KODIRANJE

Seminarska naloga

Marija Valjavec

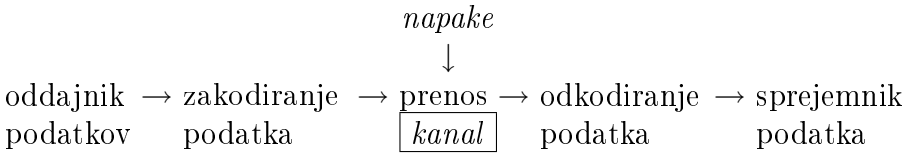
27. oktober 2009

1 Uvod v kodiranje

Predmet teorije kodiranja je prenos podatkov po poteh oz. kanalih, na katerih lahko pride do raznih motenj, zaradi česar pa se originalno sporočilo, ki ga želimo prenesti, pogosto spremeni. Namen teorije kodiranja je zato iskanje učinkovitih načinov zakodiranja originalnega sporočila, tako da napake, ki se pojavijo pri prenosu, lahko zaznanamo ali pa celo popravimo.

Pri prenosih različnih vrst sporočil uporabljamo različne kode, njihove lastnosti pa so odvisne od tega, kakšna je verjetnost, da pri prenosu informacije pride do napak. Kot bomo lahko videli v nadaljevanju, je glavno orodje pri ugotavljanju in popravljanju napak kombinatorika in pa teorija grup.

Spodaj je prikazano, kako poteka proces prenosa informacij.



2 Koda ISBN

Koda ISBN (International Standardized Book Number) je zagotovo najbolj poznana koda dandanes. To je koda oz. število, ki je zapisano v vsaki knjigi, navadno na notranji strani naslovne strani. Vsaka knjiga oz. vsak naslov ima svojo kodo, ki predstavlja njeno ti. mednarodno identiteto.

Vse ISBN kode imajo zelo zanimivo lastnost. Za demonstracijo te lastnosti si pomagajmo s primerom. Vzemimo ISBN knjige Introduction to Coding Theory, ki je 1-58488-421-5. Števke te kode smo napisali navpično v prvi stolpec Tabele 1. Nato smo v drugi stolpec zapisali po vrsti še števila od 1 do 10, tretji stoplec pa predstavlja zmnožke po vrsticah, npr. v drugi vrstici imamo v prvih dveh stolpcih števili 5 in 2, zato v tretji stolpec zapišemo njun produkt $10 = 5 \times 2$. Vse te zmnožke nato seštejemo in vsoto zapišemo na konec tretjega stolpca. V našem primeru dobimo vsoto 242. Presenetljivo je to, da je vsaka tako dobljena vsota deljiva z 11, kar pa seveda ni naključje, tako je ISBN koda namreč sestavljena. (Večkrat je v knjigi zadnje mesto te kode označeno tudi z X, kar pa predstavlja število 10.)

Tabela 1: Posebna lastnost ISBN

ISBN	množitelj	produkt
1	1	1
5	2	10
8	3	24
4	4	16
8	5	40
8	6	48
4	7	28
2	8	16
1	9	9
5	10	50
		$242 = 11 \times 22$

Da deljivost vsote z 11 tudi v našem primeru ni naključje, si poglejmo kaj se zgodi, ko npr. spremenimo zadnjo števko te kode, torej zamenjamo število 5 s 4. V tretjem stolpcu se zaradi te spremembe zadnja vrstica spremeni, in sicer se zmanjša iz 50 na 40, kar vsoto vseh zmnožkov zmanjša za 10. Dobimo torej seštevek 232, ki pa ni več deljiv z 11.

ISBN je torej vedno posebno število, vprašanje pa je 'zakaj?'. Odgovor je preprost. Predstavljajte si, da pri založniku naročate knjigo. Ko v obrazec za naročilo vpisujete kodo ISBN, se lahko zgodi, da se pri tem zmotite; npr.

po nesreči na enem mestu napišete napačno številko. Kaj se tedaj zgodi v tretjem stolpcu prej omenjene Tabele 1. V seštevku produktov pride do spremembe, in sicer takšne, da vsota ni deljiva z 11. Ko torej založnik prejme naročilo, takoj opazi (ponavadi s pomočjo programa, ki to preverja), da koda knjige ni prava oz. da taka knjiga ne obstaja, zato o napaki obvesti kupca.

ISBN je torej primer kode za ugotavljanje napak. Ta koda torej ne le da vsebuje informacije o knjigi (o založniku, naslovu itd.) ampak tudi avtomatično zazna napake: če po nesreči zamenjamo eno od številk, katerikoli prejemnik našega sporočila lahko opazi, da je prišlo do napake.

Kaj pa se zgodi, če v ISBN zamenjamo dve sosednji številki, kar se tudi pogosto zgodi? Napako zaradi različnih množiteljev v drugem stolpcu zlahka zaznamo. Pa recimo, da imamo na četrtem in petem mestu v ISBN števki a in b.

ISBN	množitelj	produkt
$\vdots$	$\vdots$	$\vdots$
a	4	$4a$
b	5	$5b$
$\vdots$	$\vdots$	$\vdots$

Če se nam pripeti, da ti dve števki a in b zamenjamo, dobimo sledečo situacijo.

ISBN	množitelj	produkt
$\vdots$	$\vdots$	$\vdots$
b	4	$4b$
a	5	$5a$
$\vdots$	$\vdots$	$\vdots$

Vsota vseh zmnožkov se ob tem poveča za a ter zmanjša za b, saj namesto $4a$ sedaj dobimo $5a$ ter $4b$ namesto $5b$. Torej, zmnožek se spremeni za a-b, kar pa očitno ne more biti deljivo z 11, razen v primeru, ko je a-b=0. V tem primeru sta a in b enaki števili in njuna zamenjava ne pomeni nobene spremembe kode.

ISBN je učinkovita prav zato, ker temelji na deljivosti z 11. Glavna vsota zmnožkov se zaradi uporabe različnih množiteljev ob zgoraj omenjenih napakah vedno spremeni, tako da ni več deljiva z 11. Poglejmo še, kaj se zgodi, če namesto deljivosti z 11 uporabimo deljivost z 10. Pa denimo, da imamo na petem mestu ISBN število 3.

ISBN	množitelj	produkt
⋮	⋮	⋮
3	5	15
⋮	⋮	⋮

Če število 3 po nesreči zamenjamo s številom 7, se število v trejtem stolpcu spremeni iz 15 na 35, kar pa pomeni, da se skupna vsota spremeni za 20, kar pomeni, da je kljub napaki še vedno deljiva z 10. V tem primeru torej založnik napake niti ne bi opazil. Problem je v tem, da je $10 = 5 \times 2$, tako da če spremenimo to našo ISBN številko za 2,4,6 ali 8, se tretji stolpec spremeni za večkratnik števila 10. Pri praštevilu, kot pa je npr. število 11, pa se to ne more zgoditi.

3 Dvojiški kanali

Ko vesoljske sonde pošiljajo informacije (slike planetov itd.) na Zemljo, jih morajo na nek način zakodirati, kar pa ponavadi storijo tako, da informacijo preoblikujejo v zaporedje ničel in enic. Radijski valovi, ki te informacije prenašajo, morajo na poti do Zemlje prečkati atmosfero, kjer pa naletijo na razne motnje, pri čemer se poslana informacija bolj ali manj spremeni. Kanal preko katerega je bilo sporočilo poslano, torej ni bil popolnoma natančen.

Poznamo več metod, katere lahko uporabimo pri zagotavljanju čimbolj točnega prenosa informacij. Ena je zagotovo ta, da vsako enico zamenjamo z dolgim nizom enic, recimo z desetimi,

$$1111111111$$

in vsako ničlo s prav tako desetimi ničlami

$$0000000000.$$

Ko sedaj zemeljska postaja sprejme zaporedje

$$1101110111$$

vemo, da to ni zaporedje, ki je bilo poslano, temveč zaporedje samih enic, kar je seveda veliko bolj verjetno kot zaporedje samih ničel, saj je le teh po številu veliko manj. Ta postopek zamenjave vsake dvojiške številke (ali bita) z dolgim nizom bitov določa kodo. Kot pri ISBN so tudi tu nekatera sporočila 'nedovoljena', za katere lahko takoj, ko jih prejmemo, povemo, da je prišlo do napake. Kakorkoli že, v primerjavi z ISBN ta koda naredi nekaj več, in sicer ne le da zazna napako, temveč nam da tudi možnost, da jo popravimo.

Pri napačni ISBN lahko založnik prosi kupca, naj ponovno naroči knjigo, nekemu satelitu, ki pa le za kratek čas obišče nek planet, pa ne moremo naročiti, naj ga ponovno slika. Taka koda oz. koda ponavljanja, kot jo lahko tudi poimenujemo, je preprost primer kode za zaznavanje in popravljanje napak.

Problem ponavljalnih kod pa je ta, da so izredno potratne. Namesto, da pošljejo en bit podatka, mora satelit vedno poslati 10 bitov podatkov. To pomeni, da je hitrost oddajanja le 10%. Ko pa vesoljska sonda pošilja podatke iz planetov, pa je zelo pomembno, da je hitrost oddajanja čim večja, saj sonda vsak planet obišče le za kratek čas in mora v tem času prenesti čim več podatkov, pri čemer je njena moč omejena. Torej, če hočemo visok odstotek natančnosti, moramo žrtvovati hitrost prenosa.

Toda leta 1948 je matematik Claude Shannon odkril, da kompromis med hitrostjo in točnostjo prenosa ni neizogiben. Shannon namreč pravi, da če jemo v pravi restavraciji, je kosilo lahko zastonj. Kar je odkril Shannon, je, da je z zelo previdnim načrtovanjem kode možno doseči skoraj popolno pravilnost prenešenega podatka in to še vedno pri konstantni hitrosti oddajanja (ki pa je odvisna le od kvalitete kanala). Če je npr. prenosni kanal le 90% natančen, zanesljiv, potem lahko pri 50% hitrosti oddajanja dosežemo kakršnokoli zanesljivost, če smo seveda oblikovali našo kodo zelo previdno. Shannonovo odkritje je bilo izredno vznemirljivo; težko je namreč verjeti, da lahko pri isti hitrosti oddajanja, dosežemo boljšo natančnost že samo s pravilno izbiro naše kode.

Vendar vseeno obstaja majhna past. Problem z visoko učinkovitimi kodami je ta, da morajo biti zelo zakomplicirane, kar pomeni, da za dekodiranje sporočil, ki jih prejmemo, potrebujemo ogromno časa. Pri vesoljskih misijah to ni težava, saj lahko prejeta sporočila posnamemo in jih kasneje v miru odkodiramo. Po drugi strani pa pri prenosu informacij med ljudmi, prejemnik hoče slišati novico zelo hitro. Zato je zelo pomembno, da izberemo tako kodo, ki ustreza našim potrebam.

Shannonova teorija govori, da učinkovite kode obstajajo, toda ne pove, kako jih oblikujemo. Sploh pa nam ne pove, kako sestaviti kodo, ki je preposta za uporabo. Torej, kmalu po tem Shannonovem izjemnem delu, so se ljudje lotili izumljanja dobrih kod.

4 Dobre kode

Kaj sploh pomeni 'oblikovati kodo'? Kaj vse vse se pri tem upošteva? Torej, ko načrtujemo novo kodo, moramo najprej izbrati 'abecedo', torej odločiti se moramo ali bodo naša sporočila sestavljena iz ničel in enic ali pa iz desetiških števk kot koda ISBN. Pomembno vprašanje pri vsem tem je, kateri nizi simbolov bodo dovoljeni in kateri ne. Npr. pri kodi ponavljanja, ki je bila omenjena v prejšnjem razdelku, je bilo sporočilo razlomljeno v nize desetih dvojiških števk, pri katerih sta bila dovoljena oz. pravilna le dva niza, in sicer 1111111111 in 0000000000. Te nize, na katere je vsako sporočilo razlomljeno, imenujemo 'besede'. Izbrati moramo, kako dolga je vsaka 'beseda' ter se odločiti katere besede so dovoljene in katere ne.

Pa zgradimo eno preprosto kodo za lažjo predstavo. Recimo, da smo se odločili, da oblikujemo neko dvojiško kodo[1] dolžine 3. Vsaka kodna beseda bo torej ena od naslednjih trojic:

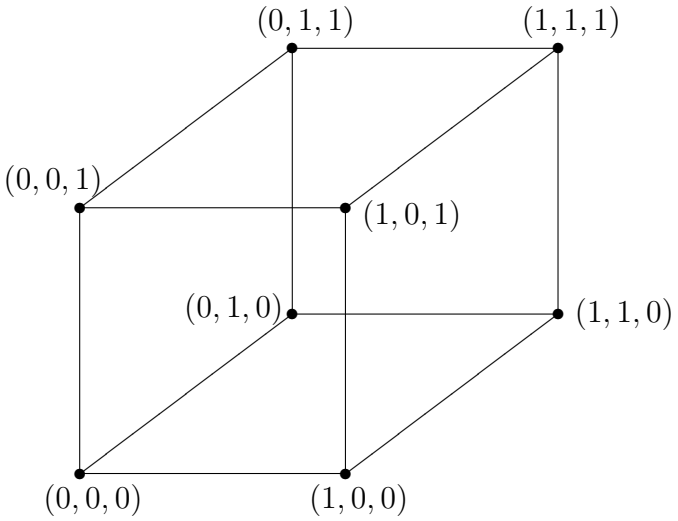
$$000\ 001\ 010\ 011\ 100\ 101\ 110\ 111.$$

Med zgornjimi trojicami moramo sedaj izbrati, katere od njih bodo kodne besede in katere ne. Imamo več načinov, kako te kodne besede izberemo. Če imamo npr. ponavljalno kodo, bi za kodni besedi izbrali le niza 000 in 111, ostale pa bi zavrgli. Vendar mi želimo nadgraditi ponavljalno kodo z nekaj novimi kodnimi besedami, ne da bi s tem izgubili preveč zanesljivosti oz. točnosti prenosa.

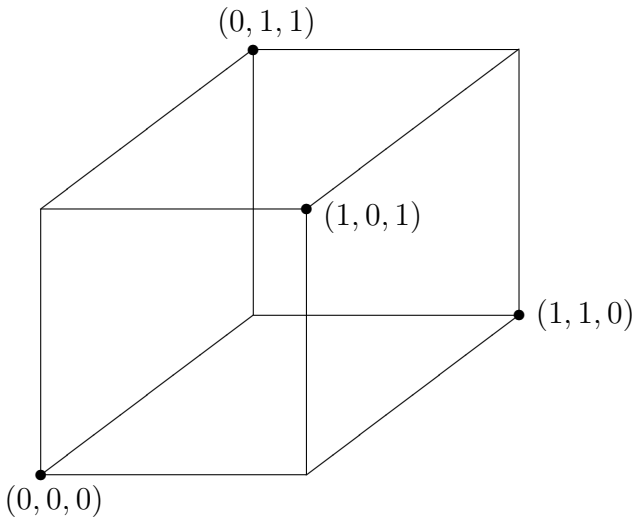
Da obdržimo zanesljivost, moramo izbrati take kodne besede, ki so si med seboj čimbolj različne, tako da je zelo malo verjetno, da bi bila ena kodna beseda ponesreči spremenjena v drugo zaradi nenatančnosti komunikacijskega kanala. Po drugi strani pa bi radi izbrali veliko besed, tako da bi lahko poslali več informacij z uporabo istega števila bitov. To, kar delamo, si lahko boljše predstavljamo, če si zamislimo, da te trojice pomenijo vogale kocke v 3-D prostoru. Npr. 000 postane točka (0,0,0), itd. (Glej Sliko 1.)

Zahtevo, da morajo biti naše kodne besede čimbolj različne, si lahko razlagamo tudi na geometrijski način: vogali kocke, ki smo jih izbrali, morajo biti čimbolj oddaljeni drug od drugega, kar je enako temu, da pridemo od enega vogala do drugega po nekaj korakih in ne direktno. Ponavljalna koda predstavlja dva nasprotna vogala, (0,0,0) in (1,1,1), ki sta drug od drugega oddaljena za tri korake, zaradi česar se koda imenuje tudi 3-oddaljena koda. Kaj še lahko storimo? Ni težko videti, da je možno izbrati tudi štiri vogale, pri čemer nobena dva nista sosedna, temveč je razdalja med vsakim parom vogalov enaka dvema korakoma. Ta primer je podan s Sliko 2. Izbira vogalov v tem primeru ustreza kodnim besedam 000 011 101 110.

[1] koda, kjer uporabimo le števki 0 in 1



Slika 1: Predstava nizov kot točk v 3-D



Slika 2: 2-oddaljena koda

Ta koda,

000 011 101 110,

je 2-oddaljena in zato tudi ni tako zanesljiva kot ponavljalna koda, ima pa dvakrat več kodnih besed. Pa izračunajmo hitrost oddajanja te kode.

Za desetkratno ponavljalno kodo je bila hitrost oddajanja 10%, ker je satelit za pošiljanje vsakega bita informacije moral poslati 10 bitov podatkov. Število bitov informacij je odvisno od števila različnih kodnih besed v naši kodi. Npr. če imamo v kodi štiri kodne besede, imamo možnost, da pošljemo štiri različna sporočila, ravno tako, kot če bi uporabljali vse možne kominacije dveh bitov:

$$00\ 01\ 10\ 11.$$

Štiri kodne besede nas torej oskrbijo z dvema bitoma informacij. Podobno nam 8 kodnih besed prinese 3 bite informacij. V splošnem: če imamo na izbiro 2^m kodnih besed, s tem dobimo m bitov informacij.

V primeru, kjer smo imeli kubično kodo (kodo na kocki), dobimo 2 bita informacij izmed treh bitov podatkov. Hitrost prenosa je torej 2/3: od tod sledi, da je kubična koda 2-oddaljena koda s hitrostjo 2/3. To se dobro primerja z 2-bitno ponavljalno kodo

$$00\ 11,$$

ki je prav tako 2-oddaljena, toda njena hitrost je le 1/2.

Tako smo torej zgradili kodo, ki nadgradi preprosto ponavljalno kodo, z uporabo geometrijske intuicije. Da bi oblikovali res uporabne kode, moramo združiti več različnih tehnik - tako geometrijske kot algebraične. Obstaja več metod, ki so bile v ta namen iznajdene in te kode, ki jih omenjene metode zgradijo, lahko zelo različno uporabimo. Ena od teh metod je opisana v naslednjem razdelku.

5 Preverjanje parnosti

Če pogledamo kodo, ki smo jo našli malo prej,

$$000\ 011\ 101\ 110,$$

lahko opazimo, da obstaja preprost način, kako opisati te kodne besede. Vidimo, da so kodne besede nizi, ki vsebujejo parno število enic. S tem postane očitno, zakaj se vse kodne besede razlikujejo vsaj na dveh mestih. Za kodo, ki jo opazujemo lahko preverimo, če je beseda dovoljena, kar s preverjanjem parnosti vsote števk. Podobna ideja je tudi pri zgradbi ISBN kode. Da bi testirali, ali je zaporedje dovoljeno, to storimo s pomočjo dodatnih števk 1-10 in preverimo ali je vsota deljiva z 11.

Ta ideja preverjanja parnosti se lahko uporabi tudi pri oblikovanju nekaterih drugih uporabnih kod. Pa razložimo, kako se zgradi 3-oddaljena 7 bitna koda, ki ima hitrost $4/7$. Ta je veliko boljša kot 3-oddaljena ponavljalna koda s hitrostjo $1/3$. Kodne besede v tem primeru bodo 7 bitni nizi:

$$b_1,b_2,\ \ldots,b_7$$

Dovoljeni nizi bodo tisti, ki zadoščajo naslednjim trem 'pravilom parnosti':

$$\begin{gathered} b_1 + b_3 + b_5 + b_7 \text{ je sodo,} \\ b_2 + b_3 + b_6 + b_7 \text{ je sodo ter} \\ b_4 + b_5 + b_6 + b_7 \text{ je sodo.} \end{gathered}$$

Trdimo, da je hitrost prenosa te kode $4/7$ - z drugimi besedami, da izmed vsakih sedmih bitov podatkov dobimo 4 bite informacij. To je isto, kot če rečemo, da imamo $2^4 = 16$ kodnih besed. Pa poglejmo, kako to vidimo. Vsak od bitov, označenih z 1, 2 in 4 se pojavi le v enem izmed pravil parnosti. To pomeni, da lahko dovoljene kodne besede sestavimo na sledeč način. Začnemo z izbiro bitov b_3, b_5, b_6 in b_7 kakor želimo. Recimo, da jih izberemo takole:

$$b_3 = 1,\ b_5 = 1,\ b_6 = 0 \text{ in } b_7 = 1.$$

Sedaj izračunamo še b_1, b_2, b_4, in sicer s pomočjo zgornjih enačb - 'pravil parnosti'. Prva enačba pravi, da mora biti vsota $b_1 + b_3 + b_5 + b_7$ soda. Pogledamo vsoto že treh podanih členov b_3, b_5 in b_7 ter vidimo, da je liha, kar pomeni, da mora biti lih tudi člen b_1, torej $b_1 = 1$. S tem je prvi pogoj izpolnjen. Iz druge enačbe na podoben način dobimo, da mora biti člen b_2 enak 0, saj je vsota že samo ostalih členov v enačbi soda. Na koncu vidimo še, da mora biti tudi $b_4 = 0$ in tako dobimo ustrezno kodno besedo

$$1\ 0\ 1\ 0\ 1\ 0\ 1.$$

Pri različnih kombinacijah vrednosti števil oz. bitov b_3, b_5, b_6 in b_7 lahko z isto metodo pridemo še do preostalih kodnih besed. Teh je 16, saj imamo le toliko različnih izbir za zadnje 4 bite (ostali trije so z njihovo izbiro enoličnо določeni). V tem primeru ni težko najti vseh 16-ih; če pa delamo z bolj zakompliciranimi kodami, pa to ni tako preprosto. Te kodne besede so podane tudi v spodnji tabeli.

Tabela 2: 3-oddaljena koda

1	0 0 0 0 0 0 0
2	1 1 1 0 0 0 0
3	1 0 0 1 1 0 0
4	0 1 1 1 1 0 0
5	0 1 0 1 0 1 0
6	1 0 1 1 0 1 0
7	1 1 0 0 1 1 0
8	0 0 1 0 1 1 0
9	1 1 0 1 0 0 1
10	0 0 1 1 0 0 1
11	0 1 0 0 1 0 1
12	1 0 1 0 1 0 1
13	1 0 0 0 0 1 1
14	0 1 1 0 0 1 1
15	0 0 0 1 1 1 1
16	1 1 1 1 1 1 1

Prej smo trdili, da je ta koda 3-oddaljena. To sedaj lahko tudi preverimo v zgornji tabeli kodnih besed: vsaki dve besedi se razlikujeta na najmanj treh mestih. (Obstaja tudi teoretični način, kako to raznolikost preveriti, vendar tega pri tako majhnih kodah večinoma ne uporabljamo.) Ta zgoraj predstavljena 7-bitna koda je primer Hammingove kode[2]. Tudi 3-oddaljena ponavljalna koda

$$\begin{matrix} 0\ 0\ 0 \\ 1\ 1\ 1 \end{matrix}$$

je Hammingova koda. Pravzaprav za vsako število oblike $2^n - 1$ (to so števila 3, 7, 15, 31, 63, 127 itd.) obstaja Hammingova koda te dolžine in vse te so 3-oddaljene.

Ko smo se lotiti iskanja kod, so nas zanimale predvsem tiste kode, ki bi izboljšale natančnost pri prenosu, ne da bi bile preveč zakomplicirane

[2]po matematiku R. W. Hammingu

za uporabo. Hammingove kode, kot tudi druge kode, ki so konstruirane s pomočjo preverjanja parnosti, imajo v tem smislu veliko prednost. Imajo namreč veliko takih struktur, ki jih je lahko odkodirati.

6 Dekodiranje Hammingove kode

Poglejmo, kako odkodiramo Hammingovo kodo iz prejšnjega razdelka. Vsakič, ko prejmemo sporočilo, ga najprej razlomimo na nize dolžine 7 (na 7-bitne nize oz nize sedmih bitov), ki jih nato pregledamo, če predstavljajo kakšno od dovoljenih kodnih besed. Če jo, potem jo sprejmemo, saj predvidevamo, da je bila poslana pravilno. Kaj pa storimo, če pridemo do niza, ki ne predstavlja nobene izmed 16-ih kodnih besed? Takoj torej vemo, da to ni originalen niz, ki je bil poslan, ampak je na poti prišlo do spremembe. Kar bi radi storili je to, da uganemo, katera kodna beseda je bila v originalu poslana. Za vsak pokvarjen niz želimo sedaj najti 'najbližjo' kodno besedo, tj. besedo, ki je najbolj podobna prejetemu nizu.

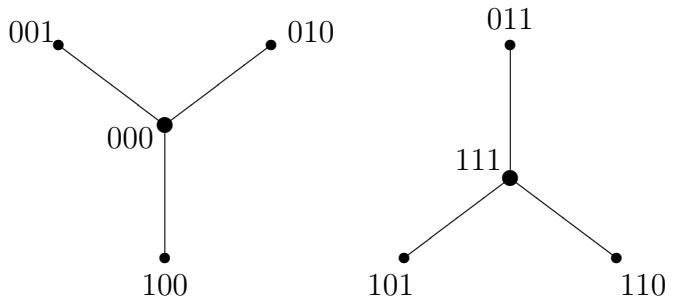
Da bi dobili idejo, kaj storiti, si to najprej poglejmo na preprosti ponavljalni kodi:

$$0\ 0\ 0$$
$$1\ 1\ 1.$$

Če bi namesto teh dveh kodnih besed prejeli eno od naslednjih šestih 3-bitnih nizov

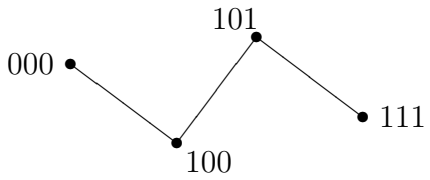
$$0\ 0\ 1$$
$$0\ 1\ 0$$
$$1\ 0\ 0$$
$$0\ 1\ 1$$
$$1\ 0\ 1$$
$$1\ 1\ 0,$$

ne bi imeli nobenih problemov pri ugibanju originalne kodne besede, ki je bila poslana. Če ima niz dve ničli in eno enko, vemo, da je bila poslana kodna beseda 0 0 0; če pa prejmemo niz z eno ničlo in dvema enkama, pa je bil poslan niz 1 1 1. Ta situacija je precej preprosta, saj je vsak niz ali kodna beseda ali pa je le en korak stran od kodne besede, torej se razlikuje le na enem mestu. (Ne more se zgoditi, da bi bil nek niz en korak stran od dveh kodnih besed.) Z drugimi besedami, vsaka kodna beseda je obkrožena z 'oblakom' sosedov, ki so vsi razlikujejo od nje na enem mestu, in ti oblaki sosedov tudi predstavljajo vse možne 3-bitne nize.



Slika 3: 'Oblaka' 3-bitne ponavljalne kode

Da bi torej dekodirali dan niz, najprej najdemo 'oblak', kateremu pripada, in v njegovem 'centru' najdemo kodno besedo, ki jo iščemo. Dejstvo, da je koda 3-oddaljena, v tem procesu igra pomembno vlogo. Če bi namreč imeli niz, ki bi bil soseden dvema različnima kodnima besedama, potem ne bi vedeli, kako ta niz odkodirati. Ker pa imamo kodne besede, ki so med seboj oddaljene za tri korake, se to ne more zgoditi.



Slika 4: 'Oblaka' se ne moreta prekrivati

Sedaj poglejmo, če je mogoče isto stvar narediti še s 7-bitno Hammingovo kodo. Ta koda je tudi 3-oddaljena, tako da ni nobenega dvoma, kateri kodni besedi je soseden nek niz. Problem pa je ta, da se nam zdijo nekateri nizi zelo različni od vseh kodnih besed. Tedaj pa je nesmiselno preverjati vseh 128 besed, da bi ugotovili, katera bi bila danemu nizu najbližja.

Poglejmo si naključen niz

1 1 1 1 0 0 1

in ugotovimo, ali je soseden kateremu izmed kodnih besed v Tabeli 2. Da. Ko pa pregledamo še nekaj drugih naključnih nizov, smo lahko prepričani, da so vsi oddaljeni največ en korak od ene izmed kodnih besed. Kako to na hitro preverimo?

Namesto da gledamo vsak niz, ki ni kodna beseda, in preverjamo, kateri izmed kodnih besedi je soseden, glejmo raje vsako kodno besedo in se vprašajmo, katere so njene sosede. Za primer vzemimo kar najbolj preprosto kodno besedo

$$0\ 0\ 0\ 0\ 0\ 0\ 0.$$

Kateri nizi so od te besede oddaljeni le za en korak? Teh je sedem; dobimo jih z zamenjavo vsakega od sedmih bitov. Tako ima ta beseda okolico, ki poleg sebe vsebuje še 7 drugih nizov:

$$\begin{array}{c} 1\ 0\ 0\ 0\ 0\ 0\ 0 \\ 0\ 1\ 0\ 0\ 0\ 0\ 0 \\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 1\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 1\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 1\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 1. \end{array}$$

Z drugimi besedami, vsaka kodna beseda je center 'oblaka' z osmimi nizi - sebe in sedmimi drugimi, ki niso kodne besede. Obstaja torej 16 'oblakov', eden za vsako kodno besedo. Še več; 'oblaki' se med seboj ne prekrivajo, saj noben od nizov ne more biti soseden dvema različnima kodnima besedama. Ti 'oblaki' ravno pokrijejo vse možne 7-bitne nize, ki jih je $128 = 16 * 8$.

Sedaj torej vemo, da ima 7-bitna Hammingova koda popoln proces dekodiranja. Za vsak pokvarjen niz, ki ga prejmemo, obstaja ena in samo ena kodna beseda, ki je za korak oddaljena od niza. Ta kodna beseda je najboljša, ki jo sploh lahko uganemo glede na originalno. Pravkar omenjeni proces pa nam ne izda nobenega učinkovitega načina za izračun, katera kodna beseda je prava.

Če prejmemo niz

$$1\ 1\ 0\ 1\ 1\ 0\ 1,$$

lahko enostavno preverimo, da to ni kodna beseda, saj ne zadošča prvemu in tretjemu pravilu parnosti. Zelo dolgotrajno bi bilo preveriti vse kodne besede, da bi ugotovili, kateri besedi je dani niz najbližji. Še bolj dolgotrajno pa bi bilo takšno preverjanje, pri še večji kodi, ki bi vsebovala na tisoče dovoljenih besed. Na srečo obstaja bližnjica, ki je odvisna od dejstva, da je koda definirana s pomočjo pravil parnosti.

Za lažje razumevanje te bližnjice zapišimo najprej pravila parnosti v matriko: vsako pravilo ustreza eni vrstici v matriki in vsak položaj bita enemu

stolpcu. Elementi v matriki so 0 ali 1, odvisno od pripadajočih pravil. Npr: na tretjem mestu v drugi vrstici je število 1, saj se b_3 pojavi v drugem pravilu parnosti: b_3 torej ustreza tretjemu stolpcu matrike. Matrika, ki pripada 7-bitni kodi je sledeča.

	b_1	b_2	b_3	b_4	b_5	b_6	b_7
Pravilo 1	1	0	1	0	1	0	1
Pravilo 2	0	1	1	0	0	1	1
Pravilo 3	0	0	0	1	1	1	1

Spomnimo se sedaj spet prejšnjega pokvarjenega niza

$$1\ 1\ 0\ 1\ 1\ 0\ 1.$$

Vemo, da ne zadošča prvemu in tretjemu pravilu parnosti (toda zadošča pa drugemu). Radi bi našli dovoljeno kodno besedo, ki je le korak stran od tega niza. Naš namen je zamenjati le en bit v našem nizu, in sicer tako, da bomo dobili kodno besedo oz. niz, ki zadošča vsem trem pravilom parnosti. Želimo najti stolpec te matrike z lastnostjo, da bo spreminjanje pripadajočega bita spremenilo parnost v prvi in tretji vrstici ne pa tudi v drugi. To pomeni, da iščemo stolpec

$$\begin{matrix} 1 \\ 0 \\ 1. \end{matrix}$$

Vprašanje je le, ali se ta stolpec pojavi v matriki preverjanja parnosti ali ne? Da, se pojavi, in sicer je to peti stolpec. Torej, če zamenjamo peti bit v našem nizu, bomo spremenili parnost prvega in tretjega pravila, ne pa drugega, saj peti bit v drugem sploh ni prisoten. Z zamenjavo petega bita iz 1 na 0 v našem nizu dobimo

$$1\ 1\ 0\ 1\ 0\ 0\ 1,$$

ki pa je zagotovo kodna beseda, in sicer deveta po vrsti v Tabeli 2.

Postopek je pri tem pokvarjenem nizu deloval, toda kako vemo, da bo deloval vedno? Je bilo to le srečno naključje, da se je stolpec

$$\begin{matrix} 1 \\ 0 \\ 1 \end{matrix}$$

pojavil v matriki preverjanja parnosti? Pa poglejmo še enkrat stolpce matrike. Ti so dvojiški prikazi števil 1-7 (napisana navpično navzgor). Za primer, dvojiški prikaz števila 4 je 100 in kot lahko vidimo, četrti stolpec matrike je

0
0
1.

Torej, stolpci matrike določajo vsa možna zaporedja treh bitov razen zaporedja 0 0 0. Katerakoli pravila hočemo spremeniti, lahko to storimo z uporabo pravilnega stolpca.

Naloga 1: Poskusi uporabiti zgoraj opisani proces za dekodiranje niza

0 1 1 1 1 0 1

Rešitev 1:
Niz

0 1 1 1 1 0 1

ne zadošča nobenemu izmed treh pravil parnosti. Sedaj želimo najti kodno besedo, ki je le korak stran od našega niza, torej zamenjati moramo le en bit. Ker pa moramo spremeniti parnost v vseh vrsticah in obenem spremeniti le en bit, to pomeni, da moramo spremeniti sedmi bit, saj ta ustreza stolpcu

1
1
1.

Odtod sedaj dobimo iskano kodno besedo

0 1 1 1 1 0 0.

7-bitna Hammingova koda ima popoln postopek dekodiranja, ki je zelo enostaven za izvedbo. Za vsak pokvarjen niz, ki ga prejmemo, obstaja ena in samo ena kodna beseda, ki je le korak oddaljena od sprejetega niza in le to lahko najdemo zelo hitro s testiranjem našega niza s pravili parnosti. Kot je bilo že omenjeno, obstajajo Hammingove kode dolžine 3, 7, 15, 31, 63, ..., ki so vse 3-oddaljene in jih je lahko dekodirati. Toda v bolj praktične namene potrebujemo kode s še večjo razdaljo (so bolj oddaljene).

Naloga 2: Ali znate zgraditi 7-bitno 4-ločeno kodo, ki ima 8 kodnih besed? Lahko poskusite zapisati 4 pravila parnosti ali pa najdete 8 kodnih besed s poskušanjem. Alternativno, lahko razmislite tudi o 7-bitni Hammingovi kodi in jo razširite na 4-ločeno.

Rešitev 2:
Med šestnajstimi kodnimi besedami Hammingove kode jih ima 8 sodo število

enic ostalih 8 pa liho. Vsaki dve 'sodi' kodni besedi se morata razlikovati v sodem številu mest. Ker je koda 3-oddaljena, vemo, da se vse kode razlikujejo najmanj v treh mestih. Tako se vsaki dve 'sodi' besedi zato razlikujeta na najmanj štirih mestih. Iz Hammingove kode lahko torej izvlečemo 4-oddaljeno kodo, kjer imamo le 'sode' besede:

$$
\begin{array}{c}
0\ 0\ 0\ 0\ 0\ 0\ 0 \\
0\ 1\ 1\ 1\ 1\ 0\ 0 \\
1\ 0\ 1\ 1\ 0\ 1\ 0 \\
1\ 1\ 0\ 0\ 1\ 1\ 0 \\
1\ 1\ 0\ 1\ 0\ 0\ 1 \\
1\ 0\ 1\ 0\ 1\ 0\ 1 \\
0\ 1\ 1\ 0\ 0\ 1\ 1 \\
0\ 0\ 0\ 1\ 1\ 1\ 1.
\end{array}
$$

To kodo lahko opišemo z matriko preverjanja parnosti tako, da vzamemo matriko za Hammingovor kodo in ji dodamo še eno vrstico (prikazana na vrhu matrike), ki preverja ali ima beseda sodo število enic.

b_1	b_2	b_3	b_4	b_5	b_6	b_7
1	*1*	*1*	*1*	*1*	*1*	*1*
1	*0*	*1*	*0*	*1*	*0*	*1*
0	*1*	*1*	*0*	*0*	*1*	*1*
0	*0*	*0*	*1*	*1*	*1*	*1*

Shannonovo odkritje zastonjskega kosila je vodilo v lov za dobrimi kodami, ki bi jih lahko uporabili za hitro in točno pošiljanje podatkov. Njegovo delo pa je vodilo tudi v razvoj nove podveje v matematiki: teorije informacij.

Literatura

[1] Keith Ball, *Strange Curves, Counting Rabbits, and Other Mathematical Explorations*. Princeton University Press 2003.