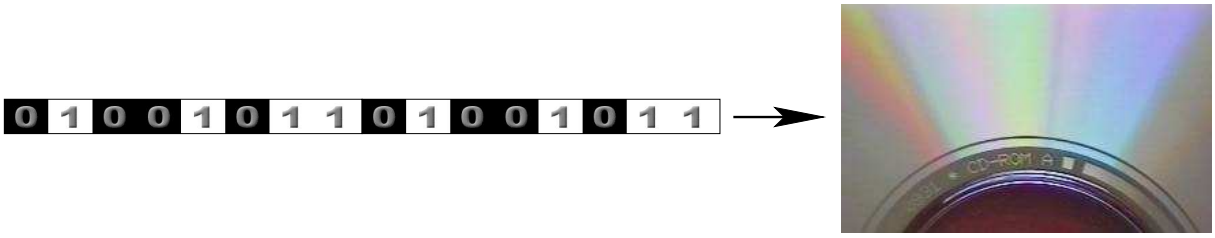
Napake niso za vedno –
Presek, zgoščenke, planeti in kode

10. apr., 2003

V času informacijske tehnologije (zgoščenke, mobilni telefoni, bančne kartice, internet) se vsi dobro zavedamo pomena hitrega in natančnega prenosa, obdelovanja in hranjenja informacij. Še tako popolne naprave delajo napake in njihovi spodrsljaji lahko spremenijo sicer izredno koristno programsko in strojno opremo v ničvredno ali celo nevarno orodje. Dolgo časa so se ljudje trudili izdelati računalnike in pomnilnike, ki bodo naredili oziroma vsebovali kar se da malo napak. Seveda so bile zato cene takih izdelkov vedno višje. Potem pa so se domislili, da bi raje naučili računalnike iskati in odpravljati napake. Tako je postala proizvodnja veliko učinkovitejša in cenejša. V resnici smo za povečanje zanesljivosti prenosa in obdelave informacij najprej dolgo časa uporabljali kontrolne bite (angl. parity-check bits), kot npr. pri številki bančnega čeka, ki so služili le za odkrivanje napak. Pred leti pa je matematik Richard Hamming vnašal v računalnik programe s pomočjo luknjanja kartic. Po mnogih neuspelih poskusih, ko mu je računalnik zavrnil paket kartic zaradi napak, se je zamislil: "Če zna računalnik sam odkriti napako, zakaj ne zna najti tudi njenega mesta in je odpraviti."



Slika 0

Odkar so računalniki začeli prevzemati večino dela na področju obdelovanja informacij in pri telekomunikacijah, nas še posebej zanima, kaj se da narediti, če pride do napake? Na začetku so bili računalniški programi dovolj enostavni, tako da so tehnične napake (ponavadi je odpovedala elektronka) hitro postale očitne. Toda z razvojem strojne opreme so postali programi vse bolj obsežni in zapleteni, s tem pa je postalo upanje, da lahko hitro opazimo majhne napake, ki spremenijo delovanje naprave, zanemarljivo in zato tudi resna skrb. Tudi elektronska vezja postajajo iz dneva v dan manjša, računalniki pa vse hitrejši, in tako je možnost, da se nam izmuzne kakšna napaka, vse večja. Tudi če je ta možnost ena sama milijardinka (npr. industrijski standard za trde diske je ena napaka na 10 milijard bitov), se bo računalnik, ki opravi 2 milijardi osnovnih operacij na sekundo, in tak računalnik je prav dolgočasno počasen glede na superračunalnike, zmotil približno dvakrat na sekundo. Glede na količino podatkov, ki jih obdelujemo dandanes, je to recept za vsakodnevne nevšečnosti.

Kaj lahko naredimo? Raziskovalci so našli odgovor v kodah za odpravljanje napak. **Koda** je skupina simbolov, ki predstavlja informacijo. Kode obstajajo že tisočletja. To so npr. hieroglifi, grška abeceda, rimske številke ali pa genetska koda za sestavljanje ribonukleinskih kislin. Nastale so za različne potrebe: za zapis govora ali glasbe, Morsejeva abeceda za prenos informacij, za shranjevanje podatkov itd. Matematična teorija kod, ki se je razvila v zadnjih petdesetih letih, je računalnikarjem in inženirjem omogočila, da sestavijo sisteme, ki so kar se da zanesljivi (kolikor pač dopušča strojna oprema). Tako lahko teorijo kodiranja smatramo za varnostno mrežo – matematično zavarovanje pred muhastim materialnim svetom, v katerem živimo.

> *Tehnologija kod za popravljanje napak je danes tako razširjena kot zgoščenke (CD). Omogoča nam, da poslušamo priljubljeni Mozartov ali Madonnin CD brez kakršnih koli motenj, četudi nam ga mačka prav pošteno spraska.*

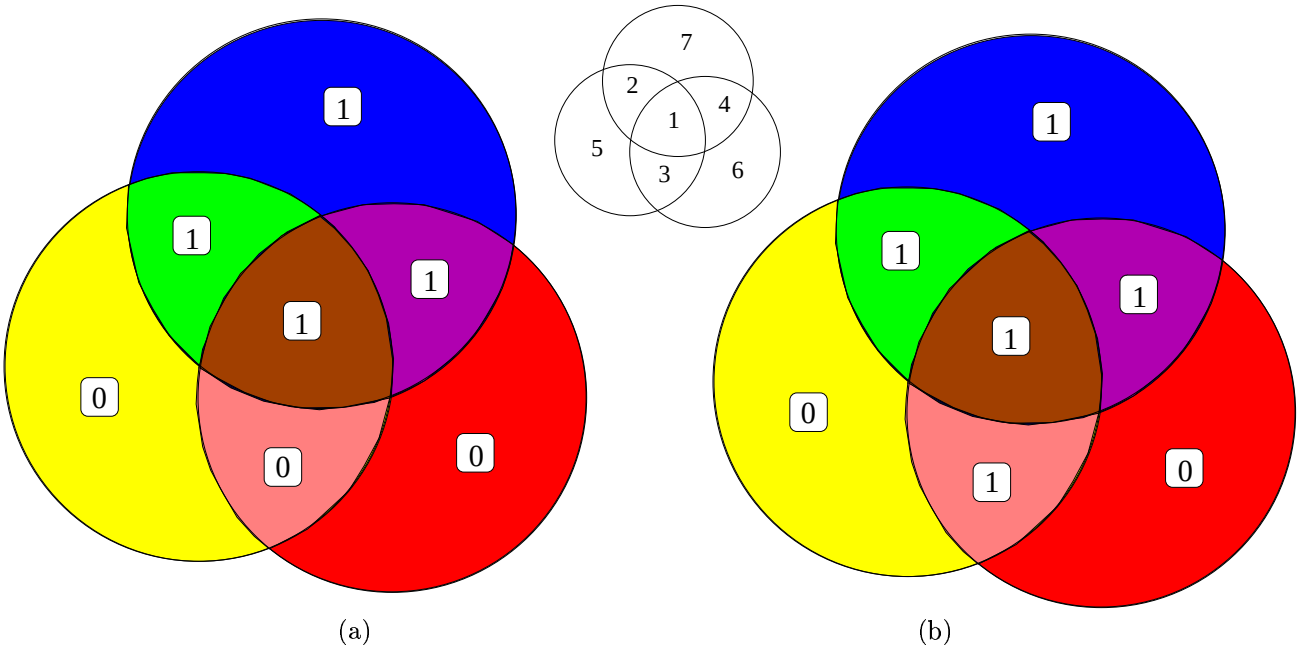
Enako tehnologijo uporabljajo za komunikacijo tudi vesoljske ladje in sonde, ki raziskujejo naše osončje. Kode za odpravljanje napak omogočajo, da kljub elektromagnetnim motnjam pridejo na Zemljo kristalno jasni posnetki oddaljenih planetov, pri tem pa za prenos porabimo manj energije kot hladilnikova žarnica (gre torej za šepetanje, ki mora prepotovati več milijard kilometrov).

V nadaljevanju prispevka bomo najprej predstavili enostavnejše kode za odpravljanje napak, npr. kode s ponavljanjem in Hammingove kode, nato pa bomo spoznali še poenostavljeno varianto t. i. Reed-Salomonovih kod. Nazadnje si bomo ogledali osnove kodiranja na zgoščenkah.

Enostavne kode za odpravljanje napak

Claude Shannon je kmalu po koncu druge svetovne vojne postavil teoretične osnove teoriji informacij in zanesljivemu prenosu digitalnih podatkov. Leta 1948 pa je Richard Hamming izumil metodo za *popravljanje* ene napake in *odkrivanje* dveh napak. Bistvo vseh metod za odpravljanje napak je dodajanje kontrolnih bitov. Najenostavnejša koda za odpravljanje napak je zasnovana na *ponavljanju*. Na primer, če pričakujemo, da pri prenosu ne bo prišlo do več kot ene same napake, potem je dovolj, da vsak bit ponovimo trikrat in pri sprejemu uporabimo "večinsko pravilo": Npr. zelo kratko "simfonijo" 1101 zakodiramo v 111 111 000 111. Če prejmemo 111 011 000 111, popravimo sporočilo v 111 111 000 111 in ga nazadnje še dekodiramo v 1101. V splošnem lahko odpravimo n napak z $(2n+1)$-kratnim ponavljanjem in uporabo večinskega pravila. Toda ta metoda je preveč potratna. V času, ko si želimo hitrega prenosa čim večje količine podatkov, je takšno napihovanje večinoma nesprejemljivo. Namesto tega si želimo dodati manjše število kontrolnih bitov, ki pa naj bodo ravno tako ali pa morda še bolj učinkoviti.

Najpreprostejši primer Hammingove kode za odpravljanje napak predstavimo kar s pomočjo Presekovega znamenja, glej sliko 1.



(a) (b)

Slika 1: (a) Našo kratko "simfonijo" 1101 spravimo tokrat zaporedoma na rjavo (1), zeleno (2), oranžno (3) in vijoličasto polje (4), preostala polja pa dopolnimo tako, da bo v vsakem krogu vsota števil *soda*. Dobimo kodo 1101001, kjer zadnja tri mesta predstavljajo zaporedoma rumeno (5), rdeče (6) in modro polje (7).

Naštejmo vseh 16 kodnih besed, ki jih lahko dobimo na ta način: 0000000, 0001011, 0010110, 0011101, 0100101, 0101110, 0110011, 0111000, 1000111, 1001100, 1010001, 1011010, 1100010, 1101001, 1110100, 1111111.

(b) Recimo, da je prišlo do ene same napake in da smo prejeli zaporedje 1111001. Potem bo prejemnik lahko ugotovil, da je napaka v rumenem (levem) in rdečem (desnem) krogu, ne pa v modrem (zgornjem), kar pomeni, da je potrebno popraviti oranžno polje (3). Ni se težko prepričati, da je možno na tak način odpraviti napako na poljubnem mestu (tudi kontrolnem), seveda ob pogoju, da je to edina napaka.

S Hammingovo kodo nam je uspelo zmanjšati število kontrolnih bitov z 8 na 3, tj. dobili smo kodo z *informacijsko stopnjo* 4/7 namesto prejšnjih 4/12=1/3. Opisano Hammingovo kodo lahko seveda posplošimo. Običajno to storimo z nekaj linearne algebre (matrike), glej [3].

Hammingova koda odkrije, da je prišlo do napake pri prenosu tudi, kadar je prišlo do dveh napak, saj ne morejo vsi trije krogi vsebovati obeh polj, na katerih je prišlo do napake. Če pa na dveh mestih zaznamo, da simbola manjkata – npr. da ju ne moremo prebrati (angl. erasure), potem znamo ti dve mesti celo popraviti.

V grobem lahko rečemo, da je cilj teorije kodiranja najti primeren kompromis med skromno metodo nadzora z nekaj kontrolnimi biti in razsipno metodo s ponavljanji. Hammingova koda predstavlja prvi korak v to smer.

Reed-Salomonove kode

Po odkritju Hammingove kode je sledilo obdobje številnih poskusov s kodami za odpravljanje napak. Ko je bila teorija kod stara deset let, sta Irving Reed in Gustave Salomon (takrat zaposlena v Lincolnovem laboratoriju na MIT) zadela v polno. Namesto ničel in enic sta uporabila skupine bitov, ki jim tudi v računalništvu pravimo kar *besede*. Ta izbira je pripomogla k odpravljanju grozdnih napak, tj. napak, pri katerih se pokvari več zaporednih bitov. Npr. če delamo s skupinami, sestavljenimi iz osmih bitov, potem celo devet zaporednih napak lahko pokvari največ dve besedi. Reed-Salomonova koda (na kratko RS-koda) za odpravljanje dveh napak torej predstavlja že precej dobro zaščito.

Naj bo $(a_0, a_1, \ldots, a_{10})$, $a_i \in \mathbb{Z}_{13}$, sporočilo, ki ga želimo prenesti. Nadomestimo ga s 13-terico $(c_0, c_1, \ldots, c_{10}, c_{11}, c_{12})$, tako da je $c_i = a_i$ za $i \in \{0, \ldots, 10\}$, besedi $c_{11}, c_{12} \in \mathbb{Z}_{13}$ pa izračunamo iz enačb

$$c_0 + c_1 + \cdots + c_{12} \mod 13 = 0 \quad \text{in} \quad c_1 + 2c_2 + \cdots + 12c_{12} \mod 13 = 0, \quad (1)$$

in pošljemo. Predpostavimo, da smo prejeli zaporedje $(r_0, r_1, \ldots, r_{10}, r_{11}, r_{12})$, pri čemer je prišlo kvečjemu do ene napake. Če je do napake prišlo pri prenosu besede c_i, potem velja $r_j = c_j$ za vse $j \in \{0, 1, \ldots, 12\} \backslash \{i\}$ in $r_i = c_i + e$ za neki $e \in \{1, \ldots, 12\}$. Iz $e = r_0 + r_1 + \cdots + r_{12} \mod 13 \neq 0$ ugotovimo, da je prišlo do napake. Nato pa iz $r_1 + 2r_2 + \cdots + 12r_{12} \equiv ie \pmod{13}$ z deljenjem z e izračunamo še i, ki nam pove, na katerem mestu moramo odšteti e, da odpravimo napako. Prišli smo do [13,11]-kode z informacijsko stopnjo 11/13, ki zna popraviti eno napačno besedo.

Primer: Naj bo $a = (1, 3, 8, 2, 7, 0, 1, 9, 11, 4, 12)$. Potem iz (1) dobimo $6 + c_{11} + c_{12} = 0 \mod 13$ in $2 - 2c_{11} - c_{12} = 0 \mod 13$ oziroma $c_{11} = 8 \mod 13$ in $c_{12} = 12 \mod 13$, torej je $c = (1, 3, 8, 2, 7, 0, 1, 9, 11, 4, 12, 8, 12)$.

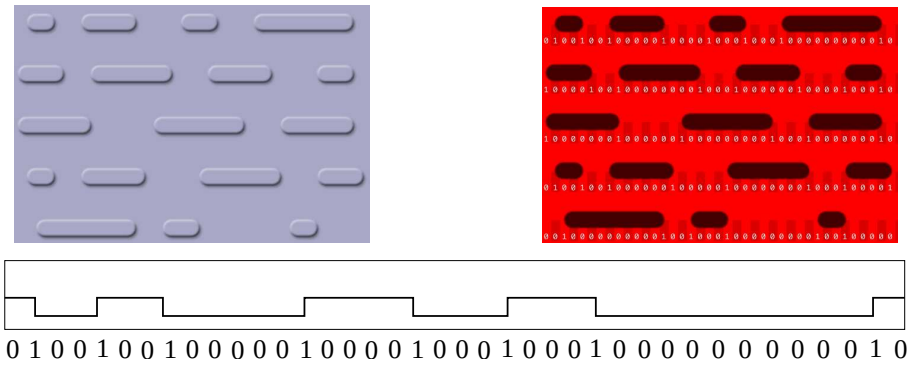
Recimo, da smo dobili $r = (1, 3, 8, 2, 1, 0, 1, 9, 11, 4, 12, 8, 12)$. Iz $r_0 + r_1 + \cdots + r_{12} = 7$ zaključimo, da je $e = 7$, nato pa iz $r_1 + 2r_2 + \cdots + 12r_{12} = 2$ še $2/7 \equiv 4 \pmod{13}$. Slednje deljenje smo seveda izvedli v končnem obsegu, glej [2]. Pravzaprav morate številu 2 v številcu prištevati 13, vse dokler se deljenje ne izide. Torej, ker 2+13=15 še ni deljivo s 7, poskusimo z 2+13+13=28 in dobimo 4, kar pomeni, da je potrebno na mestu z indeksom 4 odšteti napako 7.

Reed-Salomonove kode se da obravnavati veliko bolj splošno in priti tudi do tistih, ki odpravljajo dve in več napak. Vendar pa za to potrebujemo še več matematičnega znanja, ki presega naš okvir, pa tudi računanje v praštevilskem obsegu $\mathbb{Z}_p$ je potrebno nadomestiti z računanjem v končnem obsegu $\mathrm{GF}(2^n)$. Poleg zgoščenk se RS-kode uporabljajo tudi v telekomunikacijah. Ena izmed najpomembnejših uporab je bila kodiranje digitalnih slik, ki sta nam jih na Zemljo pošiljala vesolski sondi Voyager 1 in 2.

Slika 2: (a) Voyagerja sta leta 1979 poslikala Jupiter, leta 1980/81 Saturn, nato pa je Voyager 2 poslikal leta 1981 Uran in leta 1989 Neptun; `http://nssdc.gsfc.nasa.gov/planetary/voyager.html`.
(b) montaža: Jupiter in njegovi Galilejski sateliti, Io, Evropa, Ganimed in Kalisto;
`http://nssdc.gsfc.nasa.gov/photo_gallery/photogallery-jupiter.html`.

Zgoščenke – CD

Zapisovanje glasbe na CD-je je prevzelo ljubitelje glasbe tako rekoč čez noč. Visoko kvaliteto predvajanja zvoka je poleg laserske tehnike v veliki meri pripisati kodam za odpravljanje napak. Oglejmo si način zapisa CD nekoliko pobližje, slika 3.



Slika 3: CD ima premer 12 cm in debelino 1,2 mm, njegova osnova pa je iz prozorne plastike. Na popisanem CD-ju se nahaja spirala, ki se začenja iz notranjosti in je sestavljena iz hribčkov in dolinic. Branje poteka z laserjem (žarek ima premer 1 μm), ki zazna spremembo višine spirale (preko jakosti svetlobe, ki se odbije od diska, odboj svetlobe je ali svetlejši ali temnejši, saj je razlika v višini približno 1/4 valovne dolžine svetlobe v disku). Na ta način dobimo dvojiško zaporedje, vsaka sprememba ustreza številu 1, odsotnost spremembe višine pa ustreza številu 0.

1. Proces razdelimo na faze: digitalno/analogni konverter, kode in modulacija, glej sliko 4.

2. Pri kodiranju bomo na bajte gledali kot na elemente iz obsega GF(2^8), glej [2]. V stereotehniki posnamemo 4 bajte na vsak "tik", ki pomeni 1/44.100-ti del sekunde. Meritev 6-ih zaporednih tikov združimo v sporočilo dolžine 24 bajtov, ki ga v dveh korakih zakodiramo v 32 bajtov. Najprej uporabimo [28,24]-RS kodo, ki zna popraviti dve napaki, in nato še [32,28]-RS kodo, ki tudi zna popraviti dve napaki. Tem 32. bajtom dodamo še 33. bajt, ki je števec pesmi. Le-tega običajno vidimo na zaslonu predvajalnika.

3. Do sedaj so bili vsi omenjeni bajti bodisi nosilci informacije ali pa so bili dodani kot kontrola za odkrivanje in popravljanje napak. Z besedami smo računali za potrebe kodi-

ranja, sedaj pa se zaradi fizikalnih lastnosti materialov spustimo na nivo bitov. Potrebno je paziti, da zaporedne enice niso ne preveč blizu (med njima morata biti vsaj dve ničli) in ne preveč narazen (med njima je lahko največ deset ničel). RS-kode seveda nimajo takih lastnosti, vendar pa ima to lastnost natanko 267 dvojiških besed dolžine 14 (preverjeno z računalnikom). Zato preslikamo 256 elementov končnega obsega s pomočjo dobro izbrane tabele v 256 besed, preostalih 11 besed pa zavržemo. Ta postopek se imenuje EFM (angl. eight to fourteen modulation). Da pa bi zgoraj omenjena lastnost veljala tudi med besedami dolžine 14, mednje postavimo še tri dodatne bite (ničle ali enice; pri tem pazimo še, da sta števili doslej prebranih ničel in enic čim bolj skupaj) in dobimo kodne besede dolžine $33 \cdot 17 = 561$ bitov. Končno, da zabeležimo začetek novega niza, priključimo vsaki kodni besedi še 27 sinhronizacijskih bitov, ki imajo zgornjo lastnost in so izbrani tako, da nikakor ne morejo sestavljati kodirnega podzaporedja. Ena sekunda glasbe se torej pretvori v $(561+27) \cdot 44.100/6 = 4.321.800$ bitov glasbe na spirali.

Slika 4: Med snemanjem izmerimo glasbo 44.100-krat na sekundo, po enkrat na levi in enkrat na desni. Amplitudo zvoka opiše naravno število med 0 in $2^{16}-1$, ki ga predstavlja v dvojiškem sistemu 16 bitov. Ker gre za stereo glasbo, dobimo pri vsaki meritvi 32 bitov (t. i. audio-biti) oziroma 4 bajte podatkov.

Pri branju zgoščenke se lahko pojavi tudi čez 100.000 napak. Le-te lahko povzročajo nezaželeni delci, mikroskopski mehurčki v plastiki, prstni odtisi, praske ali celo manjše luknje v CD-ju ipd. Za primerjavo vzemimo knjigo z 200 stranmi, izpisano v pisavi, ki dopušča 3000 znakov na stran. Recimo, da je tiskalnik le $99,9\%$ zanesljiv. V povprečju lahko torej pričakujemo do 3 napake na stran. Vrnimo se k CD-jem. Če bi bila verjetnost, da predvajalnik prebere napačen bit, enaka 10^{-4}, bi še vedno imeli na stotine napak vsako sekundo. Kvaliteto zvoka seveda izboljšajo kode. Napake se običajno pojavijo v gručah (t. i. grozdne napake). Da zmanjšamo njihov vpliv, je kodiranje narejeno v dveh korakih. Pri tem druga koda kot vhod uporabi nekoliko prepleten izhod prve kode, tako da bajti, ki so sosedni v kodnih besedah (glasbi), niso sosedni tudi na disku. Kodirna shema za CD-je se imenuje CIRC (Cross-Interleaved Reed-Salomon Scheme Code) in pravilno odpravi vse grozdne napake do dolžine 8.871 bitov, kar ustreza približno 2,5 mm spirale na CD-ju.

V članki smo si ogledali le nekaj najbolj osnovnih vidikov kodirnega procesa, praksa pa je seveda še nekoliko bolj zapletena reč.

Viri in dodatno branje

[1] B. Cipra, The Ubiquitous Reed-Solomon Codes, *SIAM News* **26**/1 (1993) (`http://www.siam.org/siamnews/mtc/mtc193.htm`).

[2] A. Jurišić, Računala nove dobe, 2. del, *Presek* **30** (2002-03), str. 291–296.

[3] S. Klavžar, O teoriji kodiranja, linearnih kodah in slikah z Marsa, *Obzornik mat. fiz.* **45** (1998) 4, str. 97–106.

[4] Več avtorjev, For all practical purposes: introduction to contemporary mathematics, 5. izdaja, New York, W. H. Freeman, 2000.

Aleksandar Jurišić