

REED-SOLOMONOVE KODE

Aleksandar Jurišić

Arjana Žitnik

11. dec. 2003

V sestavku obravnavamo Reed-Solomonove kode. Opišemo originalen pristop Reeda in Solomona ter pristop z linearimi cikličnimi kodami. Povezava med njima je izpeljana s končno Fourierovo transformacijo. Predstavimo še različne načine za njihovo odkodiranje in ključno enačbo.

REED-SALOMON CODES

Reed-Solomon codes are described via two different approaches: the original approach of Reed and Solomon using interpolation polynomials and the classical approach with linear cyclic codes. The connection between these two approaches is established through finite Fourier transform. We also present different decoding procedures and the key equation.

VSEBINA

1. Mitologija – Uvod	1
2. Pluralizem – Polinomi	2
3. Poljedelstvo – Računanje v končnih obsegih	5
4. Idealizem – RS kode kot linearne ciklične kode	6
5. Centralizem – Ključna enačba	7
6. Futurizem – Zaključek	11
Literatura	12

1 Uvod

Namen transformiranja sporočila pred pošiljanjem (ali hranjenjem) informacije je lahko različen:

- dodajanje kontrolnih bitov, ki nam omogočajo odkrivanje in odpravljanje morebitnih napak po prejetju ali ponovnem branju,
- zakritje teksta pred nepooblaščeno osebo,
- kompresija teksta, da se izognemo nepotrebni informaciji.

Teorija kodiranja se ukvarja s prvo možnostjo, z drugo **kriptografija**, medtem ko ostaja zadnja običajno v domeni inženirjev. Prenosni mediji, na primer telefonske linije, omrežja in satelitske povezave, ter mediji za shranjevanje, na primer optični in magnetni disk, trakovi, običajno še zdaleč niso popolni. Če želimo zanesljive prenose in hranjenje v podatkovnih bazah, potem so kode za odpravljanje napak nepogrešljive. Eden izmed zgodnjih uspehov teorije kodiranja je bila kvaliteta slik, ki so jih poslali sateliti. Če ne bi uporabljali kod za odpravljanje napak, pošiljke ne bi vsebovale slik, na Zemljo bi prišel le naključen šum!

Koda je podmnožica nekega prostora, v katerem je definirana razdalja. Elementom kode bomo rekli tudi **kodne besede**. **Razdalja kode** je najmanjsa razdalja med različnimi kodnimi besedami. Običajno razbijemo dano sporočilo na bloke fiksne dolžine (n), ki jih nato povežemo s kodnimi besedami z neko bijektivno korespondenco. V tem primeru rečemo, da gre za **bločne kode dolžine n** . Najpogosteje si za prostor izberemo množico vseh n -teric s simboli iz neke končne množice F , imenovane tudi **abeceda**:

$$F^n = \{(a_0, a_1, \dots, a_{n-1}) \mid a_i \in F, i = 0, 1, \dots, n-1\}.$$

V tem primeru je **razdalja** med dvema n -tericama število mest, na katerih se razlikujeta. Pravimo ji tudi **Hammingova razdalja**, po enem izmed pionirjev teorije kodiranja. Lahko pa bi si za prostor izbrali tudi graf, kjer je razdalja med dvema vozliščema dolžina najkrajše poti med njima (dolžina poti je seveda enaka številu njenih povezav). Za uvod v teorijo kodiranja glej Jurišić [8], Klavžar [11] ter Vanstone in Van Oorschot [20], za referenčno knjigo pa priporočamo Pless et al. [5]. Pri kodi nas najbolj zanima, koliko napak lahko odpravimo glede na to koliko kontrolnih bitov smo dodali osnovni informaciji. V tej smeri nam pomaga naslednji rezultat.

Lema 1.1 (Singletonova meja [19]) *Naj bo C bločna koda dolžine n nad abecedo s q elementi in d njena razdalja. Potem velja*

$$|C| \leq q^{n-d+1}.$$

DOKAZ. Naj bo C' koda, ki jo konstruiramo iz kode C tako, da izbrišemo skupino katerihkoli $d-1$ koordinat v vseh kodnih besedah. Ker je razdalja kode C enaka d , velja $|C| = |C'|$. Dolžina kode C' pa je $n-d+1$, zato ima največ q^{n-d+1} kodnih besed, kar smo žeeli pokazati. ■

Če so sporočila vse možne k -terice nad abecedo s q elementi ter obstaja bijekcija med sporočili ter kodnimi besedami, je $|C| = q^k$ in pravimo, da gre za **(n, k)-kodo**. V tem primeru se Singletonova meja prevede v zgornjo mejo za razdaljo kode:

$$d \leq n - k + 1. \tag{1}$$

Za odkodirni princip si običajno izberemo metodo, ki prejeti besedi poišče *najbližjo* kodno besedo. V tem primeru ima koda, ki odpravi t napak, razdaljo $d \geq 2t + 1$, saj morajo biti krogle s središčem v kodnih besedah in radijem t disjunktne, in iz neenakosti (1) sledi, da ima koda vsaj $2t$ kontrolnih bitov, tj.

$$t \leq \left\lfloor \frac{n-k}{2} \right\rfloor. \tag{2}$$

Torej lahko (n, k) -koda odpravi kvečjemu $\lfloor (n - k)/2 \rfloor$ napak.

Ta članek predstavlja kratek uvod v **Reed-Solomonove kode** (na kratko RS-kode), ki dosežejo Singletonovo mejo. V naslednjem razdelku je opisan originalen pristop Reeda in Solomona in se prepričamo, da gre za linearne kode. Po krajšem razdelku o končnih obsegih pokažemo s končno Fourierovo transformacijo, da so RS-kode ekvivalentne posebnemu razredu linearnih cikličnih kod. V 5. razdelku predstavimo še različne načine za odkodiranje, ki uporabijo ključno enačbo in so polinomske časovne zahtevnosti. V zaključku pa omenimo številne rabe RS-kod in predstavimo odprt problem o perkeftnih kodah.

2 Polinomi

V tem razdelku bomo predstavili kode, ki sta jih v 60. letih prejšnjega stoletja vpeljala Irving Reed in Gustave Solomon [18]. Takrat sta bila zaposlena v enem izmed Lincolnovih laboratoriјev na slovitem MIT. Kar nekaj časa je bilo potrebno, da je tehnologija ujela korak s teorijo in omogočila učinkovite implementacije teh kod (v 60. letih prejšnjega stoletja ni bilo hitre digitalne elektronike, vsaj ne glede na današnje standarde). Reed in Solomon sta uporabila za svoje kode več kot 100 let staro teorijo končnih obsegov francoskega matematika Evariste Galoisa. Reed-Solomonov članek je predlagal zelo eleganten način procesiranja podatkov, vendar pa se nihče ni zavedal pravega pomena, npr. ali je tak način res praktičen (in tedaj verjetno sploh ni bil praktičen).



Slika 2.1: Irving Reed in Gustave Solomon v laboratoriju Jet Propulsion opazujeta srečanje Voyagerja 2 z Neptunom leta 1989.

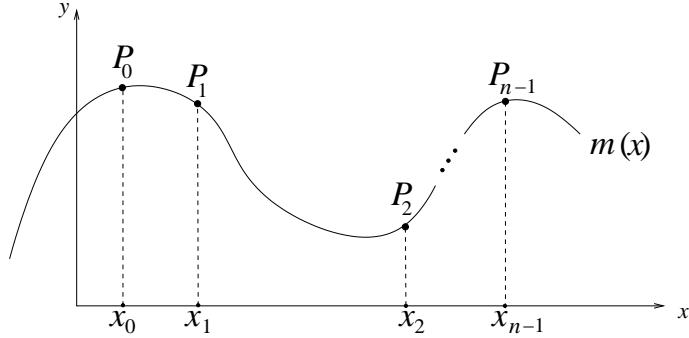
Vpeljimo nekaj oznak, ki jih bomo uporabljali skozi celoten razdelek. Za abecedo si izberimo elemente končnega obsega s q elementi, kjer je q potenca nekega praštevila, oznaka $\mathbb{F} = \text{GF}(q)$ (angl. Galois Field). Če je q praštevilo, je to kar praobseg \mathbb{Z}_q . Potem je \mathbb{F}^n z običajnim seštevanjem in množenjem po komponentah vektorski prostor nad \mathbb{F} . Čeprav ne bi bilo nujno, bomo obravnavo poenostavili in v nadaljevanju privzeli, da je dolžina kodnih besed enaka kar $n = q - 1$. Dobro je znano, da je multiplikativna grupa končnega obsega \mathbb{F} ciklična. To pomeni, da obstaja v \mathbb{F} **primitiven** element α , tj. tak element $\alpha \in \mathbb{F}$, da je $\alpha^n = 1$ in $\alpha^i \neq 1$ za vsak $i \in \{1, \dots, n - 1\}$.

Sedaj pa si oglejmo, kako sta Reed in Solomon vpeljala RS(n, k)-kode. Za **sporočilo** $m = (m_0, m_1, \dots, m_{k-1}) \in \mathbb{F}^k$ s prirejenim polinomom $m(x) = m_0 + m_1 x + \dots + m_{k-1} x^{k-1}$ izračunamo vrednosti $c_i = m(\alpha^i)$, $i \in \{0, \dots, n - 1\}$ in iz njih sestavimo **kodno besedo**:

$$c = (c_0, c_1, \dots, c_{n-1}).$$

Da bo odkodiranje možno, mora seveda veljati $k < n$. V tem primeru nas slika 2.2 oziroma dobro znana formula za polinomsko interpolacijo (glej npr. [6]) prepriča, da ni preveč pričakovati

obstoj odkodirnega algoritema za RS-kode, ki bi opazil morebitne nepravilnosti in jih odpravil. Bistveno vprašanje pa je, ali je tak algoritem učinkovit.



Slika 2.2: V bistvu predstavlja RS kodiranje risanje grafa polinoma sporočila s tiskanjem vseh njegovih točk. Če se pri prenosu oziroma hranjenju kodne besede premakne ali celo izbriše manjše število točk, ki ustrezajo koordinatam kodnih besed, bi lahko, v primeru, če bi računanje potekalo v obsegu realnih števil, tako rekoč s prostim očesom takoj ugotovili, kje so nastale napake oziroma izbrisali informacije in kako jih odpraviti.

Prvi postopek za odkodiranje sta predlagala Reed in Solomon. Le-ta temelji na reševanju velikega števila sistemov enačb. Ko sprejmemo kodno besedo $c = (c_0, c_1, \dots, c_{n-1})$, lahko sporočilo $m = (m_0, m_1, \dots, m_{k-1})$ izračunamo iz naslednjega (predoločenega) sistema enačb

$$\begin{aligned} c_0 &= m_0 + m_1 &+ m_2 &+ \cdots + m_{k-1} \\ c_1 &= m_0 + m_1\alpha &+ m_2\alpha^2 &+ \cdots + m_{k-1}\alpha^{k-1} \\ c_2 &= m_0 + m_1\alpha^2 &+ m_2\alpha^4 &+ \cdots + m_{k-1}\alpha^{2(k-1)} \\ &\vdots \\ c_{n-1} &= m_0 + m_1\alpha^{n-1} &+ m_2\alpha^{(n-1)\cdot 2} &+ \cdots + m_{k-1}\alpha^{(n-1)(k-1)}. \end{aligned} \quad (3)$$

Poglejmo množico poljubnih k enačb, ki ustrezajo k -elementni podmnožici

$$\{a_1, a_2, \dots, a_k\} \subseteq \{1, \alpha, \dots, \alpha^{n-1}\}.$$

Njihovi koeficienti tvorijo Vandermondovo matriko z determinanto

$$\left| \begin{array}{ccccc} 1 & a_1 & a_1^2 & \dots & a_1^{k-1} \\ 1 & a_2 & a_2^2 & \dots & a_2^{k-1} \\ \vdots & & \vdots & & \vdots \\ 1 & a_k & a_k^2 & \dots & a_k^{k-1} \end{array} \right| = \prod_{1 \leq i < j \leq k} (a_j - a_i).$$

Le-ta je v obsegu \mathbb{F} različna od 0, saj je $a_i \neq a_j$ za vse $i, j \in \{1, \dots, k\}$, za katere velja $i \neq j$. Zato ima sistem enolično rešitev v \mathbb{F} .

Če se pri prenosu ne bi pojavila napaka, bi lahko z izbiro poljubne k -elementne podmnožice obrnljivih elementov v \mathbb{F} dobili sistem enačb, iz katerega bi lahko določili celotno sporočilo (m_0, \dots, m_{k-1}) . Tako k -elementno podmnožico lahko izberemo na $\binom{n}{k}$ načinov. Če pa pri prenosu nastanejo napake, nam lahko različni sistemi enačb dajo različne rešitve. Naslednja lema nam zagotavlja, da se prava rešitev pojavi največkrat, če le število napak ni preveliko.

Lema 2.1 Če pride pri prenosu ali branju kodne besede (c_0, \dots, c_{n-1}) RS(n, k)-kode do s napak, se pri reševanju podsistema k -tih enačb iz (3) pojavi napačna rešitev (k -terica) največ

$$\binom{s+k-1}{k}-krat.$$

DOKAZ. Enačbe sistema (3) ustrezajo k -razsežnim hiperravninam. Zaradi linearne neodvisnosti poljubnih k vektorjev, ki določajo te hiperravnine, se poljubnih k hiperravnin seka v eni točki. V napačni točki pa se lahko seka največ $s + k - 1$ hiperravnin, saj je med njimi lahko največ $k - 1$ takih, ki se pri prenosu niso spremenile (k nespremenjenih enačb nam namreč že da pravo rešitev) in največ s takih, ki so se spremenile. Napačno točko (tj. k -terico) lahko dobimo torej na največ toliko načinov, kot smo želeli dokazati. ■

Linearna (n, k) -koda je vsak k -razsežen linearen podprostor vektorskega prostora \mathbb{F}^n , $0 \leq k \leq n$. Pri zgornji obravnavi smo se izognili tradicionalnemu pristopu z linearimi kodami, kjer običajno izračunamo posebno $((n-k) \times n)$ -razsežno matriko, ki ji pravimo **nadzorna matrika**, glej npr. Klavžar [11]. Vseeno pa omenimo, da je razdalja d poljubne linearne (n, k) -kode vsaj r natanko tedaj, ko je poljubnih $r - 1$ stolpcev te matrike linearne neodvisnih. Od tu namreč dobimo alternativen dokaz Singletonove meje (2) (s protislovjem: če je $d - 1 > n - k$, potem je poljubnih $d - 1$ stolpcev nadzorne matrike linearne odvisnih, kar seveda ni možno po definiciji razdalje kode).

Izrek 2.2 $RS(n, k)$ -koda je linearna (n, k) -koda.

DOKAZ. Naj bosta c in c' poljubni kodni besedi RS-kode ter $m(x)$ in $m'(x)$ polinoma sporočila, katerima ustreza ti dve kodni besedi. Potem za $\lambda, \lambda' \in \mathbb{F}$ in $i \in \{0, 1, \dots, n-1\}$ velja

$$(\lambda c + \lambda' c')_i = \lambda m(\alpha^i) + \lambda' m'(\alpha^i) = p(\alpha^i),$$

kjer je $p(x) = \lambda m(x) + \lambda' m'(x)$. Od tod sledi, da je $\lambda c + \lambda' c'$ kodna beseda, ki ustreza sporočilu $\lambda m + \lambda' m'$ in je RS-koda linearne. Kodne besede $a_i := (1, \alpha^i, \alpha^{2i}, \dots, \alpha^{(n-1)i})$ s prirejenimi polinomi x^i , $i \in \{0, 1, \dots, k-1\}$ so linearne neodvisne, saj jih lahko zložimo v Vandermondovo matriko, katere determinanta je različna od nič, ker so števila $1, \alpha, \alpha^2, \dots, \alpha^{k-1}$ paroma različna.

Potrebno je le še preveriti, da je poljubna kodna beseda c , ki ustreza nekemu polinomu sporočila $m(x) = \sum_{i=0}^{k-1} m_i x^i$, linearna kombinacija le-teh:

$$c = \left(\sum_{i=0}^{k-1} m_i (\alpha^0)^i, \sum_{i=0}^{k-1} m_i (\alpha^1)^i, \dots, \sum_{i=0}^{k-1} m_i (\alpha^{n-1})^i \right) = \sum_{i=0}^{k-1} m_i ((\alpha^0)^i, (\alpha^1)^i, \dots, (\alpha^{n-1})^i) = \sum_{i=0}^{k-1} m_i a_i.$$

Torej je RS-koda res k -razsežna. ■

Sedaj pa se prepričajmo, da za $RS(n, k)$ -kode v Singletonovi oceni velja enakost, tj. za dani naravni števili n in k $RS(n, k)$ -kode odpravijo največje možno število napak.

Izrek 2.3 $RS(n, k)$ -koda odpravi $\lfloor (n - k)/2 \rfloor$ napak, njena razdalja pa je $n - k + 1$.

DOKAZ. Privzemimo, da je pri prenosu RS-kodne besede prišlo do s napak. Potem dobimo po Lemi 2.1 pri reševanju vseh možnih podsistemov k -tih enačb vsako napačno rešitev največ $\binom{s+k-1}{k}$ -krat, prav pa $\binom{n-s}{k}$ -krat. Slednje število je večje natanko tedaj, ko je $n - s > s + k - 1$ oziroma $s < (n - k + 1)/2$. Ker je s celo število, lahko RS-koda na ta način odpravi poljubnih $\lfloor (n - k)/2 \rfloor$ napak. Torej je njena razdalja vsaj $n - k + 1$. Iz izreka 2.2 sledi, da ima RS-koda q^k elementov. Zaradi Singletonove meje (1) oziroma (2) pa je razdalja enaka $n - k + 1$. ■

Seveda je ta način za odkodiranje prepočasen, saj zahteva reševanje $\binom{n}{k}$ sistemov enačb velikosti $k \times k$, kar je eksponentna časovna zahtevnost glede na k . Toda tehnologija je napredovala in številni raziskovalci so začeli delati na učinkoviti implementaciji kod. Med njimi je bil tudi Elwyn Berlekamp, profesor elektrotehnike na kalifornijski univerzi v Berkeleyu, ki je v začetku

80-ih let prejšnjega stoletja odkril učinkovit algoritem za odkodiranje RS-kod [2]. Danes ga poznamo pod imenom Berlekamp-Masseyev algoritmom in je izrednega pomena tudi v kriptografiji, glej npr. Novak [14], [15] in Praprotnik [17] za testiranje linearnosti naključnih zaporedij in za študijo LFSR-ja. V 5. razdelku bomo podali polinomski algoritmom za odkodiranje, ki uporabi t.i. ključno enačbo. Le-to uporablja tudi Berlekamp-Masseyev algoritmom, vendar pa njegova predstavitev žal presega okvire tega članka. Odkodiranje porabi tipično do 10-krat več strojne opreme (npr. logike, pomnilnika, procesorjevih ciklov) kot kodiranje.

3 Računanje v končnih obsegih

Končne obsege je vpeljal Galois v tridesetih letih 19. stoletja. V slovenski literaturi so predstavljeni že v učbeniku *Algebra* Ivana Vidava [21], pred kratkim pa smo v *Preseku* [8] lahko brali tudi o računanju v (manjših) končnih obsegih. Zato si le na konkretnem primeru, ki ga bomo uporabljali v naslednjih razdelkih, oglejmo, kako v praksi seštevamo in množimo.

Obseg $\text{GF}(2^4)$ konstruiramo iz kolobarja $\mathbb{Z}_2[x]$ z uporabo nerazcepnega polinoma $p(x) = x^4 + x + 1$. Elementi tega obsega so polinomi stopnje največ 3 nad \mathbb{Z}_2 . Le-te lahko predstavimo kot vektorje s štirimi komponentami. Medtem ko je seštevanje v tem obsegu čisto običajno seštevanje polinomov (aritmetika koeficientov se izvaja v \mathbb{Z}_2), pa pri množenju običajni produkt polinomov še zreduciramo po modulu $p(x)$ nad obsegom \mathbb{Z}_2 , tj. poiščemo ostanek pri deljenju s polinomom $p(x)$ v \mathbb{Z}_2 . Množica neničelnih elementov $\text{GF}(2^4)$ ima 15 elementov in je za operacijo množenje ciklična grupa. Izkaže se, da lahko neničelne elemente tega obsega predstavimo kar kot potence neke ničle polinoma $p(x)$, glej npr. Vidav [21]. Naj bo α taka ničla. Potem velja $\alpha^4 + \alpha + 1 = 0$ oziroma, ker gre za obseg karakteristike 2, tudi $\alpha^4 = \alpha + 1$. S pomočjo te relacije dobimo tabelo 3.1 elementov obsega $\text{GF}(2^4)$, kateri dodamo še element 0, ki ga predstavimo z α^∞ . Množenje je pri t. i. eksponentni predstavitvi obsega seveda enostavnejše, npr.

$$\alpha^8 \cdot \alpha^{10} = \alpha^{18} = \alpha^3,$$

(pri zadnjem enačaju smo upoštevali $\alpha^{15} = 1$, saj je α generator multiplikativne grupe $\text{GF}(2^4)$ in ima zato red 15), pri seštevanju v eksponentni predstavitvi pa uporabimo ZechLog tabelo (tabela 3.1), s katero prevedemo seštevanje na množenje, npr. elementa α^3 in α^5 seštejemo na naslednji način:

$$\alpha^3 + \alpha^5 = \alpha^3(1 + \alpha^2) = \alpha^3\alpha^{z(2)} = \alpha^3\alpha^8 = \alpha^{11}.$$

Morda velja opomniti, da je pri zelo velikih končnih obsegih (ki jih uporabljam npr. v kriptografiji) sestavljanje ZechLog tabele, pa tudi njeno hranjenje, absolutno prezahtevna naloga.

α^i	i	$z(i)$	α^i	i	$z(i)$
0000	∞	0	1101	7	9
1000	0	∞	1010	8	2
0100	1	4	0101	9	7
0010	2	8	1110	10	5
0001	3	14	0111	11	12
1100	4	1	1111	12	11
0110	5	10	1011	13	6
0011	6	13	1001	14	3

Tabela 3.1: **ZechLog tabela** oziroma eksponentna in polinomska predstavitev obsega $\text{GF}(2^4)$, ki je generiran z ničlo α nerazcepnega polinoma $p(x) = 1 + x + x^4$. Z njima lahko prevedemo seštevanje na množenje elementov, saj iz enakosti $\alpha^k + \alpha^h = \alpha^{\min(k,h)}(1 + \alpha^{\max(k,h)-\min(k,h)})$ sledi, da je dovolj za vsak i najti tako število $z(i)$, da bo veljalo $1 + \alpha^i = \alpha^{z(i)}$.

Hitrost računanja lahko povečamo z dodatno uporabo strojne opreme. Zato je Galoisova aritmetika idealna za implementacije v FPGA (Field-Programmable Gate Array) ali ASIC (Application Specific Integrated Circuit). Malo manj primerna pa je za DSP-je (Digital Signal Processor) ali mikroprocesorje, kjer običajno ne moremo uporabiti čipov, ki izračunajo produkt v enem samem ciklu (angl. single cycle binary multipliers). V ta namen lahko uporabimo knjižnice v Matlabu, Mathematici in Maplu ali pa tudi 'C' knjižnice za nižje nivojsko programiranje. Vendar pa se tudi tu reči spreminjajo na bolje. Tako so npr. Galoisovi aritmetični operatorji že v množici inštrukcij za TMS320C6400 DSP.

4 Reed-Solomonove kode kot linearne ciklične kode

Linearna koda $C \subseteq \mathbb{F}^n$ je **ciklična**, če je za vsako kodno besedo $c = (c_0, c_1, \dots, c_{n-1})$ tudi njen ciklični pomik, tj. beseda $c' = (c_{n-1}, c_0, c_1, \dots, c_{n-2})$, kodna beseda. Zelo uporabno reprezentacijo ciklične kode dobimo, če kodne besede predstavimo s polinomi. Kodni besedi c podobno kot pri sporočilu priredimo polinom $c(x) = c_0 + c_1x + \dots + c_{n-1}x^{n-1}$. Cikličnemu pomiku potem ustreza polinom

$$c'(x) = c_{n-1} + c_0x + c_1x^2 + \dots + c_{n-2}x^{n-1} = x \cdot c(x) - c_{n-1}(x^n - 1).$$

V kolobarju polinomov $R_n = \mathbb{F}^n[x]/(x^n - 1)$, kjer gledamo polinome po modulu polinoma $x^n - 1$, dobimo ciklični pomik kar z množenjem s polinomom x . Zato bomo pogosto enačili kodne besede s polinomi po modulu polinoma $x^n - 1$, tj. delali v kolobarju R_n .

Za obravnavo cikličnih kod je izdelana lepa in zanimiva teorija, katere osnove lahko najdemo v večini učbenikov iz teorije kodiranja, glej npr. Vanstone et al. [20], mi pa se omejimo le na najnajnejše.

Izrek 4.1 *Naj bosta n in k naravni števili, $n > k$, $g(x)$ moničen polinom (tj. polinom z vodilnim koeficientom 1) stopnje $n - k$, ki deli polinom $x^n - 1$. Potem je $S = \{a(x)g(x); \deg(a) < k\}$ cikličen podprostor vektorskega prostora R_n in $B = \{g(x), xg(x), \dots, x^{k-1}g(x)\}$ baza podprostora S .*

DOKAZ. Očitno je S podprostor v R_n . Pokažimo, da je S cikličen, tj. za polinom $p(x) := a(x)g(x) \in S$ moramo pokazati, da je tudi polinom $p_1(x) := x p(x) \bmod (x^n - 1)$ v S . To je očitno, saj je razlika $p_1(x) - x p(x)$ deljiva z $x^n - 1$, ki je deljiv z $g(x)$, polinom $p(x)$ pa je tudi deljiv z $g(x)$. Zato je z $g(x)$ deljiv tudi polinom $p_1(x)$.

Sedaj pa pokažimo, da je množica B baza podprostora S . Predpostavimo, da je poljubna linearne kombinacija $\sum_{i=0}^{k-1} \lambda_i x^i g(x)$ enaka 0. Če obstaja največji indeks j , za katerega je $\lambda_j \neq 0$, potem je koeficient ob x^{n-k+j} enak λ_j , kar pomeni, da mora biti $\lambda_j = 0$. Torej je B res množica linearne neodvisnih vektorjev.

Enostavno preverimo še, da vektorji iz B napenjajo cel podprostor S . Vzamemo poljuben $p(x) \in S$, potem je $p(x) = a(x)g(x)$ za nek $a(x) = a_0 + a_1x + \dots + a_{k-1}x^{k-1}$. Torej je $p(x) = a_0g(x) + a_1xg(x) + \dots + a_{k-1}x^{k-1}g(x)$ res linearne kombinacija polinomov iz $\{g(x), xg(x), \dots, x^{k-1}g(x)\}$.

■

Podmnožica S je **ideal** komutativnega kolobarja R , če je zaprta za seštevanje in je za vsak $r \in R$ in za vsak $s \in S$ tudi $rs \in S$. Izrek nam torej pove, da je vsak k -razsežni podprostor S , ki ustreza idealu/kodi v R_n , generiranemu s polinom $g(x)$ stopnje $n - k$, ki deli $x^n - 1$, cikličen. Polinom $g(x)$ imenujemo **generatorski polinom**.

Kolobar R_n je **glavni kolobar**, tj. vsak njegov ideal je generiran z enim samim polinomom. Ni se težko prepričati, da ideali v R_n ustrezajo natanko cikličnim kodam v \mathbb{F}^n . Ustrezen polinom

ciklične kode deli modul $x^n - 1$. **Kodiranje** pri cikličnih kodah je potem kar množenje polinoma sporočila z generatorskim polinomom $g(x)$.

RS-kode lahko opišemo tudi kot posebne vrste linearne ciklične kode.

Izrek 4.2 *Naj bo \mathbb{F} končen obseg s q elementi in $n := q - 1$. Naj bo k tako število, da velja $1 \leq k < n$ in $d := n - k + 1$ ter α primitiven element v \mathbb{F} . Koda C_1 naj bo linearne ciklične koda z generatorskim polinomom $g(x) = (x - \alpha)(x - \alpha^2) \dots (x - \alpha^{d-1})$, koda C_2 pa naj bo RS-koda, pri kateri sporočilu $m \in \mathbb{F}^k$ s pripredjenim polinomom $m(x) = m_0 + m_1 x + \dots + m_{k-1} x^{k-1}$ pripredimo kodno besedo $(m(\alpha), m(\alpha^2), \dots, m(\alpha^n))$. Potem kodi C_1 in C_2 sestavljajo iste kodne besede.*

Velja opozoriti, da zgornji izrek ne trdi, da istemu sporočilu v obeh primerih pripredimo isto kodno besedo in da izrek velja tudi, če pogoj $n = q - 1$ zamenjamo s $q - 1 \mid n$.

DOKAZ. Najprej se prepričajmo, da beseda $c = (c_0, c_1, \dots, c_{n-1})$, katere pripredjeni polinom $c(x) = c_0 + c_1 x + \dots + c_{n-1} x^{n-1}$ je oblike $c(x) = m(x)g(x)$, tj. beseda iz kode C_1 , ki pripada sporočilu m , tudi v kodi C_2 . Torej je treba poiskati tak polinom $f(x)$ stopnje $k - 1$, da bo $c_i = f(\alpha^i)$ za $i \in \{0, \dots, n - 1\}$. Naj bo $f(x) = f_0 + f_1 x + \dots + f_{n-1} x^{n-1}$, tako da velja

$$f_j = \frac{c(\alpha^{-j})}{n}, \quad j = 0, \dots, n - 1. \quad (4)$$

Polinom $c(x)$ je deljiv s polinomom $g(x)$, zato so $\alpha, \alpha^2, \dots, \alpha^{d-1}$ tudi njegove ničle. Ker je $d - 1 = n - k$, to pomeni, da za $j \in \{n - 1, n - 2, \dots, k\}$ velja $c(\alpha^{-j}) = c(\alpha^{n-j}) = 0$ in zato tudi $f_j = 0$. Torej ima polinom $f(x)$ stopnjo največ $k - 1$.

Izračunajmo še vrednosti $f(\alpha^i)$, $i \in \{0, \dots, n - 1\}$. Iz (4) sledi

$$f(\alpha^i) = \sum_{j=0}^{n-1} \frac{c(\alpha^{-j})}{n} \cdot (\alpha^i)^j = \frac{1}{n} \sum_{j=0}^{n-1} \left(\sum_{h=0}^{n-1} c_h \alpha^{-jh} \right) \alpha^{ij} = \frac{1}{n} \sum_{h=0}^{n-1} c_h \cdot \left(\sum_{j=0}^{n-1} \alpha^{(i-h)j} \right) = c_i.$$

Pri zadnjem enačaju smo upoštevali, da je izraz v zadnjem oklepaju enak n za $h = i$, sicer pa 0. To vidimo takole: α je primitiven element, zato je $\alpha^n = 1$ in $\alpha \neq 1$, se pravi, da je α ničla polinoma $(x^n - 1)/(x - 1) = 1 + x + x^2 + \dots + x^{n-1}$; enako velja tudi za vse potence α , ki so različne od ena. Ker je bilo m poljubno sporočilo iz \mathbb{F}^k , zaključimo, da velja $C_1 \subseteq C_2$.

Končno iz izrekov 2.2 in 4.1 sledi, da sta kodi C_1 in C_2 linearni in k -razsežni, torej mora veljati enakost $C_1 = C_2$. ■

Pravkar dokazani izrek nam da alternativno definicijo RS-kod. Le-ta omogoča enostavno in hitro odkodiranje s pomočjo t.i. ključne enačbe, ki jo bomo predstavili v naslednjem razdelku. Zgoraj opisana transformacija, ki preslikava $c(x)$ v $f(x)$, je znana kot **(inverzna) Fourierova transformacija** v končnih obsegih in je diskreten analog Fourierove transformacije v analizi.

5 Ključna enačba

Naj bo \mathbb{F} končen obseg s q elementi in $n := q - 1$. Naj bo k tako število, da velja $1 \leq k < n$ in $d := n - k + 1$. Naj bo α primitiven element v \mathbb{F} in

$$g(x) = (x - \alpha)(x - \alpha^2) \dots (x - \alpha^{d-1}).$$

Obračnavamo odkodiranje pri $RS(n, k)$ -kodi, generirani s polinomom $g(x)$. Naj bo $c(x) = a(x)g(x)$ poslana kodna beseda, $r(x)$ pa prejeta beseda. Lahko jo zapišemo v obliki

$$r(x) = c(x) + e(x), \quad (5)$$

kjer je $e(x)$ **polinom napake**. Če pri prenosu ni prišlo do napake, je $e(x)$ enak nič in je polinom $r(x)$ deljiv z $g(x)$. Polinom sporočila $a(x)$ dobimo iz $r(x)$ kar z deljenjem s polinomom $g(x)$. V primeru, da je prišlo do napake, pa bo odkodiranje težje in različnim načinom odkodiranja je posvečen ta razdelek. Najprej bomo odkodiranje prevedli na reševanje sistema linearnih enačb.

Vemo, da obstajata taka polinoma $h(x)$ in $s(x)$, da je

$$r(x) = h(x) \cdot g(x) + s(x) \quad \text{in} \quad \deg(s(x)) < \deg(g(x)). \quad (6)$$

Polinom $s(x)$ imenujemo **sindrom** prejete besede $r(x)$. Ker so $\alpha, \alpha^2, \dots, \alpha^{d-1}$ ničle polinoma $g(x)$ in zato tudi polinoma $c(x)$, velja zaradi (5) in (6) naslednja zveza:

$$r(\alpha^i) = e(\alpha^i) = s(\alpha^i) \quad \text{za } i = 1, \dots, d-1. \quad (7)$$

Predpostavimo, da pri prenosu ni prišlo do več kot $\ell \leq \lfloor (d-1)/2 \rfloor$ napak, kolikor jih koda največ lahko odpravi. Naj bodo $a_0, a_1, \dots, a_{\ell-1} \in \{0, \dots, n-1\}$ mesta v kodni besedi, na katerih je prišlo do napake. Potem lahko polinom $e(x)$ zapišemo v obliki

$$e(x) = \sum_{j=0}^{\ell-1} \lambda_j x^{a_j}.$$

Količino $s(\alpha^i)$ označimo s S_i . Eksponenti a_j v potenci α^{a_j} nam povedo položaje napak, zato števila α^{a_j} imenujemo **lokatorji napak**. Vrednosti λ_j pa so **velikosti napak**. Iz (7) dobimo sistem enačb

$$S_i = \sum_{j=0}^{\ell-1} \lambda_j (\alpha^i)^{a_j} = \sum_{j=0}^{\ell-1} \lambda_j (\alpha^{a_j})^i \quad \text{za } i = 1, \dots, d-1 \quad (8)$$

z neznankami λ_j in α^{a_j} , $j = 0, \dots, \ell-1$. Z uvedbo oznak $X_j = \alpha^{a_j}$, $j = 0, \dots, \ell-1$, sistem (8) zapišemo v obliki

$$\begin{aligned} S_1 &= \lambda_0 X_0 + \lambda_1 X_1 + \cdots + \lambda_{\ell-1} X_{\ell-1}, \\ S_2 &= \lambda_0 X_0^2 + \lambda_1 X_1^2 + \cdots + \lambda_{\ell-1} X_{\ell-1}^2, \\ &\vdots \\ S_{d-1} &= \lambda_0 X_0^{d-1} + \lambda_1 X_1^{d-1} + \cdots + \lambda_{\ell-1} X_{\ell-1}^{d-1}. \end{aligned} \quad (9)$$

Ta sistem $d-1$ enačb z 2ℓ neznankami (λ_j in X_j) se je v preteklosti pojavil pri reševanju različnih problemov, glej Barg [1]. Prvi se je verjetno z njim ukvarjal baron de Prony že okrog leta 1795 pri reševanju nekega interpolacijskega problema. Zanimivo je, da so različni avtorji predlagali precej podoben način za reševanje sistema (9), ki ga bomo opisali spodaj. Najprej poiščemo vrednosti X_j , nato pa lahko iz sistema (9) poiščemo še velikosti napak, saj je sistem enačb za λ_i , $i = 0, \dots, \ell-1$, linearen.

Naj bo $\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2 + \cdots + \sigma_\ell x^\ell$ **polinom lokatorjev napake** oziroma bolj precizno polinom, ki ima za ničle ravno inverzne vrednosti lokatorjev napak, tj. $\prod_{i=0}^{\ell-1} (1 - X_j x)$. Zato velja:

$$\lambda_j X_j^{\ell+u} \sigma(X_j^{-1}) = 0 \quad \text{za } j = 0, \dots, \ell-1, \quad (10)$$

kjer je u naravno število manjše ali enako ℓ . Seštejmo enačbe (10), upoštevajmo še sistem (9) in dobimo

$$0 = \sum_{j=0}^{\ell-1} \lambda_j X_j^{\ell+u} \left(1 + \sum_{i=1}^{\ell} \sigma_i X_j^{-i} \right) = S_{u+\ell} + \sum_{i=1}^{\ell} \sigma_i \sum_{j=0}^{\ell-1} \lambda_j X_j^{\ell+u-i} = S_{u+\ell} + \sum_{i=1}^{\ell} \sigma_i S_{\ell+u-i},$$

kar je rekurzivna enačba za zaporedje $\{S_i\}$:

$$\sigma_1 S_{u+\ell-1} + \sigma_2 S_{u+\ell-2} + \cdots + \sigma_\ell S_u = -S_{u+\ell}. \quad (11)$$

Ko u teče od $1, \dots, \ell$, dobimo sistem linearnih enačb za σ_i , $i = 1, \dots, \ell$, ki ga lahko zapišemo v matrični obliki

$$\begin{bmatrix} S_1 & S_2 & \dots & S_\ell \\ S_2 & S_3 & \dots & S_{\ell+1} \\ \vdots & \vdots & & \vdots \\ S_\ell & S_{\ell+1} & \dots & S_{2\ell-1} \end{bmatrix} \begin{bmatrix} \sigma_\ell \\ \sigma_{\ell-1} \\ \vdots \\ \sigma_1 \end{bmatrix} = - \begin{bmatrix} S_{\ell+1} \\ S_{\ell+2} \\ \vdots \\ S_{2\ell} \end{bmatrix}. \quad (12)$$

Vnaprej ne poznamo ℓ , zato namesto z ℓ računamo z $\lfloor (d-1)/2 \rfloor$. Izkaže se, glej npr. [3, str. 149], da je rang matrike sistema v tem primeru enak številu napak. Ko poznamo število napak, lahko iz sistema (12) izračunamo koeficiente polinoma $\sigma(x)$. Da dobimo lokatorje napak, moramo poiskati ničle $\sigma(x)$ in njihove inverze. Ker smo v končnem obsegu, ničle lahko poiščemo tudi tako, da kar po vrsti preizkušamo elemente obsega (v praksi namreč obseg nima več kot 32 elementov).

Da bi se izognil reševanju dodatnega sistema enačb za izračun velikosti napak (λ_i), je Forney [4] izkoristil zvezo med $\sigma(x)$ in t.i. **polinomom izračuna vrednosti napak** (tj. interpolacijskim polinomom, se pravi, da gre v bistvu za Lagrangeovo interpolacijo):

$$\omega(x) = \sum_{i=0}^{\ell-1} \lambda_i X_i x \prod_{k=0, k \neq i}^{\ell-1} (1 - X_k x).$$

Naj bo $S(x) = \sum_{i=1}^{d-1} S_i x^i$, potem velja enakost

$$\boxed{\omega(x) = S(x) \cdot \sigma(x) \pmod{x^{d-1}}}, \quad (13)$$

ki sledi iz sistema (9):

$$\begin{aligned} S(x) \cdot \sigma(x) &= \sum_{i=1}^{d-1} \sum_{j=0}^{\ell-1} \lambda_j X_j^i x^i \cdot \prod_{k=0}^{\ell-1} (1 - X_k x) = \\ &= \sum_{j=0}^{\ell-1} \lambda_j X_j x \sum_{i=1}^{d-1} X_j^{i-1} x^{i-1} \cdot \prod_{k=0}^{\ell-1} (1 - X_k x) = \sum_{j=0}^{\ell-1} \lambda_j X_j x \prod_{k=0, k \neq j}^{\ell-1} (1 - X_k x) \cdot \left[(1 - X_j x) \sum_{i=1}^{d-1} X_j^{i-1} x^{i-1} \right], \end{aligned}$$

saj je izraz v oglatih oklepajih enak $1 - X_j^{d-1} x^{d-1}$ in zato kongruenten 1 po modulu x^{d-1} .

Ko enkrat poznamo $\omega(x)$, lahko enostavno izračunamo velikosti napak:

$$\lambda_j = \frac{\omega(X_j^{-1})}{\prod_{k \neq j} (1 - X_k X_j^{-1})} = -\frac{X_j \omega(X_j^{-1})}{\sigma'(X_j^{-1})}, \quad (14)$$

kjer smo z σ' označili običajen odvod polinoma $\sigma(x)$.

Algoritem za odkodiranje Reed-Solomonovih kod, ki smo ga predstavili zgoraj, je bistveno hitrejši od tistega iz drugega razdelka, saj je polinomski. Rešimo le dva sistema enačb (12) in (9) velikosti $O(d \times d)$, iščemo inverze ℓ elementov, ki so lahko shranjeni tudi v tabeli, ter vrednosti polinoma $\sigma(x)$ v največ n točkah. Skupna zahtevnost algoritma je v najslabšem primeru enaka $O(n^3)$. Zahtevnost algoritma s Forneyevim izboljšavo še vedno ostane reda $O(n^3)$.

Enačbi (13) pravimo tudi **ključna enačba** [16], saj igra pomembno vlogo v teoriji kodiranja. Iz nje lahko tudi brez reševanja sistema (12) poiščemo polinom $\sigma(x)$. Najhitrejši takšen postopek, ki ga je predlagal Berlekamp, ima časovno zahtevnost $O(\ell \cdot d)$. Sugiyama in sodelavci pa so predlagali postopek reševanja ključne enačbe na osnovi razširjenega Evklidovega algoritma, glej [10], [7], ki je sicer nekoliko počasnejši, reda velikosti $O(d^2) = O(n^2)$, a tudi manj zapleten.

Primer 5.1 Oglejmo si kodiranje z RS(15, 9)-kodo nad obsegom GF(2^4), ki smo ga opisali v tretjem razdelku. Za primitivni element obsega pa si zopet izberemo ničlo α modulskega polinoma. Razdalja kode je enaka $d = 15 - 9 + 1 = 7$, tako da koda popravi do tri napake. Stopnja generatorskega polinoma $g(x)$ je $n - k = 15 - 9 = 6$. Z uporabo Tabele 3.1 izračunamo

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4)(x - \alpha^5)(x - \alpha^6) = \alpha^6 + \alpha^9 x + \alpha^6 x^2 + \alpha^4 x^3 + \alpha^{14} x^4 + \alpha^{10} x^5 + x^6.$$

Kodiranje je množenje s polinomom $g(x)$. Besedo $m = (0, 0, 1, 0, \alpha^{10}, 0, \alpha^2, 0, 0)$ zakodiramo torej kot $c(x) = m(x) \cdot g(x) = \alpha^6 x^2 + \alpha^9 x^3 + \alpha^{11} x^4 + \alpha^{11} x^6 + \alpha^{11} x^8 + \alpha^9 x^9 + \alpha^8 x^{10} + \alpha^{12} x^{11} + \alpha^2 x^{12}$ oziroma $c = (0, 0, \alpha^6, \alpha^9, \alpha^{11}, 0, \alpha^{11}, \alpha^9, \alpha^8, \alpha^{12}, \alpha^2, 0, 0)$.

Poglejmo sedaj še, kako poteka odkodiranje. Če je prirejeni polinom $c(x)$ kodne besede c deljiv s polinomom $g(x)$, potem je polinom sporočila $m(x)$ enak $c(x)/g(x)$. Poskusimo odkodirati še prejeto besedo r s prirejenim polinomom $r(x) = \alpha^6 x^2 + \alpha^9 x^3 + x^4 + x^5 + x^6 + \alpha^{10} x^7 + \alpha^3 x^8 + \alpha^3 x^9 + \alpha^2 x^{12}$. Polinom $r(x)$ ni deljiv z $g(x)$, saj je ostanek enak $s(x) = \alpha^5 + \alpha^{10} x + \alpha x^2 + \alpha^{10} x^3 + \alpha^3 x^4 + \alpha^9 x^5$. Izračunamo $S_i = s(\alpha^i)$ za $i = 1, \dots, 6$ in dobimo naslednje vrednosti

S_1	S_2	S_3	S_4	S_5	S_6
α^{12}	0	α^3	α^2	α^3	1

Sestavimo matriko iz sistema (12).

$$\begin{bmatrix} \alpha^{12} & 0 & \alpha^3 \\ 0 & \alpha^3 & \alpha^2 \\ \alpha^3 & \alpha^2 & \alpha^3 \end{bmatrix} \quad (15)$$

Matriko (15) enostavno prevedemo na zgornje-trikotno obliko. Od tretje vrstice odštejemo prvo, pomnoženo z α^6 , in nato še drugo, pomnoženo z α^{14} (ker ima obseg karakteristiko 2, je odštevanje kar enako seštevanju). Dobimo matriko ranga 2, kar pomeni, da je pri prenosu kodne besede prišlo do dveh napak. Zato je treba rešiti sistem dveh enačb z dvema neznankama

$$\begin{bmatrix} \alpha^{12} & 0 \\ 0 & \alpha^3 \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = - \begin{bmatrix} \alpha^3 \\ \alpha^2 \end{bmatrix}, \quad (16)$$

ki nam da rešitev $\sigma_1 = \alpha^{14}$ in $\sigma_2 = \alpha^6$. Sedaj poznamo polinom $\sigma(x) = 1 + \alpha^{14} x + \alpha^6 x^2$. Z računanjem njegovih vrednosti v vseh elementih obsega GF(2^4) preverimo, da sta njegovi ničli α^4 in α^5 . Njuna inverza α^{11} in α^{10} nam povesta, da sta napaki pri prejeti besedi na desetem in enajstem mestu. Preostane nam le še, da izračunamo velikosti teh napak. V našem primeru bo to najenostavnejše kar z reševanjem sistema (9). Le-ta je predoločen; če nima rešitve, je bila predpostavka, da je prišlo do največ treh napak, napačna. Mi bomo velikosti napak izračunali iz prvih dveh enačb (torej brez uporabe ključne enačbe)

$$\begin{aligned} \alpha^{12} &= \lambda_0 \alpha^{11} + \lambda_1 \alpha^{10} \\ 0 &= \lambda_0 (\alpha^{11})^2 + \lambda_1 (\alpha^{10})^2 \end{aligned} \quad (17)$$

in z deljenjem s polinomom $g(x)$ preverili, da smo res dobili kodno besedo. Velikosti napak sta $\lambda_0 = \alpha^{12}$ in $\lambda_1 = \alpha^{14}$. Polinom poslane kodne besede je potem $c_1(x) = \alpha^6 x^2 + \alpha^9 x^3 + x^4 + x^5 + x^6 + \alpha^{10} x^7 + \alpha^3 x^8 + \alpha^3 x^9 + \alpha^{14} x^{10} + \alpha^{12} x^{11} + \alpha^2 x^{12}$. Ker velja $c_1(x) = g(x) \cdot (x^2 + \alpha^7 x^4 + \alpha^2 x^6)$, je polinom sporočila enak $x^2 + \alpha^7 x^4 + \alpha^2 x^6$, samo sporočilo pa je enako $(0, 0, 1, 0, \alpha^7, 0, \alpha^2, 0, 0)$.

6 Zaključek

Precej raziskav na področju teorije kodiranja je usmerjeno v izboljšavo razmerja med verjetnostjo, da je ugibanje odkodirnega postopka pravilno, in kompleksnostjo kodiranja in odkodiranja. Šele ko so raziskovalci ugotovili, da so Reed-Solomonove kode samo poseben primer širšega razreda cikličnih BCH-kod in je Berlekamp našel učinkovit algoritem za odkodiranje BCH-kod, se je začel razcvet RS-kod. Danes so RS-kode primer učinkovitih in popularnih kod v številnih področjih, kot so

- naprave za skladiščenje/hranjenje podatkov (trdi disk, CD, DVD, barkode, tape-backup sistemi, digital audio tape - DAT,...) in njihovo branje (npr. predvajalniki, digitalna televizija ...),
- brezžične komunikacije (mobilni telefoni, mikrovalovne povezave ...),
- satelitske komunikacije (npr. Voyager, Mariner, Mars Lander, Cassini ...),
- modemi za širokopasovne povezave (ADSL, xDSL,...).

Primer: Za DVB (digital video broadcasting) uporabljamo RS(204, 188)-kodo z 8-bitni simboli (tj. $n = 204$, $k = 188$, $s = 8$). $n - k = 16 = 2t$, se pravi, da dodamo blokom po 16 simbolov in lahko pri tem popravimo do 8 ($t = 16/2$) napačnih simbolov. Zaporedje do 56 zaporednih bitov lahko pokvari največ 8 simbolov.

Prednost RS-kod je v tem, da znajo z enako lahkoto popraviti simbol z eno samo bitno napako kakor tudi simbol, pri katerem so napačni vsi biti. Zato so RS-kode posebej primerne za odpravljanje *grozdnih napak* (tj. napak, kjer se napačni biti držijo skupaj). To pa pomeni, da so RS-kode občutljive na enakomerno porazdeljene napake. Druge kode, kot npr. konvolucijske kode, so boljše za odpravljanje naključnih napak, zato pogosto bloke RS-kod še prej zakodiramo s konvolucijskimi kodami in na ta način omogočimo odpornost tako na grozdne kakor tudi na naključne napake. Običajno se uporablja strojne implementacije RS-kod, vendar pa so danes zaradi občutno povečane hitrosti mikroprocesorjev možne tudi programske implementacije (npr. Texas Instruments daje na voljo brezplačen program za odkodiranje).

Kot smo videli, je študija enakosti v Singletonovi oceni pripeljala do RS-kod in nato še do ključne enačbe ter Berlekamp-Masseyevega algoritma. Podoben pomen imajo tudi druge meje/ocene v teoriji kodiranja. Poglejmo si še en primer zanimive ocene. Če zna (n, k) -koda odpraviti t napak, so krogle z radijem t in središčem v kodnih besedah disjunktne, tako da velja

Hammingova ocena:

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

Kodi z radaljo d pravimo, da je **perfektna**, kadar v zgornji neenakosti velja enakost za $t = \lfloor(d-1)/2\rfloor$. Priponimo, da je razdalja perfektnih kod nujno liha. Perfektne kode v Hammingovih grafih [9] so že klasificirane (rezultat so dokazale leta 1973 neodvisno dve skupini raziskovalcev, glej npr. [12, razdelek 6.10]):

- linearna $[(q^m - 1)/(q - 1), q^{n-m}, 3]$ Hammingova koda \mathcal{H}_m nad abecedo s q elementi,
- binarna linearna $[23, 2^{12}, 7]$ Golayeva koda \mathcal{G}_{23} ,
- ternarna linearna $[11, 3^6, 5]$ Golayeva koda \mathcal{G}_{11} .

Še vedno pa ni želenega napredka na področju klasifikacije perfektnih kod v Johnsonovih grafih [9], glej Martin [13].

Literatura

- [1] A. Barg, *At the dawn of the theory of codes*, *Math. Intelligencer* **15** (1993), 20–26.
- [2] E. R. Berlekamp, *Algebraic Coding Theory*, Aegean Park Press, revised edition, 1984.
- [3] D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger in J.R. Wall, *Coding Theory: The Essentials*, Marcel Dekker, Inc., 1991.
- [4] G.D. Forney, On decoding BCH codes, *IEEE Trans. Inform. Theory* IT11(4) (1965), 549–557.
- [5] W. C. Huffman, V. S. Pless and R. A. Brualdi (uredniki), *Handbook of Coding Theory*, Vol. 1 & 2, North-Holland, 1998.
- [6] A. Jurišić, *Kako deliti skrivnost*, *Presek* **29** (2001-02), 358–364.
- [7] A. Jurišić, *Računala nove dobe, 1 in 2. del*, *Presek* **30** (2002-03), str. 226-231 in 291–296.
- [8] A. Jurišić, *Napake niso za vedno*, *Presek*, 30 (2002-03), 361-366.
- [9] A. Jurišić, Š. Miklavič, *Asociativne sheme*, OMF **50/3** (2003), 65-81.
- [10] M. Juvan, *Razširjen Evklidov algoritem*, *Presek* **21** (1993-94), str. 116–121.
- [11] S. Klavžar, *O teoriji kodiranja, linearnih kodah in slikah z Marsa*, *Obozornik mat. fiz.* **45** (1998) 97–106.
- [12] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*, Part 1, North-Holland, 1977.
- [13] W. J. Martin, *Completely Regular Codes*, Ph.D. Thesis, University of Waterloo (1992).
- [14] T. Novak, Psevdonaključnost v kriptografiji, diplomska delo, Ljubljana, 2001.
- [15] T. Novak, Uporaba Berlekamp-Masseyevega algoritma v kriptografiji in teoriji kodiranja, magistrsko delo, Ljubljana, 2003.
- [16] W. W. Peterson, *Encoding and error-correction procedures for Bose-Chaudhuri codes* IRE Transactions on Information Theory, Vol. IT 6 (1960) 459–470.
- [17] M. Praprotnik, Kriptoanaliza časovno kontroliranih pomicnih registrov, diplomska delo, Ljubljana, 2002.
- [18] I. S. Reed and G. Solomon, *Polynomial codes over certain finite fields*, *J. Soc. Indust. Appl. Math.* **8** (1960) 300–304.
- [19] R. C. Singleton, *Maximum distance q-nary codes*, *IEEE Trans. Inform. Theory* IT-10(2) (1964), 116-118.
- [20] S. A. Vanstone and P. C. van Oorschot, *An Introduction to Error Correcting Codes with Applications*, Kluwer Academic Publishers, 1989.
- [21] I. Vidav, *Algebra*, Mladinska knjiga, Ljubljana 1972.

Predlog slike za naslovnico (posebej če bo članek izšel v OMF okoli julija 2004):

Vesoljska sonda Cassini je na svoji poti k Saturnu posnela Jupiter. Saturn bo dosegla julija 2004, glej <http://www.ssd.rl.ac.uk/news/cassini/mission/cas.html>. Cassini je največja medplanetarna sonda doslej, uporablja pa naslednje kode za prenos na Zemljo:

- 1) Konvolucijska koda s parametri bodisi ($k = 7, r = 1/2$) ali ($k = 15, r = 1/6$). Če primerjamo ($k = 7, r = 1/2$) kodo z nekodiranim kanalom, potem je 4.5 dB boljša; ($k = 15, r = 1/6$) koda pa je 2 dB boljša kot ($k = 7, r = 1/2$) koda. Konvolucijska koda tipično omogoča BER (bit error rate) 1 na 200.
- 2) RS(255, 223)-koda je uporabljena v povezavi s konvolucijsko kodo (tj. vesolsko plovilo najprej zakodira podatke z RS-kodo, potem pa naredi konvolucijsko kodiranje RS simbolov). BER od te kode je vsaj 1 na milion, kar projekt Cassini tudi potrebuje.

Cassini je skupen projekt NASA-e ter evropske in in italijanske vesoljske agencije, ki vključuje 4.300 ljudi. Na dan lahko pošlje na Zemljo do 4Gb podatkov.

7 TO DO

Sasa Section 1: Uvod

- motivation, reklama za kode (sledi Barry Cippra) {\bf (1 stran)}

Sasa Section 2: ideja s polinomi (sledi Barry Cippra)

- polinom stopnje n dolo"ca n+1 to"ck
(utemeljitev z Vandermonde...)
- interpolacija
- uporaba za R-S kode
- povezava z deljenjem skrivnosti (najvec en odstavek in kazalec na Presek) {\bf (1 stran)}

Section 3: Kon"cni obsegci in ra"cunanje v razlicnih bazah

- konkretna konstrukcija GF(2^{eN}) - samo en primer {\bf (1/2 strani)}

ARJANA Section 4: pristop z linearnimi kodami, ciklicnimi, ...

preko polinomov {\bf (2 strani)}

ARJANA Section 5: Fourierova transformacija {\bf (1-2 strani)}

- dokaz ekvivalentnosti
- uporaba/povezava z drugimi podrocji;
"ce ne najdemo, skrijemo Fourierovo transformacijo v dokaz ekviv.
v prej"snji razdelek.

Section 6: Klju"cna ena"cba {\bf (2-3 strani)}

- izpelji

- re"si

"Casovna zahtevnost:

RS: $O(n^k) == \text{eksponentna}$

Sistem ena"cb, Peterson 60, $O(n^3)$

Raz"sirjen Evklidov algoritem, Sugiyama 75, $O(n^2)=O(d^2)$

Berlekamp 69, $O(l*d)$

List decoding: Sa"sa preberi

Raz"sirjen Evklidov algoritem: Juvan, Presek

- kje vse naletimo nanjo (Barg, Sudan?)

Section 7: Zaklju"cek