódová vzdálenost D :

$$D = \min d(A_i, A_j), \tag{3.2}$$

která je důležitou charakteristikou kódu z hlediska jeho zabezpečující schopnosti proti chybám. Čím je D větší, tím mohou být vlastnosti bezpečnostního kódu lepší.

3.4 Zabezpečující schopnost kódu

Vyslaná kódová složka A_i může být v důsledku rušivých vlivů v přenosovém kanálu přijata jako složka B_x, tj. jako součet modulo 2 vyslané složky A_i a chybové složky E:

$$B_x = A_i \oplus E \tag{3.3}$$

Počet chybných prvků v přijaté kódové složce je roven vzdálenosti mezi přijatou a vyslanou kódovou složkou :

$$q = w(E) = w(A_i \oplus B_x) = d(A_i,B_x) \tag{3.4}$$

Např A_1 = 001 se změní na A_1' = 101, pak :
$d(A_1',A_2) = 3$ $d(A_1',A_3) = 1$ $d(A_1',A_4) = 1$ $d(A_1,A_1') = 1$

Chyba v q prvcích (q – násobná chyba) určité kódové složky mění její vzdálenost od všech složek kódu o q jednotek. K některým se o q jednotek přibližuje, od ostatních se o q jednotek vzdaluje.
Pro D = 1 a q = 1 vznik nedetekovatelné chyby,
D = 2 lze detekovat všechny jednoduché chyby,
D = 3 lze detekovat všechny dvojnásobné chyby.
Označme p pravděpodobnost výskytu chybného prvku. Pak pravděpodobnost správného příjmu prvku je 1 – p. Pravděpodobnost, že budou přijaty správně všechny prvky n-prvkové kódové složky je $(1 – p)^n$. Pravděpodobnost vzniku jednoduché chyby na libovolném místě n-prvkové složky je $p_1 = n \cdot p \cdot (1-p)^{n-1} \approx n \cdot p$. Podobně pro dvojnásobné chyby: $p_2 = C_2(n)p^2(1-p)^{n-2} \approx C_2(n)p^2$. Obecně pro q-násobnou chybu je pravděpodobnost výskytu:

$$p_2 = C_q(n)p^q(1-p)^{n-q} \tag{3.5}$$

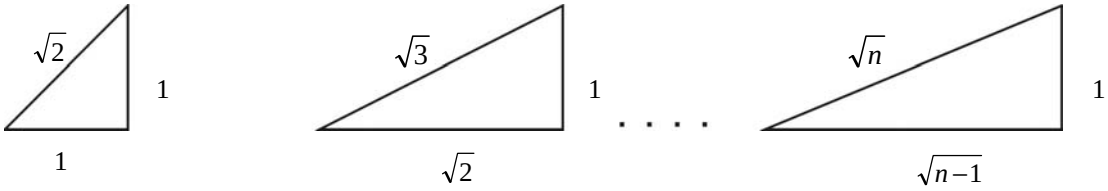
Pro vzájemně nezávislé chyby při malé hodnotě p platí:

$$p_1 > p_2 > p_q > p_n \tag{3.6}$$

V případě nezávislých chyb je proto nutné detekovat nebo korigovat především chyby nízké násobnosti, neboť se vyskytují nejčastěji.

Vliv q-násobné chyby na přenos n-prvkové kódové složky lze geometricky znázornit v n-rozměrném prostoru jako posunutí po hranách jednotkové n-rozměrné "krychle" o q jednotek, tj. do vzdálenosti $\sqrt{q}$, neboť všechny vrcholy krychle ve vzdálenosti q mají kódovou vzdálenost q od původní složky.

Pro : q = 2 q = 3 q = n využitímPythagorovy věty :



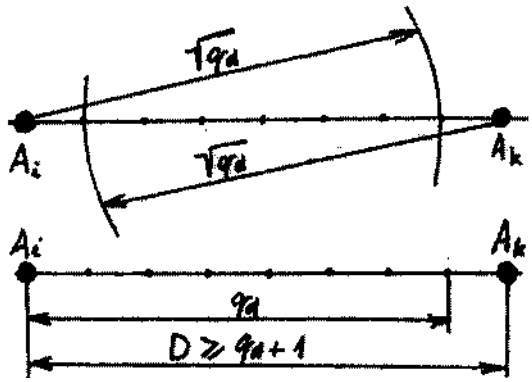
$$\sqrt{q} = \sqrt{1^2 + 1^2} = \sqrt{2} \quad \sqrt{q} = \sqrt{\left(\sqrt{2}\right)^2 + 1^2} = \sqrt{3} \quad \Rightarrow \quad \sqrt{q} = \sqrt{\left(\sqrt{n-1}\right)^2 + 1^2} = \sqrt{n}$$

Detekční schopnost kódu

Aby bylo možné detekovat chyby násobnosti q_d a menší, nesmí již uvnitř ani na povrchu koule o poloměru $\sqrt{q_d}$ ležet žádný jiný vrchol n-rozměrné jednotkové krychle, odpovídající platné kódové složce. Cestu po hranách n-rozměrné krychle, která je ve skutečnosti lomená, lze pro jednoduchost zobrazení nahradit přímkou viz obr. 3.5.

Kód který je schopen detekovat q_d chybných prvků musí mít kódovou vzdálenost $0 \geq q_d + 1$. naopak kód s kódovou vzdáleností D je schopen detekovat chybu s násobností $q_d \leq D - 1$.

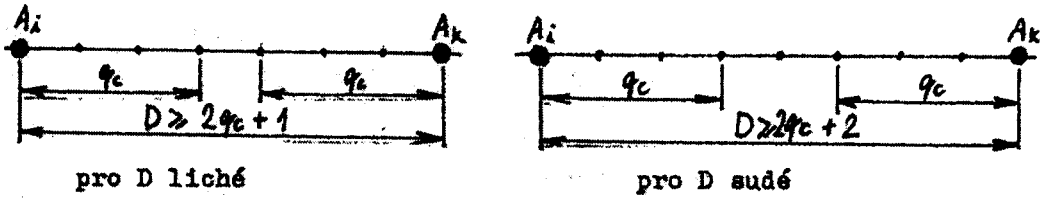
Korekční schopnost kódu



Obr. 3.5 Detekční schopnost kódu

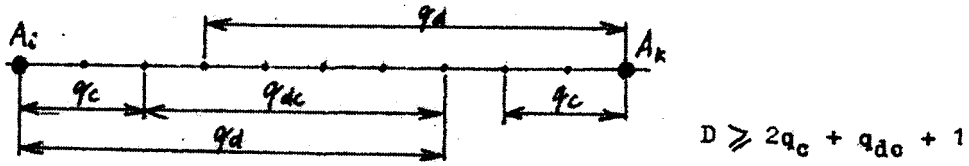
Pravděpodobnost výskytu chyby klesá s rostoucí násobností chyby za předpokladu, že výskyty chyb v jednotlivých prvcích kódové složky jsou vzájemně nezávislé.Při příjmu nepoužité (chybné) kódové složky je možné proto předpokládat, že s největší pravděpodobností byla vyslána ta platná (používaná) kódová složka, která má od přijaté nejmenší vzdálenost.Kód schopný korigovat q_c chybných prvků musí mít kódovou vzdálenost

$D \geq 2q_c + 1$ pro liché D a $D \geq 2q_c + 2$ pro sudé D viz obr. 3.6.



Obr. 3.6 Korekční schopnost kódu

Kód schopný korigovat q_c prvků a pouze detekovat q_{dc} musí mít kódovou vzdálenost $D \geq 2q_c + q_{dc} + 1 = q_c + q_d + 1$ jak vyplývá z obr. 3.7.



Obr. 3.7 Schopnost kódu současně korigovat a detekovat chyby

Čím větší je požadovaná kódová vzdálenost D, tím větší je redundance kódu. Vztah mezi kódovou vzdáleností a redundancí kódu není zatím prozkoumán natolik, aby se mohlo říci jak velký musí být celkový objem kódu N, aby bylo možno vybrat k kódových složek, jejichž kódová vzdálenost je nejméně D. Existují pouze odhady dolní a horní meze potřebné redundance. Účinnost zabezpečení je dána poměrem počtu odhalených kódových složek k celkovému počtu chybných kódových složek.

3.5 Isokódy – blokové kódy nesystematické s vnitřním zákonem

Mezi tyto kódy patří kódy konstantní váhy MTA č.3 a kód 3 z 5, které jsou uvedeny v tabulce 3.1. Počet k možných (použitelných) kódových složek kódu s konstantní váhou w a n-prvky v kódové složce je:

$$k = C_{w(n)} = \frac{n!}{(n-w)!\,w!}$$

Pro MTA č.3 (kód 3 ze 7) je k rovno.

:

$$k = \frac{7!}{(7-3)!3!} = 35 \qquad R = 1 - \frac{\log_2 35}{\log_2 27} = 0{,}27$$

D(A, B) = w(A ⊕ B) = 0011 010 ≈ ”A”

⊕ 0011 001 ≈ ”B” d(A, B) = w(A ⊕ B) = 2

0000 011

Kódová vzdálenost D = 2 umožňuje detekci pouze jednoduché chyby (q = 1). Pro zabezpečení číslic při dálnopisovém přenosu se používá kód 3 z 5 (viz tabulka 3.1).

$$k = \frac{5!}{(5-3)!3!} = 10 \quad \Rightarrow \textit{deset číslic} \quad D = 2 \Rightarrow q = 1$$

Kódy w z n vykazují dobrou zabezpečovací schopnost při použití v silně asymetrických kanálech. V čistě asymetrických kanálech, kde např. dochází pouze k chybám typu změn nul na jedničky, ale ne naopak, umožňují zjištění všech chybně přenesených značek.

3.6 Inversní kód – blokový kód systematicky nelineární

Vysílaná kódová složka se opakuje beze změny, je-li v informační části sudý počet nul, v případě lichého počtu nul v informační části se vyšle a v druhé části inversního kódu inverse první informační části, tj. té, kterou zabezpečujeme.Inversní kód je uveden v tabulce 3.1.
Inversní kód má redundanci R = 0,5 a kódovou vzdálenost D = 4. Může tedy detekovat až trojnásobné chyby. Lze použít i pro korekci jednoduché chyby. Princip korekce je patrný z následující tabulky.

Nachází-li se chybný prvek v první části (informační polovině), pak při součtu modulo 2 se

Kódová složka	G	L	X
vyslaná	01011 01011	01001 10110	10111 01000
přijatá	00011 01011	01011 10110	10111 00000
zaznamenaná	00011 10100	01011 10110	10111 11111
rozbor přijaté kódové složky	00011 ⊕ 10100 10111	01011 ⊕ 10110 11101	10111 ⊕ 11111 01000
místo korekce	chyba	chyba	chyba

shoduje jediný pár prvků (bitů) v místě chyby. Naopak došlo-li k chybě v zabezpečovací části, neshoduje se při součtu modulo 2 pár prvků v místě chyby.

3.7 Kódy systematické lineární

Kódy systematické lineární mají následující vlastnosti:
- jsou grupové, neboť tvoří grupu k operaci sčítání modulo 2,
- mezi použité kódové složky patří vždy nulová složka,
- ostatní kódové složky jsou určeny tzv. generační (vytvořující) maticí [G], která u kódu (n ,k) má k řádků a n sloupců, kde n je celkový počet prvků ve značce a k je počet nezabezpečených informačních prvků,
- jako řádky matice [G] může být použito k libovolných nenulových lineárních nezávislých n-prvkových kódových složek, přičemž lineárně nezávislé jsou takové složky kódu A_{r1} až A_{rk} pro něž platí : $c_1A_{r1} \oplus c_2A_{r2} \oplus \ldots \oplus c_kA_{rk} \neq 0$, kde c_1 až c_k mohou libovolně nabývat 0 nebo 1
- řádky generační matice tvoří dalších k kódových složek,
- zbývající kódové složky $2^k - k - 1$ se získají jako možné lineární kombinace kódových složek tvořících generační matici,
- aby kód měl požadovanou kódovou vzdálenost D, musí každá složka A_r tvořící generační matici být vzdálena od nulové složky minimálně D :

$$d(A_r, 0) = w(A_r \oplus 0) = w(A_r) \geq D \qquad (3.9)$$

- stejný požadavek musí splňovat libovolné dvě složky A_{r1} , A_{r2} navzájem :

$$d(A_{r1}, A_{r2}) = w(A_{r1} \oplus A_{r2}) \geq D \qquad (3.10)$$

- zamění-li se sloupce generační matice, vzniká sice jiný kód, ale zabezpečující schopnost

kódu se nemění (w = konst). Tak lze soustředit prvky nesoucí informaci na prvních k místech (jednotková matice)

:

$$\begin{bmatrix} 1\ 0\ 0\ 0 \ldots 0, r_{11} \ldots r_{1r} \\ 0\ 1\ 0\ 0 \ldots 0, r_{21} \ldots r_{2r} \\ 0\ 0\ 1\ 0 \ldots 0, \\ . \\ . \\ 0\ 0\ 0\ 0 \ldots 1, r_{k1} \ldots r_{kr} \end{bmatrix} = [I_k R] \cdot [G] = \begin{bmatrix} 10000\ 1001 \\ 01000\ 1110 \\ 00100\ 0110 \\ 00010\ 1010 \\ 00001\ 1100 \end{bmatrix}$$

k sloupců r sloupců např.: pro D = 3

- kódová vzdálenost lineárního systematického kódu (n, k) je rovna váze jeho nejlehčí nenulové složky A_k :D = min w(A_k),

- vysílaná zabezpečená kódová složka [V] se vytváří z nezabezpečené informace [I] :

$$[V] = [I] \cdot [G] \tag{3.11}$$

např. [V] = [11000] . [G] = [11000 0111] ... součet 1. a 2. řádku [G],
- matice [H] se nazývá maticí kontrolní, má n sloupců a (n - k) řádků,
- kódová složka (značka) náleží kódu tehdy a jen tehdy, když :

$$[S] = [V] \cdot [H]^T = [0] \tag{3.12}$$

kde $[H]^T$ je transponovaná matice [H] , řádková matice [S] se nazývá syndrom (soubor příznaků). Syndrom může sloužit k detekci, případně i k opravě chyb.

$$[H] = [R^T I_{n-k}] = \begin{bmatrix} 11011\ 1000 \\ 01101\ 0100 \\ 01110\ 0010 \\ 10000\ 0001 \end{bmatrix}$$

bezchybný příjem [S] = [0000]

- každá přijatá kódová složka [P] musí splňovat podmínky sudé parity :

např. $p_1 \oplus p_2 \oplus p_4 \oplus p_5 \oplus p_6 = 0$

$p_2 \oplus p_3 \oplus p_5 \oplus p_7 = 0$

$p_2 \oplus p_3 \oplus p_4 \oplus p_8 = 0$

$p_1 \oplus p_9 = 0$ prvky syndrome kódu (9, 5)

pak vyslaná zabezpečená informace bude : [V] = [I] . [G]

$v_1 = i_1$
$v_2 = i_2$
$v_3 = i_3$
.
.
.
$v_k = i_k$
$v_6 = i_1 \oplus i_2 \oplus i_4 \oplus i_5$
$v_7 = i_2 \oplus i_3 \oplus i_5$
$v_8 = i_2 \oplus i_3 \oplus i_4$
$v_9 = i_1$

k, r, n

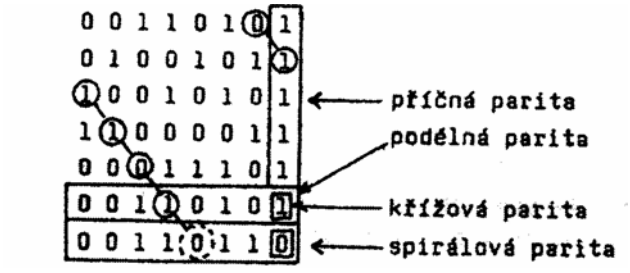
$$[H] \cdot \begin{bmatrix} p_1 \\ p_2 \\ . \\ . \\ p_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix} = [S]^T$$

neboli $[H] \cdot [P]^T = [S]^T$

Lineárním blokovým kódem je jednoduchý paritní kód (sudá parita). Kód má n míst ve značce, z toho k = n – 1 informačních. Vytvořující matice [G] a kontrolní matice [H] jednoduchého paritního kódu mají tvar :

$$[G] = \begin{bmatrix} 100000001 \\ 010000001 \\ ..... \\ 000000011 \end{bmatrix} \qquad [H] = [111111111] \quad \text{pro} \quad n = 9$$

Zabezpečení informace (např. kódu ISO - 7) jednoduchou paritou spočívá v přiřazení dalšího místa ke značce tak, aby počet jedniček ve značce byl sudý (sudá parita) nebo lichý (lichá parita = nelineární kód). Paritní kódy mají kódovou vzdálenost D = 2, proto umožňují zjistit všechny jednoduché chyby a chyby s lichou násobností.Kombinací lineárních paritních kódů dostáváme iterační kódy, z nichž nejznámější je kód s tzv. křížovou paritou.
Zabezpečení řádků se nazývá příčná parita a zabezpečení sloupců podélná parita. Iterační kód může být vypočítán jako kód opravný (korekční) pro opravu jedné chyby. Umožňuje zjistit (detekovat) všechny chyby s lichou násobností a dvojité chyby. Iterační kódy jsou sice realizačně jednoduché, zavádějí však většinou větší přídavnou nadbytečnost než kódy cyklické.Další varianta iteračního kódu spočívá v doplnění dalšího řádku vytvořeného kontrolními znaky tzv. spirálové parity. Kontrolní znaky se vypočítávají z úhlopříčně vybíraných informačních a paritních znaků. Kódová vzdálenost kombinovaných kódů je rovna součinů kódových vzdáleností výchozích kódů.Např. u křížové parity $D = D_1 \cdot D_2 = 2 \cdot 2 = 4$.



Tabulka 3.4 Iterační kód

Hammingovy kódy

Hammingovy kódy jsou lineární kódy, které jsou určeny pro zabezpečování údajů ukládaných do operačních pamětí číslicových počítačů nebo ke zvýšení výtěžnosti velkokapacitních polovodičových pamětí, případně pro kanály s nezávislými chybami. Kód byl sestaven na základě požadavku, aby syndrom [S] , pokud není nulový , udával svými prvky (souřednicemi) , chápanými jako čísla ve dvojkové soustavě , polohu chybného místa ve značce (kódové složce). Protože platí

$$[S] = [P] \,.\, [H]^T = ([V] \oplus [E]) \,.\, [H]^T = [V] \,.\, [H]^T \oplus [E] \,.\, [H]^T = [E] \,.\, [H]^T \qquad (3.13)$$

má při jednonásobné chybě matice chyb [E] pouze jednu jedničku, proto je syndrom roven řádku matice $[H]^T$, který odpovídá jedničce v chybové řádkové matici [E]. Je proto výhodné uspořádat sloupce kontrolní matice [H] tak, aby v i-tém sloupci bylo dvojkové vyjádření čísla i. Pak bude syndrom udávat pořadové číslo místa chybného znaku.

$$[H] = \begin{bmatrix} 000000011 \\ 000111100 \\ 011001100 \\ 101010101 \end{bmatrix} \qquad [S] = [S_4\, S_3\, S_2\, S_1]$$

$$D = 3$$

Např. pro pět informačních prvků (k = 5) a n = 9 je počet prvků zabezpečení r = n – k = 9 – 5 = 4 (= počet řádků [H]) a $2^r - 1 = 2^4 - 1 = 15$ sloupců kontrolní matice [H]. Nechť : $[V] = [v_1\ v_2\ v_3\ v_4\ v_5\ v_6\ v_7\ v_8\ v_9] = [v_1\ v_2\ \underline{v_1}\ v_4\ \underline{v_2}\ \underline{v_3}\ \underline{v_4}\ v_8\ \underline{v_5}]$

Prvek syndromu	Pořadové čísla místa chyby, která mají vliv na S_i
S_1	$\underline{1}$ 3 5 7 9
S_1	$\underline{2}$ 3 6 7
S_1	$\underline{4}$ 5 6 7
S_1	$\underline{8}$ 9

Jako zabezpečovací prvky volit ty místa chyb, které se vyskytují jen jednou v tabulce, tj. v_1 , v_2 , v_4 , v_8. V každé kontrolní rovnici se zabezpečení provádí sudou paritou :

$S_1 = 0 = v_1 \oplus v_3 \oplus v_5 \oplus v_7 \oplus v_9$
$\rightarrow v_1 = v_3 \oplus v_5 \oplus v_7 \oplus v_9 = i_1 \oplus i_2 \oplus i_4 \oplus i_5$
$S_2 = 0 = v_2 \oplus v_3 \oplus v_6 \oplus v_7 \rightarrow v_2 =$
$= v_3 \oplus v_6 \oplus v_7 = i_1 \oplus i_3 \oplus i_4$
$S_3 = 0 = v_4 \oplus v_5 \oplus v_7 \rightarrow v_4 = v_5 \oplus v_6 \oplus v_7 =$
$= i_2 \oplus i_3 \oplus i_4$
$S_4 = 0 = v_8 \oplus v_9 \rightarrow v_8 = v_9 = i_5$

Pořadové číslo místa chyby	S_4 S_3 S_2 S_1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

Poznámka: pro lichý počet jedniček v součtu informačních prvků operací modulo 2 je součet roven 1 a zabezpečující prvek sudé parity musí být roven také 1, takže součet kontrolní rovnice určuje zabezpečovací prvek (obdobně pro sudý počet jedniček je roven nule).

Např. pro $[I] = [i_1\, i_2\, i_3\, i_4\, i_5] = [11000]$ je $[V] = [011110000]$

Vyhodnocení při příjmu [P] = [011110010]

$$
\begin{array}{lcl}
S_1 = p_1 \oplus p_3 \oplus p_5 \oplus p_7 \oplus p_9 = 0 \\
S_2 = p_2 \oplus p_3 \oplus p_6 \oplus p_7 = 0 \\
S_3 = p_4 \oplus p_5 \oplus p_6 \oplus p_7 = 0 \\
S_4 = p_8 \oplus p_9 = 1
\end{array}
\Bigg\}
$$

nastala chyba na osmé pozici přijaté značky (kódové složky)

Příklady Hammingeova kódu jsou uvedeny v [23] na str. 45, 46.

Rozšířený Hammingeův kód s kódovou vzdáleností D = 4 je odvozen z Hammingeova kódu s D = 3 tak, že všechny značky jsou doplněny přídavným prvkem zajišťujícím navíc sudou paritu. Nová matice [H] bude mít o jeden řádek a jeden sloupec více. V novém posledním řádku budou samé jedničky, čímž je vyjádřena nově zavedená kontrola parity všech značek. Zbytek prvního (nového) sloupce je doplněn nulami, aby byl nový sloupec lineárně nezávislý vůči ostatním. Např.:

$$
[H] = \begin{bmatrix}
0\,0\,0\,0\,0 & 0\,0\,0\,1\,1 \\
0\,0\,0\,0\,1 & 1\,1\,1\,0\,0 \\
0\,0\,1\,1\,0 & 0\,1\,1\,0\,0 \\
0\,1\,0\,1\,0 & 1\,0\,1\,0\,1 \\
1\,1\,1\,1\,1 & 1\,1\,1\,1\,1
\end{bmatrix}
$$

pro [I] = [11000] je [V] = [0111100000]

sudá parita

Vyhodnocení rozšířeného Hammingeova kódu je následující:

syndrom	parita	vyhodnocení
není roven nule	souhlasí	došlo k dvojnásobné chybě
není roven nule	nesouhlasí	došlo k jednoduché (korigovatelné) chybě
je roven nule	souhlasí	značka je bezchybná
je roven nule	nesouhlasí	došlo k chybě vyšší liché násobnosti

Cyklické kódy

Cyklické kódy patří k blokovým systematickým lineárním kódům. Jejich specifická vlastnost, podle které se též nazývají, spočívá v tom, že cyklickou záměnou prvků použité kódové složky vzniká opět použitá (platná) kódová složka. Je-li:

$$[V_1] = [v_1\ v_2\ v_3\ \ldots\ v_{n-2}\ v_{n-1}\ v_n] \qquad (3.14)$$

použitá složka cyklického kódu, pak i kódové složky

$$[V_2] = [v_2\ v_3\ v_4\ \ldots\ v_{n-1}\ v_n\ v_1] \qquad (3.15)$$

$$[V_3] = [v_3\ v_4\ v_5\ \ldots\ v_n\ v_1\ v_2]$$

patří k použitým složkám kódu.

Kódové složky cyklických kódů je vhodné reprezentovat matematicky jako mnohočleny V(x). K´dová složka podle (3.14) délky n je pak vyjádřena polynomem n-1. stupně:

$$V_1(x) = v_1 . x^{n-1} + v_2 . x^{n-2} + \ldots + v_{n-2} . x^2 + v_{n-1} . x + v_n \qquad (3.16)$$

Cyklický posun prvků kódové složky o jeden prvek je pak ekvivalentní násobení daného polynomu členem x:

$$V_2(x) = x \,.\, V_1(x) = v1 \,.\, x^n + v_2 \,.\, x^{n-1} + \ldots + v_{n-1} \,.\, x^2 + v_n \,.\, x \qquad (3.17)$$

Operace násobení se zde definuje modulo x^n, kde platí $x^{n+k} = x^k$

pak $x^{n+k} = [(x^n - 1) \,.\, x^k] + x^k$, $x^n - 1 = 0$ a tedy $x^n = 1$, takže:

$$V_2(x) = v_2 \,.\, x^{n-1} + v_3 \,.\, x^{n-2} + \ldots + v_{n-1} \,.\, x^2 + v_n \,.\, x + v_1 \qquad (3.18)$$

$$V_3(x) = x \,.\, V_2(x) = x^2 \,.\, V_1(x) = v_3 \,.\, x^{n-1} + v_4 \,.\, x^{n-2} + \ldots + v_n \,.\, x^2 + v_1 \,.\, x + v_2 \qquad (3.19)$$

Cyklický (n, k) kód je takový kód, jehož kódově složky lze vyjádřit mnohočleny stupně n-1 a menšího, jež jsou dělitelné beze zbytku generačním (vytvářejícím) mnohočlenem G(x) stupně $r = n - k$. Dále dvojčlen $x^n + 1$ musí být dělitelný G(x) beze zbytku.

Označme I(x) mnohočlen jež reprezentuje přenášenou informaci k prvky. Postup zabezpečení je následující:

Každý mnohočlen I(x), vyjadřující nezabezpečenou informaci, se nejprve násobí členem x^r: $I(x) = I(x) \,.\, x^r$, čímž se stupeň každého členu polynomu I(x) zvýší o r, tj. na i+r. To je ekvivalentní připsání r nul na konec kódové složky [I]. Potom se I(x) dělí generačním mnohočlenem G(x):

$$\frac{I(x)}{G(x)} = \frac{I(x).x^r}{G(x)} = Q(x) \oplus \frac{R(x)}{G(x)} \qquad (3.20)$$

Dělením vznikne podíl Q(x) a zbytek R(x), který může být nejvýše stupně r-1. Úpravou (3.20) dostaneme:

$$I(x) \,.\, x^r = Q(x) \,.\, G(x) \oplus R(x) \quad \text{nebo} \quad I(x) \,.\, x^r \oplus R(x) = Q(x) \,.\, G(x), \qquad (3.21)$$

protože v aritmetice modulo 2 se odečítání shoduje se sčítáním. Pravá strana vztahu (3.21) zajišťuje dělitelnost $I(x) \,.\, x^r \oplus R(x)$ beze zbytku. Levá část rovnice (3.21) tvoří zabezpečenou složku n-prvkového kódu:

$$V(x) = I(x) \,.\, x^r \oplus R(x) \qquad (3.22)$$

Zbytek po dělení R(x) tedy určuje zabezpečující prvky. Do přenosového kanálu se vysílají nejprve informační prvky a za nimi prvky zabezpečující.

Například pro n = 7 lze realizovat tolik cyklických kódů délky n = 7, kolik existuje dělitelů dvojčlenu $x^7 + 1$. Protože dvojčlen $x^7 + 1$ lze rozložit na:

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

existuje celkem 6 různých polynomů G(x), kterým odpovídají kódy uvedené v tabulce 3.5.

Tabulka 3.5

Generační mnohočlen	r	kód (n, k)
$G_1(x) = x + 1$	1	(7, 6)
$G_2(x) = x^3 + x + 1$	3	(7, 4)
$G_3(x) = x^3 + x^2 + 1$	3	(7, 4)
$G_4(x) = (x + 1) . (x^3 + x + 1)$	4	(7, 3)
$G_5(x) = (x + 1) . (x^3 + x^2 + 1)$	4	(7, 3)
$G_6(x) = (x^3 + x + 1) . (x^3 + x^2 + 1)$	6	(7, 1)

Pro ilustraci zabezpečení přenosu informace cyklickými kódy použijeme kód (7, 4) s generačním $G_2 = x^3 + x + 1$. Určeme zabezpečení pro kódovou složku [I] = [0001], reprezentovanou mnohočlenem I(x) = 1. Podle (3.20) musíme I(x) nejprve násobit členem $x^r = x^3$, tedy $I'(x) = 1 . x^3 = x^3$, což představuje doplnění kódové složky [I] třemi nulami: [I´] = [0001000]. Nyní vydělíme I´(x) generačním mnohočlenem $G_3(x)$:

$$\frac{I'(x)}{G_3(x)} = x^3 : (x^3 + x + 1) = 1 + \frac{x+1}{x^3 + x + 1} \qquad (3.23)$$

$$\frac{x^3 + x + 1}{x + 1}$$

Zbytek R(x) = x + 1, tvoří redundantní část, takže vyslaná složka V(x) bude:
$V(x) = I'(x) + R(x) = x^3 + x + 1$, tj. [V] = [0001011]. Generační matici cyklického kódu (7, 4) lze zapsat v maticovém tvaru:

$$[G] = \begin{pmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{pmatrix} \qquad D = 3 \qquad (3.24)$$

Uvedený kód je schopen detekovat všechny jednoduché a dvojnásobné chyby nebo korigovat všechny jednoduché chyby. Správnost přenosu lze na přijímací straně kontrolovat dělením polynomu P(x) přijaté kódové složky generačním mnohočlenem G(x):

$$\frac{P(x)}{G(x)} = \frac{I(x).x^r + R(x) \oplus E(x)}{G(x)} = Q(x) \oplus \frac{E(x)}{G(x)} = Q(x) \oplus M(x) \oplus \frac{R_p(x)}{G(x)} \qquad (3.25)$$

Jestliže mnohočlen chyby $E(x) \neq 0$ není dělitelný polynomem G(x) beze zbytku, pak zbytek $R_p(x) \neq 0$ indikuje chybu přenosu. Zbytek $R_p(x)$ je roven nule v případě přenosu bez chyby, tj. E(x) = 0 nebo je-li mnohočlen $E(x) \neq 0$ dělitelný G(x) beze zbytku, čímž chyba v tomto případě zůstane nezjištěna.

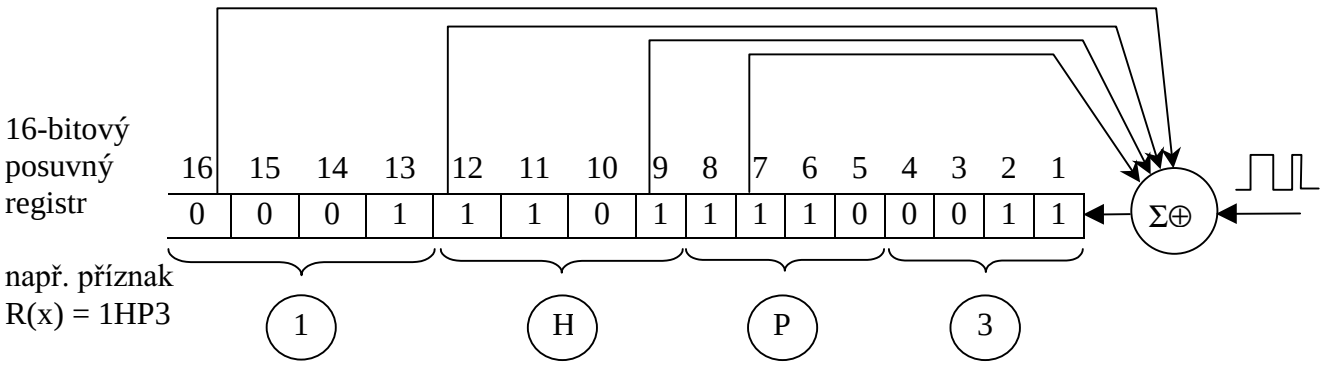
Z hlediska rozložení chybných míst v přenesené zprávě dělíme kanály na kanály s nezávislým výskytem chyb a se shluky chyb. Shlukem chyb délky b rozumíme skupinu b po sobě jdoucích prvků posloupnosti zprávy, z nichž alespoň první a poslední jsou chybné a vzdálenost k dalšímu shluku chyb je větší než b. Je-li generační mnohočlen cyklického kódu stupně r, pak detekuje shluky chyb $b \leq r$. Podle doporučení mezinárodní normy V – 41 CCITT se používají cyklické kódy (n, k): (260, 244), (500, 484), (980, 964), (3860, 3844) s generačním mnohočlenem (CCITT – 16) $G(x) = x^{16} + x^{12} + x^5 + 1$ (označovaný také SDLC).

Dále se používá cyklický kód CRC-16 s generačním mnohočlenem $G(x) = x^{16} + x^{15} + x^2 + 1$. Oba kódy mají r = 16 zabezpečovacích míst a detekují následující chyby:

- všechny jednoduché, dvojnásobné a trojnásobné chyby
- jakýkoliv lichý počet chyb
- libovolný shluk o délce b = 16 nebo kratší
- 99,9968 % shluků chyb o délce 17 prvků, tj. kromě 1 z 2^{15}
- 99,9984 % všechny shluky chyb delší než 17 prvků, tj. kromě 1 z 2^{16}

U příznakových analyzátorů se používají seště cyklické kódy s generačními mnohočleny $G(x) = x^{16} + x^{12} + x^9 + x^7 + 1$ nebo $G(x) = x^{16} + x^9 + x^7 + x^4 + 1$. Cyklické kódy se snadno realizují pomocí posuvných registrů a sčítačky modulo 2, jak ukazuje obrázek 3.8.

U příznakové analýzy se využívá skutečnosti, že určitému signálu (sledu jedniček a nul) odpovídá určitý zbytek R(x), který zůstane v posuvném registru a je zobrazen pomocí čtyř šestnáctkových číslic (označovaných jako příznak). Písmenové symboly čísloc jsou pro jednoznačné čtení 7-segmentových zobrazovačů použity jiné než se běžně používá u počítačů (A, C, F, H, P, U místo A, B, C, D, E, F),



Obr. 3.8 Princip realizace cyklického kódu (např. u příznakového analyzátoru BM 578)

TESLA Rožnov ☺ vyrábí obvod MH 101 pro vytváření a kontrolu cyklických kódů:

LRC – 8:	$x^8 + 1$	CRC-16:	$x^{16} + x^{15} + x^2 + 1$
CCITT:	$x^{16} + x^{12} + x^5 + 1$	TS:	$x^{14} + x^{10} + x^3 + 1$

V NDR se vyráběl vstup-výstupní obvod pro sériovou komunikaci U 856D (ekvivalent Z80 SIO) s protokolem HDLC nebo SDLC s implementací generačních polynomů CRC-16 a CCITT–16. U protokolu HDLC se posuvný registr na počátku neplní nulami, ale jedničkami. (CRC = Cyclic Redundancy Check)

U lokálních sítí typu ETHERNET se používá zabezpečení cyklickým kódem s 32 zabezpečovacími prvky a generačním mnohočlenem:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

Příklady cyklických kódů jsou v [23] na str. 47 až 51.

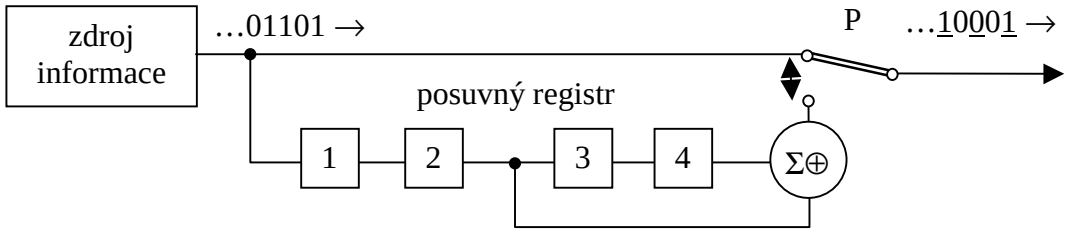
Řetězové kódy

Při snižování chybnosti přenosu dat pomocí spojitých bezpečnostních kódů se posloupnost prvků nedělí na vzájemně izolované bloky, ale proces kódování a dekódování je nepřetržitý, přičemž mezi informačními prvky jsou v pravidelných intervalech vkládány prvky zabezpečující. Spojité kódy se obvykle označují jako kódy (k/n), neboť tento poměr přímo vyjadřuje efektivnost kódu. Nadbytečnost spojitého kódu je R = 1– k/n. Spojité kódy jsou schopny korigovat shluky chyb.

U řetězových kódů je z každých n prvků spojité posloupnosti jeden prvek zabezpečující a k = n–1 prvků informačních. Redundantní prvky se vytvářejí operací modulo 2 ze dvou informačních prvků, které jsou od sebe vzdáleny o tzv. krok sčítání L. Takový řetězový kód je schopen korigovat shluky chyb délky b≤2L. Nevýhodou řetězového kódu je velká redundance. Na obr. 3.9 je uveden princip činnosti kodéru řetězového kódu (1/2) s krokem L = 2, který je schopen korigovat shluky chyb až do délky b = 4. Přepínač P vytvoří na výstupu kodéru posloupnost, ve které se střídají informační prvky s prvky zabezpečujícími, např.:

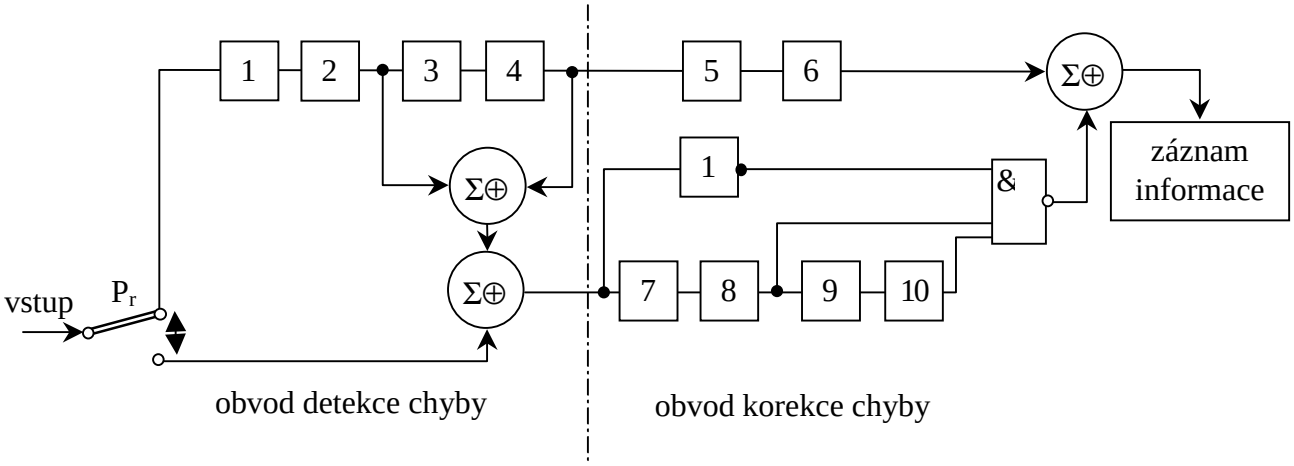
?

1 0 0 0 1 1 1 0 0 1 1 1 1 1 0 0 1 0 0 1 1 … (informační prvky jsou podtrženy)



Obr. 3.9 Kodér řetězového kódu (1/2) pro korekci shluků chyb do délky b = 4

Dekodér řetězového kódu se skládá ze dvou částí: obvodu detekce chyby a obvodu pro korekci chyby. Na obr. 3.10 je nakreslen princip dekodéru řetězového kódu:



Obr. 3.10 Dekodér řetězového kódu (1/2) pro korekci shluků chyb do délky

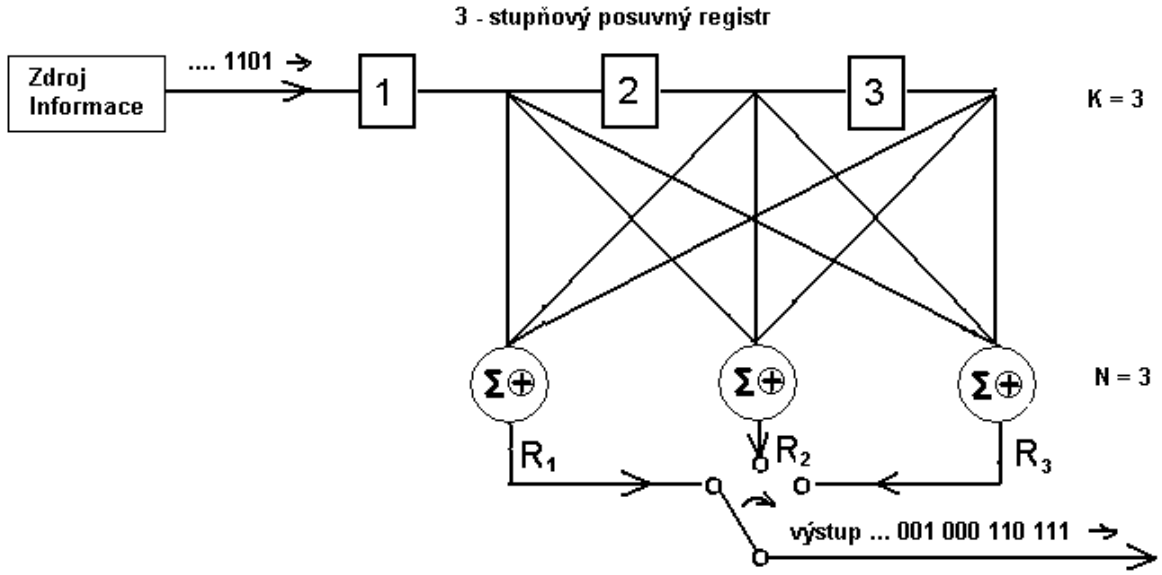
Konvoluční kódy

Kodér konvolučního kódu je tvořen K-stupňovým posuvným registrem a N sčítačkami modulo 2 připojenými k některým prvkům posuvného registru s přepínačem P, který postupně snímá výstupy sčítaček. Na obr. 3.11 je nakreslen příklad kodéru pro K = 3 a N = 3. Jednomu prvku na vstupu kodéru odpovídá uspořádaná trojice prvků na výstupu kodéru, která závisí na předchozích stavech kodéru. Pro znázornění činnosti kódu se používá buď kódový strom nebo mřížový diagram.

Při dekódování konvolučních kódů se vychází buď z kódového stromu (tzv. sekvenční dekódování) nebo z mřížového diagramu (Viterbiův dekódovací algoritmus).

Minimální kódy

Ve sdělovacích kanálech, kde pravděpodobnost výskytu chyby je malá, lze použít pro kódování zprávy kódu, který by zaručoval, že zakódovaná zpráva bude co nejkratší a tím i čas potřebný k přenosu zakódované zprávy bude minimální. Takový kód se bude konstruovat tak, že jednak kódování jím generované bude jednoznačně dekódovatelné



Obr. 3.11 Kodér konvolučního kódu

a nejčastěji se vyskytujícím znakům se přiřadí kódované slovo s nejkratší délkou. Rozhodujícím faktorem pro posouzení kvality kódu z hlediska minimální redundance bude průměrná délka kódovaných slov. Roku 1952 D.A. Hoffman dokázal, že takový kód existuje a popsal i metodu konstrukce takového kódu.

Jestliže známe četnost jednotlivých zdrojových znaků $a_1, a_2, \ldots, a_k$

(tj. pravděpodobnost P_i, že na náhodně zvoleném místě zprávy stojí "a_i"), můžeme určit střední délku kódového slova při daném kódování:

$L = d_1p_1 + d_2p_2 + \ldots + d_kp_k$,

kde d_l je délka kódovaného slova $K(a_i)$.

Nejkratším (minimálním) kódem dané zdrojové abecedy A se rozumí kód (přesněji kódování) s co nejmenší průměrnou délkou slova, $L = L_{min}$. Hodnota L_{min} závisí nejen na rozdělení pravděpodobnosti zdrojové abecedy, ale i na počtu n kódovaných znaků. Jednoduchou konstrukci kódu s poměrně hodnotou L navrhli Fano a Shannon: znaky a_1, a_2, …, a_k napíšeme pod sebe rozdělíme je odshora dolů na skupin tak, aby každá skupina měla přibližně stejnou pravděpodobnost jako ostatní (tj. $\frac{1}{n}$). První skupině přiřadíme znak b_1, první znak zdrojové abecedy, druhé skpině b_2, atd., poslední skupině přiřadíme b_n. Ve druhém kroku rozdělíme libovolnou (i-tou) skupinu, v níž je víc než jeden znak, na n skupin přibližně stejně pravděpodobných. První z těchto podskupin přiřadíme slovo b_ib_l, druhé b_ib_2 atd.

Postupujeme tak dlouho, až není co rozdělit, tj. v každé skupině je jediný znak. Výsledný kód je určitě prefixový, ale není obecně nejkratší.

Huffmanova konstrukce je složitější konstrukce kódování než Fanova-Shannonova, ale její předností je, že vždy vede k nejkratšímu kódování.

I. Binární kódování. Nejkratší binární (n=2) kódování dané zdrojové abecedy najdeme takto: Srovnáme abecedu podle pravděpodobností, tj. tak, aby platilo $p_1 \geq p_2 \geq \ldots \geq p_k$.

   Znaky a_1, … a_k napíšeme pod sebe a vedle nich jejich pravděpodobnosti. Sečteme poslední dvě a výsledek (p_{k-1} + p_k) zařadíme podle velikosti mezi ostatní pravděpodobnosti – graficky provádíme vodorovnou čárou. Potom zase sečteme dvě poslední pravděpodobnosti a výsledek zařadíme. To provádíme tak dlouho, až dojdeme k součtu 1. Sčítancům tohoto součtu přiřadíme slova 0 a 1. Kdykoli jsme některému přiřadili binární slovo $b_1 b_2 \ldots b_m$, potom sčítancům přiřazujeme slova $b_1 b_2 \ldots b_m$ 0 a b1 b2 … bm l Skončíme tehdy, když jsme všem zdrojovým znakům přiřadili binární slovo.

II. N-ární kódování. Kódujeme abecedou b_1, b_2, …, b_n. Postup je zcela analogický jako v binárním případě; nejprve provádíme součty (posledních n dosud nesečtených pravděpodobností) a zařazujeme je až do získání součtu l. Potom posledním sčítancům přiřadíme slova b_1, b_2, … , b_n a kdykoli jsme přiřadili součtu slovo B, potom sčítancům přiřazujeme slova Bb_1, Bb_2, Bb_n.

   Poznámka: jedinou výjimkou je první součet. Budeme sčítat r sčítanců, kde r = 2, 3, … , n, tak aby součet $n-r$ nesčítaných znaků byl dělitelný číslem $n-l$. To obvykle realizujeme tak, že graficky odtrhneme prvních $n-l$ zdrojových znaků, pak dalších $n-l$, atd. až zbude nejvýše n (a nejméně 2) znaky, které sečteme. Při dalším sčítání již sčítáme n znaků.

   Příklady jsou v [23] na str. 35 až 40.

3.11 Reedovy-Mullerovy kódy

Reedovy-Mullerovy (čti rídovy-malerovy) kódy jsou nejstarší známou třídou kódů opravujících volitelný počet chyb. Byly objeveny v roce 1954 a jejich význam spočívá ve velmi jednoduché dekódovací metodě, která se také snadno implementuje. Ve srovnání s BCH kódy, kterým věnujeme kap. 3.12, mají slabší parametry. Přesto jsou prakticky významně. Například kód R(1,5) použil kosmický koráb Mariner 9 při vysílání fotografií z Marsu.

Definice Reedových-Mullerových kódů je založena na boolovských funkcích.

Boolovské funcke jsou funkce, nabývající jen hodnot 0 a 1 při proměnných, které jsou také jen 0 nebo 1. Podrobněji, boolovská funkce m proměnných je předpis f, který každé m-tici x_1,

$x_2, \dots, x_m$ nul a jedniček přiřazuje hodnotu $f(x_1, x_2, \dots, x_m) = 0$ nebo 1; f je tedz zobrazení z množiny Z_2^m do množiny Z_2

. Tato zobrazení zapisujemě buď pomocí pravdivostní tabulky jako slova délky 2^m, nebo jako boolovské polynomy.

Pravdivostní tabulka je tabulka, ve které vypíšeme všechny možné kombinace hodnot $x_1, x_2, \dots, x_m$ a u každé kombinace uvedeme hodnotu funkce f. Zuápis kombinací proměnných provádíme systematixky tak, že sloupce tvoří čísla 0, 1, … , $2^m - 1$ v binárním zápisu (odshora).

Příklad boolovské funkce f dvou proměnných:

x_1	0	1	0	1
x_2	0	0	1	1
f	1	1	0	1

Příklad boolovské funkce g tří proměnných:

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
g	1	1	0	1	1	1	0	1

Na boolovských funkcích m proměnných zavádíme obvyklé logické operace:

název	označení	= l, právě když
f a g	fg	f = l a g = l;
f vel g	f + g	f = l nebo g = l, ale ne oboje;
f nebo g	f + g + fg	f = l nebo g = l nebo f = g = l;
negace f	~f	f = 0.

Například pro f = 1101 a g = 0011 platí:

fg = 0001,

f + g = 1110,

f + g + fg = 1111,

~f = 0010.

Mezi logickými operacemi platí jednoduché vztahy, např.

~f = f + 1

Jestliže má boolovská funkce tří proměnných $f(x_1, x_2, x_3)$ binární zápis $f_0\ f_1 \dots f_7$, potom první polovina tohoto slova (tj. slovo $f_0\ f_1\ f_2\ f_3$) je binárním zápisem boolovské funkce dvou proměnných $f(x_1, x_2, 0)$ a druhá polovina (tj. slovo $f_4\ f_5\ f_6\ f_7$) je zápisem funkce

$f(x_1, x_2, 1)$. Například pro funkci

$$f(x_1, x_2, x_3) = x_2 + x_3$$
$$= 00110011 + 00001111$$
$$= 0011 \mid 1100$$

platí

$$f(x_1, x_2, 0) = x_2 = 0011$$

a

$$f(x_1, x_2, 1) = x_2 + 1 = 1100.$$

Jiný způsob, jak reprezentujeme boolovskou funkci, je pomocí součtů a součinů funkcí x_i a 1. Například reprezentujeme boolovskou funkci f = 1101. Abychom si to usnadnili, přejdeme k negaci ~f = 0010. Ta je rovna 1, právě když $x_1 = 0$ a $x_2 = 1$, takže

$$\sim f = \sim x_1 x_2 = (1 + x_2)x_2 = x_2 + x_1 x_2.$$

Odtud plyne

$$f = 1 + \sim f = 1 + x_2 + x_1 x_2.$$

Tomuto tvaru říkáme boolovský polynom (dvou proměnných).

Pro každou boolovskou funkci $m + 1$ proměnných

$$f(x_1, x_2, ..., x_m, x_{m+1})$$

plati

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

Důkaz: Stačí ověřit, že obě strany se sobě rovnají jak pro $x_{m+1} = 0$, tak pro $x_{m+1} = 1$.

Boolovskou funkci f = 0111 (dvou proměnných) převedeme na boolovský polynom:

$$f = 01 + (01 + 11)x_2 = 01 + 01x_2$$

Ovšem stejným způsobem vidíme, že $01 = 0 + (0 + 1)x_1 = x_1 + 10 = 1 + (1 + 0)x_1 = 1 + x_1$

Takže

$$f = x1 + (1 + x1)x_2 = x_1 + x_2 + x_1 x_2$$

Podobně pro f = 11000111 platí

$$f = 1100 + 1011x_3 = x_1 + x_2 + x_3 + x_1 x_3 + x_1 x_2 x_3$$

Reedovým-Mullerovým kódem stupně r a délky 2^m se nazývá množina R(r,m) všech Boolovských polynomů m proměnných stupně nejvýše r.

Příklad: Všechny Reedovy-Mullerovy kódy délky 4

0	0000	R(-1,2)
1	1111	R(0,2)
x_1	0101	
x_2	0011	
$x_1 + x_2$	0110	
$1 + x_1$	1010	
$1 + x_2$	1100	
$1 + x_1 + x_2$	1001	R(1,2)
$x_1 x_2$	0001	
$1 + x_1 x_2$	1110	
$x_1 + x_1 x_2$	0100	
$x_2 + x_1 x_2$	0010	
$x_1 + x_2 + x_1 x_2$	0111	
$1 + x_1 + x_1 x_2$	1011	
$1 + x_2 + x_1 x_2$	1101	
$1 + x_1 + x_2 + x_1 x_2$	1000	R(2,2)

Vidíme, že R(-1,2) = {0} a R(0,2) je opakovací kód. Dále R(1,2) je (4,3)-kód,

Který je tedy kódem celkové kontroly parity.

Kód R(r,m) je ovšem lineární, protože součet dvou polynomů stupně $\leq$ r je také polynom stupně $\leq$r. Řádky generující matice tvoří všechny součiny $1, x_i, x_{i1}, x_{i2}, ..., x_{is}$ pro $s \leq r$.

Například kód R(2,3) (boolovských polynomů stupně $\leq$ 2) má tuto generující matici:

$$[G] = \left|\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right| \begin{array}{l} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_3 \end{array}$$

Všimněte si, že první čtyři řádky této matice tvoří generující matici kódu R(1,3), a první řádek je generující matice opakovacího kódu R(0,3).

Počet informačních znaků kódu R(r,m) je číslo

$$k = \left|\begin{matrix} m \\ 0 \end{matrix}\right| + \left|\begin{matrix} m \\ 1 \end{matrix}\right| + \left|\begin{matrix} m \\ 2 \end{matrix}\right| + ... + \left|\begin{matrix} m \\ r \end{matrix}\right|$$

Např. kód R(1,3) má tuto generující matici:

$$[G] = \left|\begin{array}{cccccccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & x_1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & x_2 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & x_3 \end{array}\right|$$

Počet informačních znaků je $\left|\begin{array}{c} 3 \\ 0 \end{array}\right| + \left|\begin{array}{c} 3 \\ 1 \end{array}\right| = 4$. Jde o binární (8,4)-kód, který je rozšířeným Hammingovým kódem.

Reedovy-Mullerovy kódy jdou důležité zejména pro svou jednoduchou a snadno implementovatelnou metodu dekódování. Je založena na většinové logice: pro hledanou hodnotu najdeme soustavu rovnic dekódování. Je založena na většinové logice: pro hledanou hodnotu najdeme soustavu rovnic a rozhodneme se buď pro 0, nebo pro 1 tak, aby většina těchto rovnic platila.

Např. používáme kód R(1,3) a chceme dekódovat slovo w = 00010101. Platí

$$v = q_0 1 + q_1 x_1 + q_2 x_2 + q_4 x_3, \qquad q_i = w_s + w_{i+s}$$

K určení koeficientu q_1 máme rovnice

$q_1 = w_0 + w_1$ (s = 0)

$= w_2 + w_3$ (s = 2)

$= w_4 + w_5$ (s = 4)

$= w_6 + w_7$ (s = 6)

V našem případě jsou všechny hodnoty kromě prvé rovny 1, a tedy

$q_1 = 1$

Podobně pro q_2 máme

$q_2 = w_0 + w_2$ (s = 0)

$= w_1 + w_3$ (s = 1)

$= w_4 + w_6$ (s = 4)

$= w_5 + w_7$ (s = 5)

V našem případě jsou všechny hodnoty kromě druhé rovny 0, a tedy

$q_2 = 0$

Dále pro q_4 platí

$q_4 = w_0 + w_4$ (s = 0)

$= w_1 + w_5$ (s = 1)

$= w_2 + w_6$ (s = 2)

$= w_3 + w_7$ (s = 3)

a většina určuje

$q_4 = 0$

Zbývá určit koeficient q_0. Od přijatého slova odečteme q_1x_1 (protože $q_2 = q_4 = 0$).

Platí $x_1 = 01010101$ a tedy

$$w' = w - q_1x_1 = 00010101 - 01010101 = 01000000 .$$

Odtud hlasováním dostáváme

$$q_0 = 0 .$$

Výsledný vektor je

$$v = q_1x_1 = 01010101 .$$

Jestliže nedošlo k více než jedné chybě, víme, že při přijetí slova w = 00010101 bylo vysláno slovo v = 01010101 .

3.12 Golayův kód

Mezi nejvýznamnější binární kódy patří Golayův (čti golejův) kód G_{23}, který má délku 23, z toho 12 informačních a 11 kontrolních znaků. Tento kód je perfektní pro opravy trojnásobných chyb. Hluboký výsledek teorie kódování je ten, že kromě Hammingových a opakovacích kódů je G_{23} jediný perfektní binární kód.

Golayovy kódy zavedeme jako systematické binární kódy délky 23 a 24. Levou polovinu generující matice tedy tvoří matice jednotková. Pravá polovina sestává ze čtvercové matice B řádu 11, doplněné o řádek 11…1 v případě G_{23} :

$$[G_{23}] = \left[\begin{array}{c|c} E & B \\ \hline & 11\ldots11 \end{array}\right]$$

Matice B vznikne cyklickými posuvy svého prvého řádku, což je slovo

11011100010 .

(Toto slovo má jedničku na místě i = 0 ,1 , . . ., 10, právě když je i čtvercem modulo 11, | tj. pro 0^2 , 1^2 , 2^2 , 3^2 , $4^2 \equiv 5^2$ a $5^2 \equiv 3$). Zde je celá matice :

$$[B] = \left[\begin{array}{cccccccccccc} b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 & b_8 & b_9 & b_{10} \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ \hdashline 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}\right]$$

Kód G_{24} vznikne tak, že kódu G_{23} přidáme celkovou kontrolu parity matice G_{24} .
Minimální vzdálenost D = 8 .

BCH kódy

Tyto kódy objevili koncem padesátých let R.C.Bose a D.K. Ray-Chaudhuri a nezávisle na nich A. Hocquenghem; podle iniciál autorů se nazývají BCH kódy.mezi jejich významné vlastnosti patří velká volitelnost parametrů, dobrý vztah mezi počtem informačních znaků a počtem opravovaných chyb a detailně vypracované dekódovací metody.

Ve srovnání s Reedovými-Mullerovými kódy o něco náročnější dekódování, zato ale mají lepší parametry. BCH kódy délky 255 , opravující 31 chyb, mají 55 informačních znaků, zatímco odpovídající Reedův-Mullerův kód má jen 37 informačních znaků.

BCH kód s plánovanou vzdáleností d = D lze definovat jeho kontrolní maticí

$$[H]=\begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & ..... & \alpha^{n-1} \\ 1 & \alpha^3 & (\alpha^3)^2 & (\alpha^3)^3 & ..... & (\alpha^3)^{n-1} \\ 1 & \alpha^5 & (\alpha^5)^2 & (\alpha^5)^3 & ..... & (\alpha^5)^{n-1} \\ \hdots & \hdots & \hdots & \hdots & \hdots & \hdots \\ 1 & \alpha^{d-2} & (\alpha^{d-2})^2 & (\alpha^{d-2})^3 & ..... & (\alpha^{d-2})^{n-1} \end{bmatrix}$$

Bližší popis těchto kódů je v [30].