Krátky abstrakt s kľúčovými slovami a deleniami k predmetu Teória kódovania

Peter Csiba, petherz@gmail.com

27.05.2011

Contents

1 Úvod

Rozdelenie hlavných informačných komponentov

- Údaje - dáta.
- Informácia - interpretovaný údaj.
- Správa - údaje v nejakom formáte.
- Komunikácia - výmena údajov, resp. informácií.

Hlavné komponenty informačného prenosu (Shannon)

- Zdroj S- Generuje znaky, ktoré treba preniesť.
- Kóder K1- Transformácia znakov.
- Kóder K2-
- Modulátor- Kódovú abecedu do fyzických signálov.
- Vysielač- Vysiela signály.
- Prenosový kanál- Prenosové médium medzi vysielačom a prijímačom.
- Prijímač- Zachytáva signály.
- Demodulátor- Transformuje signál (vlny) do kódovej abecedy.
- Dekóder D2-
- Dekóder D1- Dekóduje.
- Príjemca- Príjemca vysielanej informácie.

Atribúty kódovacíh prvkov

- Dĺžka kódu - ? koľko kódových slov má kód.
- Zdrojová abeceda - vo všeobecnosti nám dáva zdroj znaky a nie slová. Počet znakov sa tradične označuje m.
- Kódové slova - slová nad kódovou abecedou (väčšinou 0,1) priradené zdrojovým znakom.
- Entropia - koľko informácie nesie. Resp.: "Entropia je veličina ktorá popisuje neurčitosť náhodnej premennej." (wiki). Konkrétnejšie pre pravdepodobnostný zdroj:

$$Entropia = -\sum_{i=0}^{m-1} p_i \cdot log_q(p_i) \tag{1}$$

  Kde m je počet kódových slov a q je veľkosť kódovej abecedy.
- Redundacia kódu - koľko nadbytočnej informácie kód nesie. $Redundancia = 1 - entropia$.
- Index koincidencie - ?

- Cena kódu - model z pravdepodobnostným rozložením zdrojových znakov. Cena kódu $L(P, V)$ je potom stredná hodnota dĺžky kódového slova. Zaujíma nás, aká je minimálna cena a kedy je ju možné dosiahnuť.
- Pole $GF(n)$ - Galoise Field. Inštancia konečného poľa veľkosti n. Definuje sa pre $n = p^k$ kde p je prvočíslo. Všetky polia takej veľkosti sú izomorfné s $GF(n)$.
- Hammingova váha vektora $wt(v)$ - počet nenulových prvkov (bitov) vektora v.
- Hammingova vzdialenosť vektorov $d(u, v)$ - počet rôznach korešpondujúcich prvkov (bitov).
- Minimálna vzdialenosť kódu - minimálna vzdialenosť dvoch kódových slov.
- Data compaction (bezstratová kompresia)- napr. ZIP.
- Data compression (istá strata informácie)- napr. JPEG.
- Hard quantization - Demodulátor vždy príjme rozhodnutie o interpretácií signálu.
- Soft quantization - Demodulátor sa môže rozhodnúť o signály, že je príliš nejednoznačný (- napr. veľký šum).
- MLD (Maximum Likehood Decoding) - (D2) prijaté slovo dekódujeme na slovo, ktoré je najpravdepodobnejšie z množiny kódových slov.
- Úplne dekódovanie - vždy dekódujeme.
- Neúplne dekódovanie - môžeme nedekódovať.
- IMLD (Incomplete MLD) - MLD pričom sa môžeme rozhodnúť, že nedekódujeme.
- Chyba dekódera - nesprávne interpretovanie získanej správy.
- Zlyhanie dekódera - chyba dekódera alebo neschopnosť dekódovať.
- Rozdeliteľný kód - slovo vieme jednoznačne rozdeliť na kódové slová.
- Blokové (rovnomerné) kódy - každé kódové slovo je rovnako dlhé.
- Nerovnomerné kódy - opak blokových, napr. sú to prefixové.
- Guľa vo vektorovom priestore s metrikou - $B_r(v)$ pričom $|B_r(v)| = \sum_{j=0}^{r} \binom{n}{j}$.
- Repetition code (kód s opakovaním) - to isté pošleme viackrát. Zvyšujeme si tým šance detekcie chyby.
- Burst error - ak je pri viacerých chybách v správe očakávané, že budú nasledovať približne za sebou.

2 Nerovnomerné kódy

- Prefixové kódy - Majú viacero výhod. Rozdeliteľnosť a dekódovanie z ľava doprava (v poradí, v akom dostávame údaje). Vieme ho akceptovať DKA (to sa volá aj automatické dekódovanie).
- Sufixové kódy - sú iným príkladom rozdeliteľných kódov. V ich prípade ale potrebujeme mať celú správu dočítanú.
- Kraft-McMillanova nerovnosť - nutná a postačujúca podmienka na existenciu rozdeliteľného kódu (a že prefixový má takú vlastnosť).

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

  Dôkaz spočíva v uvážení sumy ako generujúcej funkcie a následne rozšírením kódu (n - krát zreťaziť). Ďalej sa odhadne dĺžka najkratšieho slova a označí dĺžka najdlhšieho. Potom sa snažíme za predpokladu rozdeliteľnosti dostať na ľavej strane exponenciálnu funkciu od n na ľavej strane a lineárnu funkciu od n na pravej strane nerovnosti. Tým nám vznikne spor.
- Shannonov kód pre dĺžky kódových slov - zoberú sa prefixové súčty z Kraft-McMillana a reprezentujú sa v binárnej sústave. Desatinnú časť doplníme na dĺžku l_i nulami a budeme ju považovať za kódové slovo.
- Úplny kód - binárny rozdeliteľný kód je úplny práve vtedy, ak ku každému binármu reťazcu β existuje kódové slovo v_i. Pričom buď β je prefixom v_i, alebo v_i je prefixom β. Celkom triviálne musí v Kraft-McMillanovi byť rovnosť, aby bol kód úplny (a práve vtedy).
- Binárne kódové stromy - zakorenený strom a jeho hrany sú ohodnotené číslami 0 alebo 1. Strom dobre reprezentuje prefixové kódy. Doplneným hrán (a listov) do stromu vieme z každého prefixového kódu vyrobiť úplny kód.
- Optimálny kód - cena optimálneho kódu sa označuje $L(P)$. Veta o optimálnom kóde.

$$\sum_{i=0}^{m-1} p_i \cdot lg\frac{1}{p_i} \leq L(P) \leq \sum_{i=0}^{m-1} p_i \cdot lg\frac{1}{p_i} + 1. \tag{3}$$

  Rovnosť nastáva práve keď $p_i = 2^{-l_i}$.
- Kvázioptimálne kódy - Shannonov kód, Fanov kód (nedá sa o nich dokázať, že sú optimálne, ale sú "dosť dobré").

Shannonov kód (pre pravdepodobnostný zdroj) Usporiadame si pravdepodbnosti od najväčšej po najmenšiu: $p_0 \geq p_1 \geq \cdots \geq p_{m-1}$.

- Dĺžka kódového slova: $l_k = \lceil lg1/p_k \rceil$.
- Prefixové pravdepodobnosti: $q_k = \sum_{i=0}^{k-1} p_j$.

Kódové slová pozostávajú z prvých l_i bitov v desatinnom rozvoji čísla q_i. Pomerne jednoducho sa dá ukázať, že Shannonov kód je prefixový. Hodnota l_k je jednoznačne určená pozíciou prvej jednotky v binárnom zápise p_k. Môže sa stať, že sa po vytvorení Shannonovho kódu dá tento kód skrátiť (ak nebude žiadne kódové slovo prefixom iného). Takže to nie je úplne optimálna konštrukcia.

Fanov kód (pre pravdepodobnostný zdroj) Rozdeľujeme stĺpec pravdepodobností na dve časti tak, aby sa súčty pravdepodobností líšili čo najmenej. Následne vytvárame binárne ohodnotený strom podľa deliacich čiar.

Huffmanov optimálny kód Je založený na greedy vlastnosti: po zlúčení dvoch najnepravdepodobnejších znakov do "jedného" riešime problém pre menší prípad.

Algoritmus: Zoradíme pravdepodobnosti a vyberieme dve najmenšie. Sčítame a vrazíme naspäť do prioritnej fronty. Týmto postupom nám vznikne binárne ohodnotený strom (synmi sčítanej pravdepodobnosti budú sčítance). Huffmanov kód nie je jednoznačný (pri rovnosti pravdepodobností sa musíme rozhodovať - v oboch prípadoch ale dostaneme kód s rovnakou cenou). Huffmanov kód je celkom odolný voči malým odchýlkam v predpokladaných pravdepodobnostiach. (SKIP vyjadrenie chyby v závislosti od predikovanej pravdepodobnosti). Poznámka: v skriptách sa implicitne pri rovnosti dvoch pravdepodobností dáva nová navrh - tým sa výhyba dlhým slovám (a teda kód je odolnejší voči odchylkám v predpokladaných pravdepodobnostiach).

Rozšírenia kódu V niektorých prípadoch sme obmedzený dĺžkou kódovej abecedy (nemôžeme kódovať kratšie ako jedným znakom napríklad). Napr. ak rozdelenie pravdepodobnosti zdroja je $\{0.9, 0.1\}$ tak minimálna cena kódu je 1, ale entropia je 0.4689955936. Tento prípad riešime tak, že kódujeme dlhšie slová. S veľkosťou rozšírenia (z jedného na n znakov) konverguje cena Huffmanovho kódu k entropií zdroja.

Kódovanie Markovovského zdroja Markovovský zdroj pridáva pravdepodobnosti medzi nasledujúcimi znakmi. Tieto pravdepodobnostné vzťahy vieme kódovať v matici kde stĺpec - ktorý znak bol, riadok - ktorý znak bude nasledovať (riadky a stĺpce sa v literatúre/ na prednáškach často zamieňajú). Niektoré pojmy a vlastnosti:

- Homogénny - pravdepodobnosť nezávisí od n a teda sa dá reprezentovať v matici. (Zase to mýli v skriptách.)
- Ergodický - umocňovanie matice speje k limitnej matici, ktorá ak má kladný riadok, tak existuje a je práve jedna nezávisle na počiatočných podmienkach.
- Stacionárne rozdelenie pravdepodobnosti je limita umocnenej matice. Dá sa vyrátať aj pomocou sústavy lineárnych rovníc (ale len keď je zdroj ergodický).

Na konštrukciu optimálneho kódu *ne*používame limitné rozdelenie, ale pre každý riadok skonštruujeme zvlášť Huffmanov kód - pri dekódovaní používame inštanciu Huffmana pre znak, ktorý je práve na rade. Využitím vzťahov dostávame výrazne lepší (cenovo) kód ako kódovaním limitného rozdelenia.

Kódovanie pomocou orákula . Pomocou orákula generujeme binárne postupnosti a snažíme sa uhádnúť správu. Posielame potom XOR skutočnej a orákulom generovanej správy. Ak tipujeme dobre, tak sa v tejto správe bude vyskytovať pomerne veľa 0. Tieto nuly pretransformujeme na počty za sebou idúcich 0 (aj nula za sebou idúcich núl). Vyjadríme strednú hodnotu dĺžky kódového slova v závislosti od p (=pravdepodobnosť uhádnutia). Je analyticky ťažké nájsť minimum výsledného výrazu a teda ho pre určité p binárne vyhľadávame.

Chyby v pravdepodobnostiach výskytu zdrojových symbolov (SKIP)

3 Metódy kompresie údajov (SKIP)

4 Kolmogorovská zložitosť a hranice kompresie údajov (SKIP)

5 Samoopravné kódy

Snažíme sa o vhodný kompromis medzi entropiou (resp. mohutnosťou kódu) a samoopravnou schopnosťou. Všetkých možných kódov je sup. exp. veľa a preto sa snažíme nájsť "optimálne" algoritmy pre ich vytváranie.

Používané pojmy

- q - počet znakov kódovej abecedy.
- Informačné symboly - pomocou nich zapisujeme informáciu, ktorú chceme preniesť.
- Kontrolné symboly - zaznamenávajúce štruktúru kódového slova.
- Absolútna redundancia kódu - počet kontrolných symbolov.
- Relatívna redundancia kódu - podiel počtu kontrolných a informačných symbolov. Vo všeobecnosti: $R(v) = log_q(v)/n$
- –Tieto pojmy nevieme definovať pre všeobecný kód (lebo nevieme striktne rozlíšiť ktoré sú ktoré symboly).
- Hranica sférického uloženia kódu (sphere packing bound) - pre každý prvok z poľa je vo vzdialenosti t maximálne jedno kódové slovo.
- Hranicou pokrytia kódu V - maximálna vzdialenosť prvkov priestoru od kódových slov.
- Dokonalý kód - hranica sférického uloženia kódu = hranica pokrytia kódu. Je ich málo. Hammingov pre opravenie jednej chyby a Golayov (23,12) kód opravujúci 3 chyby. Nevieme ich vo všeobecnosti hľadať.
- q-nárny symetrický prenosový kanál bez pamäte ($q = 2$ - binárny). Každý odvysielaný znak má rovnakú pravdepodobnosť, že bude počas prenosu cez kanál zamenený za iný znak kódovej abecedy. Vo všeobecnosti predpokladáme, že bude vznikať málo chýb počas prenosu. Nikdy nevieme s určitosťou opravovať chyby vzniknuté počas prenosu, preto vždy volíme najpravdepodobnejšiu možnosť (pomocou podmienenej pravdepodobnosti). Resp. ak tušíme, že veľa riskujeme, tak nerobíme nič.
- Testovanie parity. Paritný bit.
- Obdĺžnikové kódy opravujúce jednu chybu. Detekujú aj väčší počet chýb.

Hammingov kód . Ide o zovšeobecnenie obdĺžnikového prístupu na viacero rozmerov. Na špeciálnych pozíciách si ukladáme kontrolné súčty - špeciálne pozície pomáhajú pri implementácií dekódovania. Syndróm chyby - vektor kontrolných súm (nie kontrolných symbolov). Ak nenulový, tak v binárnom zápise udáva pozíciu na ktorej vznikla chyba. Hamingové kódy $(2^m - 1, 2^m - 1 - m)$ opravujúce jednu chybu sú dokonalé. Pre názorné príklady nám stačí Hammingov kód $(15, 11)$. Kontrolné symboly má na pozíciách $1, 2, 4, \cdots, 2^{m-1}$ a ostatné symboly sú informačné.

6 Lineárne kódy

Aby sme mohli jednoznačne opravovať chyby váhy t potrebujeme, aby vzdialenosť medzi každými dvoma kódovými slovami bol aspoň $2t$.

Definície hlavných pojmov

- Grupové kódy - kódové slová tvoria podgrupu jazyka q^n.
- Lineárny kód - je podpriestorom vektorového priestoru $GF(q)^n$. Veta: minimálna vzdialenosť kódu je minimálna váha vektoru (okrem nulového). Pomocou matice bázy kódového priestoru a informačného vektoru vieme ľahko generovať správy (násobením inf. vektora a matice). Rovnako, overovať vieme maticou ortogonálneho doplnku – keďže všetky kódové vektory sú naň kolmé (a žiadne iné). Veta: vzťah medzi lineárnou závislosťou stĺpcov kontrolnej matice a najmenšou váhou nenulového kódového slova.
- Duálny kód - kód, ktorý je ortogonálnym doplnkom pôvodného kódu. (Ortogonálny doplnok nie je komplement).
- Kontrolná matica H - $uH^T = 0$ práve vtedy, keď u je kódovým vektorom. Generuje ortogonálny priestor ku priestoru kódových slov. Veta: lineárny kód C nad poľom $GF(q)$ obsahuje nenulové kódové slovo váhy menšej alebo rovnej w práve vtedy, ak jeho kontrolná matica H obsahuje w lineárne závislých stĺpcov. Dôsledok: lineárny kód C s kontrolnou maticou H má minimálnu vzdialenosť w práve vtedy, ak je v matici H ľubovolných $w - 1$ stĺpcov lineárne nezávislých a v H existuje w lineárne závislých stĺpcov.
- Singeltonova hranica. Pri minimálnu vzdialenosť lineárneho (n, k) kódu platí $d^* \leq 1 + n - k$. Kódy, pri ktorých nastáva rovnosť nazývame *lineárne kódy s maximálnou vzdialenosťou*.
- Ekvivalentné kódy - ak je priestor kódových slov rovnaký pre oba kódy. Tento názov sa zaviedol pre to, že bázu (generujúce vektory) kódových slov môžeme štandardne upravovať a dostávame stále rovnaký kód.
- Systematický kód - generuje ho matica v systematickom tvare $G = [I_k|P]$. To znamená, že najprv sú v kódovom slove informačné symboly a potom kontrolné. Odteraz môžeme bez ujmy na všeobecnosti predpokladať, ak sa nám to hodí, že informačné symboly sú na začiatku. (Jacobus van Lint definuje systematicky kód slabšie.)
- Dekódovanie pomocou tabuľky dekódovania - faktorizuje sa scítačia grupa vektorového priestoru podľa priestoru kódových slov. Najmenší reprezentanti sú chybové vektory (okrem nulového). Prijaté slovo sa nájde v tabuľke dekódovania, v ktorej sú prvky $v_j + e_i$ a podľa toho sa určí chyba e_i. Parafráza jednej vety: ak používame maticu

dekódovania, tak dekódujeme správne práve vtedy, ak chyba $w - u$ je reprezentantom triedy rozkladu. Ak rozdiel dekódovaného a vstupného vektora nie je reprezentantom, tak sme dekódovali zle. Kód je dokonalý a opravuje t chýb ak všetky vektory váhy t a menšej sú jeho reprezentantmi chýb. Uchovávať ale takú veľkú tabuľku má zmysel len pre matematikou. Informatici používajú takzvané *syndrómy chyby* (ktoré tiež v istom zmysle reprezentujú chybovú triedu, lebo závisia iba od reprezentantov – viac o syndrómoch nižšie).

- Dekódovanie pomocou syndrómu chyby - syndróm chyby je výsledok prenásobenia získaného vektora kontrolnou maticou. Vieme, že syndróm záleží len od chybového vektora. A teda konštrukciou tabuľky chybový [vektor — syndróm] vieme podľa syndrómu nájsť chybový vektor a jeho odčítaním od vstupu získame kódový vektor.

- Dokonalý lineárny kód - $\sum_{j=0}^{t} \frac{n}{j}(q-1)^j = q^{n-k}$. Jediné známe dokonalé kódy sú Hammingove kódy a Golayov $(23, 12)$ kód opravujúci 3 chyby. To, že dokonalé kódy neexistujú sa dá dokázať pre celkom veľkú podtriedu lineárnych kódov.

- Kvázidokonalý lineárny kód - má všetkých reprezentantov chýb váhy t a menšej, ale môže mať aj reprezentantov váhy $t + 1$. Takže kvázioptimálne kódy vedia dekódovať aj niektoré slová poznačené chybami váhy $t + 1$.

- Reed-Mullerové kódy $R(r, m)$ - $n = 2^m$; $k = \sum_{0 \leq j \leq r} \frac{m}{j}$; $n - k$; $d = 2^{m-r}$. Generujúca matica kódu sa zostrojí postupne. Riadky sú "súčiny" (and) niekoľkých základných vektorov. Hneď z konštrukcie vieme rýchlo vypočítať minimálnu vzdialenosť kódových slov (keďže váhy všetkých bázových vektorovo delia d). Používajú sa hlavne kvôli "jednoduchému" dekódovaniu, ktoré nevyžaduje výpočet syndrómu chyby. Stačí vedieť, že existujú.

- Nechápem dekódovaciemu algoritmu dosť dobre. Pomocou nejakých rovníc pre informačný vektor ktoré majú argument prijaté slovo.... dokonca ani príklad mi nedáva zmysel. (SKIP).

7 Cyklické kódy

Úvod Na kódových slovách si navyše zavedieme operáciu cyklického shift-u. Zvolíme $GF(q)[x]/(x^n - 1)$ ako vhodnú štruktúru na reprezentáciu týchto operácií. Ku každému vektoru priradíme polynóm s koeficientami rovnými prvkom vektora. Cyklický kód je teda špeciálnym prípadom lineárneho kódu.

Cyklický kód sa potom ekvivalentne definuje tak, že je to faktorový okruh polynómov. Cyklický kód je aditávna podgrupa, násobok polynómu z okruhu a polynóm z podgrupy je tiež v podgrupe. Takže je to vlastne ideál. Neskôr sa ukáže, že je to dokonca hlavný ideál (to som už v skriptách nenašiel).

Hľadáme teraz generujúci polynóm tejto podgrupy. Našim kandidátom je normovaný polynóm minimálneho stupňa $g(x)$, ktorý je jednoznačne daný (konečný, ak by boli dva, vieme odčítať). A kvôli tomu, že vieme deliť so zvyškom modulo $g(x)$, ukážeme, že každý polynóm v C je násobkom $g(x)$. Veta s dôkazom: $g(x)|x^n - 1$. Chceme, aby stupeň $g(x)$ bol $n - k$ - to zatiaľ neviem zaručiť, ale časom budeme vedieť vypočítať k na základe t - počtu chýb ktoré kód opravuje.

Hlavné pojmy

- Kontrolný polynóm cyklického kódu C - keďže $g(x)|x^n - 1$, tak existuje $h(x)$ taký, že $g(x)h(x) = x^n - 1$ a $h(x)$ nazývame kontrolným polynómom.
- Ak uvažujeme ireducibilné faktory $x^n - 1$ nad poľom $GF(q)$, tak každý zo súčinov niekoľkých týchto faktorov je generujúcim polynómom. Ak je stupňa $n - k$, tak potom veľkosť kódu je 2^k (keď uvážime všetky polynómy, ktorými môžeme generujúci vynásobiť).
- Maticový popis cyklických kódov. Bázu priestoru polynómu tvoria $g(x), xg(x), \cdots, x^{k-1}g(x)$ (z lin. kódov vieme, že báza nie je väčšia, a vo vete ukážeme že sú LN). Pomocou tejto bázy sme schopný na základe koeficientov generujúceho polynómu vyrobiť generujúcu maticu priestoru (cyklický kód je špeciálny lineárny). Potom sa ukáže, že podobná matica koeficientov $h(x)$ je kontrolnou maticou. Rozmery matíc sú podľa mňa $n \times k$, lebo k kódových znakov na n znakov v správe a naopak. V skriptách sa to dosť mýli. Ešte ukážeme, že $x^k h(x^{-1})$ je generujúcim polynómom duálneho kódu $C^\perp$.
- Kódovanie pomocou cyklických kódov. Kódový polynóm vytvoríme buď ako $u(x) = i(x)g(x)$ alebo systematicky $x^{n-k}i(x) - (x^{n-k}i(x) mod g(x))$.

Dekódovanie cyklických kódov

- generujúci polynóm $g(x)$ - vzniká ako súčin niektorých ireducibilných
- kontrolný polynóm $h(x) - g(x)h(x) = x^n - 1$
- informačný polynóm $i(x)$ - užívateľ vytvára
- kódový polynóm $u(x)$ - napr. nesystematicky u(x)=i(x)g(x)
- chybový polynóm $e(x)$ - šum
- prijatý polynóm $v(x) = u(x) + e(x)$
- syndrómový polynóm $s(x)$ - napr. nesystematicky $s(x) = v(x) mod g(x)$

Veta o tom, že chyby do váhy $< d/2 (d = n - k)$ vieme jednoznačne určiť zo syndrómu. Potom si pre každý bázový polynóm aditívnej grupy vieme vypočítať syndrómový polynóm a tým vytvoríme podobnú tabuľku ako pri lineárnych kódoch. A konečne využijeme silnejšiu štruktúru na jednoduchšie dekódovanie (lebo aj v tomto prípade potrebujeme potenciálne rozsiahlu tabuľku syndrómov).

Meggittov algoritmus dekódovania cyklických kódov . Použiteľný pre všetky cyklické kódy, teda aj BCH (ktoré len špeciálne vytvárajú generátor $g(x)$). Pri najoptimálnejšej verzií máme ešte tri zlepšenia:

1. $v(x)$ sa cyklicky posúva a hľadá sa chyba len pri najvyššom bite. Dokázali sme, že to takto môžeme robiť - chybový polynóm sa posúva spolu s $v(x)$. Tým sa zmenší tabuľka syndrómov o jeden rád, čo pri kódoch opravujúcich viacero chýb je veľké zlepšenie (lebo nemusíme mať syndróm pre každý chybový polynóm).
2. Po posune nemusíme modulovať celý $xv(x)$, ale stačí $xs(x) mod g(x)$. Hovorí o tom ďalšia veta.

3. Keď sa opraví najvyšší byt, je nutné prepočítať celý syndróm. Zase, aby sme sa vyhli modulovaniu celého $v(x)$, priradíme $s(x) := s(x) + \sigma(x)$ kde $\sigma(x) = x^{n-1} modg(x)$.

Celý Meggittov algoritmus (1960) je možné efektívne realizovať pomocou LFSR (linear feedback shift register) a teda umožňuje rýchlu HW implementáciu. Keďže chyby opravujeme po jednom, stačí si pamätať všetky syndrómy (collection:set), namiesto toho, aby sme si pamätali aj k nim prislúchajúce chybové polynómy (collection:map). Nevýhodou je len to, že chyby o jedna vyššieho rádu ako boli cielené, Meggitov dekóder nevie dekódovať a lineárny áno. Toto nám nevadí, keďže zložitosť lin. dekódera by bola v tomto prípade príliš veľká. Postup v prehľadnej tabuľke:

1. (Init) Vytvor dekódovaciu tabuľku T obsahujúcu všetky syndrómy, zodpovedajúce predstaviteľom tried rozkladu faktorového okruhu $GF(2)[x]/x^n - 1$, podľa cyklického kódu C; chybovým polynómom stupňa $n-1$. (Predstavitelia tried rozkladu sú chybové polynómy, v ktorých je koeficient pri najvyššej mocnine x nenulový.) $\sigma(x) \leftarrow x^{n-1} modg(x)$, $s(x) \Leftarrow v(x) modg(x)$, PocetPosunov $\Leftarrow 0$.
2. (Existencia chyby) Ak $s(x) = 0$ pokračuj krokom 6, ináč pokračuj krokom 3.
3. (Zistenie chyby) Zisti, či sa $s(x)$ nachádza v tabuľke T. Ak áno, pokračuj krokom 4, ak nie, pokračuj krokom 5.
4. (Opravenie chyby) $v(x) \Leftarrow (v(x) + x^{n-1}), s(x) \Leftarrow s(x) + \sigma(x)$, pokračuj krokom 5.
5. (Posun v hľadaní) $v(x) \Leftarrow xv(x) modx^n$, $s(x) \Leftarrow xs(x) modg(x)$, ++PocetPosunov.
6. (Finalize, posun späť) $v(x) \Leftarrow x^{n-PocetPosunov} v(x) modx^n - 1$.

Error trapping Oveľa efektívnejšia metóda. Táto metóda sa snaží nájsť všetky chyby naraz hľadaním (cyklické posúvanie) $v(x)$ a jeho modulovaním. Funguje len keď je rozpätie chýb dosť malé - to ale zvykne byť kvôli cyklicite kódu. Je založená na pozorovaní, že keď má syndróm váhu menšiu rovnú t (počtu opravovaných znakov), tak je identický s chybovým vektorom (až na veľkosť). Pri rátaní syndrómu sa dá použiť rovnaké zlepšenie ako pri Meggittovi. Odporúčam na písomku. Jediná nevýhoda je, že nemusí výjsť.

- *Lema 1*: Ak je váha syndrómu nanajvýš t, tak je syndróm rovný chybovému polynómu.
- *Lema 2*: Ak sú chyby v rozpätí $n-k$, tak existuje cyklický posun taký, že jeho syndróm má váhu nanajvýš t. Napr. pri kódoch (15,7) aj (15,5) sme schopný zakaždým použiť error trapping.

8 Boseove-Chandhuryove-Hoquenghemove kódy

Úvod Mierne zovšeobecníme Hammingove kódy. Najprv "mergeneme" stĺpce kontrolnej matice a vzniknuté vektory transformujeme na prislúchajúce polynómy. Tieto polynómy sú nad vhodným faktorovým okruhom, ktorý je izomorfný s nejakým konečným poľom (všetky konečné polia rovnakej veľkosti sú navzájom izomorfné). Okrem iného ich pochopenie je vhodné pre pochopenie zložitejších kódov.

Konštrukcia BCH Určíme generátor a tým určíme aj celý kód.

$$g(x) = lcm\{m_{beta^{l+1}}(x), m_{beta^{l+2}}(x), \cdots m_{beta^{l+2t}}(x)\} \tag{4}$$

kde $m_{beta^{l+1}}(x)$ je minimálny polnynóm prvku β^j. Práve tieto generátory zodpovedajú BCH kódom. Pre ozrejmenie viď nasledujúce odrážky.

Hlavné pojmy

- Kontrolná matica pozostáva z mocnín primitívneho prvku (generátor cyklickej multiplikatívnej grupy). Môžeme ju rozšíriť o ďalšie primitívne prvky - aby sme vedeli odhalovať chyby väčšej váhy.
- Syndrómy sú v tomto prípade dosadenia prvkov $GF(2^{n-k})$ do kontrolných polynómov. Podľa toho, ktorej mocnine primitívneho prvku prislúchajú, sa určí pozícia chyby. Ak je chyba pri x^i, tak syndróm bude α^i a potom sa pomocou logaritmu (implementovaný napr. lookup tabuľkou) v konečnej cyklickej grupe zistí i. Pri kódoch opravujúcich viacero chýb je nutné riešiť nelineárnu sústavu rovníc nad konečným poľom, ktorú vieme riešiť pomocou polynómov.
- Primitívny prvok - α
- Lokátory chýb - $X_k = \alpha^{i_k}$ keď chyba pri x^{i_k}.
- Polynóm lokátorov chýb - polynóm z $GF(2)[x]$ ktorý má za korene lokátory chýb. Pomocou Vietových vzťahov vieme vyrátať jeho koeficienty pri jednotlivcýh mocninách. Napr. pre dve chyby je to $\Lambda(x) = (x - X_1)(x - X_2) = x^2 + (X_1 + X_2)x + X_1X_2$.
- Konštrukčná vzdialenosť BCH kódu - garantovaná minimálna vzdialenosť vyplývajúca z konštrukcie. Skutočná vzdialenosť (= minimálna vzdialenosť kódu d^*) môže byť aj väčšia, keďže poznáme $2t$ nie nutne lineárnych koreňov $g(x)$. Napríklad pre BCH kód v úzkom zmysle pre (31,23) zistíme, že v skutočnosti opravuje až 5 chýb. Primitívny BCH kód - jeho dĺžka je rovná $n = q^m - 1$. BCH kód v úzkom zmysle - $l = 0$.

PGZ algoritmus dekódovania PGZ (Petersonov-Gorensteinov-Zierlerov).

9 Citáty

- Kódu priradíme generujúcu funkciu (enumerátor dĺžok kódových slov).
- Faktorový okruh polynómov $GF(2)[x]/x^4 + x + 1$ predstavuje hľadané pole.

10 Poznatky z algebry

- Ak má polynóm v poli charakteristiky 2 koreň β, potom jeho koreňom je aj β^2.