Úvod do teórie kódovania

Daniel Olejár
Martin Stanek
Jaroslav Janáček
Richard Ostertág

20. mája 2002

Obsah

1 Úvod

Úvod do teórie kódovania je prvou zo série kníh Vybrané kapitoly z informatiky a aplikovanej matematiky, ktorú plánuje vydávať DevínLab, spoločné pracovisko Katedry informatiky FMFI UK a firmy ArtInApples, s.r.o. Séria je určená predovšetkým poslucháčom univerzitného štúdia informatiky a informatikom z praxe. Jej cieľom je aspoň čiastočne zaplniť medzeru v ponuke odbornej literatúry a prístupným spôsobom oboznámiť čitateľov so základmi dôležitých a zaujímavých informatických a matematických disciplín a možnosťami ich použitia.

Samotná teória kódovania je pekným príkladom využitia abstraktnej matematiky na riešenie praktických problémov vznikajúcich pri spracovaní informácie. Pri prenose údajov pomocou komunikačného kanála a pri uchovávaní údajov na pamäťových médiách vznikajú chyby, ktoré môžu zostať neodhalené a pri automatickom spracovávaní údajov spôsobovať vážne problémy. Teória kódovania ponúka riešenie v podobe samoopravných kódov, schopných odhaľovať a opravovať chyby spôsobené náhodnými vplyvmi. Druhým aktuálnym problémom je efektívnosť zápisu informácie. Často je potrebné prenášať alebo uchovávať také množstvá údajov, ktoré presahujú existujúce komunikačné alebo archivačné kapacity. Sú známe metódy kódovania, ktoré využívajú napr. štatistické zákonitosti v údajoch na ich transformáciu do „stručnejšej“ podoby.

Kniha „Úvod do teórie kódovania“ je venovaná základom klasickej teórie kódovania: efektívnym a samoopravným kódom.

2 Základné pojmy a označenia

Pojmy ako „kód“, „kódovanie“, „informácia“, „údaje“, „prenos“, „prenosový kanál“ a ďalšie sa v informatike považujú za všeobecne známe a možno aj preto sa často používajú v rozličných významoch. Aby sme sa vyhli nedorozumeniam spôsobeným rozličnou interpretáciou základných pojmov, vybudujeme si potrebný pojmový aparát exaktne. Začneme uvedením potrebných pojmov teórie formálnych jazykov.

2.1 Abecedy, slová a jazyky

Abeceda je ľubovoľná konečná neprázdna množina. Prvky abecedy budeme nazývať *znakmi* alebo *symbolami*. Abecedu budeme označovať symbolom Σ; ak bude potrebné rozlišovať rozličné abecedy, budeme symbol Σ indexovať. Ľubovoľná konečná postupnosť znakov z abecedy Σ sa nazýva *slovom nad abecedou* Σ. Ak nebude podstatné o akú abecedu ide alebo z kontextu bude známe, o ktorú abecedu sa jedná, budeme kvôli stručnosti slová „nad abecedou Σ“ vynechávať. Zjednodušíme aj zapisovanie slov; symboly v postupnosti nebudeme oddeľovať čiarkami a slovo (napr.) a, b, e, c, e, d, a budeme zapisovať v štandardnom tvare: $abeceda$. Nech je w slovo nad abecedou Σ, potom počet znakov slova w nazveme „dĺžkou slova w“ a budeme ju označovať symbolom $l(w)$. Tak napríklad $l(slovo) = 5$, $l(abeceda) = 7$, $l(a) = 1$. Postupnosť znakov nad abecedou Σ môže byť aj prázdna. Takáto postupnosť sa nazýva *prázdnym slovom* a označuje symbolom

λ. Pre dĺžku prázdneho slova platí $l(\lambda) = 0$. Teraz definujeme operácie nad slovami, pomocou ktorých bude možné vytvárať nové slová. Nech sú u, v dve slová nad abecedou Σ; $u = a_1 \dots a_n; \quad v = b_1 \dots b_m$. *Zreťazením slov* u, v je slovo $w = uv = a_1 \dots a_n b_1 \dots b_m$ nad abecedou Σ. (Je zrejmé, že operácia zreťazovania slov je asociatívna, ale vo všeobecnosti nie je komutatívna; prázdne slovo λ je obojstranným neutrálnym prvkom vzhľadom na operáciu zreťazenia slov: pre ľubovoľné slovo w platí $w\lambda = \lambda w = w$). Slovo možno zreťaziť aj so sebou samým, napr. $uu = a_1 \dots a_n a_1 \dots a_n$. Pre ľubovoľné slovo w a ľubovoľné číslo $k \in \mathcal{N}$ definujeme:

1. $w^0 = \lambda$,
2. $w^{k+1} = w^k w$.

Nech $u = a_1 \dots a_n$ je ľubovoľné slovo, súvislú podpostupnosť $z = a_i a_{i+1} \dots a_{i+k-1}; \quad 1 \leq i, i + k < n$ nazveme *podslovom slova* u. Ak $0 < k < n$ slovo z nazveme *vlastným podslovom slova* u. Slová u a λ sú triviálnymi podslovami slova u a v kódovaní sa nimi zvlášť zaoberať nebudeme. Zato však v teórii kódovania zohrávajú dôležitú úlohu podslová, ktoré sú začiatkom alebo koncom nejakého slova. Zavedieme pre ne špeciálne pomenovania. Nech $u = a_1 \dots a_n$ je ľubovoľné slovo, slovo $z = a_1 \dots a_k, \quad 0 < k \leq n$ nazveme počiatočným podslovom (prefixom) slova u a slovo $x = a_i a_{i+1} \dots a_n; \quad 1 \leq i = n$ nazveme *koncovým podslovom (sufixom) slova* u. Znaky v slove možno aj preusporiadať. Dôležitým prípadom preusporiadania znakov je otočenie slova: *zrkadlovým obrazom* slova $u = a_1 \dots a_n$ nazveme slovo $a_n \dots a_1$

Slová môžu byť prvkami množín. Ľubovoľnú množinu slov nad abecedou Σ nazveme *jazykom nad abecedou* Σ. Okrem bežných množinových operácií s jazykmi zavedieme aj operácie odvodené od zreťazovania slov. Nech sú $\mathcal{L}_1, \mathcal{L}_2$ jazyky nad abecedou Σ, potom $\mathcal{L} = \mathcal{L}_1\mathcal{L}_2$ je jazyk nad abecedou Σ definovaný nasledovne: $\mathcal{L} = \{uv \quad | \quad u \in \mathcal{L}_1, v \in \mathcal{L}_2\}$. Jazyk možno zreťazovať so sebou samým; pre ľubovoľný jazyk $\mathcal{L}$ a ľubovoľné číslo $k \in \mathcal{N}$ definujeme:

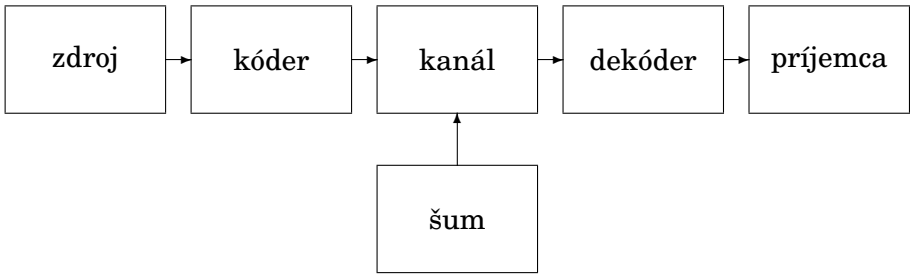
1. $\mathcal{L}^0 = \{\lambda\}$,
2. $\mathcal{L}^{k+1} = \mathcal{L}^k\mathcal{L}$.

Na záver uvedieme ešte dve operácie nad jazykmi, ktoré nám umožnia popísať množinu všetkých možných slov, ktoré sa dajú vytvoriť pomocou operácie zreťazovania jazyka. Nech $\mathcal{L}$ je ľubovoľný jazyk, potom jazyky $\mathcal{L}^+ = \bigcup_{i>0}^{\infty} \mathcal{L}^i$ a $\mathcal{L}^* = \bigcup_{i\geq 0}^{\infty} \mathcal{L}^i$ sa nazývajú kladná, resp. nezáporná iterácia jazyka $\mathcal{L}^i$. Všimnite si, že abecedu Σ možno chápať aj ako jazyk pozostávajúci zo všetkých slov dĺžky 1 nad abecedou Σ. Potom Σ^* označuje množinu všetkých slov nad abecedou Σ.

Ilustrujeme teraz zavedené pojmy na príkladoch.

2.2 Model prenosového kanála

Využijeme pojmový aparát, ktorý sme vybudovali v predchádzajúcej časti a zavedieme základné pojmy z teórie informácie a teórie kódovania. Budeme predpokladať, že pre-



Obrázok 1: Model prenosového kanála

nášané alebo uchovávané informácie majú podobu slova alebo slov nad nejakou abecedou. Transformácie, ktorými prechádza informácia pri prenose, resp. pri uchovávaní na pamäťovom médiu a čítaní z neho popíšeme pomocou modelu prenosového kanála uvedeného na obr. 1 Aby sme sa nemuseli zaoberať tým, odkiaľ informácia pochádza a ani zvláštnosťami jednotlivých zdrojov informácie, predpokladáme, že informácia je zapísaná v tvare postupnosti symbolov z abecedy Σ_S a vytvára ju nejaký *generátor–zdroj informácie*. Abecedu $\Sigma_S = \{s_1, \ldots, s_q\}$ budeme nazývať *abecedou zdroja,* alebo *zdrojovou abecedou*; informáciu, ktorá sa prenáša z jedného miesta na druhé budeme nazývať *správou* a informáciu, ktorá sa uchováva na pamäťovom médiu, *údajmi*. Existuje viacero matematických modelov zdroja informácie (generátora), ktorými sa podrobnejšie budeme zaoberať v kapitole 3. Správa, ktorú vytvoril zdroj informácií je určená pre vzdialeného príjemcu. Na prenos správy od zdroja informácie k príjemcovi/adresátovi slúži *prenosový kanál*. Prenosový kanál má vlastnú abecedu, kanálovú abecedu Σ_C, ktorá je vo všeobecnosti rôzna od abecedy zdroja Σ_S. Prenosový kanál z matematického hľadiska predstavuje transformáciu, ktorá vstupnú hodnotu (správu m) zobrazí na výstupnú hodnotu (správu m'). V ideálnom prípade je transformácia realizovaná prenosovým kanálom identická, t.j. $m = m'$, v reálnom živote však činnosť prenosového kanálu komplikuje šum. Šum spôsobuje v správach prenášaných prenosovým kanálom zmeny (chyby) dvojakého typu:

1. nahradenie jedného symbolu prenášanej správy iným symbolom (z abecedy Σ_C),
2. výpadok symbolu, doplnenie nového symbolu (z abecedy Σ_C) do prenášanej správy (poruchy synchronizácie).

V tejto knihe sa budeme zaoberať kódmi, odhaľujúcimi, resp. opravujúcimi chyby prvého typu. Poruchy synchronizácie ??? Výskyt chýb v prenesenej správe závisí od šumu. Šum je však výsledkom celého radu rozličných vplyvov, ktoré nie je možné deterministicky popísať. Budeme preto predpokladať, že chyby v jednotlivých symboloch prenášanej správy vznikajú nezávisle na sebe s rovnakými pravdepodobnosťami.

Ak neplatí vzťah $\Sigma_S \subseteq \Sigma_C$, správu zapísanú v zdrojovej abecede bude pred vstupom do prenosového kanálu potrebné zapísať pomocou symbolov z abecedy Σ_C. Táto transfor-

mácia je čiastočným zobrazením[1], ktoré sa nazýva kódovanie správy ($\mathsf{ENC} : \Sigma_S^* \to \Sigma_C^*$) a realizuje ju *kóder*. Výsledkom kódovania správy je *kódovaná* alebo *zakódovaná správa*. Zakódovaná správa vstupuje do prenosového kanála a po prenesení sa spätne transformuje v *dekóderi* na pôvodnú správu nad abecedou Σ_S. Transformácia, ktorú realizuje dekóder je čiastočné zobrazenie ($\mathsf{DEC} : \Sigma_C^* \to \Sigma_S^*$), ktoré sa nazýva dekódovanie (správ) a je inverznou transformáciou ku kódovaniu. Transformácie $\mathsf{ENC}, \mathsf{DEC}$ by mali spĺňať požiadavku jednoznačnosti dekódovania:

$$\forall m \in \Sigma_S^* : \mathsf{DEC}(\mathsf{ENC}(m)) = m.$$

Všimneme si, že hoci sme v modeli prenosového kanála hovorili o prenose správ, možno tento model priamo použiť aj na popis uchovávania a opätovného čítania údajov. Zaznamenávanie údajov a ich opätovné čítanie možno chápať ako prenos informácie v čase, zatiaľ čo pri prenose správ sa jedná o prenos informácie v priestore. Vzhľadom na túto podobnosť sa v ďalšom budeme zaoberať problémami vznikajúcimi pri prenose informácie v priestore.

Poznámka. V modeli prenosového kanála sme abstrahovali od technickej realizácie jednotlivých častí systému. V reálnych systémoch sa však kanálom neprenášajú nejaké abstraktné symboly

2.3 Kódovanie

Vráťme sa k modelu prenosového kanála z predchádzajúcej časti. Predpokladajme, že prenosový kanál je odolný voči šumu; t.j. že realizuje identickú transformáciu a prenáša správy bez zmeny. Aj v tomto prípade však zostáva vyriešiť, ako zapisovať správy nad abecedou Σ_S pomocou kanálovej abecedy Σ_C. Existuje viacero riešení tohto problému. Začneme tým najjednoduchším; kódovaním znakov zdrojovej abecedy. Nech $\Sigma_S = \{s_0, \ldots, s_{m-1}\}$ je zdrojová abeceda a $\Sigma_C = \{b_0, \ldots, b_r\}$ je abeceda prenosového kanálu (v ďalšom ju budeme nazývať *kódovou abecedou*) a nech sú $v_0, \ldots, v_{m-1}$ navzájom rôzne slová nad kódovou abecedou Σ_C. Potom zobrazenie

$$\begin{aligned} s_0 &\to v_0 \\ s_1 &\to v_1 \\ &\vdots \\ s_{m-1} &\to v_{m-1} \end{aligned}$$

budeme nazývať *kódovaním symbolov zdrojovej abecedy* slovami nad abecedou Σ_C. Množina $V = \{v_0, \ldots, v_{m-1}\}$ sa nazýva *kód* a prvky množiny V sa nazývajú *kódovými slovami*. Všimneme si, že na rozdiel od kódovacej transformácie z predchádzajúcej kapitoly, je

[1] kódovanie môže byť definované na množine správ, ktorá sa nemusí zhodovať s množinou Σ_S^*. Na druhej strane, keďže ENC nemusí byť surjekcia, DEC nemusí byť všade definované zobrazenie.

kódovanie po písmenách totálnym (všade definovaným) zobrazením a keďže zdroj S generuje len postupnosti znakov nad abecedou Σ_S, každá správ vytvorená zdrojom S sa dá vyjadriť pomocou postupnosti kódových slov kódu V. problém však vzniká pri dekódovaní kódovaných správ. Predpokladajme, že je daná nejaká správa $s_{i_1}, \ldots, s_{i_n}$ nad zdrojovou abecedou a jej prislúchajúca kódovaná správa vyjadrená ako postupnosť kódových slov $v_{i_1}, \ldots, v_{i_n}$. Postupnosť $v_{i_1}, \ldots, v_{i_n}$ sa však prenáša po znakoch a pred dekódovaném je potrebné ju rozdeliť na kódové slová. Ak sa to podarí, nie je problém dekódovať jednotlivé kódové slová a získať pôvodnú správu $s_{i_1}, \ldots, s_{i_n}$. Nasledujúci príklad ukazuje, že existujú kódy, pre ktoré sa nie každá postupnosť kódových symbolov dá rozdeliť na kódové slová jednoznačne.

Príklad 2.1 *Abeceda zdroja* $\Sigma_S = \{0, 1, 2, 3\}$ *pozostáva zo štyroch symbolov (prvých štyroch desiatkových číslic) a kódová abeceda je binárna;* $\Sigma_C = \{0, 1\}$. *Kódovanie priraďuje desiatkovej číslici jej binárny zápis:*

$$\begin{array}{cccccc} 0 & \to & 0 & 1 & \to & 1 \\ 2 & \to & 10 & 3 & \to & 11 \end{array}$$

Binárna postupnosť 001011 *sa dá interpretovať viacerými spôsobmi, ako binárny zápis desiatkových postupností* $001011, 00103, 00211, 0023$.

Jednoznačnosť dekódovania je základnou požiadavkou, ktorá sa kladie na kódovanie. Nutným predpokladom jednoznačnosti dekódovania je *rozdeliteľnosť kódu*.

Definícia 2.1 *Kód* $V = \{v_0, \ldots, v_{m-1}\}$ *nad abecedou* Σ_C *sa nazýva rozdeliteľným, ak pre ľubovoľnú rovnosť postupností kódových slov*

$$v_{i_1} \ldots v_{i_k} = v_{j_1} \ldots v_{j_l}$$

platí $l = k, \quad i_1 = j + 1, \ldots, i_k = j_k$.

Čo vlastne vyjadruje rozdeliteľnosť kódu? Ak je kód V rozdeliteľný, znamená to, že ľubovoľnú postupnosť nad Σ_C^* buď môžeme rozdeliť na postupnosť kódových slov jednoznačným spôsobom, alebo ju nemôžeme rozdeliť vôbec. Jednoduchým riešením problému rozdeliteľnosti sú *blokové* alebo *rovnomerné kódy*. Blokový kód sa vyznačuje tým, že všetky jeho kódové slová majú rovnakú dĺžku.

Príklad 2.2 *Rozšírime predchádzajúci príklad a uvedieme dva spôsoby kódovania de-*

siatkových číslic.

desiatkový zápis	binárny zápis	blokový kód
0	0	0000
1	1	0001
2	10	0010
3	11	0011
4	100	0100
5	101	0101
6	110	0110
7	111	0111
8	1000	1000
9	1001	1001

Dekódovanie postupnosti znakov kódovej abecedy bude v prípade blokového kódu relatívne jednoduché: postupnosť sa najprv rozdelí na slová dĺžky rovnej dĺžke bloku a potom sa (napríklad na základe tabuľky) jednotlivým kódovým slovám priradia symboly zdrojovej abecedy.

Príklad 2.3 *Postupnosť* 10000100110010010011 0010 *rozdelíme na kódové slová:* 1000 0100 1100 1001 0011 0010 *a dekódujeme pomocou tabuľky z predchádzjúceho príkladu: 843932. Všimneme si, že existujú aj binárne postupnosti, ktoré sa nedajú dekódovať, nakoľko slová* $1111, 1110, 1101, 1100, 1011, 1010$ *nie sú kódové slová.*

S rozdeliteľnosťou majú problémy kódy pozostávajúce zo slov nerovnakej dĺžky; tzv. *nerovnomerné kódy*. Aj medzi nerovnomernými kódmi existujú rozdeliteľné kódy. Tieto kódy umožňujú zapisovať informáciu častokrát úspornejšie ako blokové kódy a používajú sa najmä na kompresiu údajov. Skôr, ako sa nimi budeme zaoberať podrobnejšie, zovšeobecníme pojem kódovania znakovo zdrojovej abecedy.

Nech je Σ_S zdrojová abeceda, nech je množina $U = \{u_0, \ldots, u_M\}$ nejakých slov nad zdrojovou abecedou a nech sú $v_0, \ldots, v_M$ slová nad kódovou abecedou Σ_C. Zobrazenie

$$\begin{aligned} u_0 &\to v_0 \\ u_1 &\to v_1 \\ &\vdots \\ u_M &\to v_M \end{aligned}$$

budeme nazývať kódovaním množiny U kódom V. Všimneme si, že táto definícia kódovania pokrýva aj kódovanie znakov zdrojovej abecedy; stačí položiť $U = \Sigma_S$.

Príklad 2.4 *Nech je $\mathcal{U}$ rovná množine všetkých podmnožín množiny prirodzených čísel* $\{0, \ldots .99\}$, $V = \{0,1\}^{100}$ *je množina binárnych vektorov dĺžky* 100. *Podmnožine* $\{i_0, \ldots, i_k\}$

z $\mathcal{U}$ je priradené slovo v_j, ktoré má jednotkové hodnoty na pozíciách $i_0, \ldots, i_k$ a nuly na ostatných pozíciách. Je zrejmé, že V kóduje množinu $\mathcal{U}$ a že toto kódovanie je bijekciou. Poznajúc mohutnosť kódu V vieme určiť aj mohutnosť množiny $\mathcal{U}$: $|\mathcal{U}| = 2^{100}$.

3 Nerovnomerné kódy

Základná motivácia pre používanie nerovnomerných kódov spočíva v tom, že sa prvky množiny ktorú kódujeme, v správach nevyskytujú rovnako často. To umožňuje priradiť často sa vyskytujúcim prvkom kratšie kódové slová a tak dosiahnuť, že kódovaná správa je v priemernom prípade kratšia, ako keby sa na kódovanie používali napríklad blokové kódy.

V tejto kapitole sa budeme zaoberať rôznymi rozdeliteľnými nerovnomernými kódami. Najprv zavedieme prefixové kódy, potom dokážeme Kraftovu-McMillanovu nerovnosť, ktorá poskytuje kritérium na rozdelenie dĺžok kódových slov rozdeliteľného kódu. Nakoniec zavedieme pojem ceny kódu, skonštruujeme dolný odhad na cenu kódu a budeme sa zaoberať konštrukciami optimálnych a kvázioptimálnych kódov. Kvôli zjednodušeniu budeme v priebehu tejto kapitoly predpokladať, že kódová abeceda je binárna.

3.1 Rozdeliteľné kódy

3.1.1 Prefixové kódy

Prefixové kódy predstavujú rozsiahlu triedu nerovnomerných kódov s dobrými vlastosťami, ktoré budeme často využívať v ďalších konštrukciách.[2]

Definícia 3.1 *Kód $V = v_0, \ldots, v_{m-1}$ sa nazýva prefixovým kódom, ak pre ľubovoľné $v_i, v_j \in V$, $i \neq j$; v_i nie je prefixom slova v_j.*

Kód z príkladu 2.1 nebol prefixový; kódové slovo 1 bolo prefixom kódového slova 10. Dokážeme, že prefixové kódy sú rozdeliteľné.

Veta 3.1 *Ak je kód $V = v_0, \ldots, v_{m-1}$ prefixový, potom je rozdeliteľný kód.*

Dôkaz. Predpokladajme, že V je prefixový, ale nie je rozdeliteľný kód. Potom existuje aspoň jedna binárna postupnosť, ktorá je rozdeliteľná na postupnosť kódových slov aspoň dvoma rozličnými spôsobmi. Vyberieme zo všetkých takých binárnych postupností postupnosť β s minimálnou dĺžkou. Pre postupnosť β teda platí :

$$v_{i_1} \ldots v_{i_k} = v_{j_1} \ldots v_{j_l}.$$

[2] z hľadiska praktického použitia je zaujímavé aj to, že pre prefixové kódy existujú efektívne metódy dekódovania.

Z toho, že postupnosť β má minimálnu dĺžku vyplýva, že $v_{i_1} \neq v_{j_1}$. V opačnom prípade by bolo totiž možné slovo v_{i_1} z postupnosti β vynechať a dostali by sme kratšiu binárnu postupnosť, pre ktorú by platilo:

$$v_{i_2} \dots v_{i_k} = v_{j_2} \dots v_{j_l}.$$

To je však v spore s predpokladom o minimálnej dĺžke postupnosti β. Ak však $v_{i_1} \neq v_{j_1}$, potom buď slovo v_{i_1} je prefixom slova v_{j_1} alebo slovo v_{j_1} je prefixom slova v_{i_1}. To je zasa v spore s predpokladom o tom, že kód V je prefixový. To znamená, že postupnosť spĺňajúca podmienku nemôže existovať a kód V je rozdeliteľný. ■

Prirodzenou otázkou je, či existujú aj iné rozdeliteľné kódy okrem prefixových. Uvedieme príklad rozdeliteľného kódu, ktorý nie je prefixový.

Príklad 3.1 *Kód* $V = \{0, 01, 11\}$ *nie je prefixový, ale napriek tomu to je rozdeliteľný kód.*

Kód z predchádzajúceho príkladu je tzv. sufixový kód, ktorý je charakteristický tým, že žiadne kódové slovo nie je sufixom iného kódového slova. Vytvorili sme ho tak, že sme „otočili“ slová prefixového kódu $\{0, 10, 11\}$. Postupnosť kódových symbolov budeme rozdeľovať na kódové slová „odzadu“. Príkladom takejto postupnosti, ktorá sa nedá rozdeliť na postupnosť kódových slov, kým sa nedočíta do konca, je postupnosť:

$$0111\dots1$$

Ak táto postupnosť obsahuje párny počet jednotiek (napr. $2k$), dá sa rozdeliť nasledovne: $0(11)^k$; ak obsahuje nepárny počet jednotiek $(2k + 1)$, tak sa rozdelí na kódové slová nasledovne: $01(11)^k$.

- kódové stromy
- silne rozdeliteľné kódy
- automatové dekódovanie

3.1.2 Kraftova - McMillanova nerovnosť

Kvôli zjednodušeniu výkladu prijmeme niektoré predpoklady: budeme predpokladať, že zdrojová abeceda Σ_S obsahuje aspoň dva symboly, t.j. $m \geq 2$, že kódová abeceda je binárna a že sa v rozdelení pravdepodobností P nevyskytujú nulové pravdepodobnosti.

Veta 3.2 (Kraftova-McMillanova nerovnosť) *Nech sú* $l_0, \dots, l_{m-1}$ *ľubovoľné nenulové prirodzené čísla. Potom rozdeliteľný kód* $V = \{v_0, \dots, v_{m-1}\}$ *s dĺžkami kódových slov* $l_i = l(v_i)$; $i = 0, \dots, m-1$ *existuje práve vtedy, ak platí nasledujúca nerovnosť*

$$\sum_{i=0}^{m-1} 2^{-l_i} \leq 1. \tag{1}$$

Dôkaz. Najprv dokážeme, že podmienka je nutná. Nech je $V = \{v_0, \ldots, v_{m-1}\}$ ľubovoľný kód. Priradíme mu generujúcu funkciu (enumerátor dĺžok kódových slov):

$$h_V(x) = \sum_{i=0}^{m-1} x^{-l(v_i)}. \tag{2}$$

Zavedieme teraz n-násobné zreťazenie (rozšírenie) kódu V;

$$V^n = \{w_i;\ w_i = v_{i_1} \ldots v_{i_n},\ v - i_j \in V,\ j = 1, \ldots, n\}$$

Príklad 3.2 *Uvažujme kód* $V = \{0, 10, 11\}$. *Jeho vytvárajúca funkcia je* $h_V(x) = x^{-1} + x^{-2} + x^{-2} = x^{-1} + 2x^{-2}$. *Dvojnásobným zreťazaním kódu* V *dostávame kód*

$$V^2 = \{00, 010, 011, 100, 1010, 1011, 110, 1110, 1111\}$$

s enumerátorom

$$h_{V^2}(x) = x^{-2} + 4x^{-3} + 4x^{-4} = (x^{-1} + 2x^{-2})^2.$$

Pokračovanie dôkazu. Matematickou indukciou možno dokázať, že

$$h_{V^n}(x) = \left(\sum_{i=0}^{m-1} x^{-l(v_i)} \right)^n. \tag{3}$$

Predpokladajme teraz, že $V = \{v_0, \ldots, v_{m-1}\}$ je rozdeliteľný kód a že $l_i = l(v_i),\ i = 0, \ldots, m-1$. Symbolom M_i označíme počet slov dĺžky i v kóde V^n a namiesto toho, aby sme v sume ???? sčítavali cez jednotlivé slová, budeme sčítavať cez dĺžky slov (pozri príklad 3.2). Maximálnu dĺžku slova v kóde V označíme symbolom l_{max}. Dosadíme namiesto premennej x hodnotu 2 a dostávame:

$$h_{V^n}(2) = \sum_{i=1}^{n \cdot l_{max}} 2^{-i} M_i. \tag{4}$$

Je zrejmé, že ak V neobsahuje slovo λ, tak každé slovo v kóde V^n bude mať dĺžku minimálne $n \cdot l_{min}$, kde l_{min} je minimálna dĺžka kódového slova kódu V. Potom $M_1 = M_2 = \cdots = M_{n \cdot l_{min}-1} = 0$. Teraz využijeme skutočnosť, že kód V je rozdeliteľný. Z toho vyplýva, že všetky slová kódu V^n sú rôzne, a keďže V^n je binárny kód, znamená to, že $M_i \leq 2^i$. V opačnom prípade by sa aspoň jedna binárna postupnosť dĺžky i musela dať poskladať zo slov kódu V rôznymi spôsobmi. Ale to je v spore s predpokladom o rozdeliteľnosti kódu V. Na druhej strane, niektoré binárne postupnosti sa nemusia dať poskladať zo slov kódu V a v tomto prípade $M_i < 2^i$. Dosadíme horný odhad hodnoty M_i do vzťahu ???? a po jednoduchých úpravách dostávame:

$$h_{V^n}(2) = \sum_{i=1}^{n \cdot l_{max}} 2^{-i} M_i \leq \sum_{i=1}^{n \cdot l_{max}} 2^{-i} 2^i = n \cdot l_{max}. \tag{5}$$

Na druhej strane, zo vzťahov xxx xxx vyplýva

$$h_{V^n}(2) = \left(\sum_{i=0}^{m-1} x^{-l(v_i)} \right)^n \leq n \cdot l_{max}. \tag{6}$$

Posledná nerovnosť platí pre ľubovoľné n. Ak by teda

$$\sum_{i=0}^{m-1} x^{-l(v_i)} = a > 1,$$

tak by existovalo také n_0, že pre všetky $n > n_0$ by

$$a^n > n \cdot l_{max},$$

čo je v spore so vzťahom xxx . To znamená, že

$$\sum_{i=0}^{m-1} x^{-l(v_i)} \leq 1.$$

Dostatočnosť. Predpokladajme, že $l_0 \leq l_1 \cdots < l_{m-1}$; dĺžky kódových slov sú usporiadané vzostupne, a že platí Kraftova-McMillanova nerovnosť. Ukážeme, že je možné zostrojiť rozdeliteľný kód s dĺžkami kódových slov $l_0, \ldots, l_{m-1}$.
1. konštrukcia [1]. Použijeme matematickú indukciu.

1. Vyberieme ľubovoľné binárne slovo dĺžky l_0 ako kódové slovo v_0.

2. Predpokladáme, že sme už vybrali slová $v_0, \ldots, v_{k-1},\ k \leq m-1$, dĺžok $l_0, \ldots, l_{k-1}$ ktoré tvoria rozdeliteľný (prefixový) kód.

3. Nájdeme také slovo v_k dĺžky l_k, pre ktoré žiadne zo slov $v_0, \ldots, v_{k-1}$ nie je prefixom. Ukážeme, že také slovo existuje. Všetkých binárnych slov dĺžky l_k, ktoré majú prefix v_0 dĺžky l_0 je $2^{l_k - l_0}$. (Prvých l_0 bitov sa zhoduje so slovom v_0, ostatných $l_k - l_0$ bitov možno vybrať ľubovoľným spôsobom.)Všetkých binárnych slov dĺžky l_k, ktorých prefixom je niektoré zo slov $l_0, \ldots, l_{k-1}$ je

$$\sum_{i=0}^{k-1} 2^{l_k - l_i}. \tag{7}$$

Z Kraftovej-McMillanovej nerovnosti však vyplýva, že

$$\sum_{i=0}^{m-1} 2^{-l_i} = \sum_{i=0}^{k-1} 2^{-l_i} + 2^{-l_k} + \cdots + 2^{-l_{m-1}} \leq 1, \tag{8}$$

resp.

$$\sum_{i=0}^{k-1} 2^{-l_i} + 2^{-l_k}. \tag{9}$$

Vynásobíme rovnosť 9 hodnotou 2^{l_k} a upravíme

$$\sum_{i=0}^{k-1} 2^{l_k - l_i} \leq 2^{l_k} - 1. \tag{10}$$

Zo vzťahu 10 vyplýva, že, existuje aspoň jeden binárny vektor dĺžky l_k, ktorého prefixom nie je žiadne zo slov $\nu_0, \ldots, \nu_{k-1}$. Vyberieme tento vektor ako kódové slovo ν_k.

Predchádzajúci dôkaz mal skôr existenčný ako konštruktívny charakter. Dokážeme ešte raz dostatočnosť Kraftovej-McMillanovej nerovnosti pomocou Shannonovskej konštrukcie.

2. konštrukcia [4]. Rovnako ako v predchádzajúcom dôkaze budeme predpokladať, že $l_0 \leq l_1 \cdots < l_{m-1}$; dĺžky kódových slov sú usporiadané vzostupne, a že platí Kraftova-McMillanova nerovnosť. Zavedieme čísla $q_i,\ i = 0, \ldots, m-1$ nasledovne:

$$q_0 = 0, \quad q_k = \sum_{i=0}^{k-1} 2^{-l_i}; \quad k = 1, \ldots, m-1.$$

Z Kraftovaej-McMillanovej nerovnosti vyplýva, že $0 \leq q_k < 1$. Zapíšeme q_k v binárnom tvare. Zo spôsobu vytvárania q_k a skutočnosti, že $l_0 \leq l_1 \cdots < l_{m-1}$, vyplýva, že q_k sa dá zapísať ako $q_k = (0.b_{k,1} \ldots b_{k,l_{k-1}})_2$, kde $b_{k,i} \in \{0, 1\}$. Kód $V = \{\nu_0, \ldots, \nu_{m-1}\}$ vytvoríme z binárne zapísaných čísel q_k nasledujúcim spôsobom:

$$\nu_i = \underbrace{b_{i,1} \ldots b_{i,l_{i-1}} 0 \ldots 0}_{l_i};$$

t.j. slovo ν_i pozostáva z prvých l_i binárnych číslic nasledujúcich po rádovej čiarke v rozvoji čísla q_i. Tvrdíme, že takto zostrojený kód je prefixový, a teda aj rozdeliteľný. Predpokladajme opak, t.j. že kód V nie je prefixový. Potom obsahuje kódové slová, z ktorých jedno je prefixom druhého. Nech h je najmenšie také číslo, že pre slovo ν_h existuje kódové slovo (označme ho symbolom ν_i), ktoré je jeho prefixom. Keďže ν_i je prefixom ν_h, $l_i < l_h$, a teda aj $i < h$. Z toho že ν_i je prefixom ν_h a zo spôsobu konštrukcie kódových slov vyplýva, že ν_i je prefixom slov $\nu_{i+1}, \ldots, \nu_{h-1}, \nu_h$. Keďže ν_h je prvé kódové slovo, ktoré má prefix ν_i, $h = i + 1$. Pozrieme sa teraz na slová $\nu_i,\ \nu_{i+1}$ podrobnejšie, preskúmame čísla $q_i,\ q_{i+1}$.

$$\begin{aligned} q_i &= 0.\overbrace{\underbrace{b_{i,1} \ldots b_{i,l_{i-1}}}_{l_{i-1}} 0 \ldots 0}^{l_i} \\ q_{i+1} = q_i + 2^{-l_i} &= 0.\underbrace{b_{i,1} \ldots b_{i,l_{i-1}} 0 \ldots 10 \ldots 0}_{l_{i+1}} \end{aligned}$$

Môžu nastať dve možnosti:

1. $l_i < l_{i+1}$; v tomto prípade má slovo v_i na mieste l_i znak 0 a slovo v_{i+1} znak 1;
2. $l_i = l_{i+1}$. Pripočítaním hodnoty 2^{-l_i} ku q_i sa zmení niektorá z l_i číslic čísla q_i, a teda slová v_i a v_{i+1} sa odlišujú aspoň v jednom z prvých l_i znakov.

To znamená, že v_i nemôže byť prefixom slova v_{i+1}, a teda kód V je prefixový. ■

Dôsledok 1 *Pre ľubovoľný rozdeliteľný kód* $V = \{v_0, \ldots, v_{m-1}\}$ *existuje prefixový kód* $W = \{w_0, \ldots, w_{m-1}\}$*, taký, že* $l(v_i) = l(v_i),\ i = 0, \ldots, m-1$.

Z uvedeného dôsledku vyplýva, že ak nám nezáleží na konkrétnej podobe kódových slov, môžeme rozdeliteľný kód nahradiť prefixovým kódom s rovnakými dĺžkami kódových slov. Túto možnosť budeme v ďalších úvahách využívať. Skôr ako budeme pokračovať v skúmaní vlastností nerovnomerných kódov, uvedieme príklad Shannonovho kódu, ktorý sme použili v dôkaze Kraftovej-McMillanovej nerovnosti.

Príklad 3.3 *Uvažujme nasledujúce dĺžky kódových slov: 2,3,3,4,5,5,6,6. Keďže* $2^{-2} + 2^{-3} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-5} + 2^{-6} + 2^{-6} < 1$*, z vety 3.2 vyplýva, že existuje prefixový kód s týmito dĺžkami kódových slov. Vypočítame hodnoty* q_i *a vyjadríme príslušné kódové slová.*

$$\begin{array}{lll} q_0 = 0.00 & l_0 = 2 & v_0 = 00 \\ q_1 = 0.01 & l_1 = 3 & v_1 = 010 \\ q_2 = 0.011 & l_2 = 3 & v_2 = 011 \\ q_3 = 0.1 & l_3 = 4 & v_3 = 1000 \\ q_4 = 0.1001 & l_4 = 5 & v_4 = 100010 \\ q_5 = 0.10011 & l_5 = 5 & v_5 = 100011 \\ q_6 = 0.101 & l_6 = 6 & v_6 = 101000 \\ q_7 = 0.101001 & l_7 = 6 & v_7 = 101001 \end{array}$$

3.1.3 Úplné kódy

Kód z príkladu 3.3 je prefixový, ale má jeden nedostatok. Existujú binárne postupnosti, ktoré sa nedajú rozbiť na kódové slová. Okrem triviálnych postupností dĺžky 1 sú to napríklad postupnosti začínajúce dvojicou symbolov 11. Nemá však zmysel požadovať, aby platilo $V^* = B^*$, pretože to je možné len v prípade, ak $B \subseteq V$. Intuitívnej požiadavke, aby každá binárna postupnosť predstavovala alebo sa dala doplniť na postupnosť kódových slov, vyhovujú tzv. *úplné kódy*.

Definícia 3.2 *Binárny rozdeliteľný kód* V *sa nazýva úplným kódom práve vtedy, ak pre ľubovoľnú binárnu postupnosť* $\beta \in B^*$ *existuje také kódové slovo* $v_i \in V$*, že buď postupnosť* β *je prefixom slova* v_i*, alebo slovo* v_i *je prefixom postupnosti* β.

Príklad 3.4 *Uvažujme blokový kód* $V = \{00, 01, 10, 11\}$*. Keďže kód* V *obsahuje všetky binárne slová dĺžky 2, spĺňa podmienky definície 3.2 a je úplný. Každú binárnu postupnosť párnej dĺžky možno jednoznačne rozdeliť na postupnosť kódových slov.*

Overovať, či nejaký kód s veľkým počtom kódových slov spĺňa podmienky definície 3.2, by nemuselo byť jednoduché. Našťastie úplnosť kódu úzko súvisí s Kraftovou-McMillanovou nerovnosťou a prefixovými kódmi.

Veta 3.3 *Binárny rozdeliteľný kód* $V = \{v_0, \ldots, v_{m-1}\}$ *je úplný práve vtedy, ak je prefixový a platí*

$$\sum_{i=0}^{m-1} 2^{-l_i} = 1. \tag{11}$$

Dôkaz. Predpokladajme, že $V = \{v_0, \ldots, v_{m-1}\}$ je prefixový kód, pre ktorý platí rovnosť (11), ale V nie je úplný. To znamená, že existuje binárna postupnosť $\beta \in B^*$ taká, že žiadne kódové slovo $v_i \in V$ nie je prefixom postupnosti β a postupnosť β nie je prefixom žiadneho kódového slova kódu V. Potom však môžeme zostrojiť nový prefixový kód $V' = V \cup \{\beta\}$, pre ktorý platí

$$\sum_{i=0}^{m-1} 2^{-l_i} + 2^{-l(\beta)} = 1 + 2^{-l(\beta)} > 1. \tag{12}$$

Ale nerovnosť (12) je v rozpore s Kraftovou-McMillanovou nerovnosťou (9). To znamená, že postupnosť β požadovaných vlastností nemôže existovať, a teda kód V je úplný.

Dokážeme druhú časť tvrdenia sporom. Nech je V úplný rozdeliteľný kód. Predpokladajme, že V nie je prefixový, alebo preň neplatí rovnosť (11). Keďže z rozdeliteľnosti kódu V vyplýva platnosť Kraftovej-McMillanovej nerovnosti (9), znamená to, že pre kód V platí

$$\sum_{i=0}^{m-1} 2^{-l_i} < 1. \tag{13}$$

Zhrnieme naše predpoklady: kód V je úplný a platí $\sum_{i=0}^{m-1} 2^{-l_i} < 1$. Z úplnosti kódu V vyplýva, že každá binárna postupnosť dĺžky $n > l_{max}$, kde

$$l_{max} = \max_{v_i \in V}\{l(v_i)\}$$

musí mať ako prefix nejaké kódové slovo. Spočítame počet takýchto postupností:

$$\sum_{i=0}^{m-1} 2^{n-l_i} \geq 2^n. \tag{14}$$

Ak je kód V prefixový, potom je kódové slovo, ktoré je prefixom nejakej binárnej postupnosti dĺžky n dané jednoznačne. Potom však platí

$$\sum_{i=0}^{m-1} 2^{n-l_i} = 2^n, \tag{15}$$

a

$$\sum_{i=0}^{m-1} 2^{-l_i} = 1, \tag{16}$$

čo je v spore s predpokladom (13) To znamená, že platí $\sum_{i=0}^{m-1} 2^{-l_i} < 1$, kód V je úplný ale nie je prefixový. Potom však existujú kódové slová $v_i \neq v_j$ také, že (napr.) v_i je prefixom v_j. Z úplnosti kódu V vyplýva, že každá binárna postupnosť dĺžky $n > l_{max}$ musí začínať nejakým kódovým slovom kódu V. Potom

$$\sum_{i=0}^{m-1} 2^{n-l_i} > 2^n, \tag{17}$$

lebo postupnosti začínajúce slovom v_j sú už zarátané v sume (17) ako postupnosti začínajúce slovom v_i. Na druhej strane nemôže platiť nerovnosť

$$\sum_{k=0}^{m-1} 2^{n-l_k} - 2^{l(v_j)} < 2^n, \tag{18}$$

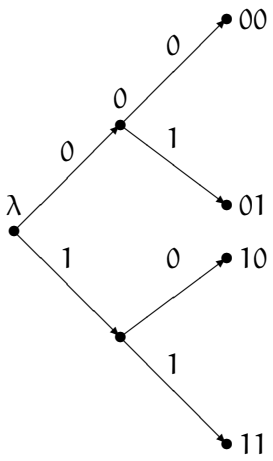
pretože to by znamenalo, že odstránením slova v_j z kódu V sa stratí úplnosť kódu; t.j. potom bude existovať binárna postupnosť β dĺžky $n > l_{max}$, ktorej prefixom nie je žiadne kódové slovo kódu V. Ale to znamená, že jej prefixom nemohlo byť odstránené slovo v_j, pretože v tom prípade by prefixom postupnosti β bolo slovo v_i, a teda kód V by nebol úplný. To znamená, že platí nerovnosť (17). Platnosť nerovnosti (17) je však v rozpore s tvrdením vety 3.2. Dostávame spor, ktorý dokazuje platnosť nášho tvrdenia. ■

3.1.4 Kódové stromy

Na skúmanie vlastností nie príliš rozsiahlych nerovnomerných kódov je možné výhodne používať orientovaný ohodnotený graf, nazývaný *kódovým stromom*. Uvažujme orientovaný binárny strom $\mathcal{T}$ hĺbky n s hranami a vrcholmi ohodnotenými nasledujúcim spôsobom: najprv ohodnotíme jeho hrany, pričom budeme postupovať od koreňa k listom; ak z vrcholu vychádzajú dve (neohodnotené) hrany tak jednej z nich priradíme hodnotu 0 a druhej hodnotu 1. Ak z vrcholu vychádza jediná (neohodnotená) hrana, priradíme jej jednu z hodnôt $\{0, 1\}$. Po ohodnotení hrán ohodnotíme vrcholy binárneho stromu $\mathcal{T}$: koreňu priradíme prázdne slovo λ a vrcholu v priradíme postupnosť binárnych hodnôt, ktoré boli priradené hranám ležiacim na ceste, spájajúcej koreň s vrcholom v.[3] Keďže $\mathcal{T}$ je súvislý acyklický graf, medzi ľubovoľnými dvoma vrcholmi v ňom existuje jediná cesta, a teda binárna postupnosť priradená vrcholu je určená jednoznačne. Binárny kódový strom binárneho kódu V; $\mathcal{T}(\mathcal{V})$ dostaneme tak, že z binárneho stromu $\mathcal{T}$ hĺbky $n \geq l_{max}$, kde $l_{max} = \max_{v_i \in V}\{l(v_i)\}$ a binárny strom $\mathcal{T}$ je ohodnotený spôsobom uvedeným vyššie, odstránime všetky podstromy, ktoré neobsahujú vrchol s ododnotením zodpovedajúcim niektorému kódovému slovu kódu V. Na obr. 2 je zobrazený binárny (ohodnotený) strom hĺbky 2.

Binárny kódový strom kódu V z príkladu 3.3 je zobrazený na obr. Všimneme si, že všetky vrcholy zodpovedajúce kódovým slovám, sú listy (vrcholy, z ktorých nevychádzajú žiadne hrany). To nie je náhoda. Ak by nejaké slovo v_i bolo prefixom iného slova v_j, vrchol v_i by musel ležať na ceste spájajúcej koreň s vrcholom v_j, a teda by musel byť vnútorným vrcholom kódového stromu.

[3] V ďalšom budeme vrchol v označovať binárnym slovom, ktoré mu je priradené.



Obrázok 2: Ohodnotený binárny strom

Veta 3.4 *Nech je V prefixový kód. Potom v kódovom strome $\mathcal{T}(\mathcal{V})$ zodpovedajú kódovým slovám listy.*

Dôkaz. Prenechávame čitateľovi.

Pomocou kódového stromu je možné ľahšie formulovať aj podmienku úplnosti kódu. Ako sme už ukázali, kód V z príkladu 3.3 nie je úplný; problémy spôsobujú postupnosti začínajúce dvojicou 11. Pri skúmaní kódového stromu kódu V zistíme, že z vrcholu 1 vychádza len jedna hrana, ktorej je priradená hodnota 0. Ak túto hranu odstránime a vrchol 1 stotožníme s pôvodným vrcholom 10, dostaneme kódový strom $\mathcal{T}(\mathcal{V}')$ prefixového kódu $V' = \{00, 010, 011, 100, 1010, 1011, 11000, 11001\}$.

Kódový strom $\mathcal{T}(\mathcal{V}')$ obsahuje ešte dva vnútorné vrcholy (11, 110) stupňa 1. Odstránením hrán vychádzajúcich z týchto vrcholov, vrcholu 110 a stotožnením vrcholov 11 a 1100 stromu $\mathcal{T}(\mathcal{V}')$ dostávame kódový strom $\mathcal{T}(\mathcal{V}'')$ prefixového kódu $V'' = \{00, 010, 011, 100, 1010, 1011, 110, 111\}$. Pre kód V'' platí $\sum_{v_i \in V''} 2^{l(v_i)} = 1$. Kód V'' je úplný. Každý binárny[4] prefixový kód, ktorý nie je úplný, možno týmto spôsobom transformovať na úplný kód.

3.1.5 Automatové dekódovanie.

Veľkou prednosťou prefixových kódov je to, že okamžite po dočítaní posledného symbolu kódového slova dokážeme určiť, o aké kódové slovo ide. (Pre porovnanie pripomíname sufixový rozdeliteľný kód z príkladu 3.1, pre ktorý existovali správy, ktoré bolo možné

[4]Ako uvidíme neskôr, mnohé z vlastností nerovnomerných kódov nezávisia od počtu znakov kódovej abecedy. Transformácia prefixového kódu na úplný prefixový kód, ktorú sme popísali vyššie, podstatne využíva to, že kódová abeceda je binárna; a nedá sa priamo zovšeobecniť na prípad kódovej abecedy s väčším počtom kódových symbolov.

dekódovať až po prijatí posledného symbolu správy.) Prefixové kódy sa vďaka možnosti priebežného dekódovania správy nazývajú aj okamžitými kódmi alebo automatovými kódmi. Ten druhý názov získali vďaka tomu, že na ich dekódovanie možno použiť konečný automat.

Definícia 3.3 *Konečný automat je usporiadaná šestica* $\mathcal{A} = (\Sigma_i, \Sigma_o, Q, \Phi, \Psi, q)$*, kde* Σ_i *je vstupná,* Σ_o *výstupná abeceda,* Q*, je konečná množina stavov,* $\Phi : \Sigma_i \times Q \to Q$ *je prechodová funkcia,* $\Psi : \Sigma_i \times Q \to \Sigma_o$ *je výstupná funkcia a* q *je počiatočný stav konečného automatu* $\mathcal{A}$*.*

Konečný automat si môžeme predstaviť ako zariadenie so vstupnou a výstupnou páskou, riadiacou jednotkou, čítacou a zapisovacou hlavou, obr. 3. Vstupná páska je rozdelená na políčka, v každom políčku je zapísaný symbol vstupnej abecedy. Podobne je výstupná páska rozdelená na políčka a v políčku je zapísaný jeden zo symbolov výstupnej abecedy, alebo je políčko prázdne. Čítacia hlava sa pohybuje po vstupnej páske zľava doprava, v každom kruku číta jeden symbol z políčka vstupnej pásky a po prečítaní sa presunie o jedno políčko doprava. Zapisovacia hlava v každom kroku zapisuje na políčko výstupnej pásky jeden symbol výstupnej abecedy a posunie sa o jedno políčko doprava, alebo nezapíše nič a zostáva na tom istom políčku aj v nasledujúcom kroku. Automat začína pracovať v počiatočnom stave q a skončí, keď prečíta celý vstup zo vstupnej pásky.

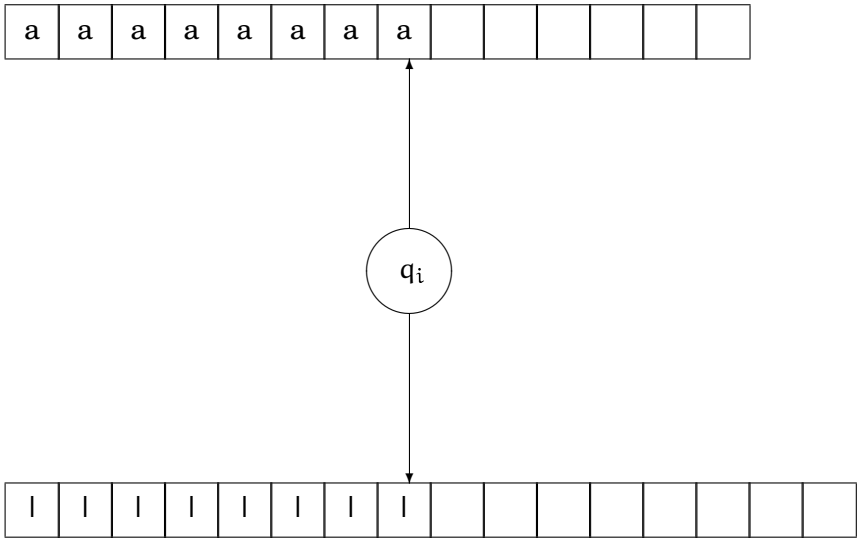
Pri dekódovaní binárnych prefixových kódov pomocou konečného automatu bude vstupná abeceda $\Sigma_i = \{0, 1\}$, výstupná abeceda sa bude zhodovať so zdrojovou abecedou; $\Sigma_o = \Sigma_S$ a prechodovú a výstupnú funkciu definujeme pomocou tabuľky. Ilustrujeme dekódovanie binárnárneho prefixového kódu na príklade.

Príklad 3.5 *Uvažujme kód* V'' *z predchádzajúceho príkladu. Predpokladáme, že kódové slová slúžia na zápis prvých písmen anglickej abecedy:*

a	*00*	*e*	*1010*
b	*010*	*f*	*1011*
c	*011*	*g*	*110*
d	*100*	*h*	*111*

Vstupná abeceda konečného (dekódovacieho) automatu $\mathcal{A}$ *je binárna:* $\Sigma_i = \{0, 1\}$*, výstupná abeceda* $\Sigma_o = \{a, b, c, d, e, f, g, h, \lambda\}$*, množina stavov* $Q = \{q, q_0, q_1, q_{01}, q_{10}, q_{11}, q_{101}\}$ *a prechodová a výstupná funkcia sú uvedené v tabuľke. Počiatočným stavom je* q*.*

stav	*vstup*	
	0	1
q	q_0, λ	q_1, λ
q_0	q, a	q_{01}, λ
q_{01}	q, b	q, c
q_1	q_{10}, λ	q_{11}, λ
q_{10}	q, d	q_{101}, λ
q_{101}	q, e	q, f
q_{11}	q, g	q, h



Obrázok 3: Konečný automat

Je daná binárne kódovaná správa 010011111. *Ukážeme, ako ju automat $\mathcal{A}$ dekóduje. Kvôli jednoduchosti budeme pozíciu čítacej hlavy na vstupnej páske a stav automatu $\mathcal{A}$ zapisovať tak, že stav automatu zapíšeme pred symbol, ktorý v danom kroku automat $\mathcal{A}$ číta. Symboly, ktoré by sa zapisovali na výstupnej páske budeme zapisovať pod dekódované slová binárnej správy.*

$$
\begin{aligned}
&q010011111 \mapsto 0q_0 10011111 \mapsto 01q_{01}0011111 \mapsto \underbrace{010}_{b} q011111 \mapsto \\
&\underbrace{010}_{b} 0q_0 11111 \mapsto \underbrace{010}_{b} 01q_{01}1111 \mapsto \underbrace{010}_{b}\underbrace{011}_{c} q111 \mapsto \underbrace{010}_{b}\underbrace{011}_{c} 1q_1 11 \mapsto \\
&\underbrace{010}_{b}\underbrace{011}_{c} 11q_{11}1 \mapsto \underbrace{010}_{b}\underbrace{011}_{c}\underbrace{111}_{h} q
\end{aligned}
$$

3.2 Cena kódu

Nerovnomerné rozdeliteľné kódy sa dajú výhodne použiť v takých prípadoch, keď sa slová (alebo znaky), ktoré sa kódujú, vyskytujú nerovnako často. Vtedy je možné často sa vyskytujúcim slovám (znakom) priradiť kratšie kódové slová a tak dosiahnuť, že kódovaná správa bude v priemernom prípade kratšia, ako keby sa na kódovanie používali blokové napríklad kódy. V ďalšom túto intuitívnu predstavu upresníme. Kvôli jednoduchosti budeme kódovať znaky zdrojovej abecedy $\Sigma_S = \{a_0, \ldots, a_{m-1}\}$. Zavedieme prvý, značne zjednodušený matematický model zdroja S. Budeme predpokladať, že zdroj S je náhodný generátor, ktorý generuje znaky zdrojovej abecedy náhodne a nezávisle na sebe. Zdroj S je charakterizovaný rozdelením pravdepodobností $P = \{p_0, \ldots, p_{m-1}\}$; $p_i \geq 0$, $i = 0, \ldots, m-1; \sum_{i=0}^{m-1} p_i = 1$ výskytu jednotlivých symbolov zdrojovej abecedy. (Z matematického hľadiska je sdroj S náhodná premenná, nadobúdajúca hodnotu a_i s pravdepodobnosťou p_i, $i = 0, \ldots, m-1$.) Je zrejmé, že existuje viacero spôsobov kódovania znakov

zdrojovej abecedy. Aby sme mohli porovnať efektívnosť jednotlivých kódov, zavedieme pojem *ceny kódu*.

Definícia 3.4 *Nech je* $P = \{p_0, \ldots, p_{m-1}\}$ *rozdelenie pravdepodobností znakov zdrojovej abecedy* $\Sigma_S = \{a_0, \ldots, a_{m-1}\}$*; nech* $V = \{v_0, \ldots, v_{m-1}\}$ *je kód kódujúci znaky kódovej abecedy,* $a_i \to v_i,\ i = 0, \ldots, m-1$ *a nech* $l_i = l(v_i)$ *sú dĺžky kódových slov kódu* V*. Potom cenou kódu* V *pri rozdelení pravdepodobností nazveme*

$$\mathcal{L}(P, V) = \sum_{i=0}^{m-1} l_i p_i.$$

Cena kódu V pri rozdelení pravdepodobností P nie je z matematického hľadiska nič iné, než stredná hodnota dĺžky kódového slova, t.j. počet symbolov kódovej abecedy pripadajúcich na zakódovanie jedného znaku zdrojovej abecedy. (V prípade kódovania slov z nejakej množiny M by to bol počet symbolov kódovej abecedy pripadajúcich na zakódovanie jedného slova z množiny M.)

Aká je minimálna hodnota $\mathcal{L}(P, V)$ pri danom rozdelení pravdepodobností? Existujú kódy dosahujúce túto minimálnu hodnotu a ak áno, sú známe metódy ich zostrojovania? Na tieto i ďalšie otázky dáme odpoveď v nasledujúcich častiach tejto kapitoly.

3.3 Kvázioptimálne kódy a optimálny kód

Kód $V = \{v_0, \ldots, v_{m-1}\}$ s dĺžkami kódových slov $l(v_i) = l_i,\ i = 0, \ldots, m-1$ nazveme optimálnym kódom pre rozdelenie pravdepodobností $P = \{p_0, \ldots, p_{m-1}\}$, ak pre ľubovoľný kód $W = \{w_0, \ldots, w_{m-1}\}$ platí

$$\mathcal{L}(P, V) \leq \mathcal{L}(P, W).$$

Cenu optimálneho kódu pri rozdelení pravdepodobností P označíme $\mathcal{L}(P)$. Prirodzená otázka je, aká je cena optimálneho kódu.

Veta 3.5 *Nech je* $P = \{p_0, \ldots, p_{m-1}\}$ *ľubovoľné rozdelenie pravdepodobností,* $p_0 \geq p_1 \geq \cdots \geq p_{m-1} > 0$*. Potom platí*

$$\sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{1}{p_i} \leq \mathcal{L}(P) \leq \sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{1}{p_i} + 1.$$

Rovnosť

$$\sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{1}{p_i} = \mathcal{L}(P) \tag{19}$$

platí práve vtedy, ak $p_i = 2^{-l_i},\ l_i \in N,\ i = 0, \ldots, m-1$.

Dôkaz. Dolný odhad. Predpokladajme, že $V = \{v_0, \ldots, v_{m-1}\}$ je ľubovoľný prefixový kód s dĺžkami kódových slov $l(v_i) = l_i,\ i = 0, \ldots, m-1$. Porovnáme cenu kódu V s entropiou zdroja $H_2(\mathcal{P}) = \sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{1}{p_i}$:

$$\sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{1}{p_i} - \sum_{i=0}^{m-1} l_i p_i = \sum_{i=0}^{m-1} p_i \cdot \left[\log_2 \frac{1}{p_i} - \log_2 2^{l_i}\right] = \sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{2^{-l_i}}{p_i}.$$

Teraz prevedieme binárny logaritmus na prirodzený, využijeme nerovnosť $\ln x \leq x - 1$ a upravíme:

$$\sum_{i=0}^{m-1} p_i \cdot \log_2 \frac{2^{-l_i}}{p_i} = \frac{1}{\ln 2} \sum_{i=0}^{m-1} p_i \cdot \ln_2 \frac{2^{-l_i}}{p_i} \leq \frac{1}{\ln 2} \sum_{i=0}^{m-1} p_i \cdot \left[\frac{2^{-l_i}}{p_i} - 1\right] = \tag{20}$$

$$= \frac{1}{\ln 2} \left[\sum_{i=0}^{m-1} 2^{-l_i} - \sum_{i=0}^{m-1} p_i\right] = \frac{1}{\ln 2} \left[\sum_{i=0}^{m-1} 2^{-l_i} - 1\right].$$

Kód V je prefixový a teda z Kraftovej-McMillanovej nerovnosti vyplýva, že $\sum_{i=0}^{m-1} 2^{-l_i} \leq 1$. Keďže $2 > 1,\ \ln 2 > 0$ platí

$$\frac{1}{\ln 2} \left[\sum_{i=0}^{m-1} 2^{-l_i} - 1\right] \leq 0.$$

To však znamená, že pre ľubovoľný prefixový kód $V = \{v_0, \ldots, v_{m-1}\}$ platí

$$H_2(\mathcal{P}) \leq \mathcal{L}(P, V).$$

Horný odhad.Dokážeme, že existuje (prefixový) kód, ktorý dosahuje cenu $H_2(\mathcal{P}) + 1$. Položíme $l_i = \lceil \log_2 \frac{1}{p_i} \rceil$. Potom platí:

$$\sum_{i=0}^{m-1} 2^{-l_i} = \sum_{i=0}^{m-1} 2^{\lceil \log_2 \frac{1}{p_i} \rceil} \leq \sum_{i=0}^{m-1} 2^{-\log_2 \frac{1}{p_i}} = \sum_{i=0}^{m-1} p_i = 1;$$

a teda existuje prefixový kód s dĺžkami kódových slov $l_i = \lceil \log_2 \frac{1}{p_i} \rceil,\ i = 0, \ldots, m-1$. Cena tohto kódu je

$$\mathcal{L}(P, V) = \sum_{i=0}^{m-1} p_i l_i = \sum_{i=0}^{m-1} p_i \lceil \log_2 \frac{1}{p_i} \rceil \leq \sum_{i=0}^{m-1} p_i \left[\log_2 \frac{1}{p_i} + 1\right] = \sum_{i=0}^{m-1} p_i \log_2 \frac{1}{p_i} + 1.$$

Vráťme sa ešte k dôkazu rovnosti (19). Ak $P = \{p_i = 2^{-l_i},\ l_i \in N,\ i = 0, \ldots, m-1\}$ je rozdelenie pravdepodobností, potom podľa vety 3.2 existuje prefixový kód s dĺžkami kódových slov

$$\lceil \log_2 \frac{1}{p_i} \rceil = \lceil \log_2 \frac{1}{2^{-l_i}} \rceil = \lceil \log_2 2^{l_i} \rceil = \lceil l_i \rceil = l_i,$$

ktorý má cenu

$$\sum_{i=0}^{m-1} p_i l_i = \sum_{i=0}^{m-1} p_i \log_2 \frac{1}{p_i}.$$

Na druhej strane, ak

$$\sum_{i=0}^{m-1} p_i l_i = \sum_{i=0}^{m-1} p_i \log_2 \frac{1}{p_i}$$

to znamená, že v odvodení 20 nastala rovnosť. To však znamená, že $\frac{2^{-l_i}}{p_i} = 1$, resp. $p_i = 2^{-l_i}$ ($\ln x = x - 1$ pre $x = 1$). ■

Jeden kód, ktorého cena sa veľmi nelíši od ceny optimálneho kódu už poznáme. Je to Shannonov kód. Skôr, ako ukážeme, ako sa konštruuje optimálny kód, uvedieme ešte jednu jednoduchú metódu konštrukcie kódu, ktorého cena je blízka k cene optimálneho kódu, Fanov kód. (Shannonov a Fanov kód sa nazývajú *kvázioptimálne kódy.*)

3.3.1 Fanov kód

Fanova konštrukcia kvázioptimálneho kódu. Predpokladáme, že je dané rozdelenie prevdepodobností $P = \{p_0, \ldots, p_{m-1}\}$

1. usporiadame pravdepodobnosti zostupne $p_0 \geq p_1 \geq \cdots \geq p_{m-1} > 0$ a zapíšeme do 1. stĺpca tabuľky. Jednotlivým pravdepodobnostiam (zastupujúcim symboly zdrojovej abecedy) priradíme prázdne slová λ.

2. Ak tabuľka obsahuje aspoň dva riadky, rozdelíme ju na 2 časti tak, aby sa súčet pravdepodobností v hornej časti tabuľky líšil čo najmenej od súčtu pravdepodobností v dolnej časti tabuľky a pokračujeme krokom 3. Ak tabuľka obsahuje jediný riadok, jej spracovanie ukončíme.

3. Slová v_i priradené pravdepodobnostiam v hornej polovici tabuľky zreťazíme sprava so znakom 0, a slová z dolnej polovice tabuľky zreťazíme sprava so znakom 1. Pokračujeme v spracovaní hornej a dolnej časti tabuľky podľa kroku 2.

Keďže tabuľka obsahuje m riadkov, krok 2 sa uplatní najviac $m - 1$-krát. Ilustrujeme konštrukciu Fanovho a Shannonovho kódu na nasledujúcom príklade.

Príklad 3.6

p_0	0.25	0	00		
p_1	0.20	0	01		
p_2	0.13	1	10	100	
p_3	0.12	1	10	101	
p_4	0.10	1	11	110	1100
p_5	0.08	1	11	110	1101
p_6	0.07	1	11	111	1110
p_7	0.05	1	11	111	1111

Fanov kód

p_i	q_i	l_i	ν_i
0.25	0.0	2	00
0.20	0.010	3	010
0.13	0.011	3	011
0.12	0.1001	4	1001
0.10	0.1011	4	1011
0.08	0.1100	4	1100
0.07	0.1110	4	1110
0.05	0.11110	5	11110

Shannonov kód

Fanov kód má cenu 2.85 *a Sahnnonov* 3.22 *a entropia je* 2.822. *Shannonov kód ešte možno upraviť (skrátiť). Keďže slovo* 100 *je prefixom jediného kódového slova, možno toto slovo* 1001 *nahradiť slovom* 100*; podobne možno skrátiť kódové slovo* 1011 *na* 101*; kódové slovo* 1100 *na* 110 *a napokon kódové slovo* 11110 *na* 1111. *Takto upravený (skrátený) Shannonov kód má cenu* 2.87.

3.3.2 Huffmanov optimalny kód

Uvedieme teraz metódu konštrukcie optimálneho kódu. Podstata Huffmanovej metódy spočíva v tom, že sa zostrojenie optimálneho kódu pre m znakov redukuje na konštrukciu optimálneho kódu pre $m-1$ znakov. Pri dôkaze budem potrebovať nasledujúcu lemu.

Veta 3.6 *Nech je* $P = \{p_0, \ldots, p_{m-1}\}$ *ľubovoľné rozdelenie pravdepodobností,* $p_0 \geq p_1 \geq \cdots \geq p_{m-1} > 0$. *Potom existuje prefixový kód* $V = \{\nu_0, \ldots, \nu_{m-1}\}$ *s dĺžkami kódových slov* $l_i = l(\nu_i),\ i = 0, \ldots, m-1$ *optimálny pre rozdelenie pravdepodobností* P*, taký, že minimálnym pravdepodobnostiam* p_{m-2}, p_{m-1} *zodpovedajú slová* ν_{m-2}, ν_{m-1} *maximálnej dĺžky* l_{m-1}*, ktoré majú spoločný prefix dĺžky* $l_{m-1} - 1$.

Dôkaz. Ak by v kóde V neboli priradené slová maximálnej dĺžky minimálnym pravdepodobnostiam, kód by nebol optimálny. To znamená, že minimálnym pravdepodobnostiam p_{m-2}, p_{m-1} musia byť priradené slová maximálnej dĺžky. Predpokladajme, že slová ν_{m-2}, ν_{m-1} nemajú spoločný prefix dĺžky $l_{m-1} - 1$. To znamená, že existuje slovo ν_j maximálnej dĺžky l_{m-1}, ktoré má spoločný prefix dĺžky $l_{m-1} - 1$ so slovom ν_{m-1}. V opačnom prípade by slovo ν_{m-1} bolo možné nahradiť jeho prefixom dĺžky $l_{m-1} - 1$, čo je v spore s optimálnosťou kódu V. (Z podobných dôvodov musí existovať slovo ν_k maximálnej dĺžky l_{m-1}, ktoré má spoločný prefix dĺžky $l_{m-1} - 1$ so slovom ν_{m-2}.) „Zámenou“ slov ν_{m-2} a ν_j dostávame kód s rovnakou cenou, ako bola cena pôvodného kódu, spĺňajúci podmienky Lemy. ■

Teraz už môžeme vysloviť a dokázať vetu, ktorá je teoretickým zdôvodnením Huffmanovej konštrukcie optimálneho kódu.

Veta 3.7 Huffmanov kód. *Nech* $V = \{v_0, \ldots, v_{m-1}\}$, $m > 1$ *je optimálny prefixový kód pre rozdelenie pravdepodobností* $P = \{p_0, \ldots, p_{m-1}\}$, *pričom* $p_j = q_0 + q_1$ *a* $p_0 \geq p_1 \geq \cdots \geq p_{m-1} \geq q_0 \geq q_1 > 0$. *Potom kód* $V = \{v_0 \ldots, v_{j-1}, v_{j+1}, \ldots, v_{m-1}, v_j 0, v_j 1\}$ *je optimálny kód pre rozdelenie pravdepodobností* $P' = \{p_0 \ldots, p_{j-1}, p_{j+1}, \ldots, p_{m-1}, q_0, q_1\}$.

Dôkaz Kód V' je tiež prefixový a jeho cena je $\mathcal{L}(P', V') = \mathcal{L}(P, V) + p_j$. Aby sme ukázali, že V' je optimálny kód, musíme dokázať, že pre ľubovoľný kód $W' = \{w_0, \ldots, w_m\}$ pre rozdelenie pravdepodobností P' platí $\mathcal{L}(P', W') \geq \mathcal{L}(P', V') = \mathcal{L}(P, V) + p_j$. Predpokladajme, že W' je optimálny kód pre rozdelenie pravdepodobností P', ktorý naviac spĺňa podmienky vety 3.6. To znamená, že minimálnym pravdepodobnostiam q_0, q_1 zodpovedajú slová maximálnej dĺžky $w1, w0$. Uvažujme teraz kód $W = \{w_0, \ldots, w_{j-1}, w, w_{j+1}, \ldots, w_{m-1}\}$ pre rozdelenie pravdepodobností P. Keďže pre rozdelenie pravdepodobností P je optimálny kód V, platí $\mathcal{L}(P, V) \leq \mathcal{L}(P, W)$. Ale potom

$$\mathcal{L}(P', V') = \mathcal{L}(P, V) + p_j \leq \mathcal{L}(P, W) + p_j = \mathcal{L}(P', W'),$$

a teda kód V' je optimálny pre rozdelenie pravdepodobností P'. ■

Popíšeme metódu konštrukcie Huffmanovho kódu pre rozdelenie pravdepodobností $P = \{p_0, \ldots, p_{m-1}\}$.

Konštrukcia optimálneho kódu.

1. usporiadaj pravdepodobnosti $p_0, \ldots, p_{m-1}$ do zoznamu zostupne. Ak $m > 1$ pokračuj krokom 2, ináč choď na krok 3.
2. Opakuj $m - 2$ krát nasledujúcu činnosť:
   - sčítaj posledné dve (minimálne) pravdepodobnosti usporiadaného zoznamu;
   - odstráň tieto dve pravdepodobnosti zo zoznamu a zaraď do zoznamu ich súčet tak, aby bol nový zoznam usporiadaný zostupne;
   - zapamätaj si miesto v zozname, na ktoré bola zaradená nová hodnota.
3. Zoznam obsahuje dve pravdepodobnosti; priraď (napríklad) väčšej z nich slovo 0 a menšej slovo 1; ak $m = 1$ skonči, ináč pokračuj krokom 4.
4. Opakuj $m - 2$ krát nasledujúcu činnosť a potom skonči:
   - urči tú pravdepodobnosť p_j v aktuálnom usporiadanom zozname, ktorá bola vytvorená ako posledná súčtom nejakých dvoch minimálnych pravdepodobností q_0, q_1;
   - odstráň pravdepodobnosť p_j zo zoznamu, doplň doň pravdepodobnosti q_0, q_1 a usporiadaj ho;
   - ak bolo pravdepodobnosti p_j priradené slovo v, priraď pravdepodobnostiam q_0, q_1 slová $v0, v1$.

Ilustrujeme Huffmanovu konštrukciu na príklade.

Príklad 3.7

0.25	0.25	0.25	0.25	0.31∗	0.44∗	0.56∗	0∗	1∗	00∗	01	01	01	01
0.20	0.20	0.20	0.24∗	0.25	0.31	0.44	1	00	01	10∗	11	11	11
0.13	0.13	0.18∗	0.20	0.24	0.25			01	10	11	000∗	001	001
0.12	0.12	0.13	0.18	0.20					11	000	001	100	100
0.10	0.12∗	0.12	0.13							001	100	101∗	0000
0.08	0.10	0.12									101	0000	0001
0.07	0.08											0001	1010
0.05													1011

Huffmanov kód

Pravdepodobnosti, ktoré vznikli sčítaním minimálnych pravdepodobností v predchádzajúcom kroku, sú uznačené hviezdičkou. Kvôli jednoduchosti sú hviezdičkou označené aj slová prislúchajúce týmto pravdepodobnostiam.

Huffmanov kód má cenu 2.85. Je zaujímavé, že aj keď sú Fanov a Huffmanov kód rôzne, majú rovnakú cenu. Vo všeobecnosti však Huffmanova metóda umožňuje získať lepšie kódy ako Fanova metóda vďaka tomu, že „preusporiadavaním“ pravdepodobností lepšie „vyvažuje“ tabuľku pravdepodobností. Čo však v tom prípade, keď je tabuľka pravdepodobností nevyvážená už na samom začiatku; ak sa jeden symbol vyskutuje veľmi často a ostatné zriedkavo? Uvedieme extrémny prípad a ukážeme, ako sa dá riešiť.

3.3.3 Rozšírenie kódu

Príklad 3.8 *Zdrojová abeceda pozostáva zo symbolov* $\{a, b\}$ *a rozdelenie pravdepodobností je* $P = \{0.9, 0.1\}$. *Optimálny kód* $V = \{0, 1\}$ *má cenu* $\mathcal{L} = 1.0$ *a entropia zdroja je* 0.4689955936. *Rozdiel medzi entropiou a cenou optimálneho kódu je príliš veľký. Podstata problému je v tom, že zdrojová abeceda je príliš malá a nemáme možnosť rozlíšiť často* (a) *a zriedkavo* (b) *sa vyskytujúce symboly, ale obom sme priradili slová rovnakej dĺžky. Pri kódovaní zdrojovej abecedy s takým extrémnym rozdelením pravdepodobností uplatníme nasledujúci postup. Namiesto jednotlivých znakov kódovej abecedy budeme kódovať* n*-tice znakov zdrojovej abecedy. Využijeme pritom predpoklady o charaktere zdroja: znaky generuje nezávisle na sebe a s nemennými pravdepodobnosťami. „Abeceda“* Σ_s^2*, jej rozdelenie pravdepodobností a príslušný Huffmanov kód* V_2 *sú uvedené v nasledujúcej tabuľke.*

aa	0.81	0
ab	0.09	11
ba	0.09	100
bb	0.01	101

Cena kódu V_2 *je* 1.29. *Treba si však uvedomiť, že to je počet binárnych symbolov pripadajúcich na jedno zdrojové slovo, ktoré má dĺžku 2, a preto cena kódu, meraná počtom kódových symbolov potrebných na zakódovanie jedného symbolu zdrojovej abecedy je* $\mathcal{L}(\mathcal{P}', \mathcal{V}_\in) = 1.29/2 = 0.645$. *Táto hodnota je už podstatne bližšia k entropii zdroja, ako*

cena pôvodného kódu. Ďalšie rozšírenie zdrojovej abecedy (kódovanie trojznakových slov nad zdrojovou abecedou) už neprinesie takú podstatnú redukciu ceny kódu:

aaa	0.729	0
aab	0.081	100
aba	0.081	101
baa	0.081	110
abb	0.009	11100
bab	0.009	11101
bba	0.009	11110
bbb	0.001	11111

$\mathcal{L}(\mathcal{P}'', \mathcal{V}_\ni) = 1.599/3 = 0.533$. *Aby sme si spravili predstavu o to, ako rýchlo sa približuje cena kódu pre narastajúcu hodnotu* n *k entropii, vypočítame cenu neskráteného Shannonovho kódu pre rozličné hodnoty* n*:*

n	3	4	5	10	20	30	50	100	200	1000	2000
$\mathcal{L}$	0.6333	0.5509	0.5163	0.5070	0.5006	0.4863	0.4789	0.4741	0.4713	0.4695	0.4692

Upozorňujeme, že n*-násobným rozšírením (dvojprvkovej) zdrojovej abecedy dostaneme množinu slov mohutnosti* 2^n*, a tak je použitie tejto metódy pre konštrukciu kódov s cenou blízkou k dolnej hranici danej entropiou pre väčšie hodnoty* n *a/alebo rozsiahlejšie abecedy zdroja prakticky nepoužiteľné.*

Metóda konštrukcie Huffmanovho kódu nie je jednoznačná. Ak v zozname pravdepodobností už existuje hodnota rovná tej, ktorú sme dostali v niektorom kroku súčtom minimálnych pravdepodobností, máme možnosť zaradiť vypožčítanú pravdepodobnosť za alebo pred pravdepodobnosť v pôvodnom usporiadanom zozname. Uplatnením rozličných stratégií zaraďovania rovnakých pravdepodobností do zoznamu, dostaneme kódy, ktoré majú rovnakú cenu, ale môžu mať rozličné dĺžky kódových slov.

Príklad 3.9

0.375	0.375	0.375	0.375	0.625∗	0	1	1	1	1
0.250	0.250	0.250	0.375∗	0.375	1	00	01	01	01
0.125	0.125	0.250∗	0.25			01	000	001	001
0.125	0.125	0.125					001	0000	0000
0.625	0.125∗							0001	00010
0.625									00011

Huffmanov kód V

0.375	0.375	0.375	0.375∗	0.625∗	0	1	00	00	00
0.250	0.250	0.250∗	0.375	0.375	1	00	01	10	10
0.125	0.125∗	0.250	0.25			01	10	11	010
0.125	0.125	0.125					11	010	011
0.625	0.125							011	110
0.625									111

Huffmanov kód V'.

ľahko sa presvedčíme o tom, že $\mathcal{L}(V, P) = \mathcal{L}(V', P) = 2.375$.

Ktorý zo zostrojených kódov je lepší? Ak sa v nejakom texte vyskytujú zdrojové symboly s pravdepodobnosťami zodpovedajúcimi pravdepodobnostiam z rozdelenia pravdepodobností P, oba kódy zakódujú daný text rovnako efektívne. Ak však kód zostrojujeme na základe predpokladaného rozdelenia pravdepodobností, ktoré sa od skutočného líši, môže sa cena skonštruovaného kódu viac alebo menej odlišovať od ceny optimálneho kódu a rozdiel medzi skutočnou a minimálnou cenou bude závisieť aj od spôsobu konštrukcie kódu.

3.3.4 Chyby v pravdepodobnostiach zdrojových symbolov

Predpokladajme, že je dané rozdelenie pravdepodobností $P = \{p_0, \ldots, p_{m-1}\}$, na základe ktorého sme zostrojili Huffmanov kód $V = \{v_0, \ldots, v_{m-1}\}$ s dĺžkami kódových slov $\{l_0, \ldots, l_{m-1}\}$. Nech je $P' = \{p'_0, \ldots, p'_{m-1}\}$, skutočné rozdelenie pravdepodobností zdrojových symbolov; $p'_i = p_i + e_i$, $i = 0, \ldots, m-1$, kde e_i je chyba v odhade pravdepodobnosti výskytu i-teho symbolu. Keďže P', P sú rozdelenia pravdepodobností, platí: $\sum_{i=0}^{m-1} p'_i = \sum_{i=0}^{m-1} p_i + e_i = 1 + \sum_{i=0}^{m-1} e_i$; a teda $\sum_{i=0}^{m-1} e_i = 0$. Zistíme, aký bude rozdiel cien kódu V pri rozdelení pravdepodobností P' a P:

$$\mathcal{L}(V, P') = \sum_{i=0}^{m-1} p'_i l_i = \sum_{i=0}^{m-1} (p_i + e_i) l_i = \sum_{i=0}^{m-1} p_i l_i + \sum_{i=0}^{m-1} l_i e_i = \mathcal{L}(V, P) + \sum_{i=0}^{m-1} l_i e_i$$

Zistíme, kedy nadobúda $\sum_{i=0}^{m-1} l_i e_i$ extrémne hodnoty. Použijeme na to metódu Lagrangeových neurčitých koeficientov [6]. Vyjadríme najprv podmienky, za ktorých budeme hľadať extrémy funkcie $\sum_{i=0}^{m-1} l_i e_i$. Odchýlky e_i od pravdepodobností výskytu symbolov budeme chápať ako výsledky náhodnej premennej e; pričom $P(e = e_i) = \frac{1}{m}$, $m = 0, \ldots, m-1$. Potom stredná hodnota chyby je

$$E(e) = \sigma^2 = \sum_{i=0}^{m-1} e_i \frac{1}{m} = \frac{1}{m} \sum_{i=0}^{m-1} e_i = 0. \tag{21}$$

Vyjadríme disperziu chýb:

$$Var(e) = \sum_{i=0}^{m-1} e_i^2 \frac{1}{m} = \frac{1}{m} \sum_{i=0}^{m-1} e_i^2. \tag{22}$$

Využijeme 21 a 22 a zostrojíme Lagrangeovu funkciu pre veličinu $\sum_{i=0}^{m-1} l_i e_i$ (λ, μ sú Lagrangeove neurčité koeficienty):

$$\mathcal{F} = \frac{1}{m} \sum_{i=0}^{m-1} l_i e_i + \lambda \left(\frac{1}{m} \sum_{i=0}^{m-1} e_i \right) + \mu \left(\frac{1}{m} \sum_{i=0}^{m-1} e_i^2 - \sigma^2 \right). \tag{23}$$

Vypočítame parciálne derivácie funkcie $\mathcal{F}$ podľa $e_i,\ i = 0, \ldots, m-1$ a položíme ich rovnými nule:

$$\frac{\partial \mathcal{F}}{\partial e_i} = \frac{1}{m}\left(l_i - \lambda - 2\mu e_i\right) = 0; \quad i = 0, \ldots, m-1. \tag{24}$$

Sčítame rovnice 24 a vyjadríme koeficient λ.

$$\sum_{i=0}^{m-1} \frac{1}{m}\left(l_i - \lambda - 2\mu e_i\right) = \frac{1}{m}\sum_{i=0}^{m-1} l_i - \lambda - \frac{2\mu}{m}\sum_{i=0}^{m-1} e_i = 0;$$

a teda

$$\lambda = \frac{1}{m}\sum_{i=0}^{m-1} l_i. \tag{25}$$

Vrátime sa k sústave rovníc 24. Jednotlivé rovnice vynásobíme zodpovedajúcimi hodnotami e_i a výsledok spočítame cez všetky hodnoty i. Dostávame

$$\frac{1}{m}\sum_{i=0}^{m-1}\left(l_i e_i - \lambda e_i - 2\mu e_i^2\right) = \frac{1}{m}\sum_{i=0}^{m-1} l_i e_i - \frac{\lambda}{m}\sum_{i=0}^{m-1} e_i - \frac{2\mu}{m}\sum_{i=0}^{m-1} e_i^2 = 0.$$

Upravíme

$$\frac{1}{m}\sum_{i=0}^{m-1} l_i e_i = \frac{2\mu}{m}\sum_{i=0}^{m-1} e_i^2,$$

a určíme koeficient μ:

$$\mu = \frac{1}{2m\sigma^2}\sum_{i=0}^{m-1} l_i e_i. \tag{26}$$

Napokon vynásobíme jednotlivé rovnice sústavy 24 príslušnými hodnotami l_i a výsledky násobenia sčítame cez všetky i:

$$\frac{1}{m}\sum_{i=0}^{m-1}\left(l_i^2 - \lambda l_i - 2\mu l_i e_i\right) = \frac{1}{m}\sum_{i=0}^{m-1} l_i^2 - \frac{\lambda}{m}\sum_{i=0}^{m-1} l_i - \frac{2\mu}{m}\sum_{i=0}^{m-1} l_i e_i = 0. \tag{27}$$

Dosadíme hodnoty konštánt μ, λ do 27 a upravíme

$$\begin{aligned}
\frac{1}{m}\sum_{i=0}^{m-1} l_i^2 - \left(\frac{1}{m}\sum_{i=0}^{m-1} l_i\right)^2 - \frac{1}{\sigma^2 m^2}\left(\sum_{i=0}^{m-1} l_i e_i\right)^2 &= 0; \\
\frac{1}{m}\sum_{i=0}^{m-1} l_i^2 - \left(\frac{1}{m}\sum_{i=0}^{m-1} l_i\right)^2 &= \frac{1}{\sigma^2}\left(\frac{1}{m}\sum_{i=0}^{m-1} l_i e_i\right)^2.
\end{aligned}$$

$$\left[\frac{1}{m}\sum_{i=0}^{m-1} l_i^2 - \left(\frac{1}{m}\sum_{i=0}^{m-1} l_i\right)^2\right]\cdot\sigma^2 = Var(l)Var(e) = \left(\frac{1}{m}\sum_{i=0}^{m-1} l_i e_i\right)^2.$$

To znamená, že pre fixovanú hodnotu disperzie chýb, σ^2, sa extrémne odchýlky ceny kódu (v kladnom alebo zápornom smere) dosahujú pre kódy, ktoré majú veľkú disperziu dĺžok kódových slov. Ináč povedané, čím väčšie sú rozdiely v dĺžkach kódových slov,

tým väčšiu ochýlku (zlepšenie alebo zhoršenie) ceny kódu môžu spôsobiť chyby v pravdepodobnostiach jednotlivých zdrojových symbolov. Príkladom kódu so stabilnou cenou je blokový kód, pre ktorý sa žiadne odchýlky v pravdepodobnostiach neprejavia zmenou ceny kódu. (Otázne však je, či pre skutočné rozdelenie pravdepodobností bude pôvodný kód optimálny. Tento problém naše odvodenie nerieši.)

Príklad 3.10 *Ilustrujeme predchádzajúce úvahy na Huffmanových kódoch z príkladu 3.9. Pripomíname, že sme zostrojili dva optimálne Huffmanove kódy pre rozdelenie pravdepodobností* P*, pričom kód* V *mal kódové slová dĺžok* $\{1,2,3,4,5,5\}$ *a kód* V' *mal kódové slová dĺžok* $\{2,2,3,3,3,3\}$*. V prvom prípade bola disperzia dĺžok kódových slov* $\mathrm{Var}(l) = \frac{20}{9}$*, v druhom* $\mathrm{Var}(l') = \frac{2}{9}$*. V nasledujúcej tabuľke uvádzame príklad chýb v pravdepodobnostiach zdrojových symbolov, ktoré viedli k rozličným odchýlkam cien kódov.*

p_i	l_i	l'_i	e_i	$\triangle_i$	$\triangle'_i$
0.375	1	2	-0.1250	-0.1250	-0.250
0.250	2	2	0	0	0
0.125	3	3	0	0	0
0.125	4	3	0	0	0
0.0625	5	3	$+0.0625$	$+0.3125$	$+0.1875$
0.0625	5	3	$+0.0625$	$+0.3125$	$+0.1875$
			0	$+0.500$	$+0.125$

Vplyv chýb v pravdepodobnostiach symbolov na cenu Huffmanovho kódu.

Huffmanov kód je pomerne odolný voči malým odchýlkam v pravdepodobnostiach zdrojových symbolov. Ak chceme minimalizovať vplyv týchto odchýlok na cenu kódu, pri konštrukcii Huffmanovho kódu budeme zaraďovať vypočítanú pravdepodobnosť do zoznamu tak vysoko, ako sa len bude dať.

3.4 Kódovanie Markovovského zdroja

Matematický model, ktorý sma používali až doteraz na popis zdroja informácie, bol značne zjednodušený. V textoch zapísaných v prirodzenom jazyku sú medzi jednotlivými znakmi závislosti, ktoré sme doteraz zanedbávali. Napríklad v slovenčine sa po mäkkých spoluhláskach takmer nikdy nepíše ypsilon, po tvrdých spoluhláskach zasa mäkké i, v textoch sa nevyskytujú viac ako tri za sebou idúce samohlásky ani dlhé postupnosti zložené zo samotných spoluhlások a pod. Popísať však dostatočne presne takéto zákonitosti prirodzeného jazyka by bolo dosť náročné. Pre naše potreby vystačíme s omnoho jednoduchším matematickým modelom a zdroj budeme popisovať pomocou Markovovských reťazcov. Budeme predpokladať, že zdroj S v diskrétnych časových okamihoch (taktoch, krokoch) generuje symboly zo zdrojovej abecedy $\Sigma_S = \{s_0, \ldots, s_{q-1}\}$; činnosť zdroja budeme popisovať pomocou postupnosti náhodných premenných S_t, $t = 0, \ldots,$[5]

[5] rozdelenie pravdepodobností symbolov $s_0, \ldots, s_{q-1}$ v t-tom kroku budeme označovať nasledovne: $\{p_0^{(t)}, \ldots, p_{q-1}^{(t)}\}$.

ktorá spĺňa nasledujúcu podmienku: pre ľubovoľné prirodzené číslo n a ľubovoľné čísla $i_0, \ldots, i_{n+1} \in \{0, \ldots, q-1\}$ platí

$$P(S_{n+1} = s_{i_{n+1}} | S_0 = s_{i_0}, \ldots, S_n = s_{i_n}) = P(S_{n+1} = s_{i_{n+1}} | S_n = s_{i_n}). \quad (28)$$

Podmienka 28 vyjadruje skutočnosť, že pravdepodobnosť výskytu symbolu v $(n+1)$-vom kroku závisí len od toho, aký symbol bol na výstupe zdroja v predchádzajúcom kroku n. Postupnosť náhodných premenných ktorá spĺňa podmienku 28 sa nazýva *Markovovský reťazec.* Analogicky, zdroj S ktorý spĺňa podmienku 28, budeme nazývať *Markovovským zdrojom.* Podmienené pravdepodobnosti $P(S_{n+1} = s_{i_{n+1}} | S_n = s_{i_n})$ sa nazývajú pravdepodobnosťami prechodu. Pravdepodobnosti prechodu vo všeobecnosti závisia od parametra n (znaky sa vyskytujú s inými pravdepodobnosťami napríklad v hlavičkých ako v textoch programov). Budeme však predpokladať, že pravdepodobnosti prechodu nezávisia od časového parametra n. Takýto Markovovský zdroj sa nazýva *homogénny.* Zdroj S popíšeme pomocou matice pravdepodobností prechodu. Kvôli zjednodušeniu zápisu budeme pravdepodobnosť $P(S_{n+1} = s_j | S_n = s_k)$ označovať symbolom $p_{j,k}$. Matica pravdepodobností prechodu zdroja S bude mať nasledujúci tvar:

$$M = \begin{pmatrix} p_{0,0} & p_{1,0} & \cdots & p_{q-1,0} \\ p_{0,1} & p_{1,1} & \cdots & p_{q-1,1} \\ & & & \\ p_{0,q-1} & p_{1,q-1} & \cdots & p_{q-1,q-1} \end{pmatrix}$$

Matica M má všetky prvky nezáporné a súčet prvkov v ľubovoľnom riadku je rovný 1. (Takáto matica sa nazýva stochastická.) Ak poznáme symbol, ktorý sa objavil na výstupe zdroja, pomocou matice pravdepodobností prechodu M vieme určiť pravdepodobnosti výskytu symbolov na výstupe zdroja v nasledujúcom i v ďalších časových okamihoch. Nech sa v 0-tom kroku objavil na výstupe zdroja symbol s_0. Potom sa v nasledujúcom kroku budú na výstupe zdroja objavovať symboly zo zdrojovej abecedy s pravdepodobnosťami

$$(1, 0, \ldots, 0) \times M = (p_{0,0}, p_{1,0}, \ldots, p_{q-1,0}).$$

Rozdelenie pravdepodobností symbolov na výstupe zdroja v ďalšom kroku by sme vypočítali ako súčin rozdelenia pravdepodobností v 1. kroku a matice M:

$$(p_{0,0}, p_{1,0}, \ldots, p_{q-1,0}) \times M = (1, 0, \ldots, 0) \times M^2.$$

(Namiesto toho, aby sme v každom kroku prácne počítali súčin vektora a matice M, využijeme poznatok, že matica pravdepodobností prechodu po m krokoch sa rovná m-tej mocnine matice pravdepodobností prechodu po jednom kroku; M. [7].) Ilustrujeme uvedené pojmy na príklade.

Príklad 3.11 *Uvažujeme Markovovský zdroj S so štvorprvkovou abecedou $\Sigma_S = \{a, b, c, d\}$.* ▮ *Vzťahy medzi symbolami sú popísané pomocou nasledujúcej matice pravdepodobností prechodov:*

$$M = \begin{pmatrix} 0.1 & 0.4 & 0.2 & 0.3 \\ 0.5 & 0.1 & 0.2 & 0.2 \\ 0.5 & 0.2 & 0.2 & 0.1 \\ 0.6 & 0.1 & 0.2 & 0.1 \end{pmatrix}$$

Nech $p = (1,0,0,0)$ *je rozdelenie pravdepodobností symbolov v kroku* 0*. Potom rozdelenie pravdepodobností symbolov v kroku* 1 *bude* $(0.1, 0.4, 0.2, 0.3)$*. Vypočítame niekoľko mocnín matice* M :

$$M^2 = \begin{pmatrix} 0.49 & 0.15 & 0.20 & 0.16 \\ 0.32 & 0.27 & 0.20 & 0.21 \\ 0.31 & 0.27 & 0.20 & 0.22 \\ 0.27 & 0.30 & 0.20 & 0.23 \end{pmatrix}$$

$$M^4 = \begin{pmatrix} 0.3933 & 0.2160 & 0.2000 & 0.1907 \\ 0.3619 & 0.2379 & 0.2000 & 0.2002 \\ 0.3597 & 0.2394 & 0.2000 & 0.2009 \\ 0.3524 & 0.2445 & 0.2000 & 0.2031 \end{pmatrix}$$

$$M^8 = \begin{pmatrix} 0.37199797 & 0.23084535 & 0.20000000 & 0.19715668 \\ 0.37092176 & 0.23159571 & 0.20000000 & 0.19748253 \\ 0.37084603 & 0.23164851 & 0.20000000 & 0.19750546 \\ 0.37059591 & 0.23182290 & 0.20000000 & 0.19758119 \end{pmatrix}$$

$$M^{16} = \begin{pmatrix} 0.3712427184 & 0.2313719296 & 0.2000000000 & 0.1973853520 \\ 0.3712414540 & 0.2313728112 & 0.2000000000 & 0.1973857348 \\ 0.3712413650 & 0.2313728732 & 0.2000000000 & 0.1973857617 \\ 0.3712410712 & 0.2313730782 & 0.2000000000 & 0.1973858506 \end{pmatrix}$$

V postupnosti matíc je vidieť istú zákonitosť—ako keby matice konvergovali k nejakej limitnej matici. Pozrieme sa na túto skutočnosť z iného hľadiska. Ako ovplyvní výskyt konkrétneho symbolu v 0-tom kroku rozdelenie pravdepodobností výskytu symbolu v n-tom kroku? V nasledujúcej tabuľke je uvedené rozdelenie pravdepodobností (náhodnej premennej) S_{16} za predpokladu, že $S_0 = a$ (b, c, d).

	p_a	p_b	p_c	p_d
$S_0 = a$	0.3712427184	0.2313719296	0.2000000000	0.1973853520
$S_0 = b$	0.3712414540	0.2313728112	0.2000000000	0.1973857348
$S_0 = c$	0.3712413650	0.2313728732	0.2000000000	0.1973857617
$S_0 = d$	0.3712410712	0.2313730782	0.2000000000	0.1973858506

Vidíme, že že rozdelenie pravdepodobností symbolov v 16-tom kroku nezávisí od toho, aký symbol bol na výstupe zdroja v nultom kroku. Markovovský zdroj popísaný v príklade 3.11 je zvláštnym prípadom tzv. *ergodického Markovovského zdroja.* Definujeme ergodický Markovovský zdroj formálne.

Definícia 3.5 *Nech rozdelenie pravdepodobností náhodnej premennej* S_n *konverguje k limitnému rozdeleniu pravdepodobností; t.j.*

$$\lim_{n\to\infty} p_k^{(n)} = p_k; \quad k = 0, \ldots, q-1$$

a limitné rozdelenie pravdepodobností $\{p_0, \ldots, p_{q-1}\}$ *nezávisí od počiatočného rozdelenia pravdepodobností symbolov, potom sa Markovovský zdroj nazýva ergodickým Markovovským zdrojom.*

Z hľadiska kódovania nás zaujíma predovšetkým spomínané limitné rozdelenie pravdepodobností. Ak je zdroj S ergodický, tak takéto limitné rozdelenie pravdepodobností existuje a musí spĺňať nasledujúci vzťah:

$$(p_0, \ldots, p_{q-1}) \times M = (p_0, \ldots, p_{q-1}). \tag{29}$$

Podmienku, ktorú musí spĺňať zdroj na to, aby bol ergodický, stanovuje nasledujúca veta.

Veta 3.8 *(Markovova, [7]) Nech je S Markovovský zdroj s abecedou $\Sigma_S = \{s_0, \ldots, s_{q-1}\}$ a $p_{j,k}^{(m)}$ je pravdepodobnosť prechodu $s_j \to s_k$ po m krokoch. Ak existujú také prirodzené čísla $s > 0,\ k_0 \geq 0$, že $\forall j,\ j = 0, \ldots, q-1$ platí $p_{j,k_0}^{(s)}$; t.j. v matici M^s existuje aspoň jeden stĺpec, ktorý má všetky prvky kladné, tak potom je Markovovský zdroj S ergodický, t.j. existujú limitné pravdepodobnosti*

$$\lim_{n\to\infty} p_{j,k}^{(n)} = p_k, \quad k = 0, \ldots, q-1,$$

nezávislé na indexe j. Postupnosť $p_0, \ldots, p_{q-1}$ je jediné nezáporné riešenie sústavy rovníc

$$p_k = \sum_{j=0}^{q-1} p_j p_{j,k}, \quad k = 0, \ldots, q-1,$$

ktoré vyhovuje podmienke

$$\sum_{j=0}^{q-1} p_j = 1.$$

T.j. limitné rozdelenie $p_0, \ldots, p_{q-1}$ je stacionárnym rozdelením pravdepodobností Markovovského zdroja.

Poznámka. Stacionárne rozdelenie pravdepodobností je počiatočné rozdelenie pravdepodobností, pri ktorom majú všetky (náhodné premenné) $S_n,\ n = 0, \ldots$ rovnaké rozdelenie pravdepodobností.

Ak teda v matici M alebo jej niektorej nenulovej mocnine existuje stĺpec, v ktorom sú všetky prvky nenulové, potom je zdroj S ergodický a riešením sústavy (29) nájdeme stacionárne limitné rozdelenie pravdepodobností. Pripomíname, že matica M nie je regulárna, a preto sústavu (29) treba riešiť za predpokladu

$$p_0 + \cdots + p_{q-1} = 1.$$

Príklad 3.12 *Nájdeme stacionárne limitné rozdelenie pravdepodobností ergodického Markovovského zdroja S z príkladu 3.11. (Ergodickosť Markovovského zdroja S vyplýva z toho, že už v samotnej matici M sú všetky prvky kladné.) Riešime sústavu rovníc* ∎

$$\begin{aligned} p_a &= 0.1 * p_a + 0.5 * p_b + 0.5 * p_c + 0.6 * p_d \\ p_b &= 0.4 * p_a + 0.1 * p_b + 0.2 * p_c + 0.1 * p_d \\ p_c &= 0.2 * p_a + 0.2 * p_b + 0.2 * p_c + 0.2 * p_d \\ p_d &= 0.3 * p_a + 0.2 * p_b + 0.1 * p_c + 0.1 * p_d \\ 1 &= p_a + p_b + p_c + p_d \end{aligned}$$

Riešením tejto sústavy je vektor

$$P = (p_a = 0.3712418301, p_b = 0.2313725490, p_c = 0.2000000000, p_d = 0.1973856209).$$

Poznanie limitného rozdelenia pravdepodobností možno využiť na zostrojenie Huffmanovho kódu Markovovského zdroja S. V našom prípade by Huffmanov kód bol blokový kód dĺžky 2 s cenou $\mathcal{L}(V, P) = 2$. Huffmanov kód Markovovského zdroja S však nevyužíval vzťahy medzi jednotlivými symbolmi. Navrhneme efektívnejšie kódovanie Markovovského zdroja S . Najprv zostrojíme Huffmanove kódy V_a, V_b, V_c, V_d pre rozdelenia pravdepodobností $P(\,|a)$, $P(\,|b)$, $P(\,|c)$, $P(\,|d)$. Jednotlivé kódy a ich ceny sú uvedené v tabuľke.

	a	b	c	d	$\mathcal{L}(V_x, P)$
V_a	001	1	000	01	1.9
V_b	1	001	01	000	1.8
V_c	1	01	000	001	1.8
V_d	0	100	11	101	1.6

Postupnosť symbolov $s_{i_0} s_{i_1} \dots s_{i_m}$ vytvorenú Markovovským zdrojom S budeme kódovať nasledovne:

1. prvý symbol, s_{i_0} zakódujeme pomocou pevne stanoveného kódu, napríklad V_{s_0};
2. i-ty symbol postupnosti zakódujeme pomocou kódu $V_{s_{i-1}}$; $i = 1, \dots m$.

Pri dekódovaní najprv dekódujeme prvý symbol,s_{i_0}, zakódovaný pomocou kódu V_{s_0}; na jeho základe určíme kód $V_{s_{i_0}}$, ktorým je kódovaný druhý symbol, s_{i_1}, atď. Kód Markovovského zdroja budeme kvôli jednoduchosti nazývať Markovovským kódom.

Príklad 3.13 *Nech postupnosť* ababcadadca *vytvoril Markovovský zdroj z príkladu 3.11.* ▮ *Jeho kódovanie je popísané v nasledujúcej tabuľke.*

znak	a	b	a	b	c	a	d	a	d	c	a
použitý kód	V_a	V_a	V_b	V_a	V_b	V_c	V_a	V_d	V_a	V_d	V_c
kódové slovo	001	1	1	1	01	1	01	0	01	11	1

Na zakódovanie postupnosti dĺžky 11 *sme potrebovali* 17*-bitový reťazec.*

Cena kódu Markovovského zdroja závisí od cien čiastkových kódov V_{s_i} a limitného rozdelenia pravdepodobností a dá sa vypočítať na základe nasledujúceho vzťahu:

$$\mathcal{L}(M, V) = \sum_{i=0}^{q-1} p_i \mathcal{L}(P(\ |s_i), V_{s_i}).$$

Porovnáme na záver cenu Hufmannovho kódu (pre limitné rozdelenie pravdepodobností), entropiu limitného rozdelenia pravdepodobností a cenu kódu Markovovského zdroja z predchádzajúceho príkladu.

entropia limitného rozdelenia	1.945755388
cena Huffmanovho kódu	2.000000000
cena Markovovského kódu	1.797647058

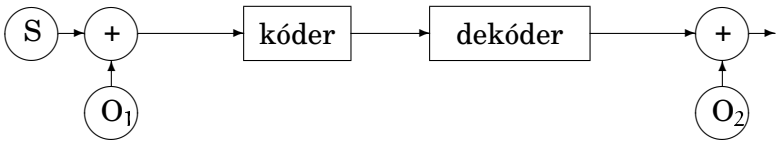
Využitím závislostí medzi jednotlivými symbolmi sme dostali cenu kódu, ktorá je výrazne nižšia ako entropia limitného rozdelenia pravdepodoboností.[6]

3.5 Kódovanie pomocou orákula

Huffmanov kód využíval to, že sme poznali rozdelenie pravdepodobností symbolov v zdrojovom texte; Markovovský kód zasa vychádzal z poznania štatistických zákonitostí medzi symbolmi, ktoré nasledovali bezprostredne za sebou. Bolo by možné zostrojiť aj iné kódy, ktoré by využívali iné zákonitosti v zdrojových textoch. Uvedieme jednu všeobecnú metódu kódovania, tzv. *kódovanie s predpoveďou*, ktorá zahŕňa viacero špeciálnych prípadov. Podstata tejto metódy spočíva v nasledujúcom (Obr. 4): Zdroj S generuje binárnu postupnosť $\alpha = a_0 a_1 \ldots$, orákulum O_1 sa pokúša „uhádnuť“ výstup zdroja S a generuje binárnu postupnosť $\beta = b_0 b_1 \ldots$. Obe postupnosti sa následne sčítavajú bit po bite modulo 2 a výsledná postupnosť $\{a_i \oplus b_i\}_{i \geq 0}$ vstupuje do kódera. Ak sa orákulu podarí „uhádnuť“ správne hodnotu symbolu a_i generovaného zdrojom, $a_i \oplus b_i = 0$. Postupnosť $\alpha \oplus \beta$ vstupujúca do kódera pozostáva zo súvislých postupností pozostávajúcich zo samých núl, ktoré sú oddelené jednotkami. Kóder zakóduje pozície jednotiek a pošle ich po prenosovom kanáli príjemcovi. Dekóder príjemcu transformuje kódovanú správu opäť do tvaru postupností núl oddelených jednotkami;$\{a_i \oplus b_i\}_{i \geq 0}$. Táto postupnosť vstupuje do člena realizujúceho sčítanie modulo 2, v ktorom sa sčítava s výstupom orákula O_2 generujúceho tú istú binárnu postupnosť $\beta = b_0 b_1 \ldots$ ako orákulum O_1. Výsledkom je postupnosť $\{a_i \oplus b_i) \oplus b_i\}_{i \geq 0} = \{a_i\}_{i \geq 0}$; t.j postupnosť generovaná zdrojom S. Doplníme niektoré predpoklady a ukážeme, aké výsledky sa dajú dosiahnuť omocou kódovania s predpoveďou. Predpokladáme, že orákulump O_1 „uhádne“ správny výsledok (jeden bit generovaný zdrojom S) s pravdepodobnosťou p a generuje opačnú hodnotu s pravdepodobnosťou $q = 1 - p$. Ďalej predpokladáme, že výsledok hádania symbolu v i-tom kroku neovplyvní výsledok hádania v ďalších krokoch.

Pozrime sa teraz na kódovanie postupnosti $\{a_i \oplus b_i\}_{i \geq 0}$. V závislosti od kvality orákula (vyjadrenej pravdepodobnosťou p) budú sa v binárnej postupnosti vyskytovať kratšie

[6] ktorá závislosti medzi jednotlivými symbolmi nezohľadňuje.



Obrázok 4: Kódovanie s predpoveďou

alebo dlhšie postupnosti núl ukončené jednotkami:

$$\alpha \oplus \beta = 0000010011000000000000100001000000100100101001 1 1...$$

Takúto postupnosť možno jednoznačne určiť postupnosťou prirodzených čísel, $n_0, n_1, \ldots$ vyjadrujúcich dĺžky nulových podpostupností. Pre vyššie uvedenú binárnu postupnosť $\alpha \oplus \beta$ bude postupnosť prirodzených čísel vyzerať nasledovne:

$$5, 2, 0, 12, 4, 6, 2, 2, 1, 3, 0, \ldots.$$

Existuje viacero možností kódovania postupnosti $n_0, n_1, \ldots$. Kvôli jednoduchosti použijeme na začiatok blokový kód dĺžky k. Keďže binárne slovo dĺžky k dokáže rozlíšiť 2^k hodnôt, postupnosti 0^n1 kde $n \geq 2^k$ sa už pomocou jedného kódového slova nedajú zakódovať. Označíme symbolom C kódovú transformáciu realizovanú kóderom, potom C možno definovať nasledovne:

$$C(0^n1) = \begin{cases} n & \text{ak } n < 2^k - 1, \\ 2^k - 1 C(0^{n-2^k+1}) & \text{ak } n \geq 2^k - 1. \end{cases}$$

Binárna postupnosť

$$\underbrace{0\ldots0}_{2^k-1}\underbrace{0\ldots0}_{2^k-1}\underbrace{0\ldots0}_{2^k-2}1$$

bude kódovaná binárne zapísanou trojicou čísel $2^k - 1, 2^k - 1, 2^k - 2$ dĺžky $3k$. Už z tohto jednoduchého príkladu je zrejmé, že efektívnosť kódovania s predpoveďou bude závisieť od výberu parametra k. Ukážeme, ako na základe p vybrať optimálnu hodnotu parametra k. Postupnosti 0^j1 sa vyskytujú s pravdepodobnosťami p^jq $j = 0, 1, \ldots$. Keďže

$$\sum_{j\geq 0} p^j q = q \sum_{j\geq 0} p^j = \frac{q}{1-p} = 1,$$

množina postupností $\{0^j1\}_{j\geq 0}$ s pravdepodobnosťami $P(0^j1) = p^jq$ tvorí pravdepodobnostný priestor. Vypočítame dĺžky kódov jednotlivých postupností a potom určíme strednú hodnotu dĺžky kódovej postupnosti potrebnej na zakódovanie jednej postupnosti 0^j1. V nasledujúcej tabuľke sú uvedené dĺžky kódov postupností 0^j1 pre jednotlivé hodnoty j (označme kvôli zjednodušeniu zápisu symbolom m hodnotu $2^k - 1$):

dĺžka postupnosti	dĺžka kódu
$0 \ldots m-1$	k
$m \ldots 2m-1$	$2k$
$2m \ldots 3m-1$	$3k$

Stredná hodnotu dĺžky kódovej postupnosti potrebnej na zakódovanie jednej postupnosti 0^j1 je

$$k\left[q + pq + \cdots + p^{m-1}q\right] + 2k\left[p^m q + p^{m+1}q + \cdots + p^{2m-1}q\right] +$$
$$+3k\left[p^{2m}q + p^{2m+1}q + \cdots + p^{3m-1}q\right] + \cdots = kq\frac{1-p^m}{1-p}\left[1 + 2p^m + 3^{2m} + \ldots\right] =$$
$$= \frac{k}{1-q^m}.$$

Na zaklade výrazu $\frac{k}{1-q^m}$ nájdeme optimálnu hodnotu dĺžky bloku k. Keďže analytický výpočet minima výrazu $\frac{k}{1-q^m}$ by mohol byť príliš zložitý,

$$\left(1-p^{2^k-1}\right)^{-1} + \frac{kp^{2^k-1}2^k\ln(2)\ln(p)}{\left(1-p^{2^k-1}\right)^2} = 0$$

vypočítame hodnoty výrazu $\frac{k}{1-q^m}$ pre rozličné p a k a určíme optimálne k:

p	0.5	0.6	0.7	0.8	0.9	0.95	0.99	0.999
$k=1$	2.0	2.5	3.3	5.0	10	20	100	1000
$k=2$	2.29	2.55	3.04	4.09	7.38	14.02	67.33	667.33
$k=3$	3.02	3.09	3.27	3.80	5.75	9.94	44.16	429.86
$k=4$	4.00	4.00	4.02	4.15	5.04	7.45	28.58	268.54
$k=5$	5.00	5.00	5.00	5.00	5.20	6.28	18.68	163.72
$k=6$	6.00	6.00	6.00	6.00	6.00	6.24	12.79	98.22
$k=7$	7.00	7.00	7.00	7.00	7.00	7.01	9.71	58.66
$k=8$	8.00	8.00	8.00	8.00	8.00	8.00	8.66	35.52
$k=9$	9.00	9.00	9.00	9.00	9.00	9.00	9.05	22.48
$k=10$	10.00	10.00	10.00	10.00	10.00	10.00	10.00	15.61
$k=11$								12.63

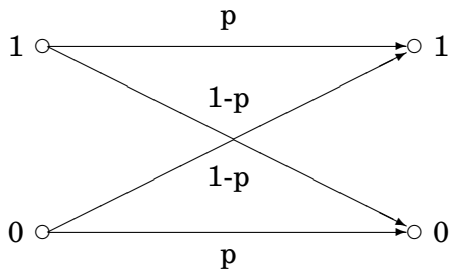
3.6 Slovníkové metódy kompresie údajov

3.7 Kolmogorovská zložitosť

4 Samoopravné kódy

Pri konštrukcii nerovnomerných kódov v predchádzajúcej kapitole sme predpokladali, že prenosový kanál realizuje identickú transformáciu; t.j. že sa správa pri prenose nemení. Tento optimistický predpoklad nebýva v reálnom živote naplnený. Preto sa budeme zaoberať takým zápisom informácie, ktorý umožní kontrolovať zmeny, ku ktorým došlo v priebehu prenosu informácie. [7] Budeme konštruovať rovnomerné (blokové) kódy, ktoré

[7] Pripomíname, že z hľadiska kódovania nie je principiálny rozdiel, či ide o prenos informácie v čase alebo v priestore. Aj preto sa v tejto kapitole budeme zaoberať kódovaním informácie pre prenášanie v priestore, ale riešenia, ktoré navrhneme budú rovnako dobré aj pre ochranu informácie prenášanú v čase.



Obrázok 5: Binárny symetrický kanál bez pamäte

budú schopné odhaľovať chyby (t.j. príjemca bude schopný zistiť, či v prijatom slove vznikli určité chyby alebo nie) alebo ich dokonca opravovať (príjemca bude pri dekódovaní schopný rekonštruovať pôvodne odvysielané kódové slovo) a popíšeme efektívne metódy kódovania a dekódovanie informácie pomocou takýchto kódov.

4.1 Princíp samoopravných kódov

4.1.1 Binárny symetrický kanál bez pamäte

Najprv upresníme predstavu o tom, aké chyby môžu vznikať pri prenose správ. Budeme predpokladať, že pri prenose správ

- dochádza k zámene jedného prenášaného symbolu kódovej abecedy na iný symbol kódovej abecedy,
- žiaden symbol nie je odolnejší voči chybe ako iný symbol; symbol sa prenáša správne s pravdepodobnosťou p a transformuje sa pri prenose na ktorýkoľvak iný symbol s pravdepodobnosťou $\frac{1-p}{q-1}$;
- výsledok prenosu jedného symbolu neovplyvňuje to, či bude nasledujúci symbol prenesený správne alebo nie.

Na popis prenosového kanála zavedieme model, ktorý budeme nazývať q-adickým symetrickým prenosovým kanálom bez pamäte. Špeciálnym a najčastejšie používaným q-adickým symetrickým prenosovým kanálom bez pamäte je binárny $(q = 2)$ symetrický kanál bez pamäte, ktorý je zobrazený na obrázku 5.

Kódy opravujúce chyby predstavujú rozličné algebraické štruktúry ako vektorové priestory, okruhy polynómov, ideály a podobne. Aby mohli kódové slová bez problémov tvoriť takéto algebraické štruktúry, budeme predpokladať, že kódová abeceda je podmnožinou prirodzených čísel; $\Sigma = \{0, \ldots, q-1\}$. Aj keď teoretické konštrukcie budeme robiť pre všeobecný prípad, v ďalšom výklade sa budeme najčastejšie zaoberať binárnymi kódmi, ktoré sa v súčasnosti najčastejšie používajú.

V čom je podstata kódov odhaľujúcich, resp. opravujúcich chyby? Ak by sme na kódovanie správ používali úplné kódy, pri prenose správ by sa jedno kódové slovo mohlo v dôsledku šumu nahradiť iným kódovým slovom a príjemca by mal problém určiť, či

prijal odvysielané kódové slovo, alebo došlo k chybe pri prenose. Preto nie je možné pri komunikácii prostredníctvom kanála so šumom používať úplné kódy. Podstata kódov odhaľujúcich a opravujúcich chyby je v tom, že množina kódových slov tvorí len podmnožinu všetkých možných slov a tak, keď dôjde počas prenosu správy ku chybe, prijaté slovo s veľkou pravdepodobnosťou nie je kódovým slovom. Zdôrazňujeme slová *s veľkou pravdepodobnosťou*, pretože nie je vylúčené, že počas prenosu vznikne chyba, ktorá prenášané kódové slovo transformuje na iné kódové slovo. Pri koštrukcii samoopravných kódov sa snažíme minimalizovať pravdepodobnosť takejto možnosti. Vychádzame z toho, že pre (binárny) prenosový kanál platí $p >> 1-p$[8]; t.j. je pravdepodobnejšie, že pri prenose kódového slova vznikne menej chýb. Ilustrujeme to na príklade.

Príklad 4.1 *Uvažujme binárny symetrický kanál bez pamäte s parametrami* $p = 0.99$, $1-p = 0.01$, *binárny blokový kód dĺžky* 15 *opravujúci tri chyby. V nasledujúcej tabuľke sú uvedené pravdepodobnosti chýb*

počet chýb	*pravdepodobnosť*	
0	$(0.99)^{15}$	0.860058354641289
1	$15 \cdot (0.99)^{14} \cdot (0.01)$	0.130311871915347
2	$\binom{15}{2} \cdot (0.99)^{13} \cdot (0.01)^2$	0.009213970741489
3	$\binom{15}{3} \cdot (0.99)^{12} \cdot (0.01)^3$	0.000403305116631
> 3	$\sum_{j>3} \binom{15}{j} \cdot (0.99)^{15-j} \cdot (0.01)^j$	0.000012497585244

Pravdepodobnosť toho, že v prenášanom slove vzniknú 4 *a viac chýb je síce nenulová* 0.000012497585244 *ale podstatne menšia ako to, že v slove budú najviac* 3 *chyby, ktoré sa pri dekódovaní dajú opraviť.*

4.1.2 Geometrická interpretácia samoopravného kódu

Skôr ako pristúpime ku popisu a konštrukcii samoopravných kódov, využijeme geometrickú interpretáciu kódu a vysvetlíme princíp kódov opravujúcich a odhaľujúcich chyby. Predpokladáme, že máme zostrojiť kód dĺžky n opravujúci t chýb. (Na začiatku kvôli zjednodušeniu popíšeme konštrukciu binárneho kódu opravujúceho 1 chybu a potom konštrukciu zovšeobecníme.) Zavedieme najprv dva dôležité pojmy, ktoré budeme pri koštrukcii samoopravného kódu potrebovať.

Definícia 4.1 *Nech sú* u, v *dva vektory vektorového priestoru* V*; nech* $u = (a_1, \dots a_n)$; $v = (b_1, \dots b_n)$. *Hammingovou váhou vektora* u *nazveme prirodzené číslo* $\boldsymbol{wt}(u)$*, definované nasledovne:*

$$\boldsymbol{wt}(u) = .$$

Hammingovou vzdialenosťou vektorov u, v *nazveme prirodzené číslo* $\boldsymbol{d}(u, v)$*;*

$$\boldsymbol{d}(u, v) = \boldsymbol{wt}(u - v) = \sum_{j=1}^{n} (a_j \neq b_j).$$

[8] V podstate však stačí, aby $p \neq 1 - p$.

Hammingova váha vektora je počet jeho nenulových zložiek a Hamminhova vzdialenosť dvoch vektorov udáva, v koľkých zložkách sa tieto dva vektory odlišujú. Vráťme sa teraz ku konštrukcii binárneho samoopravného kódu opravujúceho 1 chybu. Zostrojíme ho tak, že budeme postupne vyberať kódové slová. Ako prvé kódové slovo v_0 môžeme vybrať ľubovoľný binárny vektor z množiny $\{0,1\}^n$. Bez ujmy na všeobecnosti môžeme vybrať $v_0 = (0,\ldots,0)$; t.j. nulový vektor. Vyberieme teraz druhé kódové slovo v_1. Predpokladajme, že $\mathbf{d}(v_0, v_1) = 1$ a položme $v_1 = (1,0,\ldots,0)$. Potom však existuje chyba (ktorú budeme reprezentovať binárnym vektorom) váhy 1 ktorá transformuje kódové slovo v_1 na kódové slovo v_0:

$$\begin{array}{ll} v_0 & 000\ldots0 \\ v_1 & 100\ldots0 \\ e_1 & 100\ldots0 \\ v_1 \oplus e_1 & 000\ldots0 \quad = v_0 \end{array}$$

To znamená, že $\mathbf{d}(v_0, v_1) > 1$. Nech $\mathbf{d}(v_0, v_1) = 2$ a vyberme napríklad $v_1 = (1,1,0,\ldots,0)$. Žiadna chyba váhy 1 nemôže transformovať slovo v_1 na slovo v_0. Čo sa však stane, ak sme prijali slovo $0100\ldots0$? Existujú dve rovnako pravdepodobné možnosti (a množstvo menej pravdepodobných iných):

$$\begin{array}{lll} v_0 & 000\ldots0 & \\ v_1 & 110\ldots0 & \\ e_1 & 100\ldots0 & \\ e_2 & 010\ldots0 & \\ v_0 \oplus e_1 = & 100\ldots0 & = v_1 \oplus e_2 \\ v_i \oplus e_j = & 100\ldots0 & e_j = v_i \oplus 100\ldots0 \end{array}$$

Ak sme teda prijali slovo $0100\ldots0$, je zrejmé, že to nie je kódové slovo a odhalili sme chybu, ale nevieme ju opraviť a určiť odvysielané kódové slovo. Ak stačí, aby kód odhaľoval chyby váhy 1, vektor $v_1 = (110\ldots0)$ môže byť kódovým slovom. Ak požadujeme, aby kód opravoval chyby váhy $t \geq 1$, vektor $v_1 = (110\ldots0)$ a žiaden vektor váhy 2 nemôže byť kódovým slovom. Nech teda $\mathbf{d}(v_0, v_1) = 3$ a $v_1 = (1,1,1,0,\ldots,0)$. Chybou váhy 1 sa slovo v_1 transformuje v najhoršom prípade na slovo váhy 2, ale chybou váhy 1 sa zo slova v_0 stane vektor váhy 1:

$$\begin{array}{lll} v_0 & 0000\ldots0 & \\ v_1 & 1110\ldots0 & \\ e_1 & 1000\ldots0 & \\ e_2 & 0100\ldots0 & \\ e_3 & 0110\ldots0 & \\ v_0 \oplus e_1 = & 1000\ldots0 & \mathbf{wt}(v_0 \oplus e_1) = 1 \\ v_1 \oplus e_3 = & 1000\ldots0 & \\ v_1 \oplus e_2 = & 1010\ldots0 & \mathbf{wt}(v_1 \oplus e_2) = 2 \end{array}$$

Ak sme prijali slovo $1000\ldots0$, dekódujeme ho na základe toho, že

$$P(1000\ldots0|v_0) > P(1000\ldots0|v_1),$$

ako ν_0. (Ak použijeme hodnoty $n = 15, p = 0.99$ z predchádzajúceho príkladu, tak

$$P(1000\ldots0|\nu_0) = 0.00868745812768978 > 0.000087752102299896 8 = P(1000\ldots0|\nu_1),$$

a teda pravdepodobnosť toho, že bolo odvysielané slovo ν_0 je podstatne väčšia.) Zovšeobecníme teraz našu konštrukciu na prípad, keď má kód opravovať chyby váhy $t > 1$. Ukázalo sa, že rozhodujúcim parametrom, od ktorého závisí opravná schopnosť kódu je minimálna vzdialenosť kódových slov. Zavedieme pre tento pojem špeciálne označenie: minimálnou vzdialenosťou kódu $\mathcal{C}$ nazveme prirodzené číslo

$$d = \min_{u,\nu\in\mathcal{C}} \mathbf{d}(u,\nu).$$

Ak by bola minimálna vzdialenosť kódu $\mathcal{C}$ $d \leq t$ tak potom chybou váhy menšej alebo rovnej t by sa mohlo transformovať jedno kódové slovo na iné kódové slovo. Ak $d = t+1$, kód $\mathcal{C}$ dokáže odhaľovať chyby váhy t. Na to, aby kód $\mathcal{C}$ opravoval chyby váhy t musí byť $d \geq 2t+1$.

Popri samoopravnej schopnosti (danej minimálnou vzdialenosťou kódu) je zaujímavou kvantitatívnou charakteristikou kódu, ktorá vyjadruje jeho efektívnosť, počet kódových slov. Extrémnym prípadom je kód dĺžky $2n+1$ ktorý má dve kódové slová (napr. $0\ldots0$ a $1\ldots1$). Tento kód má síce maximálnu opravnú schopnosť n, ale na prenos jedného bitu správy potrebuje $2n+1$ kódových symbolov. Pri konštrukcii samoopravných kódov sa snažíme o kompromis medzi opravnou schopnosťou a mohutnosťou kódu. Koľko kódových slov môže vlastne obsahovať samoopravný kód dĺžky n opravujúci t chýb? Pozrieme sa najprv na binárny prípad. Množina binárnych vektorov (neskôr ukážeme, že sa jedná o vektorový priestor), z ktorej vyberáme kódové slová, má 2^n prvkov. Označíme symbolom $S(\nu, r)$ množinu vektorov;

$$S(\nu, r) = \{u | u \in \{0,1\}^n \& \mathbf{d}(u,\nu) \leq r\},$$

ktorú budeme nazývať sférou s polomerom r a stredom ν. Je zrejmé, že platí

$$\forall u, \nu \in \mathcal{C};\ (u \neq \nu) \Rightarrow S(u,t) \bigcap S(\nu,t) = \emptyset$$

a mohutnosť $S(\nu, r)$ je

$$|S(\nu,r)| = \sum_{j=0}^{r} \binom{n}{j}.$$

Ak by kód $\mathcal{C}$ mal maximálny počet kódových slov (pre dĺžku kódu n a opravnú schopnosť t), potom by množina vektorov $\{0,1\}^n$ musela byť pokrytá disjunktnými sférami polomeru t so stredami v kódových slovách. Mohutnosť kódu $\mathcal{C}$ by v takomto prípade bola

$$|\mathcal{C}| = \frac{2^n}{\sum_{j=0}^{t} \binom{n}{j}}.$$

Je zrejmé, že je len málo takých hodnôt n, t pre ktoré by bol podiel $\frac{2^n}{|S(\nu,r)|}$ celočíselný. Ak by sme sa však aj uspokojili s kódom menšej mohutnosti, zostáva otázkou, ako ho

zostrojiť. Úplné preberanie neprichádza do úvahy, nakoľko jeho zložitosť je odvodená od čísla

$$\binom{2^n}{\lfloor \frac{2^n}{\sum_{j=0}^{t}\binom{n}{j}} \rfloor},$$

ktoré je pre relatívne malé hodnoty n, t veľké (pozri nasledujúcu tabuľku). V tejto kapitole sa budeme zaoberať metódami systematického vytvárania samoopravných kódov.

n	t	mohutnosť kódu	počet možných kódov
7	1	16	93343021201262177400
7	2	4	10668000
7	3	2	8128
15	5	6	1718574240691455027134464
15	4	16	841373212397480523637905290517658013716524288174947313889 28
15	3	56	$0.9837970552 \times 10^{178}$
15	2	270	$0.7332302377 \times 10^{678}$
15	1	2048	$0.1094851418 \times 10^{3326}$

Poznámka. Aká bude mohutnosť samoopravných kódov nad inou ako binárnou abecedou? Mohutnosť sféry s polomerom t vo vektorovom priestore $\{0, \ldots, q-1\}$ je

$$\sum_{j=0}^{t}\binom{n}{j}(q-1)^j.$$

Ak $q > 2$ nestačí len vybrať j zložiek vektora, ktoré treba zmeniť, ale je potrebné aj určiť akým symbolom sa má pôvodný symbol nahradiť. Pre mohutnosť q-kódu V dĺžky n, opravujúceho t chýb platí

$$|V| \leq \frac{q^n}{\sum_{j=0}^{t}\binom{n}{j}(q-1)^j}.$$

Skôr ako sa budeme zaoberať systematicky metódami konštrukcie rozličných samoopravných kódov, uvedieme niekoľko jednoduchších príkladov kódov opravujúcich alebo odhaľujúcich chyby a ilustrujeme na nich už zavedené, resp. zavedieme niektoré nové pojmy. Odteraz sa až do odvolania budeme opäť zaoberať binárnymi kódmi.

4.1.3 Jednoduché kódy odhaľujúce/opravujúce chyby

Testovanie parity. Nech je daná množina binárnych vektorov dĺžky n. Pridáme ku každému vektoru $n+1$-bit tak aby počet jednotkových bitov vo vektore (dĺžky $n+1$) bol párny. Kódové slová budú potom vyzerať nasledovne (doplnený bit je oddelený medzerou):

$$\begin{array}{ll} 01000011100001010 & 0 \\ 01011010010101011 & 1 \end{array}$$
$$\ldots$$

Doplnený bit sa nazýva *paritným bitom.* Ak v kódovom slove vznikne pri prenose chyba nepárnej váhy $(1, 3, 5, \dots)$ počet jednotkových bitov v prijatom slove bude nepárny a príjemca bude vedieť, že nastala chyba (aj keď nedokáže určiť, kde.) Ak by však pri prenose nastala chyba nepárnej váhy $(2, 4, \dots)$, v prijatom slove bude párny počet jednotkových bitov a príjemca bude prijaté slovo považovať za kódové slovo. Ak sa vrátime k predchádzajúcemu príkladu ($p = 0.99,\ n = 15$), tak pravdepodobnosť neodhalenej chyby je 0.009226196681.

Obdĺžnikové kódy. Uvažujme opäť binárne zapísanú informáciu, ktorú chceme upraviť do formy umožňujúcej opraviť aspoň jednu chybu (chybu váhy 1). Zapíšeme informáciu do obdĺžnikovej matice typu $m \times n$ a pridáme k nej jeden kontrolný riadok a jeden kontrolný stĺpec.

0110101010	1
1110000111	0
1010101010	1
0010000111	0

Na i-tom mieste kontrolného stĺpca sa bude nachádzať paritný bit i-teho riadku ($a_{i,10} = a_{i,0} \oplus \cdots \oplus a_{i,9}$), na j-tom mieste kontrolného riadku sa bude nachádzať paritný bit j-teho stĺpca ($a_{3,j} = a_{0,j} \oplus \cdots \oplus a_{2,j}$). Predpokladajme, že nastala chyba váhy 1 napríklad na mieste $(0, 4)$. Príjemca vyčísli kontrolné sumy pre riadky aj stĺpce prijatej matice a zistí pozíciu chyby:

0110<u>0</u>01010	1	1
1110000111	0	0
1010101010	1	0
0010000111	0	0
0000100000	0	

Všimneme si, že ak vznikne chyba váhy 1 v kontrolnom riadku alebo v kontrolnom stĺpci, pozícia chyby sa určí úplne rovnako, ako v prípade chyby v "informačnom" symbole:

0110101010	1	0
1110000111	0	0
1010101010	1	0
0010000110	0	1
0000000001	0	

Obdĺžnikový kód je schopný opravovať chyby váhy 1. Čo sa stane, ak v kódovom slove vznikne chyba väššj váhy? Predpokladajme, že vznikla chyba váhy 2:

<u>0</u>110101010	1	1
1110000111	0	0
1010101010	1	0
0010000110	0	1
1000000001	0	

Vyčíslením kontrolných súm príjemca zistí, že vznikla chyba väčšej váhy. Ak by aj uhádol, že ide o chybu váhy 2, nevie či chyby vznikli v symboloch $a_{0,0}, a_{3,9}$ alebo $a_{3,0}, a_{0,9}$. Ak by chyba váhy 2 vznikla v tom istom riadku (stĺpci), na kotrolnej sume príslušného riadku (stĺpca) by sa to neprejavilo, a príjemca by vedel akurát povedať, že v niektorých stĺcoch (riadkoch) vznikla chyba väčšej váhy.

1110101011	1	0
1110000111	0	0
1010101010	1	0
0010000111	0	0
1000000001	0	

Samoopravné kódy sa zakladajú na tom, že

- nie každé možné slovo je kódovým slovom;
- kódové slová sú „dosť ďaleko od seba“.

Minimálna vzdialenosť kódu vyjadrená pomocou Hammingovej vzdialenosti kódových slov nám umožnila precizovať význam slov „dosť ďaleko od seba“. Pomocou obdĺžnikových kódov ilustrujeme pojem „redundancie (nadbytočnosti)“, ktorý upresňuje prvú požiadavku kladenú na samoopravné kódy. V kódovom slove obdĺžnikového kódu rozlišujeme dva druhy symbolov: informačné (pomocou nich sa zapisuje informácia, ktorú má kódové slovo preniesť) a kontrolné symboly (zaznamenávajúce štruktúru kódového slova.) Dĺžka kódového slova sa zvykne označovať symbolom n, počet informačných symbolov k a počet kontrolných symbolov je potom $n-k$. Samoopravný kód, ktorý má dĺžku n a počet informačných symbolov k sa označuje aj ako (n,k)-kód. Počet kontrolných symbolov sa nazýva *absolútnou redundanciou* kódu. Kódy s rozličnými dĺžkami môžu mať rozličné počty kontrolných symbolov. Aby ich bolo možné porovnávať z hľadiska redundancie, zavádzame pojem *relatívnej redundancie kódu*, definovanej ako podiel počtu kontrolných symbolov k celkovej dĺžke kódového slova; $\frac{n-k}{n} = 1 - \frac{k}{n}$. Určíme absolútnu a relatívnu nadbytočnosť obdĺžnikových kódov. Predpokladajme kvôli jednoduchosti, že kódové slovo obdĺžnikového kódu má tvar štvorcovej matice typu $m \times m$[9]. Táto matica obsahuje štvorcovú podmaticu $(m-1)\times(m-1)$ informačných symbolov a $2m-1$ kontrolných symbolov. Relatívna nadbytočnosť štvorcového kódu je $\frac{2m-1}{m^2} = \frac{2}{m} - \frac{1}{m^2}$. Pre veľké m je relatívna nadbytočnosť štvorcového kódu zanedbateľná.

V prípade obdĺžnikového kódu bolo možné rozlíšiť informačné a kontrolné symboly. Existujú samoopravné kódy, pre ktoré takéto rozdelenie symbolov kódového slova neexistuje. Aby bolo možné vyjadriť redundanciu aj pre tieto kódy, zovšeobecníme pojem redundancie nasledujúcim spôsobom.

Definícia 4.2 *Nech V je kód dĺžky n nad abecedou $\{0,\ldots,q-1\}$. (Relatívnou) redundanciou kódu V je*

$$R(V) = \frac{\log_q |V|}{n}.$$

[9] v takomto prípade hovoríme o štvorcovom kóde

Redundancia kódu úzko súvisí s ďalším dôležitým pojmom, pomocou ktorého sa vyjadruje efektívnosť kódu; s prenosovou rýchlosťou. Prenosová rýchlosť kódu je číslo z intervalu $< 0, 1 >$, ktoré je definované ako

$$\frac{\text{počet prenesených informačných symbolov}}{\text{celkový počet prenesených symbolov}} = 1 - R.$$

V ďalšom sa budem zaoberať kódmi, ktoré majú vysoké prenosové rýchlosti a zároveň dobré opravné schopnosti. Začneme zaujímavým kódom opravujúcim jednu chybu.

4.1.4 Hammingov kód

Hammingove kódy sú binárne (n, k)-kódy, kde $n = 2^m - 1,\ m \geq 3,\ m \in N,\ k = 2^m - 1 - m$ opravujúce chyby váhy 1. Princíp vytvárania Hammingových kódov, kódovanie a dekódovanie ilustrujeme na Hammingovom $(15, 11)$-kóde.

Predpokladajme, že sme už vytvorili kódové slovo $v = (v_1, \ldots, v_{15})$. Z jednotlivých komponentov kódového slova vytvoríme 4 kontrolné sumy s_0, s_1, s_2, s_3, pomocou ktorých budeme schopní rozlišovať 16 rozličných udalostí: pri prenose nenastala žiadna chyba, nastala chyba váhy 1 v $1., \ldots, 15.$ komponente kódového slova. Zavedieme dva potrebné pojmy a potom vytvoríme kontrolné sumy. Symbolom $\sigma(i, n)$ budeme označovať n-bitový vektor, ktorý je binárnou reprezentáciou čísla i. Nech sú $u = (u_1, \ldots, u_n), v = (v_1, \ldots, v_n)$ dva binárne vektory, symbolom $u\&v$ budeme označovať vektor $u\&v = (u_1 v_1, \ldots, u_n v_n)$. Pre $j = 0, 1, 2, 3$ platí:

$$s_j = \bigoplus_{\sigma(i,4)\&\sigma(2^j,4)=\sigma(2^j,4)} v_i,$$

t.j.

$$\begin{aligned}
s_0 &= v_1 \oplus v_3 \oplus v_5 \oplus v_7 \oplus v_9 \oplus v_{11} \oplus v_{13} \oplus v_{15} \\
s_1 &= v_2 \oplus v_3 \oplus v_6 \oplus v_7 \oplus v_{10} \oplus v_{11} \oplus v_{14} \oplus v_{15} \\
s_2 &= v_4 \oplus v_5 \oplus v_6 \oplus v_7 \oplus v_{12} \oplus v_{13} \oplus v_{14} \oplus v_{15} \\
s_3 &= v_8 \oplus v_9 \oplus v_{10} \oplus v_{11} \oplus v_{12} \oplus v_{13} \oplus v_{14} \oplus v_{15}.
\end{aligned}$$

Všimnime si, že komponent v_i sa vyskytuje práve vo $\mathbf{wt}(\sigma(i, 4))$ kontrolných sumách. Keďže existujú práve štyri binárne vektory dĺžky 4 s Hammingovou váhou 1 reprezentujúce čísla $1, 2, 4, 8$, každý z komponentov v_1, v_2, v_4, v_8 vystupuje v jednej kontrolnej sume. To znamená, že ak zvolíme hodnoty komponentov $v_3, v_5, v_6, v_7, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}$ kódového slova ľubovoľne, vhodnou voľbou komponentov v_1, v_2, v_4, v_8 dosiahneme, že kontrolné sumy budú pre kódové slovo nulové: $s_0 = s_1 = s_2 = s_3 = 0$. Stačí položiť:

$$\begin{aligned}
v_1 &= v_3 \oplus v_5 \oplus v_7 \oplus v_9 \oplus v_{11} \oplus v_{13} \oplus v_{15} \\
v_2 &= v_3 \oplus v_6 \oplus v_7 \oplus v_{10} \oplus v_{11} \oplus v_{14} \oplus v_{15} \\
v_4 &= v_5 \oplus v_6 \oplus v_7 \oplus v_{12} \oplus v_{13} \oplus v_{14} \oplus v_{15} \\
v_8 &= v_9 \oplus v_{10} \oplus v_{11} \oplus v_{12} \oplus v_{13} \oplus v_{14} \oplus v_{15}.
\end{aligned}$$

Kódovanie správ pomocou Hammingovho $(15,11)$**-kódu** prebieha tak, že sa správa najprv rozdelí na bloky dĺžky 11 a tie sa doplnia 4 kontrolnými symbolmi na kódové slovo:

1	1	1	1	0	0	0	0	1	1	1					informačný vektor
		1		1	1	1		0	0	0	0	1	1	1	informačný vektor
1	1		0			1									kontrolný vektor
1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	kódové slovo

Dekódovanie Hammingovho $(15,11)$**-kódu.** Predpokladajme, že pri prenose kódového slova vznikla chyba váhy 1 v i-tom komponente kódového slova; t.j. bolo prijaté slovo

$$v_0,\ldots,v_{i-1},v_i\oplus 1,v_{i+1},\ldots,v_{15}.$$

Chyba spôsobí, že všetky kontrolné sumy, ktoré obsahujú komponent v_i nadobudnú hodnotu 1. To sú však práve tie sumy s_j, pre ktoré $\sigma(i,4)\&\sigma(2^j,4)=\sigma(2^j,4)$; t.j. binárny vektor $s=(s_3,s_2,s_1,s_0)$ predstavuje číslo $\sigma(i,4)$. Vektor hodnôt jednotlivých kontrolných súm sa nazýva *syndróm chyby*. V našom prípade syndróm chyby predstavuje pozíciu, na ktorej chyba váhy 1 v kódovom slove vznikla, resp. nulová hodnota syndrómu chyby znamená, že bolo prijaté kódové slovo.

Príklad 4.2 *Predpokladajme, že chyba vznikla v* 13. *komponente kódového slova. Potom bolo prijaté slovo:*

$$v_1,\ldots,v_{12},v_{13}\oplus 1,v_{14},v_{15}.$$

Kontrolné sumy nadobúdajú hodnoty:

$$\begin{aligned}
s_0 &= v_1\oplus v_3\oplus v_5\oplus v_7\oplus v_9\oplus v_{11}\oplus(v_{13}\oplus 1)\oplus v_{15} = 1\\
s_1 &= v_2v_3\oplus v_6\oplus v_7\oplus v_{10}\oplus v_{11}\oplus v_{14}\oplus v_{15} = 0\\
s_2 &= v_4\oplus v_5\oplus v_6\oplus v_7\oplus v_{12}\oplus(v_{13}\oplus 1)\oplus v_{14}\oplus v_{15} = 1\\
s_3 &= v_8\oplus v_9\oplus v_{10}\oplus v_{11}\oplus v_{12}\oplus(v_{13}\oplus 1)\oplus v_{14}\oplus v_{15} = 1
\end{aligned}$$

Hammingov kód nie je schopný opravovať chyby váhy ≥ 2. Pri dekódovaní sa takéto chyby buď vôbec neodhalia alebo sa interpretujú ako chyby váhy 1:

1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	kódové slovo
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	chybový vektor
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	prijaté slovo
0	0	0	0												syndróm chyby
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	predpokladaná chyba
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	dekódované slovo
1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	kódové slovo
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	chybový vektor
0	0	1	0	1	1	1	1	0	0	0	0	1	1	1	prijaté slovo
0	0	1	1												syndróm chyby
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	predpokladaná chyba
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	dekódované slovo

Porovnáme ešte redundanciu Hammingových R_H a obdĺžnikových kódov R_O.

n	$a \times b$	$(n-k)_H$	$(n-k)_O$	R_H	R_O
$7^{(*)}$	2×4	3	5	0.4285	0.6250
15	3×5	4	7	0.2666	0.4666
$31^{(*)}$	4×8	5	10	0.1612	0.3125
63	7×9	6	15	0.0952	0.2380
$127^{(*)}$	8×16	7	23	0.0551	0.1796
255	15×17	8	31	0.0313	0.1215
511	16×32	9	72	0.0176	0.1409
1023	31×33	10	63	0.0097	0.0615

V prípadoch označených hviezdičkou neexistujú obdĺžnikové kódy potrebnej dĺžky (n je prvočíslo), a preto sme Hammingove kódy dĺžky n porovnávali s obdĺžnikovými kódmi dĺžky $n+1$. Aj pre $n = 511$ má obdĺžnikový kód dĺžky 512 rozmerov 16×32 menšiu redundanciu (0.0917) ako obdĺžnikový kód dĺžky 511 rozmerov 7×73.

Poznámka. Pre Hammingove kódy platí ($n = 2^m - 1,\ m \geq 3$)

$$\left[\binom{n}{0} + \binom{n}{1}\right] \mid 2^n.$$

Hammingove kódy však nie sú jedinými kódmi, pre ktoré je mohutnosť sféry mocninou dvojky. Pre $n = 23$ a $t = 3$ platí:

$$\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3} = 2^{11};$$

čím je splnená nutná podmienka pre existenciu binárneho samoopravného kódu dĺžky 23 opravujúceho chyby váhy 3. Taký kód skutočne existuje, je to Golayov (23,12)-kód, ktorým sa budeme zaoberať v časti xxx.

4.2 Lineárne kódy

4.2.1 Reed-Mullerove kódy

V tejto časti uvedieme podrobnejšie jeden špeciálny prípad lineárnych kódov, Reed-Mullerove kódy, ktoré majú jednoduchý popis a jednoduché dekódovanie. Reed-Mullerove kódy sú charakterizované dvoma základnými parametrami - rádom r a hodnotou $m; 0 \leq r < m$ určujúcou dĺžku kódového slova. Existujú Reed-Mullerove kódy s rozličnými dĺžkami kódových slov a rôznymi opravnými schopnosťmi. Reed-Mullerov kód s parametrami r, m budeme označovať symbolom $\mathcal{R}(m, r)$. V nasledujúcej tabuľke sú uvedené základné parametre kódu $\mathcal{R}(m, r)$.

$n = 2^m$	dĺžka kódu (kódového slova)
$k = \sum_{0 \le j \le r} \binom{m}{j}$	počet informačných symbolov
$n - k = \sum_{r < j \le m} \binom{m}{j}$	počet kontrolných symbolov
$d = 2^{m-r}$	minimálna váha/vzdialenosť kódu

Tabuľka xxx Základné parametre Reed-Mullerových kódov

Keďže Reed-Mullerove kódy sú lineárne kódy, možno ich zadať pomocou generujúcej matice. Generujúca matica pre Reed-Mullerov kód $\mathcal{R}(m, r)$ má zvláštny tvar:

$$G = \begin{bmatrix} G_0 \\ G_1 \\ \vdots \\ G_r \end{bmatrix}$$

Aby sme mohli popísať konštrukciu generujúcej matice G kódu $\mathcal{R}(m, r)$, zavedieme operáciu súčinu vektorov. Nech sú $u = (a_1, \ldots, a_n)$ a $v = (b_1, \ldots, b_n)$ dva vektory vektorového priestoru V. Súčinom (pozor, nejedná sa ani o vektorový ani o skalárny súčin vektorov) vektorov u, v nazveme vektor $uv = (a_1 b_1, \ldots, a_n b_n)$. (Ide o súčin vektorov po zložkách; v binárnom prípade môžeme pomocou konvencie jazyka C zapísať súčin vektorov u, v nasledovne $uv = u\&v$.) Podmatice $G_0, \ldots, G_r$ generujúcej matice G sú definované nasledovne:

1. G_0 je jednotkový vektor dĺžky 2^m;
2. G_1 je matica typu $m \times 2^m$, ktorej stĺpcami sú všetky možné binárne vektory dĺžky m;
3. G_l, $1 \ge l \ge r$ je binárna matica typu $\binom{m}{l} \times 2^m$; riadkami podmatice G_l sú všetky vektory, ktoré sú výsledkom súčinu l vektorov z matice G_1.

Ilustrujeme konštrukcie generujúcej matice Reed-Mullerových kódov na príklade $\mathcal{R}(4, 2)$.

Príklad 4.3

$$G = \begin{bmatrix} G_0 = \begin{bmatrix} 1111111111111111 \end{bmatrix} \\ G_1 = \begin{bmatrix} 0000000011111111 \\ 0000111100001111 \\ 0011001100110011 \\ 0101010101010101 \end{bmatrix} \\ G_2 = \begin{bmatrix} 0000000000001111 \\ 0000000000110011 \\ 0000000001010101 \\ 0000001100000011 \\ 0000010100000101 \\ 0001000100010001 \end{bmatrix} \end{bmatrix}$$

Kódovanie a dekódovanie Reed-Mullerových kódov. Nech je daný vektor informačných symbolov i dĺžky k. Kódové slovo vytvoríme tak, že vynásobíme informačný vektor generujúcou maticou. Pri dekódovaní prijatého slova by sme mohli použiť metódu dekódovania lineárnych kódov, vynásobiť prijaté slovo kontrolnou maticou, vypočítať syndróm chyby, na jeho základe určiť najpravdepodobnejšiu chybu a odčítať tento chybový vektor od prijatého slova. Ukážeme inú metódu dekódovania Reed-Mullerových kódov

4.3 Cyklické kódy

4.4 Bose-Chandhury-Hocquenghove kódy

5 Shannonova teoréma

Aby sme pri prenose správ pomocou kanála so šumom zaistili možnosť správneho dekódovania prenesenej (a možno modifikovanej správy), musíme k informačným symbolom pridať nejaké kontrolné symboly (a zvýšiť tak redundanciu správy). Na prvý pohľad sa zdá, že čím väčšiu samoopravnú schopnosť požadujeme, tým viac symbolov kódového slova tvoria kontrolné symboly, tým väčšia je redundancia prenášanej informácie a tým menšia je prenosovová rýchlosť. V tejto kapitole dokážeme prekvapujúci Shannonov výsledok, ktorý hovorí, že existujú kódy, pre ktoré sa pravdepodobnosť nesprávneho dekódovania blíži k 0 a prenosová rýchlosť k 1. Uvedieme Hammingov dôkaz Shannonovej teorémy [2], formálnejší dôkaz možno nájsť vo van Lintovej práci [5].

Shannonovu teorému budeme dokazovať pre prípad binárneho symetrického kanálu z viacerých dôvodov. Binárny symetrický kanál je dobrým modelom pre najčastejšie používané komunikačné kanály; dôkaz je v tomto prípade jednoduchší a použitie zovšeobecného modelu prenosového kanála dôkaz skomplikuje, ale neprinesie nejaké podstatné zovšeobecnenie výsledku.

Predpokladajme, že odosielateľ A posiela príjemcovi B binárne kódované správy prostredníctvom binárneho symetrického kanála, ktorý prenáša binárny symbol správne s pravdepodobnosťou p a nesprávne s pravdepodobnosťou $q = 1 - p$; $p > 1 - p$. Ďalej budeme predpokladať, že na kódovanie používa blokové kódy s dĺžkou kódového slova n. Nech je u odvysielané kódové slovo kódu $\mathcal{C}$ a v prijaté slovo. Príjemca B bude používať metódu dekódovania na základe maximálnej pravdepodobnosti; t.j. prijaté slovo v dekóduje ako kódové slovo u^* také, že

$$P(v|u^*) > P(v|u_i); \quad u_i \neq u^*;\ u^*, u_i \in \mathcal{C}.$$

Aká bude pravdepodobnosť nesprávneho dekódovania? Aby sme dosiahli maximálnu opravnú schopnosť kódu a minimalizovali pravdepodobnosť nesprávneho dekódovania, musíme vybrať kódové slová tak, aby miminálna vzdialenosť výsledného kódu bola maximálna. Keďže zatiaľ to nevieme spraviť, vyberieme kódové slová kódu $\mathcal{C}$ náhodne. Ak $|\mathcal{C}| = M$, tak kód $\mathcal{C}$ možno vybrať $\binom{2^n}{M}$ spôsobmi. Aby sme zjednodušili výpočty, budeme predpokladať, že sa slová pri výbere môžu opakovať. Pravdepodobnosť toho, že náhodným výberom M slov vytvoríme práve kód $\mathcal{C}$, je 2^{-nM}. Využijeme teraz matematický

model prenosového kanála a odhadneme počet chýb, ktoré môžu vzniknúť pri prenose kódového slova dĺžky n. Zavedieme náhodnú premennú ξ (= počet chýb pri prenose kódového slova dĺžky n.) Keďže chyby vznikajú nezávisle na sebe s pravdepodobnosťou $1-p$, pravdepodobnosť vzniku t chýb $t = 0, \dots n$ je $P(\xi = t) = \binom{n}{k} p^{n-t}(1-p)^t$ a náhodná premenná ξ má binomické rozdelenie pravdepodobnosti so strednou hodnotou $E(\xi) = n(1-p)$ a disperziou $Var(\xi) = np(1-p)$. Použijeme teraz Čebyševovu nerovnosť a odhadneme pravdepodobnosť toho, že pri prenose kódového slova dĺžky n vznikne aspoň $t = \lfloor n(1-p) + \sqrt{np(1-p)/(\varepsilon/2)} \rfloor$ chýb:

$$P(\xi > t) \leq \varepsilon/2.$$

Keďže $p > 1/2$, $t < n/2$ pre dostatočne veľké n. Pre ľubovoľnú konštantu $\lambda < 1/2$ platí nasledujúci asymptotický odhad

$$\sum_{i=0}^{\lambda n} \binom{n}{i} \simeq \binom{n}{\lambda n} \frac{1-\lambda}{1-2\lambda}.$$

Položíme $\lambda = 1-p < 1/2$ a odhadneme mohutnosť sféry s polomerom t:

$$\sum_{i=0}^{(1-p)n} \binom{n}{i} \simeq \binom{n}{(1-p)n} \frac{p}{2p-1}.$$

Teraz použijeme Stirlingovu formulu

$$n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n (1 + O(n^{-1}))$$

a upravíme:

$$\binom{n}{(1-p)n} \frac{p}{2p-1} = \frac{p}{2p-1} \times \frac{1}{\sqrt{2\pi np(1-p)}} \times \left(\frac{1}{p^p(1-p)^{1-p}}\right)^n \times (1 + O(n^{-1})) < 2^{nH(p)},$$

kde $H(p) = p \lg p + (1-p)\lg(1-p)$ je entropia.[10] Príjemca dekóduje prijaté slovo v neprávne vtedy, ak sa odvysielané kódové slovo u nenachádza vo sfére $S(v,t)$, resp. ak sféra $S(v,t)$ obsahuje viacero kódových slov. Pravdepodobnosť chyby pri dekódovaní bude:

$$P_E = P(u \notin S(v,t)) + P(u \in S(v,t)) \times P(\exists u'(u' \neq u)\&(u' \in \mathcal{C})\&(u' \in S(v,t)).$$

Keďže $P(u \in S(v,t)) \leq 1$

$$P_E \leq P(u \notin S(v,t)) + P(\exists u'(u' \neq u)\&(u' \in \mathcal{C})\&(u' \in S(v,t)).$$

Odhadneme zhora pravdepodobnosť

$$P(\exists u'(u' \neq u)\&(u' \in \mathcal{C})\&(u' \in S(v,t)) \leq \sum_{u' \in \mathcal{C} - \{u\}} P(u' \in S(v,t))$$

[10] Bolo by možné zostrojiť aj presnejší odhad mohutnosti sféry ale na dôkaz Shannonovej teorémy dosiahnutý odhad plne postačuje.

Pre pravdepodobnosť nesprávneho dekódovania dostávame nasledujúci odhad

$$P_E \leq \varepsilon/2 + \sum_{u' \in \mathcal{C} - \{u\}} P(u' \in S(v,t)).$$

Teraz vypočíteme strednú hodnotu chyby dekódovania. Predpokladáme, že všetky možné binárne blokové kódy sú rovnako pravdepodobné a teda že každý z nich možno vybrať s pravdepodobnosťou 2^{-nM}. Strednú hodnotu chyby dekódovania budeme označovať symbolom $\overline{P_E}$. Keďže ε je konštanta, dostávame:

$$\overline{P_E} \leq \varepsilon/2 + (M-1) \cdot \overline{P(u' \in S(v,t))} \leq \varepsilon/2 + M \cdot \overline{P(u' \in S(v,t))} \qquad (u \neq u').$$

Každé kódové slovo sme vybrali náhodne z množiny všetkých n-bitových vektorov, a preto

$$\overline{P(u' \in S(v,t))} = \frac{|S(v,t)|}{2^n} < 2^{-n(1-H(p))}, \quad (u \neq u')$$

Dokončenie neskôr.

6 Hranice kódov

7 Algebraické základy samoopravných kódov

8 Prehľad najdôležitejších pojmov

Referencie

[1] J.Adamek:*Foundations of Coding*, John Wiley, Chichester 1991

[2] R.W.Hamming:*Coding and Information Theory*, Prentice Hall, New Jersey, 1980

[3] W.W.Peterson, E.J.Weldon: *Error-Correcting Codes*, 2-nd ed., MIT Press, Cambridge, 1972

[4] S.V.Jablonskij, O.B.Lupanov: *Diskrétna matematika a matematické otázky kybernetiky* (v ruštine)

[5] J.H.van Lint: *Introduction to Coding Theory*, 3-rd ed. Springer 1998

[6] K.Rektorys et al.: *Přehled užité matematiky*, SNTL, Praha 1981

[7] A. Rényi: *Teorie pravděpodobnosti*, Academia, Praha 1972