

UNIVERZITET SINGIDUNUM



Milan Milosavljević
Saša Adamović

OSNOVI TEORIJE INFORMACIJA I KODOVANJA

Beograd, 2017.

UNIVERZITET SINGIDUNUM

Fakultet za informatiku i računarstvo

Milan Milosavljević

Saša Adamović

OSNOVI TEORIJE INFORMACIJA I KODOVANJA

Treće izdanje

Beograd, 2017.

OSNOVI TEORIJE INFORMACIJA I KODOVANJA

Autori:

dr Milan Milosavljević
dr Saša Adamović

Recenzenti:

dr Mladen Veinović
dr Branko Kovačević
dr Dejan Živković

Izdavač:

UNIVERZITET SINGIDUNUM
Beograd, Danijelova 32
www.singidunum.ac.rs

Za izdavača:

dr Milovan Stanišić

Tehnička obrada:

Saša Adamović

Dizajn korica:

Aleksandar Mihajlović

Godina izdanja:

2017.

Tiraž:

500 primeraka

Štampa:

Mobid, Loznica

ISBN: 978-86-7912-610-8

Copyright:

© 2017. Univerzitet Singidunum

Izdavač zadržava sva prava.

Reprodukcija pojedinih delova ili celine ove publikacije nije dozvoljeno.

*„Dobar naučnik je onaj ko misli kao i svi
drugi, a vidi nešto što niko drugi nije
video. Genije je onaj ko vidi isto što i svi
drugi, a misli nešto što niko drugi nije
pomislio“*

Albert Sent-Djerdji

SADRŽAJ

1.	UVOD.....	10
1.1.	Pojam informacije.....	11
1.2.	Informacijska hijerarhija	12
1.3.	Informaciono društvo.....	14
1.4.	Teorija informacija kao akademska disciplina	16
2.	ENTROPIJA I INFORMACIJA.....	19
2.1.	Entropija – izbori mogućnosti sa poznatim verovatnoćama	27
2.1.1.	Svojstva entropije.....	29
2.1.2.	Važna svojstva entropije.....	32
2.1.3.	Karakterizacija entropije.....	33
2.1.4.	Slučajne veličine - podsetnik	33
2.1.5.	Uslovne entropije	35
2.1.6.	Uzajamna entropija.....	36
2.2.	Informacija i njena mera	44
2.2.1.	Opservacije i događaji.....	44
2.2.2.	Uzajamna informacija.....	46
2.2.3.	Grafička prezentacija informaciono-teorijskih veličina	48
2.2.4.	Relativna entropija.....	51
2.3.	Šenonova matematička teorija informacije	53
2.3.1.	Modelovanje izvora	54
2.3.2.	Šenonovi modeli aproksimacije.....	54
2.3.3.	Entropija izvora u odnosu na aproksimaciju modela.....	56
2.3.4.	Šenonova teorema za kodovanje bez gubitaka	57
2.4.	Laboratorijski rad.....	60

3.	EFIKASNO KODOVANJE INFORMACIJA.....	72
3.1.	Kodovanje jedne slučajne veličine.....	73
3.1.1.	N-arna stabla kodova.....	77
3.1.2.	Kraftova nejednakost.....	80
3.2.	Efikasno Kodovanje.....	83
3.2.1.	Verovatnosno N-arno stablo.....	83
3.2.2.	Šenonova teorema o kodovanju bez prisustva šuma	87
3.2.3.	Hafmanov kod	91
3.2.4.	Optimalnost Hafmanovog koda	94
3.3.	Laboratorijski rad.....	96
4.	KANALI ZA PRENOS INFORMACIJA.....	99
4.1.	Komunikacioni kanali	100
4.1.1.	Kapacitet kanala.....	104
4.1.2.	Kanali sa simetričnim ulazom	105
4.1.3.	Kanali sa simetričnim izlazom	106
4.1.4.	Brzina prenosa.....	107
4.2.	Kodovanje u prisustvu šuma	109
4.2.1.	Kod ponavljanja.....	109
4.2.2.	Teorema o kodovanju u prisustvu šuma	110
5.	ZAŠTITNI KODOVI – KODOVI ZA ISPRAVLJANJE GREŠAKA	114
5.1.	Osnove kodova za ispravljanje grešaka	115
5.1.1.	Hemingovo rastojanje i težine kodnih reči	116
5.1.2.	Dekodovanje minimalnog rastojanja i maksimalne verodostojnosti.....	118
5.1.3.	Detekcija i ispravljanje grešaka	119
5.2.	Linearni kodovi.....	122
5.2.1.	Kodovanje linearnim kodom	124

5.2.2.	Sistemska forma linearnih kodova.....	126
5.2.3.	Dekodovanje: verifikaciona matrica.....	127
5.2.4.	Dualni kodovi.....	130
5.2.5.	Sindromi.....	131
5.2.6.	Minimalno rastojanje i verifikaciona matrica.....	134
5.2.7.	Binarni Hemingovi kodovi.....	135
5.3.	Ciklični kodovi.....	139
5.3.1.	Ciklični kodovi i polinomi.....	140
5.3.2.	Dekodovanje cikličnih kodova.....	144
5.4.	Konvolucionni kodovi.....	148
5.4.1.	Kodovanje konvolucionim kodovima.....	149
5.4.2.	Opšte definicije konvolucionih kodova.....	151
5.4.3.	Predstava preko rešetki.....	153
5.4.4.	Dekodovanje.....	156
5.5.	Kodovi niske gustine – LDPC.....	162
5.5.1.	Reprezentacija LDPC kodova.....	163
5.5.2.	Regularni i iregularni LDPC kodovi.....	164
5.5.3.	Performanse i kompleksnost LDPC kodova.....	165
5.5.4.	LDPC algoritmi dekodovanja.....	166
5.5.5.	LDPC algoritam kodovanja.....	170
5.6.	Praktična primena zaštitnih kodova.....	171
5.6.1.	ARQ.....	171
5.6.2.	FEC.....	173
5.6.3.	Hibridni ARQ.....	174
5.6.4.	Interliveri.....	175
5.6.5.	Dekodovanje sa brisanjem.....	177
5.6.6.	Konkatenacija.....	178
5.6.7.	Jedan primer kodovanja u CD.....	179
5.7.	Laboratorijski rad.....	182

6. KRIPTOGRAFIJA – INFORMACIONO TEORIJSKI UVOD	198
6.1. Perfektna tajnost.....	200
6.1.1. Perfektna šifra: definicija	201
6.1.2. Grafička prezentacija perfektne šifre	203
6.1.3. Dokaz Šenonove teoreme o perfektnoj tajnosti - OTP	204
6.2. Neperfektna tajnost i tačka jedinstvenosti	205
6.2.1. Ekvivokacija ključa i tačka jedinstvenosti.....	205
6.2.2. Povećanje tačke jedinstvenosti.....	208
6.3. Autentifikacija.....	209
BIBLIOGRAFIJA.....	211
DODATAK A – UVOD U EKSPERIMENTALNO OKRUŽENJE	212
DODATAK B – JEDNA PRIMENA KODOVA ZA ISPRAVLJANJE GREŠAKA U KRIPTOGRAFIJI.....	224
REČNIK POJMOVA	233

Predgovor

Osnove teorije informacija i kodovanja – je udžbenik koji je namenjen studentima Fakulteta za informatiku i računarstvo Univerziteta Singidunum za pripremu ispita iz predmeta „Osnovi teorije informacija i kodovanja“, a može se koristiti i za savladavanje gradiva iz predmeta u kojima informacija igra ključnu ulogu, kao što su Kriptologija 1, Kriptologija2, Veštačka inteligencija, i uopšte u svim onim stručnim predmetima u kojima memorisanje, prenos i obrada podataka zahteva kvantitativno modelovanje i analizu informacionih tokova.

U samom nazivu oblasti pojavljuje se termin Informatičar, dakle neko ko se služi informatičkim tehnologijama u cilju obrade, memorisanja i prenosa svih vidova informacionih sadržaja u okviru savremenih računarsko komunikacionih sistema. Već u samoj definiciji informatičara susrećemo tri pojma – obrada, memorisanje i prenos informacija (informacionih sadržaja).

Zamislimo da ste prvi dan na poslu u filijali jedne velike kompanije. Posao ste dobili nakon mnogo napora i rešeni ste da date sve od sebe. Šef vam zadaje sledeći zadatak: Napišite program za saldiranje dnevnih rezultata poslovanja filijale, i dostavite izveštaj centrali kompanije u što kraćem roku.

Zadatak br. 1: Pisanje programa.

Ako želim da ostavim snažan utisak moj program mora biti najbolji, a to znači da korektno radi i da je pri tome najkraći mogući (najefikasniji).

Zadatak br. 2: Obrada podataka.

Ako želim da ostavim snažan utisak morao bih da znam koji podaci su bitni ili bitniji za moj račun, a koji možda uopšte nisu bitni. Time bih ukazao kompaniji na nepotrebno prikupljanje nekih podataka.

Zadatak br. 3: Slanje podataka u centralu kompanije.

Ako želim da ostavim snažan utisak morao bih da znam koji je najefikasniji način dostave, odnosno kako da u što kraćem roku pošaljem podatke koji će biti primljeni bez greške. Moja bi se pozicija još više učvrstila ako bih znao da kažem da li podatke treba zaštititi i ako treba, kojim metodama iz skupa bezbednosnih servisa. Zasigurno bih postao zvezda kompanije ako bih na zahtev šefa umeo da podatke pošaljem apsolutno tajno (dakle da primenjenu šifru niko ne može ni sa kakvim računarskim resursima nikada razbiti) i da pri tome garantujem apsolutnu tajnost.

Promocija bi mi bila osigurana ako bih znao da generišem veliku količinu šifarskih ključeva i da pri tome poznajem metode za njihovu sigurnu distribuciju. Navedimo šta je neophodno da znam da bih sve ove zadatke uspešno obavio:

- *Šta je informacija, kako se meri?*
- *Šta je algoritamska složenost i kako se meri?*
- *Šta su granice kompresije jednog skupa podataka ispod koje se ne može ići bez gubitka dela informacije?*
- *Šta je uzajamna informacija između dva skupa podataka, odnosno koliko jedan skup podataka sadrži u sebi informacije o onom drugom?*
- *Šta je redundansa (suvišnost) jednog izvora informacija i kako se ona može iskoristiti?*
- *Kako mogu da konstruišem transformaciju podataka koje šaljem ili memorišem, tako da na prijemu ili prilikom iščitavanja mogu da ispravljam greške u prenosu ili prilikom memorisanja?*

Nakon izučavanja ovog udžbenika, čitalac će dobiti osnovna znanja, koja mu omogućavaju da sa razumevanjem reši ove zadatke. Stoga materijal izložen u okviru Osnova teorije informacija i kodovanja, ne obuhvata samo teorijsko razumevanje informacionih procesa koje susrećemo u svakom računarsko komunikacionom sistemu, već predstavlja i snažan praktični aparat koji savremenom informatičaru olakšava rešavanje problema koje mu nameće sve složenija i zahtevnija praksa.

Udžbenik se sastoji od 7 poglavlja uključujući uvod i bibliografiju korišćene literature. Osim toga radi kompletnosti, na kraju su i dva dodatka sa rečnikom osnovnih pojmova. Prvi dodatak se odnosi na kratak uvod u GNU Octave programsko okruženje, kao programsko okruženje u okviru koga se rade vežbe iz svake od izučavanih tematskih celina. Drugi dodatak se odnosi na primenu teorije informacija u modelovanju i analizi šifarskih sistema. Čitalac će kroz ovaj dodatak steći utisak o moći i adekvatnosti teorije informacija u opisivanju i analizi svih ključnih fenomena bitnih za razumevanje bezbednosnih servisa jednog šifarskog sistema. Od posebne važnosti je demonstracija efikasnosti ove teorije u formulisaju kriterijuma apsolutne tajnosti šifarskog sistema, što je od ključne važnosti kako za razumevanje mehanizama zaštite uopšte, tako i za dobijanje jasnih uputstava prilikom sinteze ovih sistema optimalnih performansi.

Savremeni razvoj bezbednosnih servisa na Internetu, pokazuje da je informaciono teorijski pristup, izložen u ovom udžbeniku, upravo na frontu najnovijih pravaca razvoja, koji već danas mogu obezbediti sisteme zaštite, kontrolisane i izmerljive snage. Time je potpuno osiguran bezbedan sadašnji i svaki budući razvoj Interenta, kao opšte računarsko komunikacione infrastrukture, ključne za razvoj digitalne ekonomije i nove globalne digitalne civilizacije.

Na kraju, zahvaljujemo se recenzentima na korisnim sugestijama i primedbama koje smo uvažili i inkorporirali u udžbenik.

Beograd, 2013. godine

Autori

1. UVOD

U teoriji informacija centralni pojam zauzima pojam informacije kao fenomen i komunikacija kao proces. Intuitivna predstava pojma informacije je gotovo svima nama bliska, ali naučna zasnovanost i interpretacija informacije utemeljena je tek polovinom prošlog veka rađanjem nove naučne discipline Teorije informacija.

Informacija je postala relevantan pojam za sve nauke koje se bave simboličkom komunikacijom u rasponu od matematike do računarske nauke, ili od logike do lingvistike, odnosno od elektronike do bibliotekarstva, kao i od humanističkih nauka i umetnosti do dokumentalistike, ali i od društvenih nauka do medicine i dr.

To je informaciji dalo interdisciplinarnu dimenziju, jer je svaka nauka pokušala i još pokušava da protumači taj kompleksan pojam. Sve ovo ukazuje da pojam informacija nije lako shvatiti niti jednostavno protumačiti. Informacija je reč latinskog porekla „*in formare*“ i izvorno je značila stavljanje u određenu formu, odnosno, davanje oblika nečemu, ali je tokom vremena izgubila prvobitno značenje.

1.1. POJAM INFORMACIJE

Danas nema jedinstvenog pristupa fenomenu informacije, a još manje jednoznačne i opšte prihvaćene definicije informacije. Navešćemo nekoliko definicija informacije:

- „Ljudi su tvorci i nosioci informacija”,
- „Informacija mora, pre svega, da ima određeni smisao, ona mora biti nosilac značenja”,
- „Informacija je sve ono što daje nove podatke, ili nova obaveštenja o nekoj činjenici ili nekom događaju, koji nisu bili ranije poznati”,
- „Informacije su oni delovi vesti koji za primaoca imaju vrednost novosti i koji mu omogućavaju da bolje izvrši svoje zadatke”,
- „Informacije su saznanja koja postaju dostupna pomoću sredstava komunikacija, a poseduju obaveštajnu vrednosti”,
- „... sadržaj onog što razmenjujemo sa spoljnim svetom dok mu se prilagođavamo i dok utičemo na njega”,
- „... svesna ili ciljna organizovanost podataka”...

U univerzitetskoj literaturi najčešće definicije pojma informacija su :

- „Informacija je inkrement znanja”, njenim postojanjem je naše znanje o nečemu uvećano
- „Informacija je značenje koje dodeljujemo podatku”
- „Informacija je skup podataka u nekom kontekstu,,
- „Informacija je primljena i shvaćena poruka”

Važno je napraviti razliku između podatka i informacije, pojmova koji se često poistovećuju. Na primer broj 17 je podatak i on kao takav nema posebno značenje. „sada je 17 časova” je informacija jer je podatku dodeljeno neko značenje. Tako možemo uvideti da se informacija sastoji od podatka i značenja koje mu je dodeljeno.

1.2. INFORMACIJSKA HIJERARHIJA

PIZM – Podaci, Informacija, Znanje, Mudrost (engl. *Data, Information, Knowledge, Wisdom*) je predlog strukturiranja podataka, informacija, znanja i mudrosti u jednu informacijsku hijerarhiju gde svaki nivo dodaje određena svojstva iznad i ispod onog prethodnog.

- Podatak je najosnovniji nivo,
- Informacija dodaje kontekst,
- Znanje dodaje kako ga upotrebiti,
- Mudrost dodaje kada i zašto ga upotrebiti.

Na slici (Slika 1.1) prikazana je hijerarhijska piramida.



Slika 1.1 Hijerarhijski model PIZM

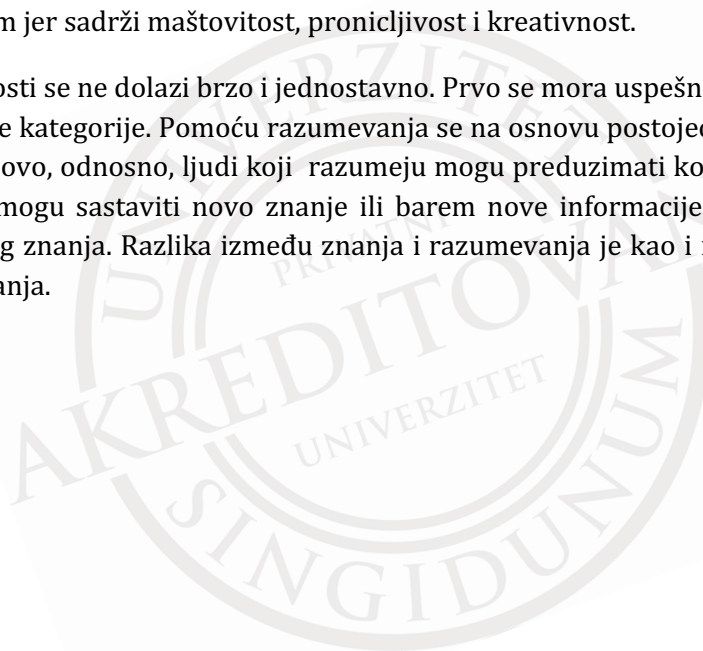
PIZM model se temelji na nizu postupaka:

- **podatak** dolazi u obliku neobrađenih zapažanja i dimenzija,
- **informacija** se oblikuje analiziranjem veza i odnosa između podataka,
- **znanje** se oblikuje koristeći informaciju za delovanje,

- **mudrost** se oblikuje kroz upotrebu znanja, kroz komunikaciju korisnika znanja i kroz razmišljanja. Opšteprihvaćeno je mišljenje da je podatak manji od informacije, a informacija je manja od znanja, odnosno, da bi se kreirala informacija potreban je podatak i samo kada postoji informacija znanje može izaći na videlo.

Sadržaj ljudskog uma prema Raselu Akofu (Russellu Ackoffu) čine: podatak, informacija, znanje, razumevanje (saznajni i analitički proces koji odgovara na pitanje zašto) i mudrost. Prve četiri kategorije odnose na prošlost i bave se onim što je bilo ili onim što je poznato. Samo se peta kategorija, mudrost, bavi budućim vremenom jer sadrži maštovitost, pronicljivost i kreativnost.

Do mudrosti se ne dolazi brzo i jednostavno. Prvo se mora uspešno proći kroz sve prethodne kategorije. Pomoću razumevanja se na osnovu postojećih znanja može stvarati novo, odnosno, ljudi koji razumeju mogu preduzimati korisne aktivnosti zato što mogu sastaviti novo znanje ili barem nove informacije na osnovu već postojećeg znanja. Razlika između znanja i razumevanja je kao i između učenja i memorisanja.



1.3. INFORMACIONO DRUŠTVO

Znanje je rezultat učenja. Povećanje nivoa znanja dovodi do boljeg odlučivanja. Nove informacione tehnologije dovode do ubrzanja i sve većih količina podataka i informacija.

Sveprisutnost informacija dovela je do toga da savremeno društvo već označavamo kao informaciono društvo. Razlog za to ne leži samo u sveprisutnosti, već i ključnom značaju informacija u funkcionisanju gotovo svih ljudskih delatnosti.

Danas već imamo informacionu teoriju sistema, informacionu teoriju odlučivanja, informacionu ekonomiju, informacionu filozofiju, informacionu teoriju svega u okviru teorijske fizike, informacionu medicinu i tako redom.

Eksperimentalno je potvrđen revolucionarni novi pristup lečenju, po kome se uzročnicima bolesti, npr. virusima šalje informacija da izvrše sopstveno samouništenje.

Razvoj tehničkih sistema za prenos i memorisanje informacionih sadržaja, kao što su računari, internet i savremeni digitalni telekomunikacioni sistemi, nisu na adekvatan način praćeni teorijskim rezultatima i relevantnim studijama fenomena informacija, kao i sistema za njeno efikasno kodovanje, prenos, memorisanje i zaštitu.

Savremena ekonomija pomera svoje težište ka informacionoj ekonomiji. Šta to zapravo znači? U klasičnoj ekonomiji cenu robe čine komponente kao što su utrošene sirovine i energija, ljudski rad, amortizacija sredstava za proizvodnju i profit.

Razvoj sredstava za proizvodnju ide ka tome da će osnovnu komponentu u ovoj ceni sve više da preuzima informacija kako nešto napraviti, a ne neposredna materijalna osnova proizvoda (rad, sirovine, energija).

Pojavom fuzionih reaktora u kojima se može sintetizovati bilo koja sirovina uz utrošak energije i sirovina zanemarljive cene, dovodi nas u do sada nezamislivi

scenario po kome će cena jednog proizvoda biti jednaka ceni informacije potrebne za pravljenje programa koji upravlja sintezom u fuzionom reaktoru.

Ovo je samo jedan primer koji potvrđuje već postojeće stanje stvari po kome je informacija izborila status resursa, istog kao što je u klasičnoj privrednoj delatnosti to sirovina, energija ili ljudski rad.

Ako informacija postaje dominantni resurs privredne delatnosti, onda se tim resursom mora pažljivo rukovati.

Stoga prenos, memorisanje i zaštita informacija dolaze na prvo mesto, kao kritične tehnologije kojima se obezbeđuje stabilan razvoj i u krajnjoj liniji opstanak jedne društvene zajednice.

Upravo teorija informacija daje teorijske i praktične okvire za sintezu sistema koji obezbeđuju efikasnost i raspoloživost informacionih resursa, shvaćenih na gore pomenuti način.

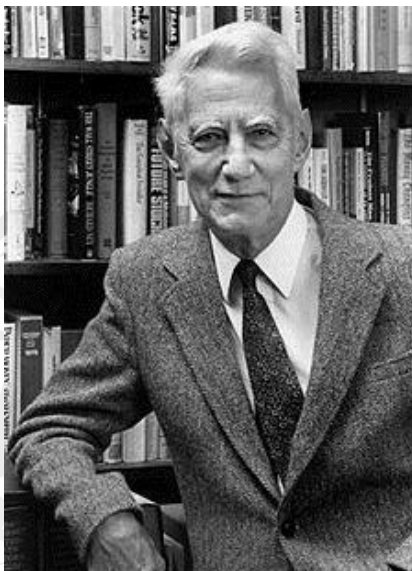
Teorija informacija u tehničkom smislu, kako se danas razume i tumači, vodi poreklo od radova Kloda Šenona objavljenih neposredno nakon drugog svetskog rata.

Šenon je definisao ovu disciplinu kao teoriju za rešavanje problema prenosa signala preko komunikacionih kanala.

Iako je ovakav pristup danas suviše uzak, pogotovu ako se ima u vidu merenje informacionog (semantičkog) sadržaja poruka koje se memorišu ili prenose, Šenonov pristup predstavlja i dalje centralno jezgro ove teorije.

1.4. TEORIJA INFORMACIJA KAO AKADEMSKA DISCIPLINA

Teoriju informacija je zasnovao Klod Šenon (*Claude Shannon*) (1916-2001). Šenon je radio u istraživačkom centru *Bell Laboratories* (1941-1956), a kasnije u *Massachusetts Institute of Technology* (1956-1978).



Slika 1.2 Klod Šenon (1916-2001)

Šenon je tražio odgovor na dva fundamentalna pitanja teorije komunikacija:

1. Do koje mere se neki skup podataka može komprimovati (odgovor: entropija izvora H)?
2. Šta je krajnja brzina komuniciranja za zadati komunikacioni kanal (odgovor: kapacitet kanala C)?

Odnos teorije informacija i telekomunikacija - Elektrotehnika (Telekomunikacije). Do Šenonovih radova smatralo se da je nemoguće slati informacije pozitivnom brzinom sa zanemarljivom verovatnoćom greške. Šenon je u svom fundamentalnom radu iz 1948. godine iznenadio celokupnu naučnu zajednicu iz domena telekomunikacija pokazavši da je grešku prenosa moguće smanjiti proizvoljno za sve brzine prenosa ispod kapaciteta kanala.

Šenonova dva fundamentalna rada su:

- Klod Šenon (C. E. Shannon), „*A mathematical theory of communication*“, *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, July and October, 1948.
- Klod Šenon (1949). „*Communication Theory of Secrecy Systems*“. *Bell System Technical Journal*, 28 (4): 656–715.

Šenon je dalje pokazao da informacioni izvori kao što su muzika ili govor poseduju kompleksnost ispod koje se signali ovih izvora ne mogu dalje komprimovati bez gubitaka. Ovu kompleksnost je nazvao entropija i razlikovao je od pojma entropije uvedenog mnogo ranije u oblasti termodinamike. Dakle, ako je entropija izvora manja od kapaciteta kanala, moguće je dizajnirati komunikacionu šemu koja je asimptotski bez greške prenosa.

Pojava turbo kodova približava nas Šenonovoj granici (dakle komunikaciji sa maksimalno mogućom brzinom-u blizini kapaciteta kanala, sa nultom greškom prenosa). Front oblasti je danas dominantno tzv. (engl. *Network coding*) mrežno kodovanje, odnosno teorija simultanog komuniciranja više čvorova u jednoj mreži različitim brzinama u prisustvu šumova i interferencija.

Odnos teorije informacija i računarske nauke - Kolmogorov, Čaitin i Solomonov su razvili ideju da je kompleksnost jednog niza jednaka dužini najkraćeg računarskog programa koji generiše date podatke. Kompleksnost je dakle jednaka dužini minimalnog opisa (deskriptivna kompleksnost). Pokazalo se da je ideja Kolmogorovljeve kompleksnosti univerzalna i nezavisna od konkretnog računara.

Kolmogorovljeva kompleksnost K je približno jednaka Šenonovoj entropiji H , ukoliko je posmatrana sekvenca za koju se ona računa izvučena slučajno u skladu sa nekom raspodelom verovatnoća koja ima entropiju H . Stoga je veza između Šenonove teorije informacija i Kolmogorovljeve kompleksnosti gotovo perfektna.

Odnos teorije informacija i fizike - Statistička mehanika je mesto rođenja entropije i drugog zakona termodinamike. Entropija uvek raste. Drugi zakon termodinamike daje teorijsku osnovu za odbacivanje svake tvrdnje o egzistenciji perpetumobila.

Odnos teorije informacija i matematike - Fundamentalne veličine teorije informacija: entropija, relativna entropija, uzajamna informacija su definisane kao funkcionali raspodela verovatnoća. Sa svoje strane ove veličine dobro opisuju ponašanje dugačkih sekvenci slučajnih veličina i omogućavaju nam procenu verovatnoća retkih događaja (teorija velikih devijacija), kao i ocene eksponenta greške (engl. *error exponent*) pri statističkom testiranju hipoteza.

Odnos teorije informacija i filozofskih nauka - Okamov sekač: Najjednostavnije objašnjenje je najbolje. Solomon i Čaitin su ubedljivi u dokazu da je univerzalno najbolja procedura predikcije ona koja uzima težinsku sumu predikcija svih programa koji dobro objašnjavaju podatke. Naravno da je ova univerzalna procedura nepraktična, budući da je pronalaženje svih programa koji mogu da objasne date podatke vremenski zahtevan postupak.

Odnos teorije informacija i ekonomije - Repetitivne investicije u stacionarno berzansko tržište rezultuje u eksponencijalnom rastu dobitka. Rast dobitka je dualan entropiji berzanskog tržišta. Postoji jednoznačna veza između teorije optimalnog investiranja u berzansko tržište i teorije informacija.

Odnos teorije informacija, računarstva i komunikacija - Pravljenjem velikih računara na osnovu malih komponenti dovodi nas i do računarskih i do komunikacionih limita. Računarstvo je limitirano komunikacijom, dok su komunikacije računarski limitirane. Stoga su svi rezultati teorije informacija u domenu komunikacija od direktnog uticaja i na računarstvo.

2. ENTROPIJA I INFORMACIJA

Neku situaciju ćemo označiti kao neodređenost ukoliko postoji više mogućnosti, pri čemu nije poznato koja će od njih biti realizovana ili izabrana. Zapravo realne situacije potpune određenosti u kojima se ishodi mogu predviđati sa sigurnošću su više izuzetak nego pravilo.

Uvedimo na formalniji način pojam neodređenih situacija, koji je početni gradivni blok savremene teorije informacija. Pretpostavimo neku situaciju S u kojoj je moguće m različitih ishoda. Označimo ove mogućnosti sa $e_1, e_2, \dots, e_m$. Ovaj skup mogućnosti ćemo označiti sa $S(e_1, e_2, \dots, e_m)$, i nazvati konačnom šemom izbora. Po nekom mehanizmu ili na osnovu nekog procesa ili na osnovu nečije volje vrši se izbor jedne od ovih mogućnosti. Neodređenost asocirana zadatoj šemi izbora S , nastaje usled našeg neznanja ili nemogućnosti da znamo koja će od m alternativa biti izabrana.

Na koji način možemo izmeriti količinu neodređenosti u datoj šemi izbora S ?

Intuitivno predosećamo da što je veća kardinalnost $|S|$ skupa S , odnosno broj elementa ovog skupa, veća je i neodređenost. Zašto onda ne bi smo jednostavno uzeli $|S|$ kao meru neodređenosti? Iako je to u principu moguće, jedan drugi pristup će se pokazati delotvornijim.

Predstavimo sebi sledeću igru: zamisli se jedna od mogućnosti iz skupa $S(e_1, e_2, \dots, e_m)$, a zatim se na osnovu pitanja koja su tako formulisana da se na njih odgovara samo sa **da/ne**, pokušava pogoditi zamišljena vrednost.

Mogućnosti uvek možemo označiti redom brojevima $1, 2, \dots, m = |S|$, tako da pitanja mogu biti tipično sledećeg oblika:

Da li je zamišljeni broj paran?

Da li je zamišljeni broj manji od 20?

Da li je zamišljeni broj veći od 12?

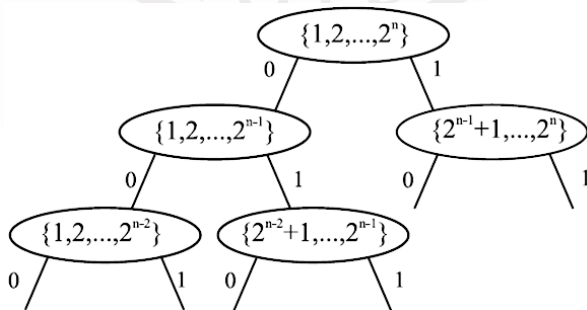
i td.

Odmah zapažamo da ukoliko nam je potrebno više pitanja, neodređenost šeme izbora S je veća. Stoga je osnovano smatrati da je broj pitanja tipa **da/ne** neophodnih za određivanje nepoznate zamišljene mogućnosti iz skupa S , dobra mera neodređenosti šeme izbora S .

Od važnosti je i kakva pitanja postavljamo. Ako su pitanja tipa da li je zamišljeni broj jednak 1, pa ako nije da li je jednak 2, pa ako nije da li je jednak 3, jasno je da nas to vodi ka postavljanju svih m mogućih pitanja. Ovo sigurno nije optimalan način dobijanja tačnog odgovora u najmanjem broju koraka. Međutim, ako je prvo pitanje da li je zamišljeni broj manji od $m/2$, u narednom pitanju sužavamo pretragu na polovinu početnih mogućnosti. Ako i u narednim koracima nastavimo da postavljamo pitanja na ovaj način, pokazaće se da je izabrana strategija vrlo efikasna.

U cilju detaljnijeg upoznavanja sa ovim postupkom, pretpostavimo prvo da je ukupan broj početnih mogućnosti stepen broja 2, odnosno da važi $m = 2^n$. Tada možemo podeliti S postavljanjem prvog pitanja (da li je zamišljeni broj veći od 2^{n-1} ?) na dva skupa iste veličine: $\{1, 2, \dots, 2^{n-1}\}$ i $\{2^{n-1} + 1, \dots, 2^n\}$. Svaka od ovih polovina će bit podeljena na dve polovine narednim pitanjem.

Ako je odgovor na prvo pitanje **ne**, tada sledeće pitanje generiše skupove $\{1, 2, \dots, 2^{n-2}\}$ i $\{2^{n-2} + 1, \dots, 2^{n-1}\}$. Ako je odgovor na prvo pitanje **da**, tada naredno pitanje pravi razliku između skupova $\{2^{n-1} + 1, \dots, 2^{n-1} + 2^{n-2}\}$ i $\{2^{n-1} + 2^{n-2} + 1, \dots, 2^n\}$. Prikazani proces postavljanja pitanja i dobijanja odgovora prikazan je odgovarajućom grafskom strukturom na Slika 2.1.



Slika 2.1 Stablo pitanje odgovor nastalo tokom binarne pretrage za nepoznatom mogućnošću od ukupno $m = 2^n$ mogućnosti.

Svakom pitanju odgovara jedan čvor, označen skupom mogućnosti identifikovanih do tog momenta. Prvi čvor je stoga označen skupom svih mogućnosti S . Čvorovi u prvom sloju su označeni sa dva skupa upola manje dimenzije, drugi sloj čvorova je označen sa četiri skupa dimenzija četiri puta manjih od početnog skupa S i td. Odgovori se označavaju granama. Odgovor – **ne** označavamo sa 0, dok odgovor – **da** označavamo sa 1.

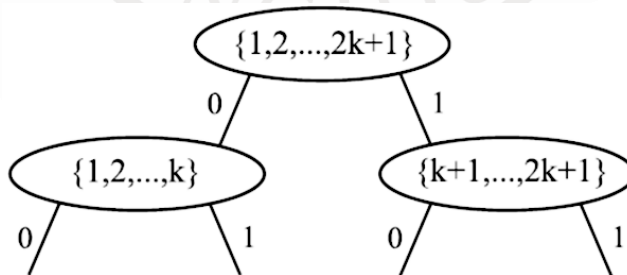
Proces sukcesivnog deljenja početnog skupa mogućnosti na polovine, završava se nakon n koraka sa ukazivanjem na tačan odgovor. Broj n nije ništa drugo do logaritam od $m = 2^n$ za osnovu 2, odnosno $n = \log_2 m$. Ovaj broj je značajno manji od m . Stoga izgleda razumno predstaviti količinu neodređenosti u šemi izbora S pomoću logaritma za osnovu dva broja svih mogućnosti $|S| = 2^n$ tog sistema.

Ako meru neodređenosti šeme izbora S označimo sa $h(|S|)$, tada je izbor $h(|S|) = \log |S|$, dobar izbor barem kada je $|S|$ stepena 2.

Šta će se desiti u opštem slučaju kada je kardinalnost skupa S proizvoljan prirodan broj?

Možemo primeniti istu šemu postavljanja pitanja, s tom razlikom što će neki generisani skupovi imati neparan broj elemenata oblika $2k + 1$, pa će njihova podela na dva dela davati podskupove nešto različite kardinalnosti k i $k + 1$.

Prikazana je ova situacija na Slika 2.2.



Slika 2.2 Stablo pitanje-odgovor nastalo tokom binarne pretrage za nepoznatom mogućnošću u opštem slučaju

Ukoliko je broj mogućnosti u šemi izbora S , između dva stepena broja 2, tj. $2^n \leq |S| < 2^{n+1}$, tada možemo ili izbaci neke mogućnosti i redukovati njihov broj na 2^n ili dodati neke mogućnosti u cilju dobijanja 2^{n+1} mogućnosti.

U prvom slučaju nam je potrebno n pitanja, a u drugom $n + 1$ da bi smo pronašli tačan odgovor. Prema tome količina neodređenosti u S je negde između ovih vrednosti, tj. $n \leq \log |S| < n + 1$. Stoga mi opet možemo uzeti $h(|S|) = \log |S|$ kao meru neodređenosti u opštem slučaju, pri čemu ova vrednost nije neophodno ceo broj i konačno usvajamo sledeću definiciju:

DEFINICIJA 2.1 *Količina neodređenosti šeme izbora.* Za šemu izbora S sa $|S|$ mogućnosti, količina neodređenosti $h(|S|)$ je definisana sa $h(|S|) = \log |S|$.

PRIMER 2.1 (Šahovska tabla) Neka je data prazna šahovska tabla. Postoji ukupno $m = 64 = 2^6$ mogućnosti za postavljanje prve figure. Stoga se odgovarajuća šema izbora može prikazati sa $S = \{1, 2, \dots, 64\}$, gde je značenje svakog broja jedno polje na šahovskoj tabli. Količina neodređenosti u postavljanju jedne figure na praznu šahovsku tablu je $h(|S|) = \log |64| = 6$.

Kao i svaka druga definicija, i naša definicija mere neodređenosti je donekle proizvoljna. Opravdanja dobrih definicija se stiču njihovim svojstvima korisnosti i elegancije u praktičnim primenama i olakšavanju teorijskih izvođenja. Budući da je ova definicija mere neodređenosti preživela prvo stoleće teorije informacija, očigledno da je zadovoljila oba ova kriterijuma.

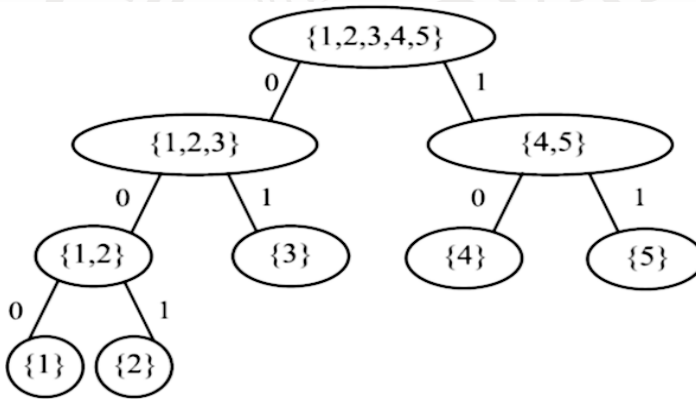
Napomenimo da pitanja možemo postavljati i tako da postoji više od dva moguća odgovora. Pretpostavimo da pitanja imaju $k > 2$ mogućih odgovora. U tom slučaju svako pitanje vrši podelu skupa od m mogućnosti u k disjunktih podskupova sa približno $\frac{m}{k}$ elemenata. Ako je $|S| = k^n$, tada nam je potrebno tačno $n = \log_k |S|$ pitanja. Ako ovog puta usvojimo logaritam za osnovu k , tada je $h(|S|) = \log_k |S|$.

Imajući u vidu da je $\log_k |S| = \log_k 2 \cdot \log_2 |S|$, Vidimo da je promena osnove logaritma po efektima ekvivalentna promeni jedinice merenja i stoga nije od suštinskog značaja.

Definicijom sistema izbora S i odgovarajućeg stabla pitanja, kao što je ono na (Slika 2.1), istovremeno je određen i sistem kodovanja svih mogućnosti sistema izbora. Kod jedne mogućnosti dobijamo spajanjem simbola 0 i 1 na putu od korena stabla do date mogućnosti. Ako su upućena binarna pitanja, tada dobijamo binarni kod za dati sistem mogućnosti. Primetimo da je dužina koda svake mogućnosti jednaka ili sledećem najmanjem ili sledećem najvećem celom broju od $h(|S|) = \log(|S|)$. Ovim se prvi put susrećemo sa bliskom vezom između mere neodređenosti i kodovanja.

PRIMER 2.2 Binarno stablo pitanja. Neka je sistem mogućnosti dat sa $S = \{1, 2, 3, 4, 5\}$. Njegova količina neodređenosti je $h(|S|) = \log 5 \approx 2,3219$ bit.

Jedno moguće stablo pitanja je dato na (Slika 2.3). Vidimo da kod 001 reprezentuje mogućnost $\{2\}$, pri čemu je njegova dužina 3 jednaka narednom najvećem celom broju od $h(|S|)$. Mogućnost $\{3\}$ ima kod dužine 2, $\{4\}$ ima kod dužine 2 i td.



Slika 2.3 Stablo pitanje-odgovor nastalo na osnovu binarnih pitanja u sistemu sa 5 mogućnosti.

Nabrajaćemo sada neke najjednostavnije osobine mere neodređenosti $h(|S|)$:

1. Ako su S_1 i S_2 dva sistema mogućnosti i $|S_1| = |S_2|$, tada je i $h(|S_1|) = h(|S_2|)$. Prema tome uticaja na neodređenost ima samo broj mogućnosti a ne i njihova priroda.

2. Ako su S_1 i S_2 dva sistema mogućnosti i $|S_1| < |S_2|$, tada je i $h(|S_1|) < h(|S_2|)$, budući da je logaritam neopadajuća funkcija. Ova osobina je u skladu sa našim intuitivnim očekivanjima: neodređenost raste sa porastom broja mogućnosti izbora.
3. Ako su S_1 i S_2 dva sistema mogućnosti S_2 ima dva puta više mogućnosti od S_1 ($|S_2| = 2 \cdot |S_1|$) tada uzimajući logaritam za osnovu 2, dobijamo $h(|S_2|) = h(|S_1|) + 1$. Ovo sledi iz svojstva aditivnosti funkcije logaritma i činjenice da je $\log_2 2 = 1$, $h(|S_2|) = \log_2 |S_2| = \log_2(2 \cdot |S_1|) = \log_2 2 + \log_2 |S_1| = 1 + h(|S_1|)$.
4. Ako je S sistem samo sa dve mogućnosti, tada ako za osnovu logaritma uzmemo osnovu 2, tada je $h(|S|) = \log_2 2 = 1$. Ova jedinica mere se naziva bit (jedinica binarne informacije). Vidimo da je neodređenost tesno povezana sa informacijom i da se mogu izraziti u istim mernim jedinicama. Budući da je u računarima informacija prirodno zapisana pomoću dva stanja, najpopularniji tip pitanja je binarni.
5. Ako su nam zadata dva sistema mogućnosti $S_1(e_{1,1}, e_{1,2}, \dots, e_{1,n})$, $S_2(e_{2,1}, e_{2,2}, \dots, e_{2,m})$, tada se ovi sistemi mogu iskombinovati u jedinstveni sistem mogućnosti, čije su mogućnosti svih $n \cdot m$ parova $\{(e_{1,1}, e_{2,1}), (e_{1,1}, e_{2,2}), \dots, (e_{1,2}, e_{2,1}), \dots, (e_{1,n}, e_{2,m})\}$. Ovakav skup svih parova se naziva dekartov proizvod dva skupa i zapisuje se u formi: $S_1 \times S_2 = \{(e_{1,1}, e_{2,1}), (e_{1,1}, e_{2,2}), \dots, (e_{1,2}, e_{2,1}), \dots, (e_{1,n}, e_{2,m})\}$.

Nazvaćemo ovaj novi sistem izbora, sistemom nezavisnih izbora. Ovim se izražava ideja da se izbor mogućnosti u svakom od sistema vrši nezavisno od drugog sistema. Ovakvim nezavisnim izborima se dobijaju kombinovane mogućnosti. Kako se količina neodređenosti kombinovanog sistema odnosi prema količinama neodređenosti pojedinačnih sistema mogućnosti? Odgovor je formulisan u sledećem stavu.

TEOREMA 2.1 Aditivnost neodređenosti. Neodređenost sistema nezavisnih izbora jednaka je sumi neodređenosti sistema konstituenata, tj. $h(|S_1 \times S_2|) = h(|S_1|) + h(|S_2|)$.

Dokaz: Dokaz se svodi na korišćenje svojstva aditivnosti funkcije logaritma, naime $h(|S_1 \times S_2|) = \log |S_1 \times S_2| = \log(|S_1| \cdot |S_2|) = \log |S_1| + \log |S_2| = h(|S_1|) + h(|S_2|)$.

Svojstvo neodređenosti izraženo Teoremom 2.1 u potpunosti podržava intuiciju da se neodređenosti u dve nezavisne situacije sabiraju, ukoliko su te situacije međusobno nezavisne, kada se razmatraju zajedno

Vratimo se Primeru 2.1 sa šahovskom tablom, u kome smo videli da je neodređenost jedne figure na celokupnoj šahovskoj tabli $h(|S|) = \log 64 = 6$.

Na isti način, možemo videti da je neodređenost pozicije jedne figure u jednom redu ili jednoj koloni jednaka $\log 8 = 3$. Iz ove činjenice možemo izvesti neodređenost pozicije na celokupnoj tabli, udruživanjem u nezavisan sistem izbora sisteme izbora po kolonama i redovima, tj. $6 = \log 64 = \log 8 + \log 8$.

Uopštenje se može direktno izvršiti na slučaj m nezavisnih sistema izbora. Tada je dekartov proizvod dužine m

$$S_1 \times S_2 \times \cdots \times S_m = \{(e_{1,1}, e_{2,1}, \dots, e_{m,1}), \dots\}$$

odgovarajući sistem nezavisnih izbora. Teorema 2.1 se direktno proširuje na ovaj opšti slučaj i u važnosti je izraz za neodređenost združenog sistema mogućnosti $h(|S_1 \times S_2 \times \cdots \times S_m|) = h(|S_1|) + h(|S_2|) + \cdots + h(|S_m|)$. (2.1)

Neka se kocka baca m puta i neka se podrazumeva da su bacanja nezavisna. Ovaj proces se može modelovati pomoću m nezavisnih sistema izbora $S_1, \dots, S_m$, pri čemu svaki sistem ima 6 mogućnosti. Prema (2.1), ukupna neodređenost ishoda prilikom bacanja kocke m uzastopnih puta je: $h(|S_1 \times S_2 \times \cdots \times S_m|) = h(|S_1|) + h(|S_2|) + \cdots + h(|S_m|) = m \cdot \log 6$.

Formalizovali smo situacije neodređenosti uvođenjem sistema S , u kome se bira jedna od mogućnosti, ali se ne zna koja.

Neodređenost pridružena sistemu mogućnosti se meri najmanjim brojem pitanja koja treba postaviti u cilju nalaženja stvarnog izbora. Na ovaj način smo došli do mere neodređenosti izražene sa $\log |S|$. Ako su upotrebljena binarna pitanja, jedinica mere neodređenosti je bit.

Sistemom postavljanja pitanja u vezi određivanja date nepoznate mogućnosti, indukuje se stablo na osnovu koga se može jednoznačno odrediti kod svake od mogućnosti sistema S . Dužine kodova su približno jednake meri neodređenosti

$\log |S|$. Ako su korišćena binarna pitanja, dobija se binarni kod. Neodredjenosti sistema nezavisnih mogućnosti se međusobno sabiraju.



2.1. ENTROPIJA – IZBORI MOGUĆNOSTI SA POZNATIM VEROVATNOĆAMA

Postoje situacije u kojima su nam poznate verovatnoće sa kojima se pojedine mogućnosti pojavljuju u nekom posmatranom sistemu mogućnosti S . Npr. ako neko kuca tekst na srpskom jeziku, pojedini znaci se pojavljuju češće od nekih drugih. Uvedimo formalno verovatnoće u sistem mogućnosti $S = \{e_1, e_2, \dots, e_m\}$ dodelom verovatnoća p_i mogućnostima e_i za $i = 1, 2, \dots, m$. Ove verovatnoće moraju da zadovolje sledeće uslove:

$$0 \leq p_i \leq 1, \text{ za } i = 1, 2, \dots, m, \quad \sum_{i=1}^m p_i = 1. \quad (1.2)$$

Sistem izbora S zajedno sa skupom verovatnoća $P = \{p_1, p_2, \dots, p_m\}$ obrazuje verovatnosni sistem izbora. Formalno, definišemo ga na sledeći način.

DEFINICIJA 2.2 Ako je S sistem izbora a P skup verovatnoća nad S , koji zadovoljava uslov (1.2), tada se par (S, P) naziva verovatnosni sistem izbora. Ukoliko je $E \subseteq S$, tada se u terminima teorije verovatnoće on naziva događaj, a njegova verovatnoća je data sa $p(E) = \sum_{e_i \in E} p_i$.

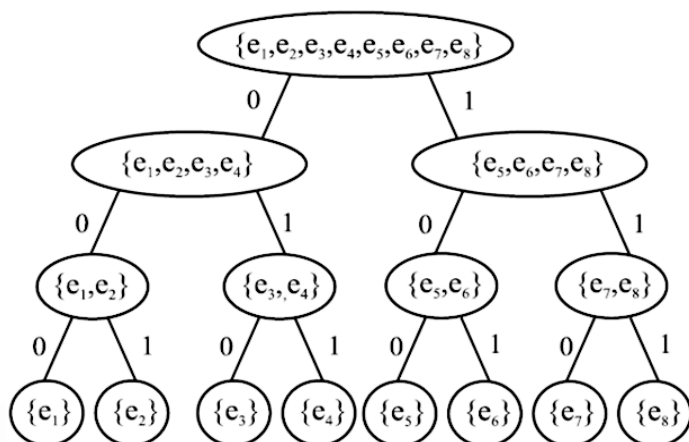
Kolika je količina neodređenosti u verovatnosnom sistemu izbora?

Možemo pristupiti na isti način postavljanjem niza odgovarajućih pitanja. Međutim, sada bi podela na skupove jednakih veličina bila neoptimalna strategija, jer njome ne bi bile obuhvaćene verovatnoće kojima raspolažemo.

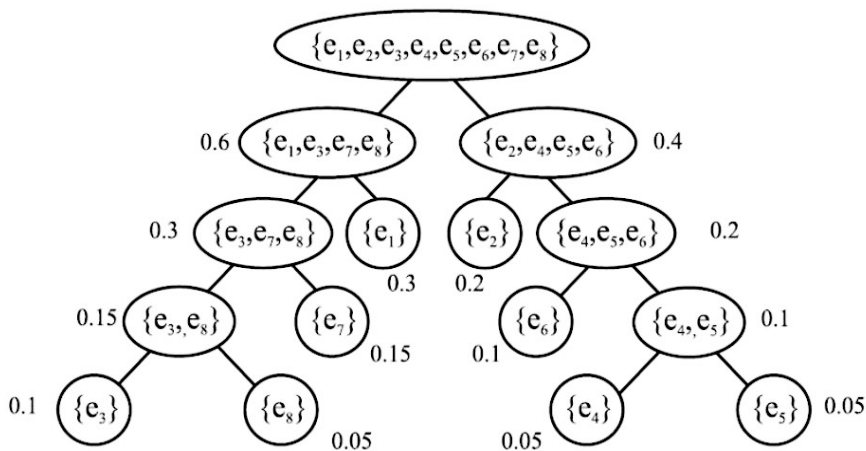
Pretpostavimo npr. da je jedna mogućnost, recimo e_1 mnogo verovatnija od svih drugih. Razumno je odmah prvo pitati da li se mogućnost e_1 realizovala, jer sa velikom verovatnoćom postoji šansa da smo pogodili samo na osnovu jednog pitanja. Samo ako je odgovor “**ne**” nastavlja se sa pitanjima.

PRIMER 2.3 Neka je dat verovatnosni sistem izbora (S, P) sa $S = \{e_1, e_2, \dots, e_8\}$ i $P = \{0.3, 0.2, 0.1, 0.05, 0.05, 0.1, 0.15, 0.05\}$. Odgovarajuće binarno stablo prikazano je na (Slika 2.4), dok je alternativno stablo koje uzima u obzir verovatnoće, prikazano je na (Slika 2.5). Jednostavnim računom dolazimo do

podatka da je očekivani broj pitanja u slučaju binarnog stabla 3, a u slučaju stabla koje uzima u obzir verovatnoće 2.75.



Slika 2.4 Binarno stablo iz primera 2.3 sa očekivanim brojem pitanja 3.



Slika 2.5 Alternativno stablo iz primera 2.3 kada su u obzir uzete verovatnoće pojavljivanja svih mogućnosti. Očekivani broj pitanja je 2.75.

Minimizacija očekivane vrednosti broja postavljenih pitanja nije trivijalan zadatak. Rešenje ovog problema je poznato i veoma se koristi u teoriji kodovanja. Ključna ideja ovog postupka je da se skup svih mogućnosti ne deli na podskupove

jednake kardinalnosti, već jednakih verovatnoća. Iz teorije kodovanja je poznato da je očekivani broj pitanja aproksimativno iznosi $-\sum_{i=1}^m p_i \log p_i$.

Ova veličina se naziva entropija, i predstavlja meru neodređenosti količine neodređenosti verovatnosnih sistema mogućnosti. Ukoliko je neka od verovatnoća $p_i = 0$, usvaja se konvencija da je $0 \cdot \log 0 = 0$, što sledi iz činjenice da je $\lim_{x \rightarrow 0} x \log x = 0$.

DEFINICIJA 2.3 Neka je (S, P) verovatnosni sistem mogućnosti. Tada je količina neodređenosti u njemu definisana entropijom

$$H(P) = -\sum_{i=1}^m p_i \log p_i.$$

U definiciji (2.3) postoji određeni stepen proizvoljnosti, budući da nije definisana osnova logaritma. Baza k logaritma odgovara broju mogućih odgovora u eksperimentu postavljanja pitanja i dobijanja odgovora. Promena baze menja samo jedinicu mere. Binarni slučaj je najpopularniji i njemu odgovara jedinica bit.

PRIMER 2.4 Izračunajmo količinu neodređenosti za verovatnosni sistem neodređenosti iz primera 2.3.

$$\begin{aligned} H(P) &= -\sum_{i=1}^8 p_i \log p_i \\ &= -0.3 \cdot \log 0.3 - 0.2 \cdot \log 0.2 - 0.2 \cdot \log 0.1 - 0.15 \cdot \log 0.05 \\ &\quad - 0.15 \cdot \log 0.15 \approx 2.7087 \end{aligned}$$

bita.

$H(P)$ je manje od očekivane vrednosti dužine koda u boljem stablu pitanja na (Slika 2.5).

2.1.1. SVOJSTVA ENTROPIJE

Budući da je entropija fundamentalan pojam teorije informacija, navešćemo njene osnovne karakteristike. U buduću ćemo za sistem mogućnosti nad

verovatnoćama $P = \{p_1, p_2, \dots, p_m\}$ označavati entropiju tog sistema sa $H(P) = H(p_1, p_2, \dots, p_m)$.

Prvo ćemo povezati opšti pojam entropije sa pojmom mere neodređenosti za sisteme mogućnosti u kojima nisu definisane verovatnoće pojedinih izbora. Ako je P uniformno raspodeljeno po svih n mogućnosti, tada je

$$H\left(\frac{1}{n}, \frac{1}{n}, \dots, \frac{1}{n}\right) = -\sum_{i=1}^n \frac{1}{n} \cdot \log \frac{1}{n} = -\log \frac{1}{n} = \log n = h(n).$$

OSOBINA 2.1 Entropija uniformne raspodele verovatnoća n mogućnosti jednaka je meri neodređenosti odgovarajućeg sistema mogućnosti bez definisanih verovatnoća mogućih ishoda. Ovaj rezultat je u tesnoj vezi sa poznatim Laplasovim principom nedovoljnog razloga, koji tvrdi da ukoliko ne znamo ništa određeno o nekoj pojavi, razumno je pretpostaviti jednake verovatnoće mogućih ishoda. U tom pogledu entropija je primenljiva kao mera neodređenosti i na one sisteme mogućnosti u kojima nam nisu poznate verovatnoće pojedinih ishoda.

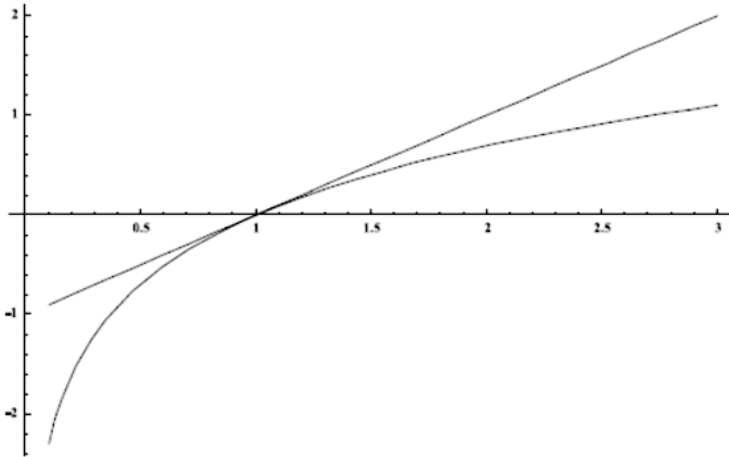
Za zadati sistem mogućnosti S u kome je $|S| = n$, imamo maksimalnu neodređenost ukoliko ne poznajemo verovatnoće ishoda ili ako su one međusobno jednake, odnosno važi $H(P) \leq h(S) = \log n$. Ovaj stav možemo i formalno dokazati. U tu svrhu potrebna nam je sledeća lema.

LEMA 2.1 Neka su $p_1, p_2, \dots, p_m$ i $q_1, q_2, \dots, q_m$ dve raspodele verovatnoća nad istim brojem mogućnosti m . Tada važi:

$$\sum_{i=1}^m p_i \cdot \log q_i \leq \sum_{i=1}^m p_i \cdot \log p_i$$

pri čemu jednakost važi ako i samo ako je $p_i = q_i$.

Dokaz. Budući da je $\log x = \log e \cdot \ln x$, gde $\ln x$ označava prirodni logaritam za osnovu e , i budući da je $\ln x$ konveksna funkcija, važi da je $\ln x \leq x - 1$. Ovu nejednačinu dobijamo iz činjenice da uslov konveksnosti znači da se data funkcija uvek nalazi ispod njene tangente u bilo kojoj tački, i ako za tu tačku izaberemo $x = 1$, upravo ćemo dobiti gornju nejednakost, videti na (Slika 2.6).


 Slika 2.6 Konveksnost funkcije $\ln x$ i tangenta u tački $x=1$.

Stoga je:

$$\ln \frac{q_i}{p_i} \leq \frac{q_i}{p_i} - 1,$$

pa je:

$$\sum_{i=1}^m p_i \cdot \ln \frac{p_i}{q_i} \leq \sum_{i=1}^m q_i - \sum_{i=1}^m p_i = 1 - 1 = 0.$$

Na osnovu gornjeg zaključujemo da je:

$$\sum_{i=1}^m p_i \cdot \log q_i - \sum_{i=1}^m p_i \cdot \log p_i = \log e \sum_{i=1}^m p_i \cdot \ln \frac{q_i}{p_i} \leq 0.$$

Gornja nejednakost važi sa znakom jednakosti ako i samo ako je $q_i/p_i = 1$, za svako i , odnosno ako je $p_i = q_i$.

Ako primenimo lemu 2.1 dobijamo:

$$H(P) - \log m = - \sum_{i=1}^m p_i \cdot \log p_i + \sum_{i=1}^m p_i \cdot \log \frac{1}{m} \leq 0,$$

gde znak jednakosti važi ako i samo ako je $p_i = 1/m$. Ovim smo ujedno dokazali sledeću teoremu.

TEOREMA 2.2

$$\max H(P) = \log m,$$

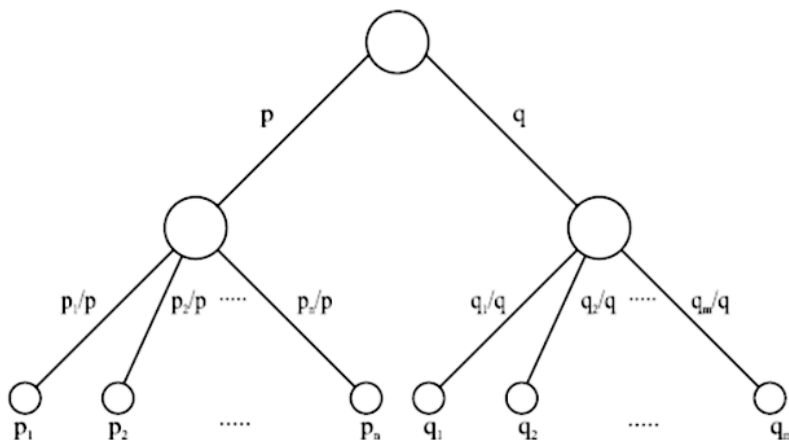
gde je maksimumu tražen po svim raspodelama verovatnoća sistema sa m mogućnosti. Maksimum se dostiže samo na raspodeli koja odgovara jednakim verovatnoćama svih mogućnosti.

2.1.2. VAŽNA SVOJSTVA ENTROPIJE

1. Ako su (S_1, P_1) i (S_2, P_2) dva verovatnosna sistema mogućnosti za koje važi $|S_1| = |S_2|$ i $P_1 = P_2$, tada je $H(P_1) = H(P_2)$. Ovim se potvrđuje intuitivan stav da entropija zavisi samo od raspodele verovatnoće, a ne i od prirode mogućnosti e_i u nekom sistemu mogućnosti.
2. U važnosti je relacija $H(p_1, p_2, \dots, p_m) = H(p_1, p_2, \dots, p_m, 0)$, što potiče od konvencije da je $0 \cdot \log 0 = 0$. Mogućnosti čija je verovatnoća nula ne utiču na ukupnu količinu neodređenosti. To je sasvim u skladu sa našom intuicijom, pošto smo sigurni da se takve mogućnosti neće nikada realizovati.
3. Posmatrajmo dvonivovsku šemu sa (Slika 2.7). Na prvom nivou, biraju se dve mogućnosti sa verovatnoćama p i $q=1-p$. Ako se izabere prva mogućnost, na drugom nivou se bira n mogućnosti sa verovatnoćama p_i/p . Ako se izabere druga mogućnost, tada se na drugom nivou bira m mogućnosti sa verovatnoćama q_i/q . Podrazumeva se da važi $\sum_{i=1}^n p_i = p$, $\sum_{i=1}^m q_i = q$.

Primetimo da to implicira $p_1 + p_2 + \dots + p_n + q_1 + q_2 + \dots + q_m = 1$, odnosno, $\{p_1, p_2, \dots, p_n, q_1, q_2, \dots, q_m\}$ je raspodela verovatnoća $n + m$ elemenata. Tada važi sledeća relacija između entropija na ova dva nivoa izbora

$$\begin{aligned} H(p_1, p_2, \dots, p_n, q_1, q_2, \dots, q_m) \\ = H(p, q) + pH\left(\frac{p_1}{p}, \frac{p_2}{p}, \dots, \frac{p_n}{p}\right) + qH\left(\frac{q_1}{q}, \frac{q_2}{q}, \dots, \frac{q_m}{q}\right). \end{aligned}$$



Slika 2.7 Dvonoovski verovatnosni sistem mogućnosti u kome se na prvom nivou biraju dve mogućnosti sa verovatnoćama p i $q=1-p$, a na drugom nivou ili n mogućnosti sa verovatnoćama p_i/p ili m mogućnosti sa verovatnoćama q_i/q .

2.1.3. KARAKTERIZACIJA ENTROPIJE

1. $H(p_1, p_2, \dots, p_n) = H(p_{\pi(1)}, p_{\pi(2)}, \dots, p_{\pi(n)})$, za svaku permutaciju π .
2. $H(p_1, p_2, \dots, p_n)$ je kontinualna funkcija po svim varijablama.
3. U važnosti je jednakost

$$H(p_1, p_2, \dots, p_n) = H(p_1 + p_2, p_3, \dots, p_n) + (p_1 + p_2)H\left(\frac{p_1}{p_1 + p_2}, \frac{p_2}{p_1 + p_2}\right),$$
za svaku raspodelu verovatnoća $p_1, p_2, \dots, p_n$, za $n \geq 2$.
4. $H\left(\frac{1}{n}, \frac{1}{n}, \dots, \frac{1}{n}\right)$ je monotono rastuća funkcija od n .

2.1.4. SLUČAJNE VELIČINE - PODSETNIK

Često se javlja situacija u kojoj se svakom elementarnom ishodu jednog eksperimenta pridružuje neki realan broj. Na takvo pridruživanje se nailazi, na primer, u igrama na sreću, gde se svakom ishodu pripisuje određeni dobitak ili gubitak za igrača. Kao što se verovatnoće mogu računati samo za događaje iz polja događaja, tako se i preslikavanja koja svakom ishodu dodeljuju neki realan

broj mogu proučavati pod uslovom da zadovoljavaju određene uslove. U tom smislu navodi se sledeća definicija.

DEFINICIJA Slučajna veličina.

Neka je dat prostor verovatnoća $(\Omega, \mathcal{F}, P)$ i neka je X funkcija koja preslikava prostor elementarnih ishoda Ω u skup realnih brojeva R tako da važi:

1. Skup $\{\omega | X(\omega) \leq x\}$ je događaj koji pripada $\mathcal{F}$ za svako x iz R .
2. $P\{\omega | X(\omega) = -\infty\} = P\{\omega | X(\omega) = \infty\} = 0$.

Tada je X slučajna veličina.

Osobina (1) se naziva merljivost i omogućava računanje verovatnoća događaja vezanih za slučajnu veličinu. Osobina (2) je **finitnost** (konačnost) slučajne veličine X .

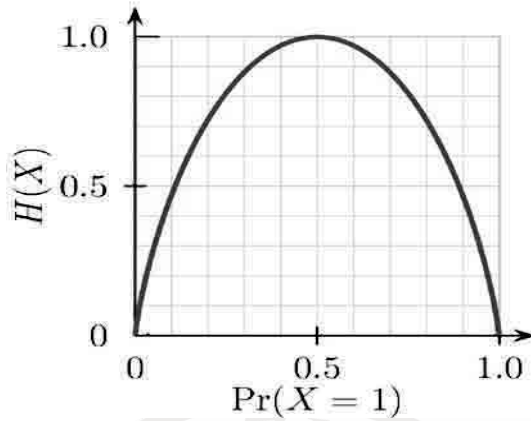
Osobine navedene karakterišu entropiju. To znači da ako bi nametnuli važenje ova 4 uslova na neku meru neodređenosti H , oni bi implicirali kao rešenje entropijsku meru neodređenosti.

U cilju daljih izlaganja uvešćemo sledeću notaciju. Verovatnosnom sistemu mogućnosti (S, P) možemo pridružiti konačnu slučajnu varijablu X , koja uzima vrednosti e_i iz $S(e_1, e_2, \dots, e_m)$. Verovatnoća da je $X = e_i$ je data sa $p_X(e_i)$, a P se naziva gustina verovatnoće slučajne varijable X . I obrnuto svaka konačna slučajna varijabla inducira jedan verovatnosni sistem mogućnosti. Stoga umesto verovatnosnog sistema mogućnosti možemo govoriti o slučajnoj veličini, a shodno tome i o entropiji slučajne veličine X sa vrednostima iz S , datoj izrazom

$$H(X) = - \sum_{x \in S} p_X(x) \log p_X(x) .$$

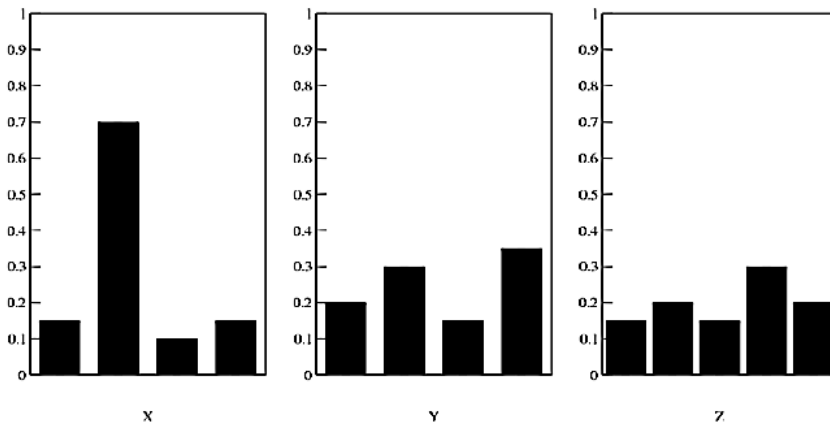
Sada entropiju možemo interpretirati kao meru neodređenosti asociranu sa datom slučajnom veličinom X .

Entropija slučajne varijable sa dva ishoda ($X = 1, X = 0$) prikazana je na (Slika 2.8).



Slika 2.8 Entropija slučajne varijable sa dva ishoda

Na (Slika 2.9), prikazane su entropije različito raspoređenih slučajnih veličina.



Slika 2.9 Entropije različito raspoređenih slučajnih veličina

2.1.5. USLOVNE ENTROPIJE

Neka je zadat verovatnosni sistem mogućnosti (S, P) i neka je ovom sistemu asocirana slučajna veličina X . Neka je observirano da se desio događaj $E \subseteq S$.

Kako poznavanje ove činjenice utiče na neodređenost sistema?

Ovaj događaj indukuje novi verovatnosni sistem mogućnosti (E, P_E) . Ovde je P_E uslovna verovatnoća $p_{X|E}(x) = \frac{p_X(x)}{p_X(E)}$, za svako $x \in E$.

Neodređenost vezana za ovaj novi sistem je data sa uslovnom entropijom slučajne veličine X kada je poznato E :

$$H(X|E) = - \sum_{x \in E} p_{X|E}(x) \log p_{X|E}(x) .$$

Neka je X slučajna veličina vezana za verovatnosni sistem mogućnosti (S, P) , gde je $S = \{1, 2, 3, 4\}$, $P = \{0.5, 0.25, 0.125, 0.125\}$ i $E = \{1, 3\}$ je jedan događaj. Tada je bezuslovna entropija data sa:

$$H(X) = -0.5 \log 0.5 - 0.25 \log 0.25 - 0.125 \log 0.125 - 0.125 \log 0.125 = 1.75$$

bit.

Kako je:

$$p_X(E) = 0.625, \quad p_{X|E}(1) = \frac{p_X(1)}{p_X(E)} = 0.8, \quad p_{X|E}(3) = \frac{p_X(3)}{p_X(E)} = 0.2$$

dobijamo:

$$H(X|E) = -p_{X|E}(1) \log p_{X|E}(1) - p_{X|E}(3) \log p_{X|E}(3) = -0.8 \log 0.8 - 0.2 \log 0.2 \\ \approx 0.7219 \text{ bit}$$

2.1.6. UZAJAMNA ENTROPIJA

U ovom poglavlju naučićemo kako se entropija združenog sistema mogućnosti ili višedimenzionih promenljivih odnosi prema entropiji komponentata sistema mogućnosti ili pojedinačnih promenljivih. Takođe, kako poznavanje izbora u jednoj komponenti sistema mogućnosti ili vrednosti jedne promenljive utiču na neodređenost preostalih komponenti sistema mogućnosti ili promenljivih.

Neka su zadati sistemi mogućnosti (S_1, P_1) uz koji je vezana slučajna varijabla X i sistem (S_2, P_2) uz koji je vezana slučajna varijabla Y . Tada možemo definisati združeni sistem $(S_1 \times S_2, P)$, gde je P opisano združenom raspodelom

verovatnoće $p_{X,Y}^{(x,y)}$, definisanom nad parovima $(x,y) \in S_1 \times S_2$. Tada možemo definisati tri entropije nad ova tri sistema:

$$H(X,Y) = - \sum_{x \in S_1} \sum_{y \in S_2} p_{X,Y}(x,y) \log p_{X,Y}(x,y)$$

$$H(X) = - \sum_{x \in S_1} p_X(x) \log p_X(x)$$

$$H(Y) = - \sum_{y \in S_2} p_Y(y) \log p_Y(y)$$

U kakvom su odnosu ove tri entropije?

TEOREMA 2.3

Za bilo koji par slučajnih veličina X i Y važi:

$$H(X,Y) \leq H(X) + H(Y)$$

Znak jednakosti važi ako i samo ako su X i Y nezavisne slučajne veličine, tj.

$$p_{X,Y}(x,y) = p_X(x) \cdot p_Y(y)$$

Odavde vidimo da je neodređenost združenog sistema uvek manja od sume neodređenosti pojedinačnih sistema. Razlog potiče od međusobne zavisnosti (korelisanosti) X i Y , koja kao da rezultuje u zajedničkom udelu neodređenosti u obe varijable. U slučaju nezavisnih X i Y , ove zajedničke neodređenosti nema, i u važnosti je znak jednakosti.

Teorema 2.3 se može generalisati na više od dve varijable. U opštem slučaju, ako imamo slučajni vektor $X = (x_1, x_2, \dots, x_m)$, čija je raspodela verovatnoća data sa $p_X(X)$, združena entropija entropija je data sa:

$$H(X) = - \sum_x p_X(x) \log p_X(x)$$

Sve marginalne raspodele pojedinih komponenti vektora X su date sa:

$$p x_i(x_i) = - \sum_{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_m} pX(x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_m)$$

Slučajne veličine $x_1, x_2, \dots, x_m$ su uzajamno nezavisne, ako i samo ako važi:

$$pX(x) = pX_1(x_1) \cdot pX_2(x_2) \cdots pX_m(x_m)$$

LEMA 2.2 Za bilo koji slučajnu višedimenzionu varijablu (slučajni vektor) $X = (x_1, x_2, \dots, x_m)$ važi:

$$H(X) \leq \sum_{i=1}^m H(X_i)$$

Jednakost važi samo u slučaju uzajamne nezavisnosti slučajnih varijabli $x_1, x_2, \dots, x_m$.

PRIMER 2.5 Neka su $x_1, x_2, \dots, x_m$ nezavisne slučajne veličine koje dobijaju vrednost 0 i 1 sa verovatnoćama 0.5, odnosno:

$$pX_i(0) = 0.5, \quad pX_i(1) = 0.5, \quad \text{za } i = 1, \dots, n.$$

Tada je:

$$H(X_1, \dots, X_n) = n \cdot H(X_1) = n.$$

Posmatrajmo slučajne veličine X i Y . Neka je opservirana realizacija $Y = y$.

Kako ovaj događaj utiče na neodređenost veličine X ?

Opserviranjem $Y = y$ menja se raspodela $p_X(x)$ u uslovnu raspodelu:

$$pX|y(x, y) = \frac{pX, Y(x, y)}{pY(y)}$$

pa je uslovna entropija za X kada je $Y = y$ data sa:

$$H(X|Y = y) = - \sum_x pX|y(x, y) \log pX|y(x, y).$$

Uvešćemo skraćeno zapisivanje :

$$H(X|Y = y) \rightarrow H(X|y).$$

Ako su dve slučajne varijable nezavisne, tada je:

$$p_{X|Y}(x, y) = p_X(x)$$

$$H(X|y) = - \sum_x p_X(x) \log p_X(x) = H(X).$$

Odnosno, kada su X i Y nezavisne, opservacija promenljive Y ne menja neodređenost promenljive X . Pored ove uslovne entropije možemo definisati očekivanu uslovnu entropiju:

$$H(X|Y) = \sum_y p_Y(y) H(X|y)$$

što je u stvari očekivana vrednost $H(X|y)$ u odnosu na y .

Do sada smo se susreli sa tri tipa uslovnih entropija:

- $H(X|y)$ – Uslovna entropija varijable X u odnosu na događaj ili opservaciju $Y = y$.
- $H(X|E)$ – Uslovna entropija varijable X u odnosu na događaj E
- $H(X|Y)$ – Očekivana vrednost uslovne entropije

TEOREMA 2.4 Za par slučajnih promenljivih X i Y uvek važi:

$$H(X, Y) = H(X) + H(X|Y)$$

ili

$$H(X, Y) = H(X) + H(Y|X).$$

DOKAZ TEOREME 2.4 Važi $p_{X,Y}(x, y) = p_Y(y)p_{X|Y}(x, y)$, pa je stoga:

$$\begin{aligned}
 H(X|Y) &= \sum_y p_Y(y) H(X|y) \\
 &= - \sum_x \sum_y p_Y(y) p_{X|Y}(x, y) \log p_{X|Y}(x, y) \\
 &= - \sum_x \sum_y p_{X, Y}(x, y) \log \frac{p_{X, Y}(x, y)}{p_Y(y)} = \\
 &\quad - \sum_x \sum_y p_{X, Y}(x, y) (\log p_{X, Y}(x, y) - \log p_Y(y)) \\
 &= H(X, Y) - H(Y).
 \end{aligned}$$

KAROLARIJ 2.1 Za svaki par slučajnih veličina X i Y važi:

$$H(X|Y) \leq H(X).$$

Jednakost važi ako i samo ako su X i Y nezavisne.

DOKAZ KAROLARIJA 2.1 Primenom pravila lančanja, datog Teoremom 2.3 dobijamo:

$$H(X|Y) = H(X, Y) - H(Y) \leq H(X) + H(Y) - H(Y) = H(X)$$

pri čemu jednakost važi ako i samo ako su X i Y nezavisne.

KAROLARIJ 2.2 (Generalizovano pravilo lančanja) Neka su $x_1, x_2, \dots, x_m$ slučajne veličine. Tada važi:

$$H(X_1, X_2, \dots, X_m) = H(X_1) + H(X_2 | X_1) + \dots + H(X_m | X_1, X_2, \dots, X_{m-1}).$$

DOKAZ KAROLARIJA 2.2 Dokazaćemo ga indukcijom. Za $m=2$, stav važi i dat je teoremom 2.4. Pretpostavimo da važi za neko m . Neka je $X_m = (X_1, X_2, \dots, X_m)$. Na osnovu teoreme 2.4 i pretpostavki indukcije, dobijamo:

$$\begin{aligned}
 H(X_1, X_2, \dots, X_m, X_{m+1}) &= H(X_m, X_{m+1}) + H(X_1, X_2, \dots, X_m | X_m, X_{m+1}) \\
 &= H(X_m, X_{m+1}) + H(X_1, X_2, \dots, X_m | X_m) \\
 &= H(X_1) + H(X_2 | X_1) + \dots + H(X_m | X_1, X_2, \dots, X_{m-1}) \\
 &\quad + H(X_{m+1} | X_1, X_2, \dots, X_m)
 \end{aligned}$$

Kako važi za $m + 1$, važi i za svako m .

PRIMER 2.6

$$p_{X,Y}(0,1) = p_{X,Y}(1,0) = p_{X,Y}(0,0) = \frac{1}{3}, p_{X,Y}(1,1) = 0, p_X(0) = p_Y(0) = \frac{2}{3} \text{ i } p_X(1) = p_Y(1) = \frac{1}{3}.$$

Sledi:

$$\begin{aligned} H(X|Y=0) &= -p_{X|Y}(0,0) \log p_{X|Y}(0,0) - p_{X|Y}(1,0) \log p_{X|Y}(1,0) \\ &= -\frac{p_{X,Y}(0,0)}{p_Y(0)} \log \frac{p_{X,Y}(0,0)}{p_Y(0)} - \frac{p_{X,Y}(1,0)}{p_Y(0)} \log \frac{p_{X,Y}(1,0)}{p_Y(0)} \\ &= -0.5 \log 0.5 - 0.5 \log 0.5 = 1, \end{aligned}$$

$$H(X|Y=1) = -\frac{p_{X,Y}(0,1)}{p_Y(1)} \log \frac{p_{X,Y}(0,1)}{p_Y(1)} - \frac{p_{X,Y}(1,1)}{p_Y(1)} \log \frac{p_{X,Y}(1,1)}{p_Y(1)} = 0,$$

$$H(X) = H(Y) = -\frac{2}{3} \log \frac{2}{3} - \frac{1}{3} \log \frac{1}{3} \approx 0.9183.$$

$$H(X|Y=1) < H(X) < H(X|Y=0).$$

$$H(X|Y) = p_Y(0)H(X|Y=0) + p_Y(1)H(X|Y=1) = \frac{2}{3} \text{ bit}$$

$$H(X,Y) = -\log \frac{1}{2} \approx 1.5850 \text{ bit}$$

Odakle možemo da vidimo da važi Teorema 2.4

$$H(X,Y) = H(Y) + H(X|Y)$$

Odnosno:

$$1.5850 = 0.9183 + 0.6666$$



Slika 2.10 Model komunikacionog kanala

Kanal se opisuje uslovnim raspodelama:

$$0 \leq p_{Y|x}(y, x) \leq 1, x \in I, y \in O,$$

$$\sum_y p_{Y|x}(y, x) = 1 \text{ za sve } x \in I.$$

koja se često naziva i transmisiona matrica. Jednakost $H(X, Y) = H(X) + H(Y|X)$ označava da se celokupna neodređenost sistema sastoji od neodređenosti ulaza $H(X)$ i neodređenosti prenosa preko kanala $H(Y|X)$.

PRIMER 2.7 (Binarni simetrični kanal) Transmisiona matrica je data sa:

$$\mathbf{P} = \begin{pmatrix} p_{Y|x}(0,0) & p_{Y|x}(1,0) \\ p_{Y|x}(0,1) & p_{Y|x}(1,1) \end{pmatrix} = \begin{pmatrix} 1 - \varepsilon & \varepsilon \\ \varepsilon & 1 - \varepsilon \end{pmatrix}$$

Odakle se vidi da je ε greška prenosa. Neka je $pX(0) = p, pX(1) = q = 1 - p$. Tada je:

$$H(X) = -p \log p - (1 - p) \log(1 - p)$$

$$pY(0) = p_{Y|x}(0,0) \cdot pX(0) + p_{Y|x}(0,1) \cdot pX(1) = (1 - \varepsilon)p + \varepsilon(1 - p),$$

$$pY(1) = p_{Y|x}(1,0) \cdot pX(0) + p_{Y|x}(1,1) \cdot pX(1) = \varepsilon p + (1 - \varepsilon)(1 - p).$$

$$\begin{aligned} H(Y|0) &= -p_{Y|x}(0,0) \log p_{Y|x}(0,0) - p_{Y|x}(1,0) \log p_{Y|x}(1,0) \\ &= -(1 - \varepsilon) \log(1 - \varepsilon) - \varepsilon \log \varepsilon, \end{aligned}$$

$$\begin{aligned} H(Y|1) &= -p_{Y|x}(0,1) \log p_{Y|x}(0,1) - p_{Y|x}(1,1) \log p_{Y|x}(1,1) \\ &= -\varepsilon \log \varepsilon - (1 - \varepsilon) \log(1 - \varepsilon) = H(Y|0) \end{aligned}$$

$$\begin{aligned} H(Y|X) &= pX(0)H(Y|0) + pX(1)H(Y|1) = pH(Y|0) + (1 - p)H(Y|0) = H(Y|0) \\ &= H(Y|1). \end{aligned}$$

što nije iznenađenje jer je kanal simetričan. Uz određeni napor na osnovu gornjih izraza možemo proveriti da važi:

$$H(X, Y) = H(X) + H(Y|X)$$

Neka je:

$$\varepsilon = 0.1, pX(0) = p = 0.2, pX(1) = q = 0.8.$$

Tada je:

$$H(X) = -0.2 \log 0.2 - 0.8 \log 0.8 \approx 0.7219 \text{ bit},$$

$$pY(0) = 0.9 \cdot 0.2 + 0.1 \cdot 0.8 = 2.46,$$

$$pY(1) = 0.1 \cdot 0.2 + 0.9 \cdot 0.8 = 0.74,$$

$$H(Y|X) = H(Y|0) = -0.9 \log 0.9 - 0.1 \log 0.1 \approx 0.4690 \text{ bit}.$$

$$p_{X,Y}(0.0) = p_{Y|x}(0.0)pX(0) = 0.9 \cdot 0.2 = 0.18,$$

$$p_{X,Y}(1.0) = p_{Y|x}(0.1)pX(1) = 0.1 \cdot 0.8 = 0.08,$$

$$p_{X,Y}(0.1) = p_{Y|x}(1.0)pX(0) = 0.1 \cdot 0.2 = 0.02,$$

$$p_{X,Y}(1.1) = p_{Y|x}(1.1)pX(1) = 0.9 \cdot 0.8 = 0.72,$$

$$H(X, Y) = -0.18 \log 0.18 - 0.08 \log 0.08 - 0.02 \log 0.02 - 0.72 \log 0.72 \approx 1.1909 \text{ bit}$$

2.2. INFORMACIJA I NJENA MERA

Šta je informacija i kako se ona meri?

Osnovna ideja je da je ona vezana za promenu neodređenosti. Mera informacije, stoga mora biti srazmerna količini promene neodređenosti.

Opservacija slučajne veličine ili događaji vezani za jednu slučajnu veličinu predstavljaju informaciju. Količina informacije dobijena na osnovu opservacija slučajne veličine ili događaja se meri na osnovu rezultujuće promene neodređenosti. Entropija i mera informacije su tesno povezani.

2.2.1. OPSERVACIJE I DOGAĐAJI

Počnimo naša razmatranja od sistema mogućnosti (S, P) , reprezentovanog slučajnom promenljivom X , koja uzima vrednosti $x \in S$, sa verovatnoćom $p_X(x)$. Ova slučajna promenljiva opisuje jedan eksperiment u kome su ishodi neodređeni. Mera neodređenosti u vezi sa ovim eksperimentom je data entropijom:

$$H(X) = - \sum_{x \in S} p_X(x) \log p_X(x).$$

Kada se izvrši eksperiment, mi opserviramo određenu vrednost $x \in S$. Nakon toga više nema neodređenosti. Prethodna neodređenost $H(X)$ redukovana na aposteriornu neodređenost 0. Razlika $H(X) - 0 = H(X)$ jednaka je količini informacije dobijene izvođenjem ovog eksperimenta. Stoga je entropija jedne slučajne veličine jednaka količini informacije dobijene opserviranjem jedne konkretne njene realizacije.

Ova ideja nam nameće dva važna zapažanja:

1. Budući da je informacija jednaka promeni entropije, ona se meri u istim jedinicama kao i entropija, dakle ako je u 2 osnova logaritma, jedinica je jedan bit.

2. Dobijena količina informacije je ista za sve moguće realizacije slučajne veličine X , bez obzira da li je verovatnoća neke realizacije velika ili mala.

PRIMER 2.8 Binarna slučajna promenljiva

Neka je zadata binarna slučajna veličina X , koja prima vrednosti 0 i 1 sa verovatnoćama p i $q = 1 - p$, respektivno. Opserviranjem jednog ishoda binarnog eksperimenta rezultuje u dobijanju količine informacije jednake:

$$H(X) = -p \log p - q \log q.$$

Na primer, u fer bacanju novčića, opserviranjem ishoda bacanja se dobija količina informacije od 1 bita.

Generališimo sada ovo razmatranje. I dalje ćemo posmatrati slučajnu promenljivu X vezanu za sistem mogućnosti (S, P) . Količina neodređenosti je i dalje $H(X)$. Sada ćemo obaviti eksperiment samo parcijalno. Nećemo opservirati direktno slučajnu veličinu X , već samo neki događaj $E \subseteq S$. Ovo sada predstavlja neku informaciju.

Šta je sada njena mera?

Opservacija događaja E menja slučajnu veličinu X u uslovnu promenljivu $X|E$, koja se odnosi na novi verovatnosni sistem mogućnosti (E, P_E) . Ovde P_E označava uslovne verovatnoće:

$$p_{X|E}(x) = \frac{p_X(x)}{p_X(E)}, \text{ za svako } x \in E.$$

Ova nova situacija, nastala opservacijom događaja E , ima neodređenost jednaku uslovnoj entropiji:

$$H(X|E) = - \sum_{x \in E} p_{X|E}(x) \log p_{X|E}(x).$$

Prema tome opserviranje događaja E , menja neodređenost sa početne $H(X)$ u $H(X|E)$. Količina informacije dobijena tom prilikom je jednaka $H(X) - H(X|E)$.

Videćemo da ovo ne mora biti u opštem slučaju dobitak, budući da $H(X|E)$ može biti i veća od $H(X)$, tako da opserviranje događaja E može da poveća neodređenost, što vodi definiciji negativne informacije.

Kada se E svodi na opservaciju tačne vrednosti jedne od mogućnosti X , tada je $H(X|E) = 0$, pa je $H(X) - H(X|E) = H(X) - 0 = H(X)$ količina informacije dobijena tom prilikom jednaka entropiji.

Zato se često entropija neke slučajne veličine naziva i informacija o samoj sebi (engl. *Self-information*). Odnosno entropija jedne slučajne veličine je u srednjem jednaka količini informacije potrebne za opis te slučajne veličine.

Da bi smo nastavili sa razvojem koncepta informacije nužno je da uvedemo dva nova pojma:

1. relativnu entropiju i,
2. uzajamnu informaciju.

2.2.2. UZAJAMNA INFORMACIJA

Posmatrajmo združeni verovatnosni sistem mogućnosti $(S_1 \times S_2, P)$ i pridružene slučajne veličine X i Y . Pretpostavimo da smo opservirali jednu vrednost $Y = y \in S_2$. Kao što smo videli ova situacija menja neodređenost u pogledu promenljive Y na nulu, odakle sledi da ova opservacija sadrži $i(y; Y) = H(Y)$ bita informacije, relativno u odnosu na prethodna znanja o Y . Ovu veličinu ćemo zvati lokalna uzajamna informacija, odnosno koliko jedna realizacija slučajne veličine Y govori o slučajnoj veličini Y u celini.

Međutim opservacija $Y = y$, menja i raspodelu verovatnoće slučajne veličine X , tako što ona sada postaje uslovna raspodela $p_{X|Y}(x, y) = \frac{p_{X,Y}(x, y)}{p_Y(y)}$, za svako $x \in S_1$.

Ovo znači da se takođe menja i neodređenost u pogledu slučajne promenljive X . Stoga opservacija $Y = y$, sadrži informaciju u odnosu na X . Količina ove informacije iznosi $i(y; X) = H(X) - H(X|y)$.

Ovu veličinu ćemo zvati lokalna uzajamna informacija slučajne veličine Y i X , a govori o tome koliko jedna opservacija $Y = y$ daje informacije o drugoj slučajnoj veličini X .

Pređimo sada sa pojedinačnih opservacija na očekivane vrednosti lokalnih uzajamnih informacija.

$$I(X; Y) = \sum_y p_Y(y) i(y; X) = \sum_y p_Y(y) (H(X) - H(X|y)) = H(X) - H(X|Y).$$

Veličina $I(X; Y)$ se naziva uzajamna informacija između slučajnih veličina X i Y . Ona predstavlja jedan od centralnih pojmova teorije informacija. Striktno govoreći ona nije informacija već očekivana vrednost količine informacije koju dobijamo o X kada opserviramo vrednosti Y .

Na osnovu Karolarija 2.3 koji tvrdi da je uvek $H(X|Y) \leq H(X)$.

TEOREMA 2.5 $I(X; Y) \geq 0$, dok je $I(X; Y) = 0$, ako i samo ako su X i Y međusobno nezavisne slučajne veličine.

Podsetimo se da lokalna uzajamna informacija $i(y; X)$ može biti i negativna, dok je uzajamna informacija kao njeno matematičko očekivanje uvek pozitivna. Na osnovu svojstva $H(X|Y) = H(X, Y) - H(Y)$, dobijamo da važi $I(X; Y) = H(X) + H(Y) - H(X, Y)$. Iz ove dobijene simetrije sledi sledeća Teorema.

TEOREMA 2.6 $I(X; Y) = I(Y; X)$. Sumirajmo sada najznačajnija svojstva uzajamne informacije i njen odnos prema entropiji.

TEOREMA 2.6a

$$I(X; Y) = H(X) - H(X|Y)$$

$$I(X; Y) = H(Y) - H(Y|X)$$

$$I(X; Y) = H(X) + H(Y) - H(X, Y)$$

$$I(X; Y) = I(Y; X)$$

$$I(X; X) = H(X)$$

2.2.3. GRAFIČKA PREZENTACIJA INFORMACIONO-TEORIJSKIH VELIČINA

Slučajne promenljive o kojim govorimo su diskretne. Za dve slučajne promenljive X i Y , Šenonove informacione mere su: $H(X)$, $H(Y)$, $H(X|Y)$ i $I(X; Y)$. Napravljena je nova prezentacija informaciono-teorijskih mera. Pogledajmo $\tilde{X}$ i $\tilde{Y}$ promenljive, koje odgovaraju promenljivoj X i Y , respektivno. $\tilde{X} \cup \tilde{Y}$ biće univerzalni set Ω , autor je u radu definisao stvarnu meru μ^* nad σ - poljem:

$$F\{(\tilde{X} \cup \tilde{Y}), \tilde{X}, \tilde{Y}, (\tilde{X} \cap \tilde{Y}), (\tilde{X} \cap \tilde{Y}^c), (\tilde{X}^c \cap \tilde{Y}), (\tilde{X} \cap \tilde{Y})^c, \phi\}$$

za:

$$\mu^*(\tilde{X} \cup \tilde{Y}) = H(X, Y)$$

$$\mu^*(\tilde{X}) = H(X)$$

$$\mu^*(\tilde{Y}) = H(Y)$$

$$\mu^*(\tilde{X} \cap \tilde{Y}) = I(X; Y)$$

$$\mu^*(\tilde{X} - \tilde{Y}) = H(X | Y)$$

$$\mu^*(\tilde{Y} - \tilde{X}) = H(Y | X)$$

$$\mu^*((\tilde{Y} - \tilde{X})^c) = H(X | Y) + H(Y | X)$$

i:

$$\mu^*(\phi) = 0$$

Primećujemo da prvih šest jednakosti sa leve strane u izrazu možemo dobiti sa izrazima sa desne strane sa kojima su predstavljene Šenonove informacione mere za X i Y , uradivši zamenu simbola:

$$H / I \rightarrow \mu$$

$$, \rightarrow \cup$$

$$; \rightarrow \cap$$

$$| \rightarrow -.$$

Ovo je formalna zamena simbola. Ne postoji značajna razlika između entropije i zajedničke informacije. Entropija se nekad odnosi na samu informaciju. Tako da za dve slučajne promenljive X i Y možemo koristiti Šenonove informacione mere na polju F . Mera μ^* je isto kao I - mera za dve slučajne promenljive X i Y . Lako je uvideti da za bilo koji teorijski identitet informacije X i Y , možemo dobiti odgovarajući teorijski identitet formalnom zamenom simbola.

Na primer, za informaciono teorijski identitet:

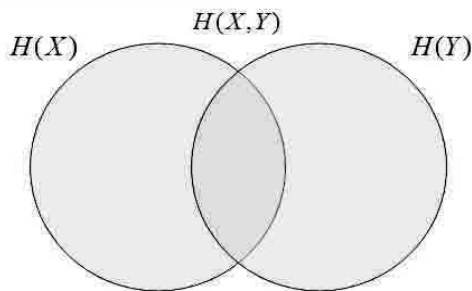
$$H(X, Y) \equiv H(X) + H(Y) - I(X; Y),$$

Sledi set teorijskog identiteta:

$$\mu(\tilde{X} \cup \tilde{Y}) \equiv \mu(\tilde{X}) + \mu(\tilde{Y}) - \mu(\tilde{X} \cap \tilde{Y}).$$

U nastavku sledi grafička prezentacija Šenonovih mera, preuzeta iz rada (1991).

Zajednička entropija dve diskretne slučajne promenljive X i Y , ili njihova entropija nakon udruživanja definisana je sa $H(X, Y)$. Ovo znači da ako su promenljive nezavisne, njihova entropija biće zbir njihovih pojedinačnih entropija $H(X) + H(Y) = H(X, Y)$, prikazano na, (Slika 2.11).



Slika 2.11 Dve slučajne promenljive

Pod uslovom da je:

$$H(X, Y) \leq H(X) + H(Y)$$

i:

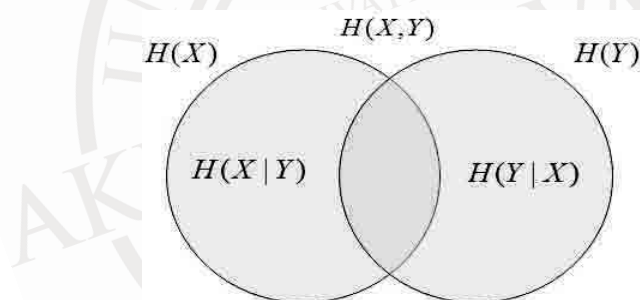
$$0 \leq H(X) \leq \log 2n$$

Gde n predstavlja broj mogućih vrednosti X .

2. Uslovna entropija ili uslovna nezavisnost promenljive X kada poznajemo promenljivu Y (često se naziva ekvivokacija) je prosečna uslovna entropija nad Y . Osnovna osobina ovog oblika entropije je:

$$H(X | Y) = H(X, Y) - H(Y).$$

Sledi grafička prezentacija ove veličine, (Slika 2.12).



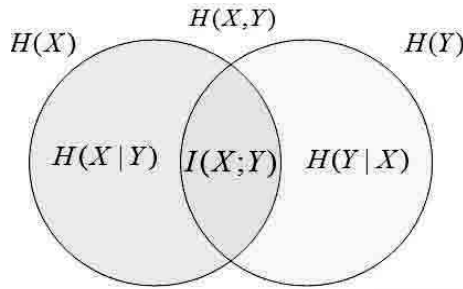
Slika 2.12 Uslovna entropija

Pod uslovom da je:

$$H(X | Y) \leq H(X),$$

$$H(Y | X) \leq H(Y).$$

3. Uzajamna informacija meri količinu informacije koju je moguće dobiti o jednoj promenljivoj, kada je opservirala ili kada je poznata druga. Ova veličina je važna u komunikacijama, gde se može koristiti u cilju povećavanja razmenjene količine informacije između poslatih i primljenih signala. Sledi grafička prezentacija ove veličine, (Slika 2.13).



Slika 2.13 Uzajamna informacija

Pod uslovom da je:

$$I(X;Y) = H(X) - H(X|Y) \quad I(X;Y) = H(Y) - H(Y|X)$$

2.2.4. RELATIVNA ENTROPIJA

Posmatrajmo dva verovatnosna sistema mogućnosti nad istim skupom mogućnosti S . Ako su X i Y korespondentne slučajne promenljive, tada se veličina:

$$K(P_X, P_Y) = \sum_{x \in S} p_X(x) \log \frac{p_X(x)}{p_Y(x)}$$

Naziva relativna entropija ili Kulbak Lejblerova divergencija (Kulbak-Lejblerovo rastojanje) između raspodele dve slučajne veličine X i Y . Iako se u jednom od naziva koristi rastojanje, primetimo da ne važi uslov simetričnosti, naime:

$$d(P_X, P_Y) \neq d(P_Y, P_X).$$

Divergencija je jednaka nuli:

$$K(P_X, P_Y) = K(P_Y, P_X) = 0,$$

Ako i samo ako su:

$$P_X = P_Y.$$

Pošto se usvaja konvencija da je:

$$0 \log(0/0) = 0, 0 \log(0/q) = 0, p \log(p/0) = \infty, K(P_X, P_Y) = \infty,$$

ukoliko postoji x za koje je $p_X(x) > 0$, a $p_Y(x)=0$.

Posmatrajmo združeni verovatnosni sistem mogućnosti $(S \times S, P)$ kome je asociran par slučajnih veličina $X, Y \in S$. Marginalni sistemi (S, P_X) i (S, P_Y) su indukovani marginalnim raspodelama verovatnoća:

$$p_X(x) = \sum_y p_{X,Y}(x, y), p_Y(y) = \sum_x p_{X,Y}(x, y).$$

Označimo sa $P_X \cdot P_Y$ raspodelu verovatnoća $p_X(x) \cdot p_Y(y)$. Tada važi:

$$I(X; Y) = I(Y; X) = K(P, P_X \cdot P_Y)$$

$$I(X; Y) = \sum_y p_Y(y) i(y; X) = \sum_y p_Y(y) (H(X) - H(X|y)) = H(X) - H(X|Y).$$

Ako u gornji izraz zamenimo vrednosti $H(X)$ i $H(X|Y)$ dobijamo:

$$\begin{aligned} I(X; Y) &= \sum_x p_X(x) \log p_X(x) + \sum_{x,y} p_Y(y) p_{X|Y}(x, y) \log p_{X|Y}(x, y) \\ &= \sum_{x,y} p_{X,Y}(x, y) \left(\log \frac{p_{X,Y}(x, y)}{p_Y(y)} - \log p_X(x) \right) \\ &= \sum_{x,y} p_{X,Y}(x, y) \log \frac{p_{X,Y}(x, y)}{p_X(x) p_Y(y)}. \end{aligned}$$

Na osnovu ovih uvida nam je jasno da uzajamna informacija $I(X; Y)$ meri stepen divergencije združene raspodele verovatnoće para (X, Y) od raspodele verovatnoće ovog para u slučaju njihove nezavisnosti.

Stoga se često sa opravdanjem uzajamna informacija uzima kao mera međuzavisnosti dve slučajne veličine koje uzimaju vrednosti nad istim skupom.

2.3. ŠENONOVA MATEMATIČKA TEORIJA INFORMACIJE

Klod Šenon (*Claude E. Shannon*) je u radu (1948) formulisao teoriju za kompresiju podataka. Šenon je utvrdio da postoji fundamentalni limit za kompresiju podataka bez gubitaka. Ovo ograničenje se naziva stopa entropije i označava sa H . Tačna vrednost entropije zavisi od izvora informacija, tačnije, od statističke prirode izvora. Šenon tvrdi da je moguće kompresovati poruke na izvoru bez gubitaka, uz stepen kompresije blizu H i da je matematički nemoguće uraditi bolje od H .

Koncept ima mnogo primena u teoriji informacija i komunikacija, a Šenon ga je primenio i da bi izrazio količinu aktuelnih informacija u engleskom tekstu, rečima: "Entropija je statistički parametar koji meri u određenom smislu, koliko se mnogo informacija proizvodi u proseku za svako slovo teksta u jeziku. Ako se jezik prevede u binarne digitalne znakove-bitove (0 ili 1) u najefikasnijem slučaju, entropija H je srednji broj bita koji se zahteva po slovu originalnog jezika."

Takođe, Šenon je razvio teoriju o gubicima prilikom kompresije podataka. Ovo je više poznato kao stopa distorzije u teoriji. U kompresiji sa gubicima, dekompresovani podaci ne moraju biti isti kao originalni podaci. Umesto toga, neka vrednost distorzije D , može da se toleriše. Šenon je pokazao da za određeni izvor sa poznatim statističkim osobinama i datom merom distorzije, postoji funkcija $R(D)$ koja se naziva funkcija za stopu distorzije. Teorija kaže da ako je D tolerantna količina distorzije, onda je $R(D)$ ajbolji stepen kompresije.

Kada se radi o kompresiji bez gubitaka (tj. bez distorzije ili $D = 0$), najbolji mogući stepen kompresije je $R(0) = H$ (odnosi se na alfabet, kao konačni izvor poruka). Drugim rečima, najbolja moguća stopa kompresije bez gubitaka je stopa entropije H .

Teorija o kompresiji podataka bez gubitaka i stopa distorzije zajedno čine teoriju o kodovanju informacionog izvora. Teorija o kodovanju izvora postavlja fundamentalna ograničenja na performanse svih algoritama za kompresiju podataka. Teorija, sama po sebi ne definiše kako de se dizajniraju i sprovedu u delo ovi algoritmi. On međutim, daje neke savete i smernice za postizanje

optimalnih performansi. U nastavku poglavlja analiziraćemo kako je Šenon modelovao izvor informacija u pogledu slučajnog procesa i njegovu teoremu za kodovanje izvora bez gubitaka.

2.3.1. MODELOVANJE IZVORA

Zamislamo da odemo u biblioteku "Amazon". Ta biblioteka ima veliki izbor knjiga, negde oko 50 miliona knjiga. Svaka knjiga ima u proseku 50 miliona znakova (slova). Izabraćemo jednu knjigu na slučajan način. U tom momentu ta knjiga je kompresovan informacioni izvor u nekom ZIP formatu, da bismo knjigu preneli do kuće na disku ili preko Interneta. Matematički, knjigu koju izaberemo označavano na sledeći način:

$$X = \{X_1, X_2, X_3, \dots\}.$$

gde X predstavlja celu knjigu, prvi znak u knjizi je X_1 , drugi znak u knjizi je X_2 i tako redom. Sada ćemo da definišemo konačan skup simbola ili poruka A (alfabet) koje generiše knjiga kao informacioni izvor.

$$A = \{a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, w, x, y, z, \text{razmak}\}$$

Stavićemo sada nas u ulogu nekog inženjera koji treba da dizajnira algoritam za kompresiju izabrane knjige. Iz naše perspektive, karakteri u knjizi su slučajne promenljive koje uzimaju vrednosti iz alfabeta A . Cela knjiga je beskonačan niz slučajnih promenljivih (slučajan proces). Postoji nekoliko načina kako možemo modelovati statističke osobine knjige.

2.3.2. ŠENONOVİ MODELI APROKSIMACIJE

Model aproksimacije nultog reda - Svaki karakter je statistički nezavistan od svih drugih karaktera i verovatnoće svih 27 karaktera su jednake. Ako je ovaj model tačan, onda bi niz karaktera iz ove knjige izgledao ovako (svi primeri su preuzeti iz rada (Shannon, 1948)) :

xfoml rxkhrjffuj zlpwcfwkcyj ffjeyvkcqsghyd qpaamkbzaacibzlhjqd

Dobijeni tekst nije proizvod jednog inteligentnog bića. Rezultat više deluje kao posledica nasumičnog udaranja po tastaturi.

Model aproksimacije prvog reda - poznato je da se u prirodnim jezicima neka slova javljaju češće od drugih. Na primer, u engleskom jeziku slova „a“ i „e“ su češći nego „k“ i „z“. Dakle, u ovom modelu, karakteri su i dalje nezavisni jedan od drugog, ali postoji verovatnoća distribucije karaktera koja odgovara jednom jeziku. Tipičan tekst za ovaj model izgleda ovako:

ocroh hli rgwr nmiehwis eu ll nbneseyba th eei alhenhttpa oobttva nah brl

Model aproksimacije drugog reda - kod prethodna dva modela pretpostavlja se statistička nezavisnost od jednog do sledećeg karaktera. Distribucija karaktera ne odražava u potpunosti strukturu jezika. Sa druge strane, postoje karakteri koji se češće nalaze uz neke karaktere. Ovo znači da postoji zavisnost između karaktera. U ovom modelu, sadašnji karakter zavisi od prethodnog karaktera, ali je uslovno nezavisan od svih prethodnih karaktera. Na primer, slovo „u“ se retko javlja (verovatnoća = 0.022). Međutim, obzirom da je prethodni karakter „q“, verovatnoća karaktera „a“ je sada mnogo veća (verovatnoća = 0.995). Tipičan primer za ovaj model će izgledati ovako:

**on ie antsoutinys are t inctore st be s deamy achin d ilonasive tucoowe at
teasonare fuso tizin andy tobe seace ctisbe**

Model aproksimacije trećeg reda - treći model je nastavak drugog modela aproksimacije. Ovde karakter zavisi od prethodna dva karaktera, ali je uslovno nezavisan od svih prethodnih karaktera. U ovom modelu, distribucija varira od toga šta predstavljaju. Tipičan tekst za ovaj model će izgledati ovako:

**in no ist lat whey cratict froure birs grocid pondenome of demonstures of
the reptagin is regoactiona of cre**

Sada je sličnost sa obični engleskim tekstom poprilično uočljiva na svakom od navedenih koraka.

Opšti model aproksimacije - U ovom modelu, knjiga je proizvoljni stacionarni slučajni proces. Statističke osobine ovog modela su previše kompleksne da bi se razmatralo praktično. Ovaj model je interesantan samo sa teorijskog stanovišta.

2.3.3. ENTROPIJA IZVORA U ODNOSU NA APROKSIMACIJU MODELA

Entropija izvora je broj koji zavisi samo od statističke prirode izvora. Ako izvor ima jednostavan model, onda se ovaj broj može lako izračunati. Sledi jedan proizvoljan izvor:

$$X = \{X_1, X_2, X_3, \dots\},$$

čije će se vrednosti odnositi na engleski jezik.

Model aproksimacije nultog reda - karakteri su statistički nezavisni, svako slovo alfabeta podjednako verovatno može doći u obzir. Neka m bude veličina alfabeta. U ovom slučaju, entropija izvor je određena sa izrazom:

$$H = \log 2m,$$

bitova po karakteru. Za engleski tekst, veličina alfabeta $m = 27$. Dakle, ako je ovo bio precizan model za engleski jezik, onda entropija izvora je $H = 4.75$ bitova po karakteru.

Model aproksimacije prvog reda - karakteri su statistički nezavisni, ukupan broj karaktera odgovara veličini alfabeta i verovatnoća svakog i -tog karaktera je poznata. Entropija izvora u tom slučaju je:

$$H = \sum_{i=1}^m p_i \log 2 p_i.$$

Koristeći aproksimaciju prvog reda, entropija engleskog teksta je 4.07 bitova po karakteru, ako bi ovo bio odgovarajući model izvora.

Model aproksimacije drugog reda - $P_{j|i}$ neka bude uslovna verovatnoća sadašnjeg karaktera j , budući da je poznat prethodni karakter i . Entropija je:

$$H = \sum_{i=1}^m P_i \sum_{j=1}^m P_{i|j} \log 2 P_{i|j},$$

Koristeći aproksimaciju modela drugog reda, entropija engleskog teksta će biti 3.36 bita po karakteru, pod uslovom da je model ispravan.

Model aproksimacije trećeg reda - $P_{k|i,j}$ neka bude uslovna verovatnoća sadašnjeg karaktera k , budući da su poznata prethodna dva karaktera i, j . Entropija je:

$$H = \sum_{i=1}^m P_i \sum_{j=1}^m P_{j|i} \sum_{k=1}^m P_{k|j,i} \log_2 P_{k|j,i},$$

Opšti model aproksimacije - B_n predstavlja prvi n karakter. Entropija u opštem slučaju data:

$$H = \lim_{n \rightarrow \infty} \frac{1}{n} \sum p(B_n \log_2 p(B_n))$$

gde je suma veća od m^n mogućih vrednosti B_n . Praktično je nemoguće izračunati entropiju sa gore navedenom funkcijom. Koristeći metod intuitivnog, Šenon je bio u stanju da proceni entropiju za 27 karaktera engleskog alfabeta, gde je entropija po jednom karakteru 2.3 bita. Više detalja o ovome, moguće je pronaći u Šenonovom zborniku.

2.3.4. ŠENONOVA TEOREMA ZA KODOVANJE BEZ GUBITAKA

Šenonova teorema za kodovanje izvora zasniva se na konceptu blokovskog kodovanja. Da bi ilustrovali ovaj koncept, uvodimo poseban izvor informacija u kojem se alfabet sastoji samo od dva slova:

$$A = \{a, b\}$$

Blok kod za aproksimaciju nultog reda – u ovom modelu svaki karakter je mapiran u jedan bit.

B_1	$p(B_1)$	Kodna reč
a	0.5	0
b	0.5	1
R=1 bit		

PRIMER

Originalni podaci	a	a	a	a	a	a	a	a	a	a	b	b	b	b	b	a	a	a	a	a
Kompresovani podaci	0	0	0	1	1	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0

U ovom slučaju koristili smo 20 bitova za predstavljanje 20 karaktera. Prosečni broj bitova po jednom karakteru je 1 bit.

Blok kod aproksimacija drugog reda – u ovom modelu svaki karakter može da bude predstavljen sa jednim, dva ili tri bita.

B_2	$p(B_2)$	Kodna reč
aa	0.45	0
bb	0.45	10
ab	0.05	110
ba	0.05	111
R=0.71 bitova		

PRIMER

Originalni podaci	aa	aa	aa	aa	aa	bb	bb	ba	aa	aa
Kompresovani podaci	0	0	0	0	0	10	10	111	0	0

U ovom slučaju koristili smo 14 bitova za predstavljanje 20 karaktera. Prosečni broj bitova po jednom karakteru je 0.7 bita.

Blok kod aproksimacija trećeg reda – po tri karaktera su predstavljena binarnom sekvencom dužine od jednog do šest bitova.

B_3	$p(B_3)$	Kodna reč
aaa	0.405	0
bbb	0.405	10
aab	0.045	1100
abb	0.045	1101
bba	0.045	1110
baa	0.045	11110
aba	0.005	111110
bab	0.005	111111
R= 0.55 bitova		

PRIMER

Originalni podaci	aaa	aaa	aaa	abb	bbb	aaa
Kompresovani podaci	0	0	0	1101	10	0

U ovom slučaju koristili smo 10 bitova za predstavljanje 18 karaktera. Prosečni broj bitova po jednom karakteru je 0.55 bitova.

2.4. LABORATORIJSKI RAD

VEŽBA 1.

Merenje količine informacije, uvodna napomena:

Ako je verovatnoća poruke veća, poruka sadrži manju količinu informacije i obrnuto. Količina informacije ili informacioni sadržaj poruke x definisan je logaritamskom funkcijom:

$$I_x = \log_2 \frac{1}{p_x} = -\log_2 p_x,$$

gde je p_x verovatnoća poruke x .

Obzirom da koristimo logaritam sa osnovom dva, tada se količina informacija meri u bitovima.

Neki informacioni izvor generiše 4 poruke jednake verovatnoće $1/4$. Odrediti količinu informacije sadržane u svakoj poruci i ukupnu količinu informacije u ove četiri poruke.

$$\begin{aligned} I_M &= \log_2 \frac{1}{p_1} + \log_2 \frac{1}{p_2} + \log_2 \frac{1}{p_3} + \log_2 \frac{1}{p_4} = \\ &= \log_2 \frac{1}{\frac{1}{4}} + \log_2 \frac{1}{\frac{1}{4}} + \log_2 \frac{1}{\frac{1}{4}} + \log_2 \frac{1}{\frac{1}{4}} = \\ &= 2 + 2 + 2 + 2 = 8 \text{ bitova} \end{aligned}$$

Sada, ako želimo da kodujemo binarno ove 4 poruke, koristi se sledeći binarni kod (svaka poruka je kodovana sa po 2 bita):

Poruke	Kodovi
M1	00

M2	01
M3	10
M4	11

Rešenje zadatka.

Niz sa verovatnoćama:

```
>> p = [0.25, 0.25, 0.25, 0.25];
```

Količina informacije u jednoj poruci:

```
>> -log2(p(1))
```

```
ans = 2
```

Ukupna količina informacije:

```
>> -log2(p(1))-log2(p(2))-log2(p(3))-log2(p(4))
```

```
ans = 8
```

VEŽBA 2

Entropija, uvodna napomena:

Entropija nam daje prosečnu količinu informacije za jedan informacioni izvor i definisana je izrazom,

$$H = - \sum_{i=0}^{L-1} p_i \log p_i,$$

bitova po jednoj poruci.

Neki izvor informacije generiše 6 poruka sa sledećim verovatnoća: $p \{1/4, 1/4, 1/8, 1/8, 1/8, 1/8\}$. Odrediti pojedinačne količine informacije u svakoj poruci, ukupnu količinu informacije i entropiju izvora.

Pojedinačne količine informacije sadržane u porukama i ukupna količina informacije:

$$I_M = \log_2 \frac{1}{4} + \log_2 \frac{1}{4} + \log_2 \frac{1}{8} + \log_2 \frac{1}{8} + \log_2 \frac{1}{8} + \log_2 \frac{1}{8} =$$

$$= 2 + 2 + 3 + 3 + 3 + 3 = 16 \text{ bitova}$$

Entropija izvora:

$$H_M = -\frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{8} \log_2 \frac{1}{8} =$$

$$= -0.25 \log_2 0.25 - 0.25 \log_2 0.25 - 0.125 \log_2 0.125 - 0.125 \log_2 0.125 -$$

$$0.125 \log_2 0.125 - 0.125 \log_2 0.125 =$$

$$= 0.5 + 0.5 + 0.375 + 0.375 + 0.375 + 0.375 = 2.5 \text{ bita}$$

Sada, ako želimo da kodujemo binarno ovih 6 poruka, koristi se sledeći binarni kod (svaka poruka je kodovana sa po 3 bita. Maksimalan broj poruka koji možemo da kodujemo sa 3 bita je 8.

Poruke	Kodovi
M1	000
M2	001
M3	010
M4	011
M5	100

M6	101
-----------	-----

Jednostavan primer kodovanja:

Poruka = **M1M4M6M2**;

Kodovana poruka = 000011101001;

Dekodovana poruka = **M1M4M6M2**;

Rešenje zadatka.

Niz sa verovatnoćama:

```
>> p = [0.25, 0.25, 0.125, 0.125, 0.125, 0.125];
```

Količina informacije u svakoj poruci:

```
>> -log2(p)
```

```
ans = 2 2 3 3 3 3
```

Ukupna količina informacije:

```
>> sum(ans)
```

```
ans = 16
```

Entropija izvora:

```
>> -sum(p.*log2(p + (p == 0)))
```

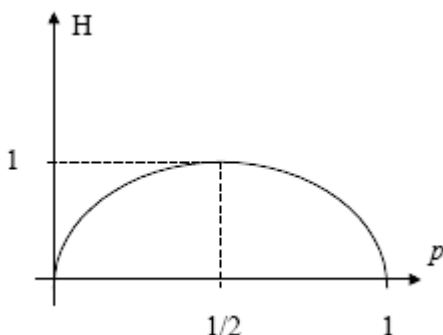
```
ans = 2.5000
```

VEŽBA 3.

Entropija binarnog izvora, uvodna napomena:

Kod binarnog informacionog izvora, izvor generiše samo dve poruke, tada entropija izvora zavisi od verovatnoće poruka. Entropija binarnog izvora definisana je izrazom:

$$H = -p \log p - (1 - p) \log(1 - p).$$



Slika 2.14 Entropija binarnog inf. izvora sa porukama jednakih verovatnoća

Binarni informacioni izvor generiše poruke sa verovatnoćama: $p\{3/4, 1/4\}$. U tom slučaju entropija izvora je:

$$\begin{aligned} H &= -\frac{3}{4} \log_2 \frac{3}{4} - (1 - \frac{3}{4}) \log_2 (1 - \frac{3}{4}) = \\ &= -0.75 \log_2 0.75 - 0.25 \log_2 0.25 = \\ &= 0.311 + 0.5 = 0.811 \text{ bitova inf./bit} \end{aligned}$$

Ukoliko binarni informacioni izvor generiše poruke sa jednakim verovatnoćama: $p\{1/2, 1/2\}$, tada je entropija izvora maksimalna i rešenje je predstavljeno na (Slika 2.14). Uvrstimo verovatnoće u sledeći izraz:

$$H = -\frac{1}{2} \log_2 \frac{1}{2} - (1 - \frac{1}{2}) \log_2 (1 - \frac{1}{2}) =$$

$$\begin{aligned}
 &= -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = \\
 &= 0.5 + 0.5 = 1 \text{ bit inf./bit}
 \end{aligned}$$

Iz rešenja vidimo da je entropija H maksimalna (1 bit).

Rešenje zadatka.

Nizovi sa verovatnoćama oba izvora:

$$>> p1 = [0.75, 0.25];$$

$$>> p2 = [0.5, 0.5];$$

Entropija prvog izvora:

$$>> H1 = -(p1(1)) \cdot \log_2(p1(1)) - (1-(p1(1))) \cdot \log_2(1-p1(1))$$

$$H1 = 0.8113$$

Entropija drugog izvora:

$$>> H1 = -(p2(1)) \cdot \log_2(p2(1)) - (1-(p2(1))) \cdot \log_2(1-p2(1))$$

$$H1 = 1$$

VEŽBA 4.

Odrediti entropiju binarnog niza A dužine 50 bitova u odnosu na njegovu uniformnu distribuciju. Niz je generisan na slučajan način, a kao izvor korišćen je atmosferski šum.

$$A = [10000110100011000111011000101010111110001010100010]$$

U prvom koraku, prebrojaćemo sve parove sačinjene od nula i jedinica (00, 01, 10, 11). Ukupan broj mogućih parova je 4. Parove ćemo tretirati kao poruke ovog verovatnosnog skupa $S = \{00, 01, 10, 11\}$.

Vrednosti koje smo predstavili sa p_{s_i} su verovatnoće poruka S u binarnom nizu.

$$p_{S_{00}} = 0.24, p_{S_{01}} = 0.12, p_{S_{10}} = 0.48, p_{S_{11}} = 0.16;$$

Entropiju sada određujemo na način kao u primeru 2, zatim dobijenu vrednost delimo sa brojem 2 da bismo dobili prosečnu vrednost entropije po jednom bitu.

$$\begin{aligned} H &= -p_{s_{00}} \log_2(p_{s_{00}}) - p_{s_{01}} \log_2(p_{s_{01}}) - p_{s_{10}} \log_2(p_{s_{10}}) - p_{s_{11}} \log_2(p_{s_{11}}) = \\ &= -0.24 \log_2(0.24) - 0.12 \log_2(0.12) - 0.48 \log_2(0.48) - 0.16 \log_2(0.16) = \\ &= 1.7925 / 2 = 0.8962 \text{ bit} \end{aligned}$$

Rezultat H od 0,8962 bita predstavlja prosečnu količinu informacije koja je sadržana po jednom bit-u za dati binarni niz. Ako pomnožimo ukupan broj bita iz niza sa dobijenom vrednosti entropije $H = 0,8962$, dobićemo ukupno 44.81 bita informacije. Možemo da zaključimo da ovaj niz nije apsolutno slučajan, obzirom da je jedinica mere za količinu informacije 1 bit.

Sada nam je možda malo jasnije zašto je toliko teško pronaći informacioni izvor koji će generisati čisto slučajne binarne nizove. U praksi je to uglavnom slučaj da se koriste pseudo slučajni generatori. Pseudo generatori otklanjaju sva prirodna nesavršenstva pravih slučajnih nizova. U tom slučaju dobijamo i pseudo sigurnost. Za potvrdu ispravnosti pseudo generatora, neophodno je sprovesti niz testova koje je predložio NIST, za proveru postojanja osobina slučajnih nizova.

VEŽBA 5:

Lokalna entropija i entropija slike, uvodna napomena:

„Šenonova entropija“ je prvi put uvedena od strane Šenona 1948. Entropija u toj upotrebi termina se definiše kao očekivani informacioni sadržaj nekog događaja. Ako je poznata verovatnoća za neki događaj, onda se entropija može izračunati kao:

$$H = - \sum_{i=1}^l \log(p_i(x_i)),$$

gde se p odnosi na verovatnoće simbola histograma slike. U aplikacijama, koristi se logaritam osnove 2 zbog lakšeg rukovanja sa binarnim informacionim sadržajem. Za posmatrane podatke dobijeni rezultat se može tumačiti kao relativna frekvencija događaja.

Lokalna entropija zavisi od izbora veličine prozora preko koga se određuje entropija svakog piksela. Broj prozora jednak je ukupnom broju piksela na slici. Metod vraća niz entropijskih vrednosti H_i gde je $i = m * n$ za 2D sliku.

Svaki piksel je centar jednog prozora kome se posebno određuje vrednost entropije. Dobijene vrednosti izražene su preko logaritma sa osnovom 2, čime je određena jedinica mere za informaciju 1 bit. Izabrani metod nam dozvoljava da menjamo veličinu prozora koji je kvadratne dimenzije.

Menjanje veličine prozora dosta podseća na Šenonove modele aproksimacije koji se primenjuju za modelovanje informacionih izvora prirodnog jezika. Vrednost lokalne entropije se povećava povećanjem veličine prozora.

U ovom primeru pokazaćemo kako možemo lokalno da izračunamo informacionu vrednost 2D slike. Metod koji ćemo da koristimo je metod za računanje lokalne entropije. Metod je dobro poznat u oblasti za obradu slika (engl. *Image processing*).

Izabrana slika:



Slika 2.15 Originalna slika (rgb 24)

Učitavanje i prikaz slike:

```
>> slika = imread('singidunum.JPG');  
  
>> imshow(slika)
```

Konvertovanje slike iz RGB formata u 8-bitnu sliku (engl. *Gray-scale*):

```
>> slika = rgb2gray(slika);
```

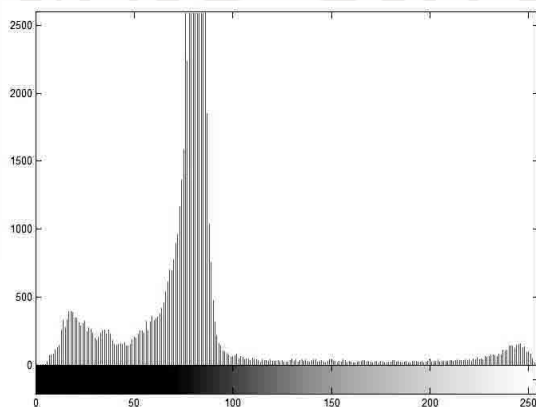
```
>> imshow(slika)
```



Slika 2.16 Gray-scale (8 bit)

Slika konvertovana iz RGB u nijanse sive koristi 8 bitova za opis jednog piksela (nijanse sive su iz opsega od 0 do 255). U nastavku prikazaćemo histogram ili dijagram frekvencija od koga zavisi vrednost entropije.

```
>> imhist(slika);
```



Slika 2.17 Histogram 8-bitne slike

Određivanje verovatnoće za svaki piksel na slici:

```
>> [brojac, simboli]=imhist(slika);
```

```
>> p = brojac / (255*341);
```

Ukoliko posmatramo sliku kao informacioni izvor a piksele kao simbole, možemo sada kada su nam poznate verovatnoće odrediti entropiju izvora:

```
>> H = -sum(p.*log2(p + (p == 0)))
```

```
H = 5.8359
```

Dobijena vrednost H predstavlja prosečnu količinu informacije po jednom pikselu.

Za originalnu sliku bilo nam je potrebno $((255 \cdot 341) \cdot 8) = 695640$ bitova, sada kada smo izračunali entropiju i ukoliko kodujemo preko Hafmanovog algoritma za kodovanje, potrebno nam je $((255 \cdot 341) \cdot 5.8359) = 507460$ bitova. Na ovaj način povećali smo brzinu prenosa, tako što smo smanjili kodnu sekvencu za 188180 bitova.

U sledećem koraku posmatraćemo lokalno prostiranje entropije na ovoj 2D slici. Postavićemo sledeće pitanje:

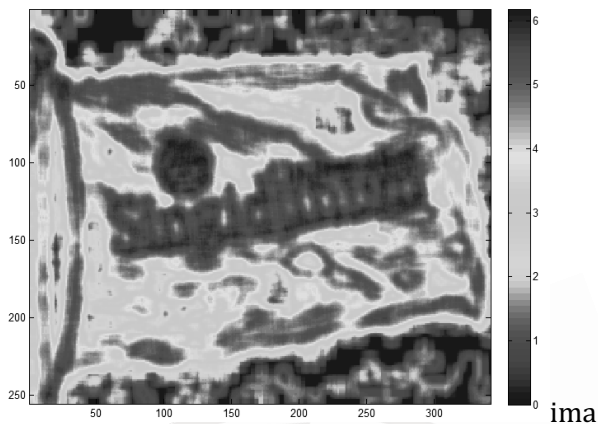
Koji regioni slike sadrže najmanji informacioni sadržaj?

Računamo lokalnu entropiju za veličinu prozora $[9 \times 9]$ i prikazujemo rezultat preko toplotne mape:

```
>>prozor = true(9);
```

```
>> LH = entropyfilt(slika);
```

```
>> imagesc(LH);figure(gcf);
```

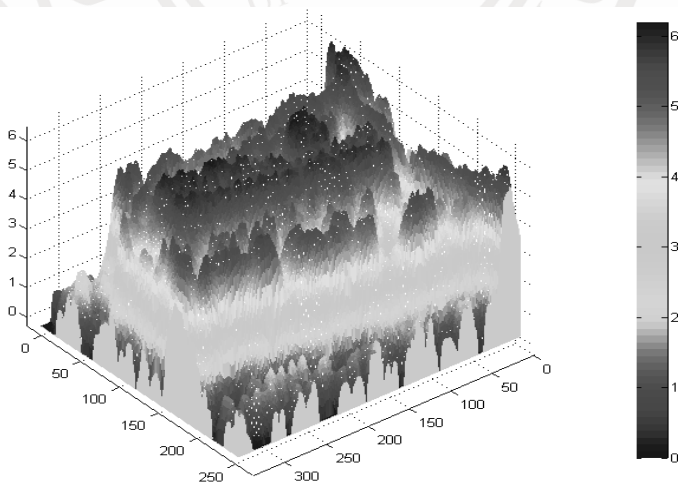


Slika 2.18 Prikaz lokalne entropije preko toplotne mape

Regioni slike označeni sa nijansama crvene boje predstavljaju regione sa najvećom lokalnom entropijom, dok sa nijansama plave boje predstavljeni su regioni sa niskom lokalnom entropijom.

Sledi 3D toplotna entropijska mapa, preko koje možemo preciznije da označimo regione sa visokom entropijom.

```
>> meshz(LH);figure(gcf);
```



Slika 2.19 3D entropijska toplotna mapa

	1	2	3
52	51	[0,0,1,0,1,1,1,0,1]	
53	52	[0,0,0,1,1,0,0,1,1]	
54	53	[0,0,0,0,1,0,0,1,1]	
55	54	[0,0,0,0,1,0,0,0,1]	
56	55	[0,0,0,0,1,1,1,1,0]	
57	56	[0,1,1,1,1,1,0,0]	
58	57	[0,0,0,0,1,0,0,0,0]	
59	58	[0,1,1,1,1,1,1,1]	
60	59	[0,1,0,0,1,1,1,1]	
61	60	[0,1,1,1,0,1,0]	
62	61	[0,1,1,0,1,0,1,0]	
63	62	[0,1,1,0,0,0,0,0]	
64	63	[0,1,0,0,1,1,0,0]	
65	64	[0,0,1,0,1,0,1,1]	
66	65	[0,0,0,1,1,0,1,0]	
67	66	[1,1,0,1,0,0,1]	
68	67	[1,0,0,0,0,1,1]	
69	68	[0,1,1,0,0,0,1]	
70	69	[0,1,1,0,0,1,1]	
71	70	[0,1,0,0,0,1,1]	
72	71	[0,0,1,0,1,0,0]	
73	72	[0,0,0,0,1,1,0]	
74	73	[1,0,0,0,1,1]	
75	74	[0,1,1,0,1,1]	
76	75	[0,1,0,0,0,0]	
77	76	[0,0,0,1,0]	
78	77	[1,0,0,1,1]	
79	78	[0,1,1,1,0]	
80	79	[0,0,1,0,0]	
81	80	[0,0,0,0,0]	
82	81	[0,1,0,1]	
83	82	[0,0,1,1]	
84	83	[1,0,1,0]	
85	84	[1,0,1,1]	
86	85	[1,1,1]	
87	86	[1,1,0,0]	
88	87	[0,0,0,1,1,1]	
89	88	[1,1,0,1,1,0]	
90	89	[0,1,0,0,1,0,1]	
91	90	[0,0,0,0,1,1,1,0]	
92	91	[1,0,0,0,0,0,0,1]	
93	92	[0,0,1,0,1,0,1,0,1]	
94	93	[0,1,1,1,1,1,1,0,0]	
95	94	[1,0,0,1,0,0,1,1,0]	
96	95	[0,0,1,0,1,1,0,0,0,0]	
97	96	[0,0,1,0,1,1,0,1,1,1]	
98	97	[0,1,1,0,0,0,0,1,0,0]	

Slika 2.20 Jedan deo Hafmanovog rečnika za sliku iz primera 5.

U prvoj koloni su nijanse sive kao simboli poruke, a u drugoj koloni verovatnoće.

3. EFIKASNO KODOVANJE INFORMACIJA

U ovom poglavlju date su osnove kodovanja diskretnog izvora informacija u cilju kompresije poruka. Postavljeni su uslovi koje moraju da zadovolje kodovi u cilju efikasne kompresije, kao i fundamentalne granice kompresije podataka i metode konstrukcije efikasnih kompresionih kodova.

Prvi praktični problem koji je moguće rešavati zahvaljujući Šenonovom konceptu mera neodređenosti jeste efikasno kodovanje diskretnog izvora informacija u formi niza simbola.

Pokazaće se da je entropija izvora fundamentalno ograničenje za kompresiju podataka, odnosno da ona određuje minimalnu očekivanu vrednost dužine koda.

Ali, pre svega razmotrimo pitanje:

Šta je razlog da uopšte nešto kodujemo?

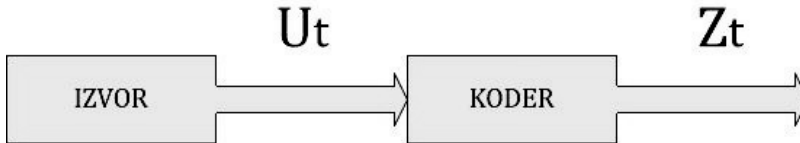
Postoje tri osnovna razloga za to:

1. Kodovanje radi kompresije (izbacivanje redundanse)
2. Kodovanje u cilju kvalitetnog prenosa u uslovima šuma na kanalu (dodavanje redundanse)
3. Kodovanje radi tajnosti - poruke su dostupne samo autorizovanim korisnicima

Ova lekcija će se baviti prvim aspektom efikasnog kodovanja diskretnog izvora informacija.

3.1. KODOVANJE JEDNE SLUČAJNE VELIČINE

Izvor informacija je generator poruka odnosno sekvence simbola.



Slika 3.1 Kodovanje informacionog izvora

Osnovna šema kodovanja izvora (engl. *Source coding*): simbol izvora U_t u trenutku t se transformiše u odgovarajuću kodnu reč Z_t , na (Slika 3.1).

Simboli su elementi skupa koga nazivamo *alfabet*.

Informacioni izvor *je diskretan* ako je alfabet konačan.

Dimenzija alfabeta se naziva i *arnost* izvora.

Razmatraćemo samo poruke konačne dužine. Na primer, novine se mogu smatrati jednim diskretnim izvorom informacija, nad alfabetom datog jezika, dok su poruke novinski tekstovi.

Koder transformiše poruke izvora u *kodne reči*.

Kodne reči su sekvence simbola iz *kodnog alfabeta*, koji je u opštem slučaju različit od alfabeta poruka.

Formalno, informacioni izvor U_t , odnosno U bez indeksa t ako vremenski indeks nije relevantan, to je zapravo slučajan proces nad datim alfabetom $\mathcal{V}_U$. Jednostavnije rečeno, to je niz slučajnih veličina iz $\mathcal{V}_U$.

Izvor emituje simbole u svakom trenutku t sa verovatnoćama $P(U_t = u_i)$.

Izvor je bez memorije, ukoliko verovatnoća simbola u_i ne zavisi od prošlih emitovanih vrednosti:

$$\forall t \geq 1 \forall u_i \in \mathcal{V}_U \quad P(U_t = u_i | U_1 \dots U_{t-1}) = P(U_t = u_i)$$

Izvor informacija je stacionaran, ukoliko verovatnoće emitovanja simbola $P(U_t = u_i)$ ne zavise od vremena t .

Koder možemo posmatrati kao preslikavanje $Z = f(U)$ iz alfabeta izvora v_U u skup kodnih reči v_Z . Ako sa Z^* označimo skup svih konačnih sekvenci sastavljenih od simbola iz kodnog alfabeta v_Z , tada su kodne reči zadatog koderu jedan podskup skupa Z^* .

DEFINICIJA 3.1 Nesingularni kodovi.

Kod jednog diskretnog informacionog izvora se naziva nesingularnim, kada se različitim simbolima izvora korespondiraju različite kodne reči. Formalno,

$$u_i \neq u_j \rightarrow z_i \neq z_j,$$

pri čemu je z_i kodna reč koja korespondira simbolu izvora u_i . U ovoj lekciji bavićemo se samo nesingularnim kodovima.

PRIMER 3.1 Kodovanje konačnog izvora.

Morzeov kod služi za kodovanje slovnih i brojčanih znakova. Alfabet Morzeovog koda čine samo dva simbola: tačka (·) i crta (-). Na primer, A, E i K se koduju sa “·”, “·-” i “-·-”, respektivno.

Budući da je cilj koderu kodovanje poruka, a ne pojedinačnih simbola, od interesa su kodovi koji se mogu na jedinstven način dekodovati. Takvi kodovi se zovu nedvosmisleni ili jednoznačno dekodivi kodovi. Kod ovih kodova bilo koja sekvenca kodnih reči korespondira jednoznačno samo jednoj poruci informacionog izvora.

DEFINICIJA 3.2 Jednoznačno dekodivi kodovi.

Kod diskretnog izvora informacija se naziva jednoznačno dekodivim, ako i samo ako bilo koja konačna sekvenca kodnih reči jednoznačno odgovara jednoj poruci izvora.

PRIMER 3.2 Nejednoznačan kod.

Neka se izvor sastoji od tri simbola a, b i c . Poruka ovog izvora je svaki niz koji se sastoji od ova tri simbola, npr. $aaabcaacc$. Kod definisan preslikavanjem

$$a \rightarrow 1 \quad b \rightarrow 00 \quad c \rightarrow 11$$

je nejednoznačan. Na primer, prilikom kodovanja niza $aaaa$, dobiće se kodna reč 1111 , što je isto kao i da kodujemo niz cc . Ne postoji način prilikom dekodovanja koji će biti u stanju da razlikuje šta je kodovana poruka $aaaa$ ili cc .

PRIMER 3.3 Jednoznačno dekodiv kod.

Neka je izvor informacija kao u prethodnom primeru. Neka je kod definisan sa

$$a \rightarrow 1 \quad b \rightarrow 00 \quad c \rightarrow 10$$

Može se dokazati da je ovaj kod jednoznačan. Npr. niz 10000 se dekoduje u niz abb , a niz 1000 u cb . Među jednoznačnim kodovima od posebnog značaja su tzv. prefiksni kodovi (engl. *Prefix – Free Code*).

Jedan niz z dužine n ($n \geq 1$) je prefiks nekog drugog niza z' ako i samo ako je prvih n simbola u z' jednako egzaktno nizu z . Npr. $abba$ je prefiks niza $abbabc$.

DEFINICIJA 3.3 Prefiksni kodovi.

Za jedan kod se kaže da je prefiksni ako u njemu ni jedna kodna reč nije prefiks nekoj dugoj kodnoj reči.

PRIMER 3.4 Prefiksni kod.

Neka je dat izvor sa tri simbola a, b i c . Za ovaj izvor primer jednog prefiksnog koda je,

$$a \rightarrow 1 \quad b \rightarrow 10 \quad c \rightarrow 11$$

dok kod:

$$a \rightarrow 1 \quad b \rightarrow 00 \quad c \rightarrow 10$$

nije prefiksni, budući da je 1 kod za simbol a , istovremeno i prefiks u nizu 10 , koji predstavlja kodnu reči za simbol c .

Odakle potiče naš interes za prefiksnim kodovima?

Odgovor leži u dve osobine, 3.1 i 3.2.

OSOBINA 3.1

Svaki prefiksni kod je jednoznačno dekodiv. Važno je napomenuti da postoje jednoznačni kodovi koji nisu prefiksni, kao što nam to pokazuje Primer 3.3.

DEFINICIJA 3.4 Trenutno dekodivi kod.

Kod je trenutno dekodiv ako i samo ako se svaka kodna reč u bilo kom nizu kodnih reči može dekodovati u istom momentu kada je dostignut kraj kodne reči.

OSOBINA 3.2

Kod je trenutno dekodiv ako i samo ako je prefiksni. Trenutno dekodivi kodovi ne zahtevaju memorisanje prispeli kodnih reči ili čekanje dospeća novih da bi dekodovanje bilo obavljeno. Ovim se štedi i vreme i memorijski prostor dekodera.

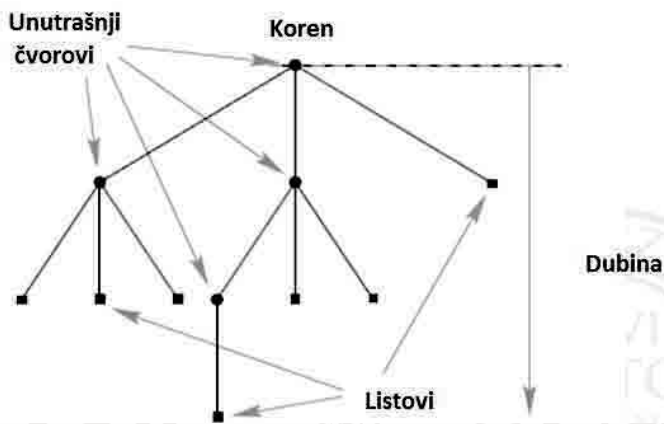
Klase kodova koje smo do sada opisali se međusobno odnose kao na (Slika 3.2).



Slika 3.2 Klase opisanih kodova

3.1.1. N -ARNA STABLA KODOVA

Pogodan način predstave i izučavanja prefiksnih kodova su n -arna stabla. Na slici (Slika 3.3) dati su sastavni elementi jednog stabla.

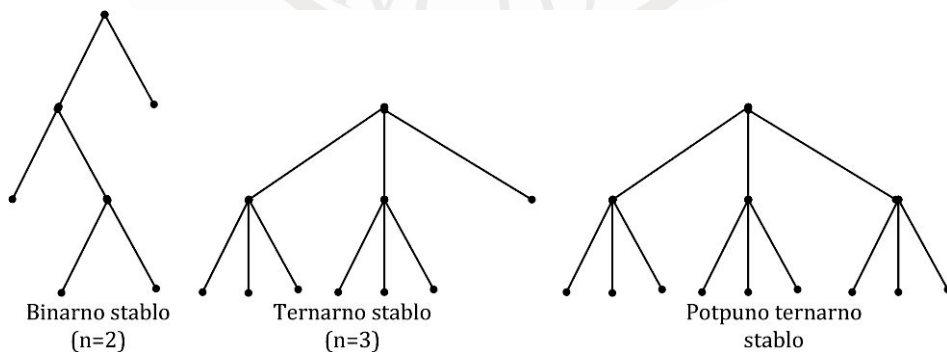


Slika 3.3 Pregled termina vezanih za stabla

DEFINICIJA 3.5 N -arno stablo.

N -arno stablo je stablo u kome svaki unutrašnji čvor ima arnost n , odnosno ima egzaktno n čvorova-dece. Potpuno n -arno stablo je n -arno stablo u kome svi listovi imaju istu dubinu.

PRIMER 3.5 N -arna stabla.



Slika 3.4 Binarno stablo ($n=2$), Ternarno stablo ($n=3$), Potpuno ternarno stablo

Čvor n_1 pokriva čvor n_2 , ukoliko put od korena do n_2 sadrži čvor n_1 .

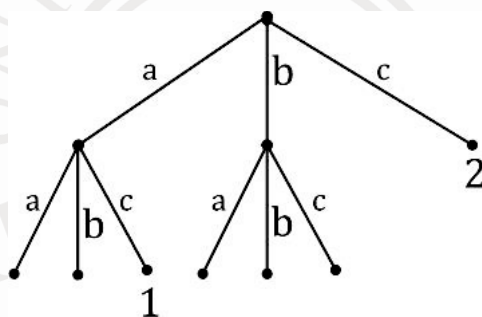
OSOBIKA 3.3

U potpunom n -arnom stablu dubine $d \geq 0$, svaki čvor dubine δ ($0 \leq \delta \leq d$) pokriva tačno $n^{d-\delta}$ listova.

DEFINICIJA 3.6 Kodno stablo

Kodno stablo je n -arno stablo čije su grane označene slovima datog alfabeta dimenzije n , na takav način da se svako slovo pojavljuje samo jedanput u izlaznim granama čvorova. Kodne reči odgovaraju sekvencama oznaka grana duž puteva od korena do listova.

PRIMER 3.6 Kodno stablo.



Slika 3.5 Ternarno kodno stablo

Ternarno kodno stablo: kodna reč predstavljena listom 1 je ac , dok list 2 predstavlja kodnu reč c , (Slika 3.5).

DEFINICIJA 3.7 N -arni kod.

Kod čiji alfabet ima n slova se naziva n -arni kod.

OSOBIKA 3.4

Za svaki n -arni prefiksni kod postoji barem jedno n -arno kodno stablo takvo da svaka kodna reč odgovara sekvenci oznaka na jedinstvenom putu od korena stabla do lista. I obrnuto, svako kodno stablo definiše jedan prefiksni kod. Kodne

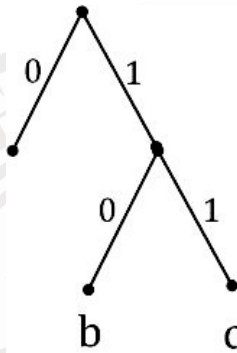
reči ovog prefiksnog koda su određene sekvencom oznaka putanja od korena do svakog lista kodnog stabla.

Ukratko, kodno stablo i prefiksni kod su ekvivalentni objekti.

PRIMER 3.7

Kodno stablo koje odgovara kodu iz primera 3.4

$a \rightarrow 0$ $b \rightarrow 10$ $c \rightarrow 11$

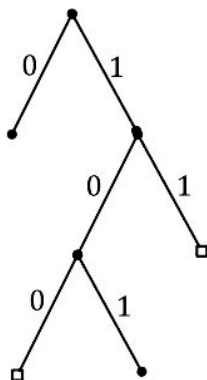


Slika 3.6 Kodno stablo

Prema standardnoj konvenciji prikazivanja, listovi su označeni simbolima izvora, a kodne reči se dobijaju kao nizovi oznaka grana na putanjama od korena do listova.

Primetimo da reprezentacijom prefiksnih kodova pomoću kodnih stabala, neki listovi ne korespondiraju nijednoj kodnoj reči. Takvi listovi se nazivaju neiskorišćeni listovi.

Na primer, binarno kodno stablo koje odgovara kodu $\{0,101\}$ ima dva neiskorišćena lista.



Slika 3.7 Kodno stablo sa dva neiskorišćena lista

Zaista nizovi 11 i 100 ne korespondiraju nijednoj kodnoj reči posmatranog koda.

DEFINICIJA 3.8 Kompletan kod.

Kada nema nekorišćenih listova u odgovarajućem n -arnom kodnom stablu, kod se naziva kompletan kod.

3.1.2. KRAFTOVA NEJEDNAKOST

Sada su nam potrebni uslovi koji mora da budu ispunjeni da bi jedan prefiksni kod postojao. Ispostavlja se da je Kraftova nejednakost potreban i dovoljan uslov.

TEOREMA 3.1 Kraftova nejednakost.

Za svaki D -arni prefiksni kod od N kodnih reči čije su dužine pozitivni celi brojevi $\{l_1, l_2, \dots, l_N\}$, mora važiti:

$$\sum_{i=1}^N D^{-l_i} \leq 1$$

i obrnuto, ako su date dužine kodnih reči koje zadovoljavaju ovu nejednakost, postoji prefiksni kod sa kodnim rečima date dužine. Kada u (3.1) važi jednakost, tada je kod kompletan.

PRIMER 3.8

Za binarni ($D = 2$) kompletni prefiksni kod iz primera 3.4 $\{0,10,11\}$ važi:

$$\sum_{i=1}^N D^{-l^i} = 2^{-1} + 2^{-2} + 2^{-2} = \frac{1}{2} + \frac{1}{4} + \frac{1}{4} = 1$$

Kraftova nejednakost nam kaže da postoji barem jedan ternarni prefiksni kod čije su dužine kodnih reči 1,2,2,4, zato što važi:

$$3^{-1} + 3^{-2} + 3^{-2} + 3^{-4} + \frac{48}{81} \cong 0.57 < 1$$

Ovaj kod ne bi bio kompletan.

Klasična pogrešna upotreba Kraftove nejednakosti je odgovaranje na pitanje da li je kod sa zadatim dužinama kodnih reči prefiksni. Ova nejednakost nam samo daje informaciju kada može postojati jedan prefiksni kod.

Na primer, prvi kod iz primera 3.4 $\{1,00,10\}$ nije prefiksni iako je:

$$\sum D^{-l_i} = 2^{-1} + 2^{-2} + 2^{-2} = 1$$

Teorema 3.1 nam samo kaže da postoji prefiksni kod istih dužina kodnih reči, i zaista on postoji i dat je u istom primeru 3.4, tj. kod $\{0,10,11\}$.

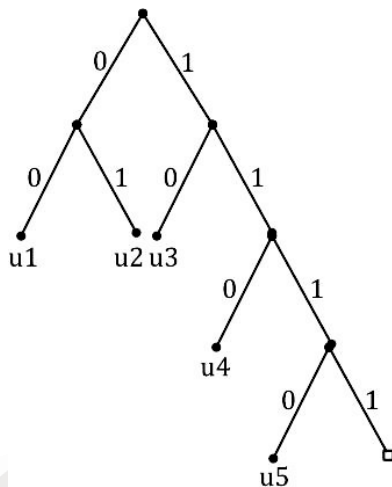
PRIMER 3.9

Da li postoji binarni prefiksni kod sa dužinama kodnih reči $l_1 = 2, l_2 = 2, l_3 = 2, l_4 = 3$ i $l_5 = 4$?

Odgovor je DA, pošto je:

$$\sum_{i=1}^5 2^{-l_i} = \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{6} = \frac{15}{16} < 1.$$

Primer jednog takvog koda je dat na (Slika 3.8).



Slika 3.8 Kodno stablo binarnog prefiksnog koda

U	u_1	u_2	u_3	u_5	u_6
Z	00	01	10	110	1110

PRIMER 3.10

Da li binarni prefiksni kod sa dužinama kodnih reči 1,2,2,3,4 postoji?

Odgovor je NE, budući da važi:

$$\sum_{i=1}^5 2^{-l_i} = \frac{1}{2} + \frac{1}{4} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} = \frac{19}{16} > 1.$$

3.2. EFIKASNO KODOVANJE

Osnovni cilj efikasnog kodovanja je kodovanje datog izvora informacija tako da je srednja dužina koda minimalna. Pod vrlo opštim uslovima na izvor informacija, koji su skoro uvek ispunjeni u praksi, minimizacija srednje dužine koda je ekvivalentno minimizaciji očekivane dužine koda.

DEFINICIJA 3.9 Očekivana dužina koda.

Neka je dat izvor, čiji se simboli u_i emituju sa verovatnoćama p_i , ($0 \leq i \leq N$) i neka su l_i dužine kodnih reči. Očekivana dužina koda, odnosno očekivana dužina kodnih reči je data sa:

$$E[L] = \sum_{i=1}^N p_i l_i$$

Naš cilj je da pronađemo prefiksni kod koji ima $E[L]$ što je moguće manje. Jasno nam je da kraćim kodnim rečima treba kodovati najverovatnije simbole.

Kako odrediti dužine kodnih reči?

Koja je najmanja teorijska vrednost za $E[L]$?

3.2.1. VEROVATNOSNO N-ARNAO STABLO

Prefiksni kodovi se mogu definisati n -arnim stablima u kojima svaka kodna reč odgovara jednom listu stabla, odnosno jedinstvenom putu od korena do tog lista. Budući da izvor emituje simbole sa određenim verovatnoćama, jasno je da će i svaka kodna reč, odnosno list, imati tačno određenu verovatnoću. Neiskorišćenim listovima se po konvenciji dodeljuju verovatnoće jednake 0.

Nakon označavanja listova odgovarajućim verovatnoćama, možemo sračunati i verovatnoće unutrašnjih čvorova, rekurzivno od listova ka korenu, imajući uvek u vidu da je verovatnoća jednog čvora jednaka sumi verovatnoća njegovih deca-čvorova. Nakon ove procedure dobili smo verovatnosno n -arno stablo.

DEFINICIJA 3.10 Verovatnosno n -arno stablo.

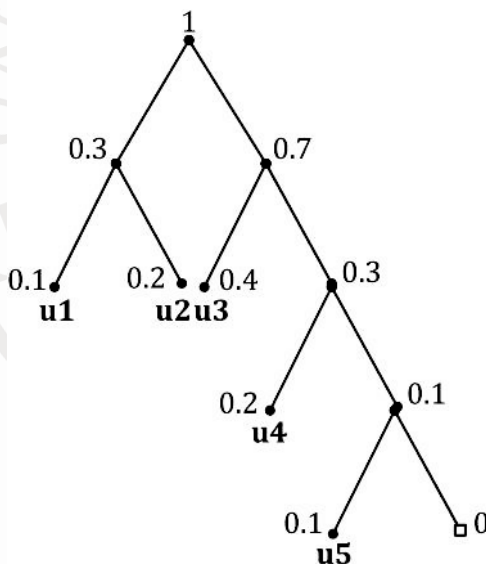
Verovatnosno n -arno stablo je n -arno stablo čijim čvorovima su dodeljene verovatnoće na sledeći način:

1. Korenu je dodeljena verovatnoća 1,
2. Verovatnoća svih ostalih čvorova, uključujući i koren, jednaka je sumi verovatnoća njegovih deca-čvorova.

PRIMER 3.11

Za binarni prefiksni kod iz primera 3.9, neka su verovatnoće kodnih reči $p_1 = p_2 = 0.1$, $p_2 = p_4 = 0.2$, $p_3 = 0.4$.

Tada kao rezultat dobijamo verovatnosno binarno stablo:



Slika 3.9 Rešenje primera 3.11

Primitimo da u verovatnosnom n -arnom stablu suma verovatnoća svih listova mora biti jednaka 1.

LEMA 3.1 Lema dužine puta.

U verovatnosnom n -arnom stablu, prosečna (očekivana) vrednost dubine listova jednaka je sumi verovatnoća unutrašnjih čvorova (isključivanjem listova i uključivanjem korena).

PRIMER 3.12 Lema dužine puta za primer 3.11.

Za primer ovog verovatnosnog binarnog stabla, očekivana dubina listova je jednaka $1 + 0.3 + 0.7 + 0.3 + 0.1 = 2.4$.

Provera, očekivana dužina kodnih reči je $2 \cdot 0.1 + 2 \cdot 0.2 + 2 \cdot 0.4 + 3 \cdot 0.2 + 4 \cdot 0.1 = 2.4$.

DEFINICIJA 3.11 Entropija listova n -arnog verovatnosnog stabla.

Neka je N broj listova verovatnosnog n -arnog stabla i neka su $p_1, p_2, \dots, p_N$ njihove verovatnoće. Pod entropijom listova podrazumevaćemo entropiju definisanu sa:

$$H = \sum_i p_i \log p_i$$

OSOBINA 3.5

Za verovatnosno n -arno stablo koje odgovara prefiksnom kodu informacionog izvora U , važi:

$$H = H(U)$$

DOKAZ OSOBINE 3.4

Neka je Z prefiksni kod izvora U . Po definiciji verovatnoća p_i je verovatnoća i -te kodne reči, pa je stoga $H = H(Z)$. S druge strane budući da je kod nesingularan, odnosno $Z = f(U)$ je injektivno preslikavanje, $H(Z) = H(U)$. Stoga je $H = H(U)$.

DEFINICIJA 3.12 Entropija grananja.

Neka je M broj unutrašnjih čvorova verovatnosnog n -stabla i neka su $P_1, P_2, \dots, P_M$ njihove verovatnoće. Neka su $q_{i1}, q_{i2}, \dots, q_{in_i}$ verovatnoće n_i čvorova-dece (uključujući i listove) unutrašnjeg čvora čija je verovatnoća P_i . Entropija grananja H_i u tom čvoru je data sa:

$$H_i = \sum_{j=1}^{n_i} \frac{q_{ij}}{P_i} \log \frac{q_{ij}}{P_i}, \quad P_i = \sum_{j=1}^{n_i} q_{ij}.$$

PRIMER 3.13

Neka je $M = 5$ čvorova verovatnosnog stabla iz primera 3.9 tako numerisano da je $P_1 = 1, P_2 = 0.3, P_3 = 0.7, P_4 = 0.3, P_5 = 0.1$. Tada je:

$$H = - \sum_{i=1}^5 p_i \log p_i \cong 2.122 \text{ bit.}$$

$$n_1 = 2, \quad q_{11} = 0.3, \quad q_{12} = 0.7,$$

$$H_1 = -0.3 \log 0.3 - 0.7 \log 0.7 \cong 0.881 \text{ bit.}$$

$$n_2 = 2, \quad q_{21} = 0.1, \quad q_{22} = 0.2,$$

$$H_1 = -0.3 \log 0.3 - 0.7 \log 0.7 \cong 0.918 \text{ bit.}$$

$$H_3 \cong 0.985 \text{ bit}, H_4 \cong 0.918 \text{ bit}, H_5 = 0.$$

TEOREMA 3.2 Entropija listova.

Entropija listova verovatnosnog n -arnog stabla jednaka je sumi po svim unutrašnjim čvorovima uključujući koren, entropija grananja čvorova sa težinskim faktorima jednakim njihovim verovatnoćama, tj.:

$$H = \sum_{i=1}^M P_i H_i$$

PRIMER 3.14

Ako nastavimo sa primerom 3.13, prema teoremi 3.2 važi:

$$\begin{aligned} H &= 1 \cdot H_1 + 0.3 \cdot H_2 + 0.7 \cdot H_3 + 0.3 \cdot H_4 + 0.1 \cdot H_5 \\ &\cong 0.881 + 0.3 \cdot 0.918 + 0.7 \cdot 0.985 + 0.3 \cdot 0.918 + 0 \cong 2.122 \text{ bit} \end{aligned}$$

Što se slaže sa direktnim računanjem obavljenim u primeru 3.13.

TEOREMA 3.3

Za bilo koja dva prefiksna koda nad istim informacionim izvorom, kod koji ima kraću očekivanu dužinu ima veću entropiju simbola. Komprimovanje izvora informacija povećava entropiju simbola.

3.2.2. ŠENONOVA TEOREMA O KODOVANJU BEZ PRISUSTVA ŠUMA

Koristeći dosadašnje rezultate, u stanju smo da odredimo fundamentalnu donju granicu očekivane vrednosti prefiksnih kodova datog informacionog izvora.

TEOREMA 3.4 Šenonova teorema kodovanja Deo I

Za bilo koji diskretni izvor informacija bez memorije čija je entropija $H(U)$, očekivana dužina D -arnog prefiksnog koda $E[L]$ tog izvora, zadovoljava:

$$E[L] \geq \frac{H(U)}{\log D}$$

Teorema 3.4 daje jednu donju granicu za očekivanu dužinu koda datog izvora informacija.

Kakva je ta granica?

Da li ima i drugih granica (npr 1)?

Jasno je da ovaj rezultat postaje visokovrednovan samo ako pokažemo da je ta granica najbolja moguća donja granica. Da bi smo ovo pokazali, potrebno je konstruisati konkretan kod čija se očekivana dužina kodnih reči proizvoljno tačno približava ovoj granici.

Prvo ćemo konstruisati efikasan prefiksni kod, koji iako nije optimalan u opštem slučaju se dovoljno dobro približava donjoj granici očekivane vrednosti kodnih reči. Ideja se svodi na izbor d užine l_i kodne reči u_i tako da važi:

$$l_i = \left\lceil -\frac{\log p_i}{\log D} \right\rceil,$$

gde $[x]$ označava celobrojnu vrednost takvu da je:

$$x \leq [x] \leq x + 1.$$

Ovakav kod se naziva Šenon-Fanoov kod, budući da ga je Šenon implicitno dao u svom radu iz 1948 godine, a Fano ga je eksplicitno konstruisao.

- *Da li ovakav kod uvek postoji?*

Odgovor je afirmativan na osnovu Kraftove nejednakosti. Zaista po definiciji operacije je:

$$l_i \geq -\frac{\log p_i}{\log D},$$

pa je:

$$\sum_i D^{-l_i} \leq \sum_i D^{\frac{\log p_i}{\log D}} = \sum_i D^{\log_D p_i} = \sum_i p_i = 1.$$

Sada ćemo istražiti koliko je ovaj kod dobar u odnosu na očekivanu dužinu koda. Po definiciji važi:

$$l_i < -\frac{\log p_i}{\log D} + 1.$$

Ako obe strane pomnožimo sa p_i i sumiramo po i , dobijamo:

$$\sum_i p_i l_i < \frac{-\sum_i p_i \log p_i}{\log D} + \sum_i p_i,$$

$$E[L] < \frac{H(U)}{\log D} + 1.$$

Vidimo da je očekivana vrednost kodnih reči Šenon-Fanoov koda udaljena najviše jedan simbol od donje granice date Teoremom 3.4. To znači da je ovaj kod prilično dobar i da što je entropija izvora veća on je sve bolji. Međutim za male entropije izvora moguće je pronaći znatno bolji kod.

TEOREMA 3.5 Šenonova teorema kodovanja, Deo II

Za bilo koji diskretni izvor informacija bez memorije, čija je entropija $H(U)$, postoji barem jedan D -arni prefiks kod čija je očekivana dužina kodnih reči $E[L]$ data sa:

$$E[L] < \frac{H(U)}{\log D} + 1.$$

PRIMER 3.15

Razmotrimo binarni ($D = 2$) Šenon-Fanoov kod za izvor U sa četiri simbola čije su verovatnoće $p_1 = 0.4, p_2 = 0.3, p_3 = 0.2$ i $p_4 = 0.1$.

Kodne reči su date dužinama:

$$l_1 = -\lceil \log_2 0.4 \rceil = 2,$$

$$l_2 = -\lceil \log_2 0.3 \rceil = 2,$$

$$l_3 = -\lceil \log_2 0.2 \rceil = 3,$$

$$l_4 = -\lceil \log_2 0.1 \rceil = 4.$$

Kod konstruišemo na osnovu postupka datog u dokazu Kraftove nejednakosti.

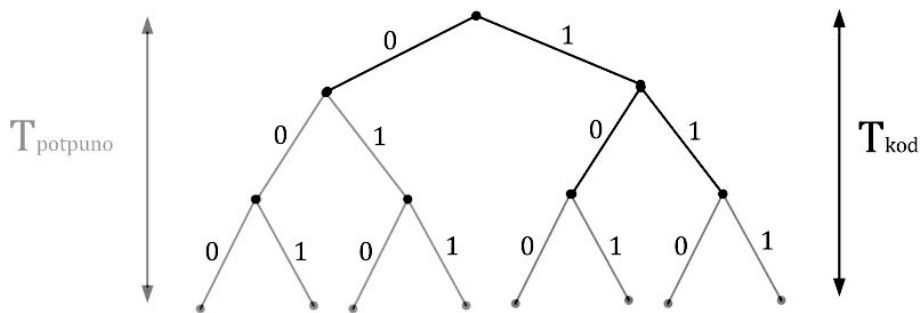
Kako se konstruiše kodno stablo (T_{kod}) ?

Neka su zadate dužine kodnih reči $l_1, l_2, \dots, l_N$

$$L = \max_i l_i + 1$$

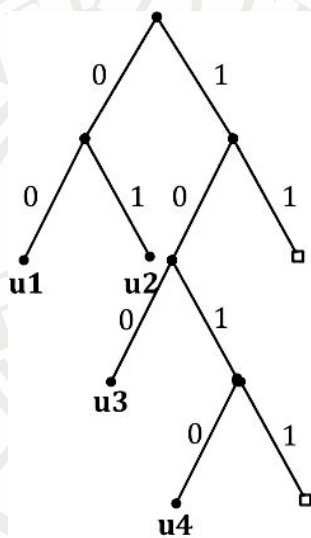
Konstruisati potpuno kodno stablo dubine L ($T_{potpuno}$).

Za čvorove koje proglasimo kodnim rečima izvršiti potkresivanje odgovarajućeg podstabla ispod tog čvora.



Slika 3.10 Potpuno kodno stablo

Nastavak primera 3.15.



Slika 3.11 Primer 3.15.

Na osnovu Leme o dužini puta, imamo:

$$R[L] = 1 + 0.7 + 0.3 + 0.3 + 0.1 = 2.4,$$

Direktan račun daje:

$$H(U) = 0.4 \log 0.4 + 0.3 \log 0.3 + 0.2 \log 0.2 + 0.1 \log 0.1 \cong 1.8 \text{ bit}.$$

Vidimo da je zadovoljena nejednakost:

$$E[L] < \frac{H(U)}{\log D} + 1.$$

Odnosno $2.4 < 1.8 + 1 = 2.8$.

Međutim vidi se da ovaj kod nije optimalan. Kada bi smo upotrebili 4 kodne reči dužine 2, imali bi smo kraću očekivanu dužinu koda $E[L] = 2$.

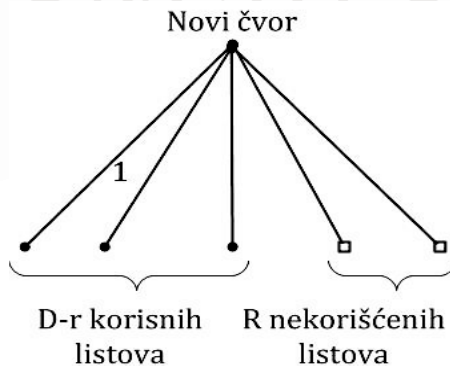
3.2.3. HAFMANOV KOD

Hafmanov algoritam kodovanja izvora informacija bez memorije, pomoću D-arnog prefiksnog koda, daje optimalan kod minimalne moguće očekivane vrednosti kodnih reči.

Neka izvor U ima n simbola $u_1, u_2, \dots, u_n$, sa odgovarajućim verovatnoćama $p_1, p_2, \dots, p_n$.

Hafmanov kod

1. Startujemo sa svih n čvorova, koji će na kraju kodovanja biti listovi. Zatim, izračunajmo ostatak r pri deljenju $1 - n$ sa $D - 1$. (Primetimo da za n parno, r će uvek biti nula) Učinimo svih n čvorova „**aktivnim**“.
2. Formirajmo novi čvor, čija su deca-čvorovi D-r najmanje verovatnih aktivnih čvorova i r neiskorišćenih čvorova (listova).



Slika 3.12 Stablo Hafmanovog koda

Označiti D-r aktivnih čvorova kao „**neaktivne**”. Novo kreirani čvor označiti kao „**aktivan**”. Dodeliti novokreiranom čvoru verovatnoću jednaku sumi upravo deaktiviranih D-r čvorova.

3. Ako postoji samo jedan aktivan čvor tada se procedura okončava. Taj čvor je ujedno i koren konačnog kodnog stabla. U suprotnom, postaviti $r = 0$ i ići na korak 2.

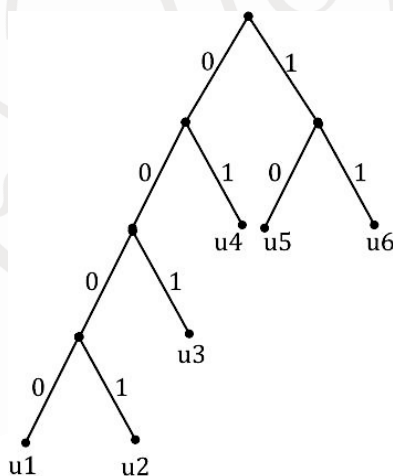
Prefiksni kod koji rezultuje u opisanoj proceduri se naziva Hafmanov kod, budući da je upravo opisani algoritam prvi dao Hafman 50-tih godina XX veka.

PRIMER 3.16 Binarni Hafmanov kod.

Neka je dat izvor U :

U	u_1	u_2	u_3	u_4	u_5	u_6
p_i	0.05	0.1	0.15	0.27	0.20	0.23

Jedan Hafmanov kod za izvor U je dat sa:



Slika 3.13 Stablo Hafmanovog koda za dati primer

z_1	z_2	z_3	z_4	z_5	z_6
0000	0001	001	01	10	11

Verovatnoće unutrašnjih čvorova su date u tabeli:

$v_1 = u_1 \oplus u_2$	$v_2 = v_1 \oplus u_3$	$v_3 = u_5 \oplus u_6$	$v_4 = v_2 \oplus u_4$	$v_5 = v_4 \oplus v_3$
0.15	0.30	0.43	0.57	1

Primetimo da je:

$$E[L] = 2(0.2 + 0.23 + 0.27) + 3(0.15) + 4(0.1 + 0.05) = 2.45$$

Ili pomoću Leme dužine puta:

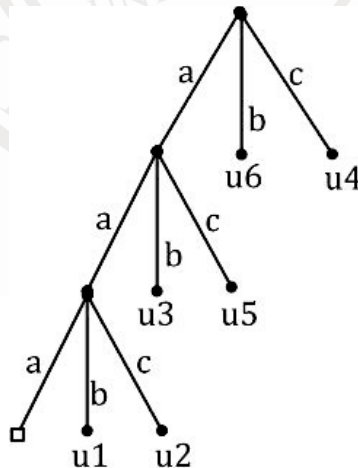
$$E[L] = 1 + 0.57 + 0.43 + 0.30 + 0.15 = 2.45$$

Dok je entropija izvora U :

$$H(U) = \sum_{i=1}^6 p_i \log p_i = 2.42 \text{ bit.}$$

PRIMER 3.17 Ternarni Hafmanov kod.

Neka je izvor informacija U , kao u prethodnom primeru 3.16. Ako upotrebimo ternarni kod $D = 3$, ostatak od $1 - n + 1 - 6 = 5$ pri deljenju sa $D - 1 = 3 - 1 = 2$ iznosi $r = 1$. (Provera $-5 = -3 \cdot 2 + 1$). Stoga treba uvesti jedan neiskorišćeni list. Hafmanov kod u ovom slučaju je dat sa:



Slika 3.14 Kodno stablo za primer 3.17

z_1	z_2	z_3	z_4	z_5	z_6
aab	aac	ab	c	ac	b

$v_1 = u_1 \oplus u_2$	$v_2 = v_1 \oplus u_3 \oplus u_5$	$v_3 = v_2 \oplus u_6 \oplus u_4$
0.15	0.50	1

$$E[L] = 1 + 0.5 + 0.15 = 1.65$$

$$\frac{H(U)}{\log 3} = \frac{2.42}{1.59} = 1.52.$$

3.2.4. OPTIMALNOST HAFMANOVOG KODA

Sada nam je cilj da pokažemo da je Hafmanov kod optimalan, a to znači da ne postoji neki drugi prefiksni kod čija je očekivana dužina kodnih reči manja od očekivane dužine kodnih reči Hafmanovog koda.

Važno je uočiti da postoji jako puno optimalnih kodova. Svaka permutacija kodnih simbola jednog optimalnog koda, takodje daje optimalni kod. Svaka zamena kodnih reči koje odgovaraju jednakoverovatnim simbolima datog izvora, takodje je optimalan kod. Hafmanov algoritam konstruiše samo jedan od njih. Prvo ćemo navesti svojstva optimalnih kodova.

Za jedan kod ćemo reći da je optimalan ukoliko je $\sum_{i=1}^n p_i l_i$ minimalna moguća na skupu svih prefiksnih kodova datog izvora informacija, gde je l_i označena dužina kodne reči simbola u_i koji se emituje sa verovatnoćom p_i .

LEMA 3.2

Za svaki optimalni kod izvora sa n mogućih simbola, važi:

$$p_i > p_j \Rightarrow l_i \leq l_j, \quad i, j = 1, 2, \dots, n.$$

LEMA 3.3

Broj listova u D -arnom stablu je $1 + k(D - 1)$, gde je k broj unutrašnjih čvorova, uključujući koren.

LEMA 3.4

Za dati izvor informacija U , u kodnom stablu koje odgovara optimalnom D -arnom prefiksnom kodu za U , postoji najviše $D - 2$ neiskorišćenih listova i svi se oni nalaze na maksimalnoj dubini kodnog stabla.

Šta više, postoji optimalni D -arni kod za U , u kome su svi neiskorišćeni listovi, deca jednog istog roditeljskog čvora.

LEMA 3.5

Broj neiskorišćenih listova u kodnom stablu koje odgovara optimalnom D -arnom prefiksnom kodu izvora U sa n simbola je jednak pozitivnom ostatku deljenja $n - 1$ sa $D - 1$.

LEMA 3.6

Postoji optimalni D -arni prefiksni kod izvora U sa n simbola ($n \geq 2$) takav da se D -r najmanje verovatnih kodnih reči razlikuje samo po poslednjem simbolu, gde je r ostatak deljenja $1 - n$ sa $D - 1$ (stoga je $D - r \geq 2$).

TEOREMA 3.6

Hafmanovo kodovanje je optimalno, što znači da ako je Z jedan Hafmanov kod informacionog izvora U , a X drugi jednoznačni kod istog izvora, tada je $E[L_X] \geq E[L_Z]$.

3.3. LABORATORIJSKI RAD

VEŽBA 6.

Hafmanov kod – entropijsko kodovanje, uvodna napomena:

Entropijsko kodovanje je metoda za određivanje jednoznačnog koda minimalne dužina. Minimizovana dužina koda omogućuje maksimalnu brzinu prenosa informacije preko komunikacionog kanala. Osnovna ideja i pretpostavka je da poruke sa većom verovatnoćom budu kodovane sa kraćim kodovima, dok poruke sa manjom verovatnoćom budu kodovane dužim kodovima.

Neki informacioni izvor S generiše 6 poruka sa sledećim verovatnoća: $p = \{0.49, 0.26, 0.12, 0.04, 0.04, 0.03, 0.02\}$.

Ukoliko nam verovatnoća nije poznata, entropija izvora je $H = \log_2(7)$, i u ovom slučaju svaku poruku bismo kodovali sa 3 bita:

$$H = 2.8074 \text{ bitova}$$

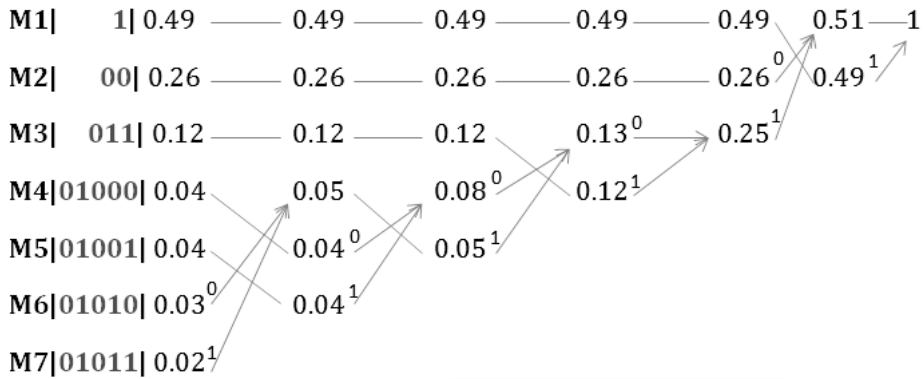
Entropija izvora za poznate verovatnoće je:

$$H_S = 2.0128 \text{ bitova}$$

Ukupna količina informacije:

$$I_S = 1.03 + 1.94 + 3.06 + 4.64 + 4.64 + 5.06 + 5.63 = 26.0 \text{ bitova}$$

Sledi metod za jednostavno generisanje Hafmanovog efikasnog koda. Za generisanje ovog koda nije potreban računarski program.



Rešenje zadatka.

Niz sa verovatnoćama:

```
>> p = [0.49, 0.26, 0.12, 0.04, 0.04, 0.03, 0.02];
```

Simboli poruka:

```
>> simboli = [1 2 3 4 5 6 7];
```

Generisanje Hafmanovog rečnika:

```
>> [recnik] = huffmandict(simboli, p);
```

```

1  {
2
3  [1,1] = 1
4  [1,2] = 0  1
5  [1,3] = 0  0  1
6  [1,4] = 0  0  0  0  0
7  [1,5] = 0  0  0  0  1
8  [1,6] = 0  0  0  1  0
9  [1,7] = 0  0  0  1  1
10
11 }
```

Slika 3.15 Hafmanov rečnik

Na slici iznad (Slika 3.15), prikazao je stanje promenljive rečnik, koja je generisana preko funkcije *huffmandict()*. U prvoj koloni su navedene poruke $M = \{1, 2, 3, 4, 5, 6, 7\}$, a u drugoj koloni su predstavljene kodne reči koje odgovaraju informacionoj vrednosti svake poruke.

Primer kodovanja preko generisanog rečnika:

Poruka:

6 6 6 1 6 7 1 3 1 6 7 1 3 1 5 5 1 7 5 7 1 3 7 1

Kodovanje poruke:

```
>> poruka = [6 6 6 1 6 7 1 3 1 6 7 1 3 1 5 5 1 7 5 7 1 3 7 1];
```

```
>> kodovana_poruka = huffmanenco(poruka, recnik);
```

Kodovana poruka:

0	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	1	0	0	1	0	1	1	1		
0	1	1	1	0	1	0	1	0	0	1	0	1	1	1	0	1	1	1	0	1	0	0	0	1	0	
0	0	1	0	1	0	1	1	0	1	0	0	0	0	1	0	1	1	1	0	1	1	0	1	0	1	1

Entropija kodovane poruke:

```
>> entropy(kodovana_poruka)
```

```
ans = 0.9372 bitova
```

Dekodovanje poruke:

```
>> dekodovana_poruka = huffmandeco(kodovana_poruka, recnik);
```

Dekodovana poruka:

```
>> dekodovana_poruka
```

6 6 6 1 6 7 1 3 1 6 7 1 3 1 5 5 1 7 5 7 1 3 7 1

4. KANALI ZA PRENOS INFORMACIJA

Ciljevi ovog poglavlja su:

1. Prezentovanje osnova kodovanja informacionih izvora u cilju pouzdanog prenosa, čak i u prisustvu šuma.
2. Formalizacija prenosa informacija preko pojma kanala
3. Upoznavanje sa dva bazična pojma vezana za prenos u prisustvu šumova: kapacitet kanala i brzina prenosa.
4. Fundamentalne granice greške prenosa u prisustvu šuma.



4.1. KOMUNIKACIONI KANALI

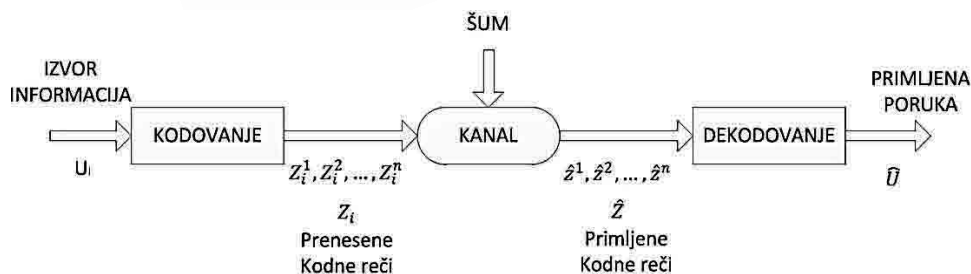
Prenos informacija podrazumeva ili njihovo prenošenje iz jedne tačke u drugu ili njihovo prenošenje kroz vreme.

Primer prenošenja od jedne tačke do druge je komuniciranje između dva mobilna telefona u jednoj mreži, dok je primer prenošenja kroz vreme memorisanje nekog sadržaja na nekom memorijskom mediju, a zatim njegovo iščitavanje u nekom budućem trenutku vremena.

U oba slučaja, susrećemo se sa problemom pouzdanog prenosa informacija, kako usled delovanja šumova na kanalu za prenos, tako i u nedostatku memorijskog prostora ili njegovog kvara.

Sveprisutni šumovi na kanalima prenosa mogu delovati na jedan od sledećih načina:

1. U uslovima izuzetno visokog šuma nije uopšte moguće obaviti pouzdan prenos poruka.
2. Nivo šuma je takav da je moguće preneti poruke sa prihvatljivim nivoom grešaka u prenosu.
3. Moguće je preneti poruke sa verovatnoćom greške koja se može učiniti po želji malom primenom kodova za ispravljanje grešaka. Osnovna ideja zaštitnog kodovanja je dodavanje redundanse u kodirane poruke, tako da i pored grešaka u prenosu preostaje dovoljno nepromenjenih informacija u kodovanim porukama na osnovu kojih se može rekonstruisati poslata poruka sa zadatim nivoom greške rekonstrukcije.



Slika 4.1 Model komunikacionog kanala

Komunikacioni kanal, ili kraće kanal, reprezentuje sve što se može dogoditi sa porukom u toku njenog prenosa od predajnika do prijemnika (Slika 4.1). Poruka je niz simbola. Simbol je element skupa koga nazivamo alfabet.

Razmatramo samo konačne alfabete. Ulazna sekvenca $X_1, X_2, \dots$ (sekvenca koju treba preneti) je određena u potpunosti informacionim izvorom. Izlazna sekvenca $Y_1, Y_2, \dots$ (primljena sekvenca) je određena uslovnom verovanoćom izlaza za poznati ulaz.

Stoga je u matematičkom smislu kanal određen uslovnim verovatnoćama različitih primljenih sekvenci, pod uslovom poznatih sekvenci koje se šalju, tj $P(y_1, y_2, \dots, y_n | x_1, x_2, \dots, x_n)$.

DEFINICIJA 4.1 Diskretni kanal bez memorije (DMC – *Discrete Memoryless Channel*)

Diskretni kanal bez memorije je najjednostavniji komunikacioni kanal. Formalno je određen sa tri veličine. Diskretni ulazni alfabet $\mathcal{Y}_X$, čiji su elementi simboli svih ulaznih poruka izvora X . Diskretni izlazni alfabet $\mathcal{Y}_Y$, čiji elementi predstavljaju moguće primljene simbole izlazne sekvence kanala.

Za svako $X \in \mathcal{Y}_X$, uslovne verovatnoće $p(y|x)$ nad skupom $\mathcal{Y}_Y$, tako da je za svako $n = 1, 2, 3, \dots$

$$p(y_n | x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_{n-1}) = p(y_n | x_n)$$

Ovo se nazivaju transmisione verovatnoće kanala.

Definicija 4.1. je matematička formulacija činjenice da je DMC bez memorije. Ono što će se desiti sa signalom poslatim u n -tom korišćenju kanala je nezavisno od onoga što se desilo u prethodnih $n - 1$ upotreba kanala.

Primetimo da je DMC vremenski nepromenljiv, pošto raspodela verovatnoća $p(y_n | x_n)$ ne zavisi od n .

Kada su $\mathcal{Y}_X$ i $\mathcal{Y}_Y$ konačni, kanal se može predstaviti dijagramom u kome:

1. čvorovi sa leve strane odgovaraju ulaznom alfabetu $\mathcal{Y}_X$,
2. čvorovi sa desne strane odgovaraju izlaznom alfabetu $\mathcal{Y}_Y$,

3. grane koje povezuju čvorove x_i i y_j su označene uslovnim verovatnoćama $p(y_j|x_i)$, osim u slučajevima kada su ove verovatnoće jednake 0 i kada se odgovarajuća grana jednostavno izostavlja.

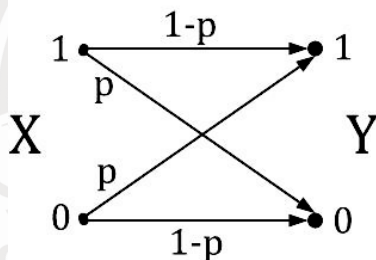
PRIMER 4.1 Binarni simetrični kanal (BSC – *Binary Symmetric Channel*).

Najprostiji DMC je binarni simetrični kanal (BSC) kod koga su $Y_X = Y_Y = \{0,1\}$ binarni, i

$$p_{Y|X=0}(1) = p_{Y|X=1}(0)$$

Vrednost $p = p_{Y|X=0}(1) = p_{Y|X=1}(0)$ se naziva verovatnoća greške (eng. *error rate*) i to je ujedno jedini parametar BSC, budući da je $p = p_{Y|X=0}(1) = p_{Y|X=1}(0)$.

BSC se prezentuje sledećim dijagramom.



Slika 4.2 Dijagram binarnog simetričnog kanala (BSC)

PRIMER 4.2 Prenos preko BSC u prisustvu šuma.

Pretpostavimo da želimo da prenesemo osam poruka: 000, 001, 010, 011, 100, 101, 110, 111. Zatim, pretpostavimo da je na kanalu za prenos prisutan šum, tako da u proseku se svaki 10-ti simbol promeni u suprotni (0 u 1 ili 1 u 0). Takav kanal je BSC sa parametrom $p = 0.1$.

Kolika je verovatnoća da prenesemo jednu poruku korektno?

Bez obzira koja se poruka šalje ova verovatnoća iznosi:

$$(1 - p)^3 = 0.9^3 = 0.719$$

Što odgovara verovatnoći da tri puta prenesemo jedan bit bez greške. Odgovarajuća verovatnoća greške je $1 - 0.719 = 0.281$, dakle prilično visoka.

Pretpostavimo sada kodovanje u kome je svaki bit prethodnog koda ponovljen (udvojen).

Poruka	000	001	010	011	100	...	111
Kod	000000	000011	001100	001111	110000	...	111111

Sada je verovatnoća tačnog prijema jedne poruke $(1 - p)^6 = 0.531$. Odnosno verovatnoća greške 0.469, što je veće nego u prethodnom slučaju. Međutim, potrebno je razmotriti još jedan tip grešaka, naime kolika je verovatnoća pogrešnog prijema koja se ne može detektovati.

Ova situacija nastaje kada se promene npr. dva simbola, npr. umesto 000000 se primi 110000. U opštem slučaju promene od 2, 4 ili svih 6 simbola dovode do pogrešnog prijema koji se ne može razlikovati od tačnog prijema.

Verovatnoća promene dva simbola je data sa:

$$\binom{3}{1} p^2 (1 - p)^4 = 3p^2 (1 - p)^4.$$

Verovatnoća promene 4 simbola je data sa $3p^4(1 - p)^2$, i konačno verovatnoća promene 6 simbola je data sa p^6 . Ukupna verovatnoća greške koja se ne može detektovati je data zbirom:

$$3p^2(1 - p)^4 + 3p^4(1 - p)^2 + p^6 = 0.020.$$

Ova verovatnoća je prilično mala. Zaključak je da se odgovarajućim kodovanjem može poboljšati prenos poruka preko kanala na kome deluju šumovi.

Za kanal kažemo da je bez povratne sprege ukoliko raspodela verovatnoće ulaza ne zavisi od izlaza, ili formalno:

$$pX_n | x_1, \dots, x_{n-1}, y_1, \dots, y_{n-1} = pX_n | x_1, \dots, x_{n-1}$$

TEOREMA 4.1

Za DMC bez povratne sprege važi:

$$H(Y_1 \dots Y_n | X_1 \dots X_n) = \sum_{i=1}^n H(Y_i | X_i)$$

4.1.1. KAPACITET KANALA

Kapacitet kanala meri sposobnost jednog kanala da prenosi informacije. To je maksimalna prosečna količina informacija koje ulaz kanala može preneti na izlaz.

DEFINICIJA 4.2 Kapacitet kanala.

Kapacitet C diskretnog kanala bez memorije jednak je:

$$C = \max_{p_X} I(X; Y),$$

gde je X ulaz u kanal, opisan raspodelom verovatnoće $p_X(x)$, dok je Y izlaz kanala.

PRIMER 4.3 Kapacitet BSC.

Po definiciji:

$$C = \max_{p_X} (H(Y) - H(Y|X)).$$

Imajući u vidu da je:

$$P(Y \neq X) = p \quad P(Y = X) = 1 - p,$$

imamo da je :

$$H(Y|X) = -p \log(p) - (1 - p) \log(1 - p) =: \tilde{h},$$

i ne zavisi od $p_X(x)$, tako da je konačno:

$$C = \max_{p_X} (H(Y)) - \tilde{h}(p).$$

Kako je Y binarna slučajna promenljiva, važi $H(Y) \leq \log 2$, odnosno $H(Y) \leq 1$ bit.

Da li se ova granica može dostići za neku raspodelu $p_X(x)$?

Odgovor je da, ukoliko je X uniformno raspodeljeno.

Tada imamo:

$$p_Y(0) = p \cdot p_X(1) + (1 - p) \cdot p_X(0) = 0.5p + 0.5(1 - p) = 0.5$$

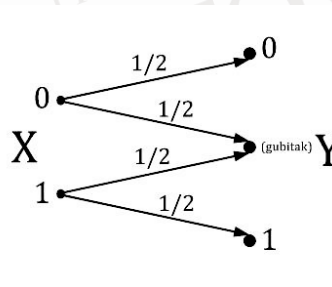
Što znači da je Y takođe uniformno raspodeljeno, odnosno da je $H(Y) = 1$ bit. Stoga je $\max_X H(Y) = 1$ bit, odnosno, konačno $C = 1 - \tilde{h}(p)$ (u bitovima).

4.1.2. KANALI SA SIMETRIČNIM ULAZOM

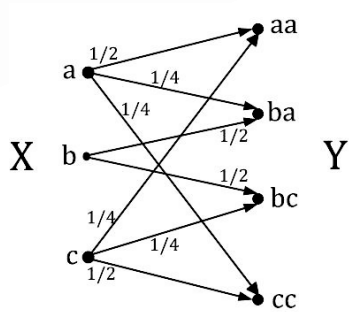
DEFINICIJA 4.3

DMC kanal je sa simetričnim ulazom ukoliko je raspodela verovatnoće greške ista za sve ulazne simbole, odnosno ukoliko je skup $\{p_{Y|x_i}(y) : y \in \mathcal{Y}\}$ nezavisan x_i

PRIMER 4.4 Kanal sa simetričnim ulazom.



Slika 4.3 Kanal sa simetričnim ulazom i nesimetričnim izlazom.



Slika 4.4 Primer DMC kanala sa simetričnim izlazom i ne simetričnim ulazom.

BSC je kanal sa simetričnim ulazom.

LEMA 4.1

Za svaki DMC sa simetričnim ulazom, $H(X|Y)$ je nezavisna od raspodele $p_X(x)$, i

$$H(Y|X) = H(Y|x_i)$$

za svako $x_i \in \mathcal{V}_X$.

Stoga je nalaženje raspodele verovatnoće ulaza za koju se dostiže kapacitet simetričnog DMC kanala, ekvivalentno nalaženju ulaza koji maksimizuje neodređenost izlaza.

OSOBINA 4.1

Za svaki DMC sa simetričnim ulazom, važi:

$$C = \max_{P_X} [H(Y)] - H_0,$$

gde je $H_0 = H(Y|X = x_i)$ za svako $x_i \in \mathcal{V}_X$.

4.1.3. KANALI SA SIMETRIČNIM IZLAZOM

Kanali sa simetričnim ulazom su oni čije verovatnoće grana koje napuštaju svaki ulazni simbol u dijagramu kanala su iste za svaki simbol (iste do na permutaciju grana).

Sada ćemo razmotriti kanale kod kojih su verovatnoće grana koje dolaze do izlaznih simbola iste (iste do na permutaciju grana).

DEFINICIJA 4.3a Kanali sa simetričnim izlazom.

DMC je sa simetričnim izlazom kada je skup $\{p_{Y|x}(y_i) : x \in \mathcal{V}_X\}$ nezavisan od y_i .

PRIMER 4.5

BSC kanal iz primera 4.1 je sa simetričnim izlazom.

LEMA 4.2

Kod DMC sa simetričnim izlazom, uniformna raspodela verovatnoća ulaza ($p_X(x_i)$) je ista za svako x_i) rezultuje u uniformnoj raspodeli verovatnoća na izlazu ($p_Y(y_i)$) je isto za svako y_i).

OSOBIKA 4.2

Za svaki DMC sa simetričnim izlazom važi:

$$\max_{p_X} H(Y) = \log |v_Y|.$$

DEFINICIJA 4.4 Simetrični kanal.

DMC je simetričan ako je istovremeno sa simetričnim ulazom i sa simetričnim izlazom.

TEOREMA 4.2

Kapacitet simetričnog kanala je dat sa:

$$C = \log |v_Y| - H_0,$$

gde je $H_0 = H(Y|X = x_i)$ za svaki ulazni simbol $x_i \in v_X$.

PRIMER 4.6

BSC iz primera 4.1 je simetričan kanal kod koga je:

$$H_0 = \tilde{h}(p) = -p \log p - (1 - p) \log(1 - p).$$

Stoga je:

$$C_{BSC} = \log 2 - \tilde{h}(p) = 1 - \tilde{h}(p) \text{ (u bitovima).}$$

4.1.4. BRZINA PRENOSA

DEFINICIJA 4.5

Brzina prenosa (za osnovu b) koda kojim se koduje diskretan izvor U sa $|Y_U|$ poruka, čije su kodne reči fiksne dužine n je definisana sa:

$$R_b = \frac{\log_b |v_X|}{n}.$$

Primetimo da je $|Y_U|$ istovremeno i broj mogućih kodnih reči determinističkih nesingularnih kodova. Po pravilu se osnova b za računanje brzine koda uzima da je jednaka arnosti koda.

PRIMER 4.7

U primeru 4.2 (binarni kod sa 8 kodnih reči i dužinom kodnih reči 6), brzina prenosa je:

$$R = \frac{\log 8}{6} = \frac{3}{6} = \frac{1}{2}.$$

Ovo izgleda razumno, budući da se u ovom kodu koristi dva puta više bita nego što emituje izvor.

4.2. KODOVANJE U PRISUSTVU ŠUMA

Osnovu poglavlja čini „Teorema o kodovanju u prisustvu šuma“ (engl. *Noisy Coding Theorem*), koja utvrđuje uslove pod kojima je moguć pouzdan prenos informacija.

Nakon ovog poglavlja znaćemo:

Koja je maksimalna brzina pouzdanog prenosa po kanalu sa šumom uz upotrebu kodova za ispravljanje grešaka?

Koja je maksimalna količina grešaka koja se čini kada se informacije prenose brzinom većom od maksimalne?

4.2.1. KOD PONAVLJANJA

Daćemo primer jednog veoma jednostavnog i naivnog koda za ispravljanje grešaka. Primer prevashodno služi da nam pokaže mogućnost konstrukcije daleko efikasnijih kodova.

Binarni kod ponavljanja R_k jednostavno svaki ulazni binarni simbol ponavlja $n = 2k + 1$ put, $k > 0$. Kod R_1 je opisan u primeru 4.2. Razmatraćemo samo neparan broj ponavljanja, budući da se takvi kodovi dekoduju na osnovu većinske logike.

U ovoj klasi kodova verovatnoća pogrešnog dekodovanja je jednaka verovatnoći da se dese najmanje $k + 1$ grešaka u razmatranom bloku podataka.

Razmotrimo situaciju u kojoj se ovaj kod koristi za komuniciranje po BSC kanalu. Broj grešaka je raspodeljen po binomijalnoj raspodeli sa parametrima (n, p) . Očekivana vrednost ovih grešaka na primer za $p < 0.5$, očekivani broj grešaka je manji od $k + 0.5$, što znači da verovatnoća da se desi najmanje $k + 1$ greška unutar jedne kodne reči teži nuli kada k (a samim tim i n) teži beskonačnosti.

Drugim rečima verovatnoća pogrešnog dekodovanja teži nuli, kada povećavamo broj ponavljanja (a samim tim i dužine kodnih reči)

Prema tome, u stanju smo da kompenzujemo gubitke usled šumova na kanalu do proizvoljne mere ukoliko primenimo dovoljno velik broj ponavljanja. Međutim, cena koja se za to plaća je efikasnost. Brzina takvog koda je $\frac{1}{n}$, i teži nuli kada se n povećava. Za veliko n , to je neprihvatljivo niska brzina prenosa.

4.2.2. TEOREMA O KODOVANJU U PRISUSTVU ŠUMA

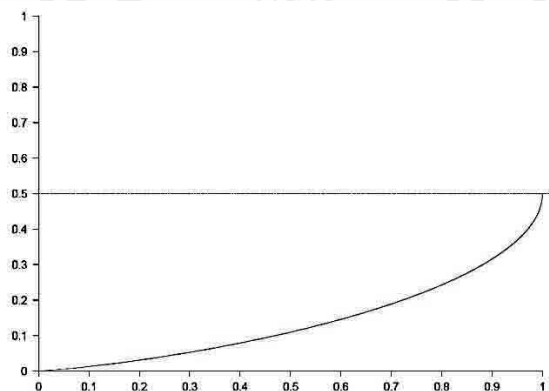
TEOREMA 4.3 Ukoliko je binarni simetrični izvor BSS brzine R ulaz u DMC bez povratne sprege kapaciteta C i ukoliko je $R > C$, tada je verovatnoća greške P_b na izlazu kanala data sa:

$$P_b \geq \tilde{h}^{-1}\left(1 - \frac{C}{R}\right)$$

gde je:

$$\tilde{h}^{-1}(x) = \min\{p: -p \log(p) - (1 - p) \log(1 - p) = x\}.$$

$\tilde{h}^{-1}$ označava inverznu binarnu entropijsku funkciju, pri čemu je min uveden zbog jedinstvenosti inverzne funkcije.



Slika 4.5 Inverzna binarna entropijska funkcija

Važna posledica ove teoreme je da kad god važi $R > C$, postoji pozitivna donja granica za verovatnoću greške P_b koju ni jedan sistem kodovanja, bez obzira koliko on bio kompleksan ne može prevazići.

PRIMER 4.9

Neka je kapacitet jednog DMC, $C = 0.25$ bita za prenos poruka BSS izvora. Ako se prenos vrši brzinom $R = 0.5$, tada na osnovu funkcije (P_b) za verovatnoću greške na izlazu kanala dobijamo grešku najmanje 11%, budući da je:

$$P_b \geq \tilde{h}^{-1} \left(1 - \frac{1}{2} \right) = .11.$$

To znači da će najmanje 11% bita biti pogrešno dekodovano bez obzira kakav kod za ispravljanje grešaka primenili.

Kako stoji stvar sa DMC sa povratnom spregom?

Pokazuje se da i u slučaju DMC sa povratnom spregom, nejednakost važi. Ova činjenica se izražava stavom da povratna sprega ne može da poveća kapacitet DMC. Međutim, to ne znači da povratna sprega nema značaja. Kada se vrši prenos na brzinama $R < C$, upotreba povratne sprege po pravilu pojednostavljuje kodovanje i dekodovanje i omogućava postizanje željene greške dekodovanja.

TEOREMA 4.4 Kodovanje u DMC uz prisustvo šuma.

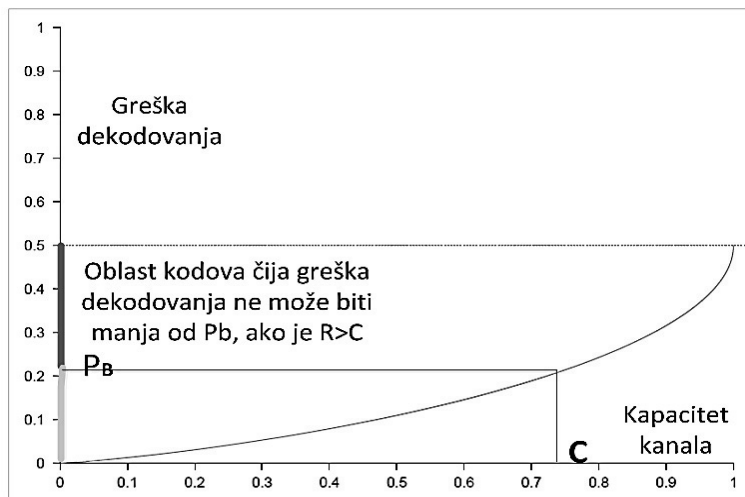
Razmotrimo prenos poruka brzinom R preko DMC bez povratne sprege, čiji je kapacitet C . Za sve $R < C$ i sve $\varepsilon > 0$, postoji kod za ispravljanje grešaka brzine R i greške dekodovanja $P_b < \varepsilon$.

Uopšteno govoreći ova važna teorema tvrdi da ako izaberemo odgovarajući način kodovanja informacije za zadati kanal, može se postići po želji mala greška dekodovanja, odnosno prenosa informacija.

Razmotrimo BSC sa verovatnoćom greške p , čiji je kapacitet stoga $C = 1 - \hat{h}(p)$.

Da li za zadate p, C i R , date u tabeli postoje kodovi koji ispunjavaju ove zahteve?

Kanal	p	5						10%					
	C	0.801						0.675					
Kod	R	2/3		3/4		9/10		2/3		3/4		9/10	
	$P_b(\%)$	1	2.2	1	2.2	1	2.2	1	2.2	1	2.2	1	2.2
Da li postoji?													



Slika 4.6 Slika predstavlja odgovor na pitanje iznad

Objašnjenje:

Ako je $R < C$ sigurni smo da postoji kod čija je greška dekodovanja bita P_b po želji mala.

Ako je $R > C$, greška dekodovanja ne može biti $\hat{h}(P_b) < 1 - \frac{C}{R}$.

Ako je $R > C$, i $\hat{h}(P_b) \geq 1 - \frac{C}{R}$, ne možemo ništa određeno zaključiti, pošto je ova situacija moguća (nije u kontradikciji sa teoremom 4.3) ali ne znamo dovoljno o konkretnoj kodnoj šemi da bi smo zaključili da li takav kod zaista može postojati.

U tabeli koja sledi, dato je rešenje na osnovu ovih prosuđivanja.

Kanal	p	5%					
	C	0.801					
Kod	R	2/3		3/4		9/10	
	P_b (%)	1	2.2	1	2.2	1	2.2
$R < C?$		da		da		ne	
$1 - C/R$		-		-		0.109	
$\tilde{h}(P_b)$		-		-		0.081	0.153
Postoji!		da	da	da	da	ne	možda

Kanal	p	10%					
	C	0.675					
Kod	R	2/3		3/4		9/10	
	$P_b(\%)$	1	2.2	1	2.2	1	2.2
$R < C?$		da		ne		ne	
$1 - C/R$		-		0.100		0.250	
$\tilde{h}(P_b)$		-		0.081	0.153	0.081	0.153
Postoji!		da	da	ne	možda	ne	ne

Teorema 4.4. je bilo veliko iznenađenje u Šenonovom radu (*C. E. Shannon: A mathematical theory of communication. Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, July and October, 1948), objavljenom 1948. godine.

Pre ovog rezultata smatralo se da u cilju pouzdanog prenosa moramo smanjivati brzinu prenosa, ili ekvivalentno povećavati odnos signal/šum.

Šenon je srušio ovaj mit pokazujući da ako je brzina prenosa ispod kapaciteta kanala, povećanje pouzdanosti prenosa se može u potpunosti ostvariti konstrukcijom kompleksnijih sistema kodovanja i dekodovanja, bez potrebe da se menja odnos signal/šum.

Ova teorema nema praktičnu dimenziju. U njoj nisu dati recepti kako efikasno konstruisati dobre kodove. Na primer, ništa se ne kaže kolika je dužina kodnih reči n u funkciji parametra ϵ .

Stoga je nastala čitava disciplina koja se zove Teorija kodovanja (engl. *Coding Theory*), čiji je cilj dizajniranje efikasnih zaštitnih kodova, odnosno kodova što veće brzine i što manje greške dekodovanja.

5. ZAŠTITNI KODOVI – KODOVI ZA ISPRAVLJANJE GREŠAKA

Ciljevi ovog poglavlja su da damo odgovore na sledeća pitanja:

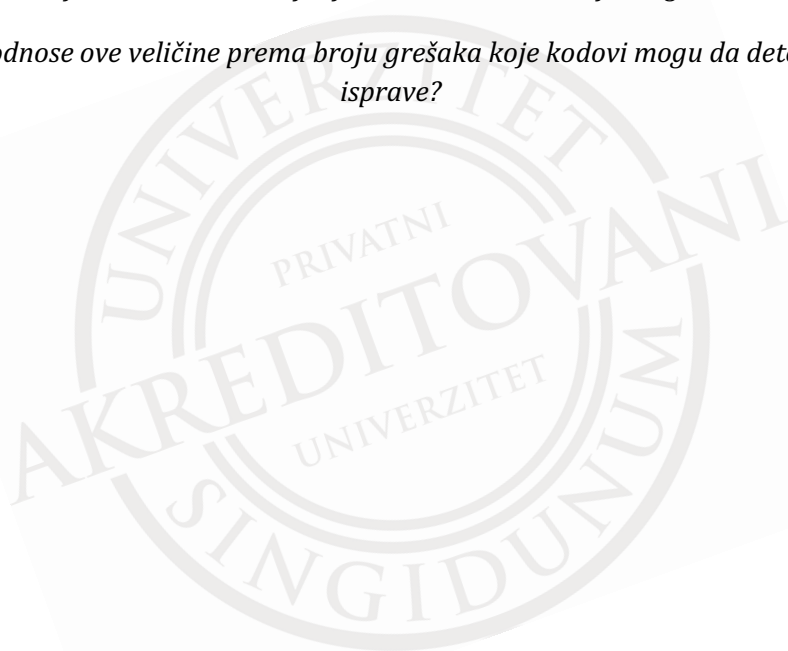
Šta su blok kodovi?

Kako se meri rastojanje između kodnih reči?

Šta je težina kodne reči?

Šta je minimalno rastojanje i minimalna težina jednog koda?

Kako se odnose ove veličine prema broju grešaka koje kodovi mogu da detektuju ili isprave?



5.1. OSNOVE KODOVA ZA ISPRAVLJANJE GREŠAKA

Kada se jedna kodna reč z_i prenosi po nekom kanalu sa šumovima i na izlazu kanala primi $\hat{z}$, tada se odgovarajuća greška u prenosu dobija kao razlika $e = \hat{z} - z_i$.

Ključna ideja algebarskog kodovanja je dodavanje algebarske strukture u skupu kodnih reči tako da se greška može lako izraziti pomoću operacija koje definišu tu algebarsku strukturu.

PRIMER 5.1 Razlike binarnih sekvenci.

Prirodno je razlike binarnih sekvenci definisati preko XOR operacije. Dakle važi:

$$11011 - 01101 = 10110$$

Tehnički govoreći operacija oduzimanja iz primera 5.1 odgovara aritmetici po modulu 2, odnosno odgovarajuća algebarska struktura je polje Galoa $GF(2)$. Kodne reči dužine n u tom slučaju pripadaju vektorskom prostoru $GF(2)^n$. Ovaj koncept se direktno proširuje na D-arne kodove korišćenjem aritmetike po modulu D.

DEFINICIJA 5.1 Blok kod.

D-arni blok kod dužine n je neprazan podskup vektorskog prostora n -torki $GF(D)^n$. Jednostavnije rečeno to su D-arne reči dužine n posmatrane kao vektori kolone.

PRIMER 5.2 Blok kod.

Skup $\{1101, 0110, 1110\}$ je jedan primer binarnog blok koda.

Drugi primer blok koda je ternarni kod sa simbolima 0, 1, 2, takav da je $1 + 2 = 0$, $1 + 1 = 2$, $1 + 0 = 1$, $2 + 2 = 1$, $2 + 0 = 2$ (aritmetika u $GF(3)$), čije su kodne reči $\{120, 201, 222, 010\}$. Skup $\{011, 110, 10\}$ nije blok kod, budući da kodne reči nisu iste dužine.

5.1.1. HEMINGOVO RASTOJANJE I TEŽINE KODNIH REČI

DEFINICIJA 5.2 Hemingovo rastojanje.

Hemingovo rastojanje $d(z_i, z_j)$, između dve reči z_i i z_j iste dužine, odnosno dva niza dužine n , je broj simbola (pozicija) u kojima se te dve kodne reči razlikuju.

PRIMER 5.3 Hemingovo rastojanje.

Hemingovo rastojanje između 101010 i 111010 je 1, pošto se ove dve reči razlikuju samo na 2. poziciji.

Hemingovo rastojanje između 1010 i 0101 je 4, pošto se ove dve reči razlikuju u svim pozicijama.

Hemingovo rastojanje između 1010 i 111010 nije definisano, budući da su reči različite dužine.

Hemingovo rastojanje je zaista rastojanje u strogom matematičkom smislu (zadovoljava tri aksioma metričkog rastojanja – simetrija, nulto rastojanje i nejednakost trougla)

DEFINICIJA 5.3 Težine kodnih reči.

Težina jedne reči jednaka je broju nenultih simbola u toj reči.

PRIMER 5.4 Težine kodnih reči

Težina reči 10110 je 3, dok je težina reči 00000000 jednaka 0. Težina reči 0001000 je jednaka 1.

OSOBINA 5.1

Hemingovo rastojanje dve kodne reči jednako je težini njihove razlike, odnosno

$$d(z_i, z_j) = w(z_i - z_j),$$

gde $d()$ označava Hemingovo rastojanje, a $w()$ težinu reči.

PRIMER 5.5

Ovaj primer pokazuje ekvivalentnost Hemingovog rastojanja i težine razlike:

$$d(10110, 11010) = 2$$

$$w(10110 - 11010) = w(01100) = 2.$$

OSOBI NA 5.2

Težina kodnih reči je uvek pozitivna ili jednaka nuli.

DEFINICIJA 5.4

Nula kodna reč je kodna reč sastavljena samo od nula. Označava se sa 0.

OSOBI NA 5.3

Težina jedne kodne reči je nula, ako i samo ako je kodna reč nula kodna reč.

OSOBI NA 5.4

Težine su simetrične funkcije, odnosno $w(z_i) = w(-z_i)$, gde se pod $-z_i$ podrazumeva reč u kojoj je svaki simbol zamenjen suprotnim simbolom.

PRIMER 5.6 Simetrija težina.

Razmotrimo ternarni kod sa simbolima 0, 1, 2 u $GF(3)$. Tada je

$$w(-1202102) = w(2101201) = 5 = w(1202102).$$

Primetimo da je u binarnom slučaju $-z_i = z_i$ ($-1 = 1, -0 = 0$).

OSOBI NA 5.5

Za svaku kodnu reč z_i i z_j važi:

$$w(z_i) + w(z_j) \geq w(z_i + z_j).$$

PRIMER 5.7

U binarnom slučaju imamo $w(110101) + w(010101) = 4 + 3 = 7 \geq 1 = w(100000) = w(110101 + 010101)$.

Razmotrimo ternarni kod sa simbolima 0, 1, 2:

$$w(01221021) + w(21002010) = 6 + 4 = 10 \geq 5 = w(01221021 + 21002010).$$

Zašto su Hemingova rastojanja i težine reči važna za algebarske kodove za ispravljanje grešaka?

Greška e koja nastaje pri prenosu jednaka je $e = \hat{z} - z_i$, gde je z_i emitovana a $\hat{z}$ primljena kodna reč. Broj grešaka pri prenosu jednostavno je jednak težini reči e , odnosno $w(\hat{z} - z_i)$. Ova vrednost je ujedno jednaka Hemingovom rastojanju $d(\hat{z}, z_i)$ između emitovane i primljene kodne reči.

Stoga je detektovanje greške ekvivalentno sa detektovanjem ne nulte težine, a Ispravljanje grešaka svodi se na izračunavanje razlike e , bez poznavanja z_i , budući da je na prijemu poznato samo $\hat{z}$.

5.1.2. DEKODOVANJE MINIMALNOG RASTOJANJA I MAKSIMALNE VERODOSTOJNOSTI

Na koji način se dekoduju blok kodovi?

Prirodan intuitivan kriterijum za dekodovanje je minimizacija greške dekodovanja, odnosno $w(e)$. S druge strane videli smo da je ta minimizacija ekvivalentna nalaženju najbliže kodne reči u odnosu na primljenu, odnosno minimizacija $d(\hat{z}, z_i)$.

Na primer, ako su moguće kodne reči 000 i 111, a primljena reč je 010, tada je ako nemamo nikakvu dodatnu informaciju, dekodovana reč 000, jer je $d(000, 010) = 1$, $d(111, 010) = 2$, dakle 010 je najbliža kodna reč.

DEFINICIJA 5.5 Dekodovanje minimalnog rastojanja.

Kaže se da kod C koristi dekodovanje minimalnog rastojanja ako je odluka D prilikom dekodovanja za svaku primljenu reč $\hat{z}$ najbliža kodna reč, tj.

$$D(\hat{z}) = \underset{z \in C}{\operatorname{Argmin}} d(z, \hat{z}).$$

Posmatrajmo sada proces dekodovanja sa stanovišta Bajesovske verovatnosne postavke problema. Kada je primljena reč $\hat{z}$, najverovatnija emitovana kodna reč z je ona koja maksimizuje verovatnoću $P(X = z|Y = \hat{z})$, gde je X ulaz, a Y izlaz iz kanala.

Ova verovatnoća se ne može izračunati bez poznavanje apriorne verovatnoće emitovanja kodnih reči $P(X = z_i)$. Stoga se obično uvode neke pretpostavke, od kojih je razumna ona po kojoj se sve kodne reči emituju sa jednakim verovatnoćama (tzv. pretpostavka maksimalne entropije).

Tada važi:

$$\begin{aligned} \operatorname{Argmax}_{z \in C} P(X = z|Y = \hat{z}) &= \operatorname{Argmax}_{z \in C} \frac{P(Y = \hat{z}|X = z)P(X = z)}{P(Y = \hat{z})} \\ &= \operatorname{Argmax}_{z \in C} P(Y = \hat{z}|X = z)P(X = z) = \operatorname{Argmax}_{z \in C} P(Y = \hat{z}|X = z) \end{aligned}$$

Član $P(Y = \hat{z}|X = z)$ je lakši za izračunavanje nego početni izraz $P(X = z|Y = \hat{z})$. Na primer u slučaju BSC ovaj član se svodi na proizvode transmisionih verovatnoća za svaki simbol.

Ako je BSC takav da svi simboli imaju jednaku verovatnoću greške p , ovaj član postaje:

$$P(X = z|Y = \hat{z}) = p^{d(z, \hat{z})} (1 - p)^{n - d(z, \hat{z})},$$

budući da je $d(z, \hat{z})$ simbola preneseno sa greškom i $n - d(z, \hat{z})$ simbola tačno preneto. Lako je videti da kodna reč koja maksimizira $P(Y = \hat{z}|X = z)$ je ona koja minimizira rastojanje $d(\hat{z}, z_i)$.

Ovim je ujedno pokazano da je dekodovanje po metodu minimalnog rastojanja i maksimalne verodostojnosti ekvivalentno u slučaju BSC.

5.1.3. DETEKCIJA I ISPRAVLJANJE GREŠAKA

Da li postoji način da se unapred za zadati kod može reći koliko grešaka može detektovati, a koliko ispraviti?

Odgovor je pozitivan i zasniva se na važnom svojstvu blok kodova: njihovog minimalnog rastojanja.

DEFINICIJA 5.6 Minimalno rastojanje koda.

Minimalno rastojanje koda $C = \{z_1, z_2, \dots, z_M\}$ je minimalno Hemingovo rastojanje $d_{\min}(C)$ između bilo koje dve različite kodne reči:

$$d_{\min}(C) = \min_{i \neq j} d(z_i, z_j).$$

Naredna teorema pokazuje zašto je minimalno rastojanje jednog koda od centralnog značaja u teoriji kodova za ispravljanje grešaka.

TEOREMA 5.1 Kapacitet detekcije i ispravljanja grešaka.

Blok kod dužine n i dekodovanjem minimalnog rastojanja, može za bilo koja dva broja t i s , takva da je $0 \leq t \leq n$, i $0 \leq s \leq n - t$, da ispravi sve oblike nizova grešaka sa ukupno t ili manje grešaka i da detektuje sve oblike nizova grešaka sa $t + 1, \dots, t + s$ grešaka, ako i samo ako je minimalno rastojanje koda striktno veće od $2t + s$.

OSOBINA 5.6 Maksimalni kapacitet detekcije grešaka.

Blok kod C koji koristi detekciju zasnovanu na minimalnom rastojanju, može se upotrebiti za detekciju svih oblika nizova grešaka sa ukupno manje od $d_{\min}(C) - 1$ grešaka.

DOKAZ

U teoremi 5.1 treba staviti $t = 0, s = d_{\min}(C) - 1$.

OSOBINA 5.7 Maksimalni kapacitet ispravljanja grešaka.

Blok kod C koji koristi dekodovanje zasnovano na principu minimalnog rastojanja, može da se upotrebi za ispravljanje svih oblika nizova grešaka sa ukupno $(d_{\min}(C) - 1)/2$ ili manje grešaka i ne može da ispravi sve oblike nizova grešaka sa ukupno $1 + (d_{\min}(C) - 1)/2$ grešaka.

Napomenimo da je deljenje sa 2 u gornjim izrazima tzv. Euklidsko odnosno celobrojno deljenje (u oznaci $\lfloor x/2 \rfloor$, tj celobrojni deo deljenja).

DOKAZ

U teoremi 5.1 treba staviti $t = (d_{\min}(C) - 1)/2$, $s=0$.

Ujedno veličina $(d_{\min}(C) - 1)/2$ je maksimalno t koje se može upotrebiti u ovoj teoremi.

PRIMER 5.8

Blok kod sa minimalnim rastojanjem 8 može se upotrebiti za sledeće zadatke:

1. Ispravljanje 3 greške i detektovanje 4 greške ($t = 3, s = 1$)
2. Ispravljanje manje od 2 greške i detektovanje 3 i 5 grešaka ($t = 2, s = 3$)
3. Ispravljanje 1 greške i detektovanje 2 i 6 grešaka ($t = 1, s = 5$)
4. Detektovanje manje ili jednako 7 grešaka ($t = 0, s = 7$)

Blok kod sa minimalnim rastojanjem 7 može se upotrebiti za sledeće zadatke:

1. Ispravljanje do 3 greške i detektovanje 4 greške ($t = 3, s = 0$)
2. Ispravljanje manje od 2 greške i detektovanje 3 i 4 grešaka ($t = 2, s = 2$)
3. Ispravljanje 1 greške i detektovanje 2 i 5 grešaka ($t = 1, s = 4$)
4. Detektovanje manje ili jednako 6 grešaka ($t = 0, s = 6$)

PRIMER 5.9

Da bi blok kod mogao da ispravlja 1 grešku, njegovo minimalno rastojanje mora biti barem 3.

5.2. LINEARNI KODOVI

Linearni kodovi predstavljaju jednu od najznačajnijih klasa kodova zbog njihove jednostavnosti i lake implementacije.

Naučićemo:

Šta su (n, m) D-arni linearni kodovi?

U ovoj klasi kodova je minimalno rastojanje ekvivalentno sa minimalnim težinama.

Kako se vrši kodovanje pomoću generator matrica?

Šta je sistematska forma generator matrica ovih kodova?

Kako se vrši dekodovanje na osnovu verifikacionih matrica i sindrom tabela?

Kako se izračunava minimalno rastojanje linearnih kodova?

Šta su binarni Hemingovi kodovi i kako se koriste?

DEFINICIJA 5.7 Linearni kod.

(n, m) D-arni linearni kod ($1 \leq m \leq n$) je m -dimenzioni podprostor vektorskog prostora $GF(D)^n$ n -dimenzionih vektora u $GF(D)$. Dakle, linearni kodovi su blok kodovi koji su vektorski prostor.

PRIMER 5.10 Linearni kod.

Kod $\{1101, 0110, 1110\}$ dat u primeru 5.2 nije linearni kod, pošto 0000 nije kodna reč ovog koda, i stoga nije vektorski prostor.

Binarni kod $\{0000, 1101, 0110, 1011\}$ je linearni kod pošto je bilo koja linearna kombinacija kodnih reči takođe kodna reč. Dalje, to je $(4, 2)$ binarni linearni kod pošto je dimenzija vektorskog podprostora 2, a njegova dužina 4.

Minimalno rastojanje ovog koda je 2, pa se može koristiti za ispravljanje samo jedne greške (prema teoremi 5.1).

OSOBI NA 5.8

Svaki linearni kod sadrži nula kodnu reč. Ovo svojstvo direktno sledi iz činjenice da je linearni kod vektorski (pod)prostor.

Koliko kodnih reči sadrži jedan (n, m) D-arni linearni kod?

OSOBI NA 5.9

(n, m) D-arni linearni kod sadrži D^m različitih kodnih reči, uključujući i nula kodnu reč.

Ova osobina je zgodna za brzu proveru da li je jedan kod sa pogrešnim brojem kodnih reči (nije stepen od D) linearni kod. Takođe se na osnovu ovog svojstva lako pogađa m – dimenzija bazisa koda.

DOKAZ

Pošto je linearni kod vektorski prostor dimenzije m , svaka kodna reč se dobija kao linearna kombinacija m bazičnih kodnih reči (to je ujedno jedna baza ovog vektorskog prostora). Za (m, n) D-arni kod, ima ukupno D^m različitih kodnih reči, zapravo svih D^m mogućih linearnih kombinacija.

OSOBI NA 5.10 Brzina linearnog koda.

Brzina prenosa linearnog (n, m) koda je:

$$R = \frac{m}{n}.$$

DOKAZ

Podsetimo se da je brzina D-arnog koda izvora sa M različitih poruka sa kodnim rečima dužine n data sa:

$$R = \frac{\log_D M}{n}.$$

Koliko se različitih poruka može kodovati (m, n) linearnim kodom?

Onoliko koliko ima kodnih reči, a to je D^m . Stoga je:

$$R = \frac{\log_D D^m}{n} = \frac{m}{n}.$$

TEOREMA 5.2 Ekvivalentnost minimalnog rastojanja i minimalne težine.

Za svaki linearni kod C :

$$d_{\min}(C) = w_{\min}(C),$$

gde je $w_{\min}(C)$ minimalna težina koda, odnosno najmanja težina nenultih kodnih reči:

$$w_{\min}(C) = \min_{\substack{z \in C \\ z \neq 0}} w(z).$$

Ovaj rezultat je značajan zato što je daleko lakše izračunati u praksi $w_{\min}(C)$ nego $d_{\min}(C)$.

PRIMER 5.11

(11,3) binarni kod $\{00000000000, 10011110000, 01000111100, 00111001111, 11011001100, 10100111111, 01111110011, 11100000011\}$ ima minimalnu težinu 5, a samim tim i minimalno rastojanje 5.

Stoga ovaj kod može da ispravi sve oblike nizova grešaka sa ukupno 1 i 2 greške (videti osobinu 5.7). Proveriti da li je ovaj kod linearan.

5.2.1. KODOVANJE LINEARNIM KODOM

Sada ćemo pokazati kako se poruke transformišu u kodne reči linearnog koda. Ako se m kodnih reči izabere za bazu (n, m) linearnog koda označe sa $z_1, z_2, \dots, z_m$, tada se bilo koja kodna reč z_i može zapisati u obliku:

$$z_i = \sum_{k=1}^{k=m} u_{i,k} z_k$$

gde su $u_{i,k}$ komponente z_i za bazičnih vektora z_k . U kompaktnom obliku, koristeći linearnu algebru imamo:

$$z_i = (u_{i,1}, \dots, u_{i,m}) \cdot G = u_i \cdot G,$$

gde je u_i vektor red $(u_{i,1}, \dots, u_{i,m})$ a G matrica čiji su redovi $z_1, z_2, \dots, z_m$.

Prema tome, poruka u_i dužine m simbola se koduje kodnom rečju z_i množenjem sa matricom G . Stoga se matrica G naziva generator matrica datog koda.

DEFINICIJA 5.8 Generator matrica.

$m \times n$ matrica G je generator matrica (n, m) linearnog koda C ako i samo ako su njenih m vektora redova baziisi vektorskog prostora C . Kodovanje poruke u dužine m se svodi na množenje $z = u \cdot G$.

PRIMER 5.12 Generator matrica.

Neka je dat $(4,2)$ binarni linearni kod iz primera 5.10 $\{0000, 1101, 0110, 1011\}$. Ovaj kod ima četiri kodne reči i može da koduje četiri poruke: četiri binarne reči od po dva bita $u_0 = 00$, $u_1 = 10$, $u_2 = 01$, $u_3 = 11$. Izvršimo kodovanje pomoću generator matrice.

Jedna moguća baza ovog koda je $z_1 = 1101$, $z_2 = 0110$, što daje generator matricu:

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

Tada se u_1 koduje u:

$$u_1 \cdot G = (1 \ 0) \cdot \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = (1 \ 1 \ 0 \ 1) = z_1.$$

Slično postupamo za transformisanje u_2 u z_2 , u_3 u 1011 i u_0 u 0000 .

Primetimo da linearni kod uvek koduje nula poruku u nula kodnu reč. Ovo je posledica linearnosti koda, odnosno činjenice da je linearni kod vektorski prostor.

5.2.2. SISTEMSKA FORMA LINEARNIH KODOVA

Od svih mogućih generator matrica, jedna ima poseban oblik, ako postoji. To je sistematska forma.

DEFINICIJA 5.9 Sistematska forma.

Generator Matrica $G(n, m)$ linearnog koda je sistematske forme ako se može zapisati u obliku:

$$G = [I_m \ P] = \begin{bmatrix} 1 & 0 & \dots & 0 & p_{1,1} & \dots & p_{1,n-m} \\ 0 & 1 & \dots & 0 & p_{2,1} & \dots & p_{2,n-m} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & p_{m,1} & \dots & p_{m,n-m} \end{bmatrix},$$

gde je I_m jedinična matrica dimenzije m , dok je P $m \times (n - m)$ matrica koja se često naziva matrica parnosti (engl. *Parity Matrix*).

Primetimo, da ukoliko postoji sistematska forma generator matrice linearnog koda, ona je jedinstvena.

DEFINICIJA 5.10 Sistematski linearni kod.

Linearni kod koji koristi generator matricu u sistematskoj formi se naziva sistematski (linearni) kod.

Kada (n, m) linearni kod koristi sistematsku formu generator matrice, tada je prvih m simbola od ukupno n simbola kodne reči egzaktno jednak simbolima poruke:

$$z_i = (u_{i,1}, u_{i,2}, \dots, u_{i,m}, z_{i,m+1}, \dots, z_{i,n}).$$

Drugim rečima, sistematski kod prvo šalje nekodovanu poruku, a zatim $(n - m)$ kodnih simbola za potrebe detekcije i ispravljanja grešaka.

PRIMER 5.13

Vratimo se na primer 5.12. Drugi mogući izbor bazisa je $z_1 = 1011$, $z_2 = 0110$, što daje:

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Što je ujedno sistematska forma generator matrice ovog koda.

PRIMER 5.14 Biti provere parnosti.

Za binarne poruke bit provere parnosti je bit koji odgovara parnosti bita poruke, tj. Binarna suma bita poruke. Na primer, bit provere parnosti za poruku 01101 je $1 + 1 + 1 = 1$, dok je za poruku 00101 jednak $1 + 1 = 0$.

Kodovanje sa bitom provere parnosti se sastoji jednostavno u slanju nekodovane poruke iza koje sledi bit parnosti. U terminima teorije kodovanja ovo odgovara $(m + 1, m)$ binarnom linearnom kodu, čija je generator matrica:

$$G = \begin{bmatrix} & 1 \\ & 1 \\ I_m & \cdot \\ & \cdot \\ & 1 \end{bmatrix},$$

koja je u sistematskoj formi. Primetimo da je minimalno rastojanje ovog koda 2 (prema teoremi 5.2), što znači da ovaj kod može da detektuje samo jednu grešku (videti teoremu 5.1).

5.2.3. DEKODOVANJE: VERIFIKACIONA MATRICA

Na koji način linearni kodovi detektuju i ispravljaju greške?

Pretpostavimo da smo našli matricu F , takvu da za svaku kodnu reč z važi da je $z \cdot F = 0$. Tada, ako se prilikom prenosa z desi greška e , što znači da je na prijemu $\hat{z} = z + e$, imamo:

$$\hat{z} \cdot F = (z + e) \cdot F = z \cdot F + e \cdot F = 0 + e \cdot F = e \cdot F.$$

Ovaj rezultat je od velike koristi, pošto proizvod $\hat{z} \cdot F$ je nezavisan od emitovane kodne reči z i zavisi samo od greške e . Prema gornjim izrazima sledi da je greška u prenosu jednaka linearnim kombinacijama redova matrice F . U cilju ispravljanja i/ili detektovanja grešaka, potrebno je jednostavno vektore iz vektorskog prostora generisanog redovima matrice F preslikati u odgovarajuće korigovane (ili detektovane) poruke. Iz matematičkih razloga jednačina $z \cdot F = 0$ se uvek daje u obliku:

$$z \cdot H^T = 0.$$

DEFINICIJA 5.11 Verifikaciona matrica.

Matrica H dimenzije $(n - m) \times n$ je verifikaciona matrica za (n, m) D-arni linearni kod C , ako i samo ako važi:

$$\forall z \in GF(D)^n \quad z \cdot H^T = 0 \Leftrightarrow z \in C.$$

Drugim rečima, verifikaciona matrica koda C je jezgro (kernel) koda C .

Primetimo da jedan kod može imati više različitih verifikacionih matrica. Svaka matrica čiji su redovi bazis vektorskog prostora koji je ortogonalan na linearni kod je verifikaciona matrica toga koda.

Kako naći verifikacionu matricu?

U slučaju sistematskog koda verifikaciona matrica se nalazi lako.

TEOREMA 6.3

Za sistematski linearni (m, n) kod C , čija je sistematska forma generator matrice data sa:

$$G = [I_m \ P],$$

matrica oblika:

$$H = [-P^T \ I_{n-m}]$$

je verifikaciona matrica.

Primetimo da je u slučaju polja $GF(2)$, $-P = P$.

PRIMER 5.15

Razmotrimo sistematski kod čija je generator matrica data sa:

$$G = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} I_2 & 1 & 0 & 1 \\ & 1 & 1 & 1 \end{bmatrix}.$$

Tada je ($n = 5, m = 2$):

$$H = \begin{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}^T & I_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

je jedna moguća verifikaciona matrica koda C .

Kako se dobija verifikaciona matrica u opštem slučaju, kada generator matrica nije u sistematskoj formi?

Neka je H verifikaciona matrica (n, m) D -arnog linearnog koda sa generator matricom G , čiji su redovi $z_1, z_2, \dots, z_m$. H možemo naći na osnovu sledećeg algoritma:

1. Za $i = m + 1, \dots, n$, izabrati z_i kao proizvoljan vektor u $GF(D)^n$ linearno nezavisan od $z_1, z_2, \dots, z_{i-1}$,
2. Izračunati inverznu matricu M^{-1} matrice M čiji su redovi $z_1, z_2, \dots, z_n$,
3. Izdvojiti H^T kao poslednjih $n - m$ kolona matrice M^{-1} .

PRIMER 5.16

Vratimo se na primer 5.12 i $(4,2)$ kod, čiji je jedan generator dat sa

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Nađimo prvo dva vektora nezavisna od ova prethodna dva. Izaberimo npr. 1000 i 0100.

Tada je:

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, M^{-1} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

Odakle sledi:

$$H^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \text{ tj. } H = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}.$$

5.2.4. DUALNI KODOVI

Verifikaciona matrica (n, m) D-arnog linearnog koda C je $(n - m) \times n$ matrice H takva da je $\ker(H)$ jednak C : $\ker(H) = C$. Na osnovu fundamentalne teoreme o dimenzijama vektorskog prostora $\dim(\ker(H)) + \text{rank}(H) = n, \dim(C) + \text{rank}(H) = n$ ili $\text{rank}(H) = n - m$.

Prema tome $n - m$ redova matrice H generiše $n - m$ dimenzionalni podprostor $G(D^n, \text{ tj. } (n, n - m) \text{ linearni kod. Ovaj kod se naziva dualni kod koda } C \text{ (i obrnuto).$

OSOBI NA 5.11

Dualni kod dualnog koda C je sam kod C .

OSOBI NA 5.12

Generator matrica jednog koda je verifikaciona matrica njegovog dualnog koda i obrnuto.

5.2.5. SINDROMI

Ponovimo još jednom ključne ideje linearnih kodova. Ako je z kodna reč koja se prenosi, a e greška prilikom prenosa, primljena reč je $\hat{z} = z + e$. Ako je H verifikaciona matrica korišćenog koda za ispravljanje grešaka, tada je:

$$\hat{z} \cdot H^T = (z + e) \cdot H^T = z \cdot H^T + e \cdot H^T = 0 + e \cdot H^T = e \cdot H^T$$

Ovo ilustruje važnu činjenicu da je $\hat{z} \cdot H^T$ zavisi samo od vektora greške e , a ne i od prenesene kodne reči $\hat{z}$. Stoga je veličina $\hat{z} \cdot H^T$ od posebne važnosti u dekodovanju i naziva se sindrom ($\hat{z}$ u odnosu na H).

DEFINICIJA 5.12 Sindromi.

Sindrom reči $\hat{z}$ u odnosu na verifikacionu matricu H je proizvod $\hat{z} \cdot H^T$.

OSOBINA 5.13

Sindrom $s = \hat{z} \cdot H^T$ primljene reči $\hat{z}$ u odnosu na verifikacionu matricu H koda C zavisi samo od greške u prenosu $e = \hat{z} - z_i$, a ne i od prenošene kodne reči z_i , $z_i \in C$.

Greška e se može dekomponovati u elementarne greške (greške na nivo svakog simbola), odnosno vektor greške ima svoje komponente $e = (e_1, \dots, e_n)$, koje se mogu izraziti sa:

$$s(\hat{z}) = \hat{z} \cdot H^T = e \cdot H^T = \sum_k e_k h_k$$

gde je h_k k -ta kolona matrice H , $H = [h_1, \dots, h_n]$.

Da bi se našao korektor (suprotno od greške) dovoljno je naći korektore koji odgovaraju pojedinačnim greškama. Ispravljanje više grešaka se vrši sumiranjem odgovarajućih korektora za pojedinačne greške.

Korigovanje se dakle vrši u dva koraka: mapiranjem kolona matrice H u korektore (koji se pamte u memoriji), dodavanjem jedinice na odgovarajuće nenulte pozicije sindroma.

PRIMER 5.17 Korekcione tablice zasnovane na sindromima.

Neka je:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

verifikaciona matrica zadatog binarnog koda.

Tada se na osnovu kolona matrice H izvode sledeći korektori, tabela ispod.

Tabela a		Tabela b	
Sindrom	Korektor	Sindrom	Korektor
101	10000	000	00000
		001	00001
111	01000	010	00010
		011	?
100	00100	100	00100
		101	10000
010	00010	110	?
001	00001	111	01000

jednostavno rednim iščitavanjem kolona matrice H , (tabela a). Ako izvršimo uređenje prema sindromima, kako se to u praksi i koristi dobijamo (tabelu b).

Primetimo sledeće:

Nula sindrom se uvek mapira i nema korigovanja, na osnovu definicije 5.11. Za 011 i 110 korektor nije jedinstven u ovom primeru.

Na primer:

$011 = 010 + 001$, što daje 00011 ($00001 + 00010$),

$011 = 111 + 100$, što daje 01100.

Razlog za to leži u činjenici da ovaj kod ima minimalno rastojanje 3 i može da koriguje samo 1 grešku i ne sve oblike vektora greške sa ukupno 2 ili više grašaka. Ova dva sindroma zaista odgovaraju dvema greškama u prenosu.

U praksi, pamti se tabela sindroma i odgovarajućih korektora. Opšta procedura dekodovanja primljene poruke $\hat{z}$ je sledeća:

1. Izračunati sindrom $s(\hat{z}) = \hat{z} \cdot H^T$.
2. Izračunati korektor $c = -e$, na osnovu linearnih kombinacija memorisanih korektora.
3. Izvršiti dekodovanje $z = \hat{z} + c$.

PRIMER 5.18 Dekodovanje linearnih kodova.

Nastavimo sa primerom 5.17, u kome je generator matrica data sa:

$$G = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Pretpostavimo da se prenosi $u = 10$. Ova poruka se koduje u reč $z = 10101$, sa datim linearnim (5,2) kodom. Pretpostavimo da je $\hat{z} = 00101$, odnosno da je prilikom prenosa prvi bit pogrešno prenet. Računamo sindrom $s(\hat{z}) = \hat{z} \cdot H^T = 101$, što vodi ka korektoru $c = 10000$, videti (tabelu a) iz primera 5.17.

Dekodovanje se vrši po formuli $\hat{z} + c = 00101 + 10000 = 10101$, što daje u prva dva bita dekodovanu poruku 10, pošto je korišćen sistematski binarni kod, i kao što vidimo ona odgovara originalnoj poruci na ulazu u kanal.

KONTROLNO PITANJE

Šta je originalna poruka, ako je primljena poruka 101011001, verifikaciona matrica upotrebljenog koda je data sa:

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

ODGOVOR

Originalna poruka je 10001. $s(\hat{z}) = \hat{z} \cdot H^T = 0101$, što odgovara trećoj koloni u H , što znači da se greška desila na trećoj poziciji. Stoga je emitovana kodna reč 100011001 (promenjen je treći bit), a budući da je kod sistematičan (videti H), originalna poruka je 10001 (uzeti prvih $m = 9 - 4 = 5$ bita).

5.2.6. MINIMALNO RASTOJANJE I VERIFIKACIONA MATRICA

Kod linearnih kodova moguće je izračunati minimalno rastojanje na osnovu verifikacione matrice.

TEOREMA 5.4 Verifikaciona matrica i minimalno rastojanje.

Ako je H verifikaciona matrica (n, m) D-arnog linearnog koda C ($0 \leq m \leq n$), tada je minimalno rastojanje koda $d_{\min}(C)$ jednako najmanjem broju linearno zavisnih kolona u H .

Za binarne linearne kodove C sa verifikacionom matricom H , ovaj rezultat znači sledeće:

1. H nema nula kolone, odnosno $d_{\min}(C) > 1$.
2. Ako H nema dva puta istu kolonu, $d_{\min}(C) > 2$.

PRIMER 5.19

Binarni linearni kod sa verifikacionom matricom:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix},$$

Ima rastojanje 3. Zaista, H nema nula kolonu niti dva puta istu kolonu, pa je stoga $d_{\min}(C) > 2$. U H postoje tri linearno zavisne kolone h_1, h_3 i h_5 .

OSOBINA 5.14

Za (n, m) linearni kod C , važi:

$$d_{\min}(C) \leq n - m + 1.$$

5.2.7. BINARNI HEMINGOVI KODOVI

Sada ćemo izučavati dobre binarne kodove koji mogu da koriguje jednu grešku. Pošto želimo da ispravimo samo jednu grešku, dovoljno je da sindrom daje indikaciju gde se ta greška nalazi. Dakle ideja je da sindrom direktno daje poziciju, npr. 00...01 za prvu poziciju, 00...10 za drugu, 00...11 za treću, i td.

Podsetimo se da jedna greška na poziciji k odgovara sindromu koji je k -ta kolona matrice H . Ova činjenica vodi ka ideji da se konstruiše verifikaciona matrica čije su kolone binarne reperezentacije njihovih pozicija.

Šta nam još uvek nije jasno:

Koku dimenziju treba da ima ova verifikaciona matrica?

Da li ova konstrukcija zaista daje verifikacionu matricu koda?

Da li ovako konstruisan kod može zaista da koriguje sve vektore grešaka sa ukupno jednom greškom?

Što se tiče prvog pitanja, podsetimo se da je sindrom linearnog (n, m) koda dužine $n - m$. Ako sindrom direktno koduje poziciju greške, trebalo bi da reprezentuje $2^{n-m} - 1$ pozicija. Prema tome, ako se izabere da je dužina kodne reči $n = 2^{n-m} - 1$, sve pozicije jedne greške će biti zastupljene u verifikacionoj matrici.

U donjoj tabeli su date moguće vrednosti dimenzije ove klase kodova.

n	m	$r = n - m$
7	4	3
15	11	4
31	26	5
63	57	6
.	.	.
.	.	.

Navedimo dva primera verifikacionih matrica za dužine koda $n = 7$ i $n = 15$.

$$H_3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$H_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Drugo pitanje ima pozitivan odgovor, dakle ova konstrukcija daje jedan linearni kod, budući da su matrice koje rezultuju u ovoj konstrukciji punog ranga, odnosno $\text{rank}(H) = n - m$. Ovo se može pokazati tako što se lako može konstruisati jedinična matrica I_{n-m} od kolona datih matrica (uzimanjem 1., 2., 4., 8. kolone). Odavde sledi da je dimenzija kernela ove matrice m , što daje (n, m) linearni kod.

Da bi smo odgovorili na poslednje pitanje: da li ovaj kod može da ispravi sve oblike vektora grešaka sa ukupno jednom greškom, dovoljno je izračunati minimalno rastojanje. Verifikaciona matrica dobijena ovom konstrukcijom nikad neće imati nula kolonu niti dve iste kolone, tako da je u važnosti $d_{\min}(C) \geq 3$. S druge strane prve tri kolone ove matrice predstavljaju binarnu reprezentaciju brojeva 1, 2 i 3 i kao takve su uvek linearno zavisne. Odavde sledi da je minimalno rastojanje ovog koda uvek 3 i da može da ispravi sve oblike vektora grešaka sa ukupno jednom greškom.

Ovaj kod se naziva binarni Hemingov kod.

DEFINICIJA 5.13 Hemingov kod.

Hemingov kod je $(2^r - 1, 2^r - r - 1)$ binarni linearni kod, ($r \geq 2$), čija je verifikaciona matrica data sa:

$$H_r = [b_r(1)^T \ b_r(2)^T \ \dots \ b_r(n)^T] = \begin{bmatrix} 0 & 0 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

gde je $b_n(i)$ binarna reprezentacija broja i pomoću n -bitne reči (npr. $b_4(5) = 0101$).

OSOBIINA 5.15

Svaki binarni Hemingov kod može da koriguje sve oblike vektora greške sa ukupno jednom greškom.

PRIMER 5.20 Hemingov kod.

Uzmimo da je $r = 3$ i konstruišimo Hemingov binarni kod $(7,4)$. Tada imamo:

$$H_3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Da bi smo našli generator matricu, potrebno je da pronađemo 4 vektora z , takvih da je $z \cdot H^T = 0$.

To su na primer vektori (ima više mogućih rešenja):

$$z_1 = 1110000$$

$$z_2 = 1101001$$

$$z_3 = 1000011$$

$$z_4 = 1111111$$

što daje generator matricu:

$$G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Pretpostavimo da je poslata poruka $u = 1001$. Ona se koduje u kodnu reč $z = u \cdot G = 0001111$. Neka se u toku prenosa desi greška na trećem bitu, tako da je primljena kodna reč sada $\hat{z} = 0011111$.

Kako se ona dekoduje?

Sindrom $s(0011111) = \hat{z} \cdot H^T = 011$, odnosno binarna reprezentacija za 3, što znači da se desila jedna greška na 3. bitu. Stoga je rezultat dekodovanja $\hat{z} - 0010000 = 0001111$, dakle kodna reč koja je stvarno i poslata.



5.3. CIKLIČNI KODOVI

Naučićemo:

Šta su ciklični kodovi?

Kako se kodovi prezentuju pomoću polinoma?

Kako se vrši kodovanje i dekodovanje pomoću cikličnih kodova?

DEFINICIJA 5.14

Ciklični kod je linearni blok kod takav da ako je z_i kodna reč sa n simbola $z_i = z_{i,1}, z_{i,2}, \dots, z_{i,n}$, tada je i reč $z_{i,2}, z_{i,3}, \dots, z_{i,n}, z_{i,1}$ koja se dobija cikličnom permutacijom (poznatom i kao pomeranje u levo) simbola kodne reči z_i , takodje kodna reč datog cikličnog koda.

Ovo praktično znači da je svaka ciklična permutacija kodne reči takodje kodna reč.

PRIMER 5.21 Ciklični kod.

Sledeći linearni blok kod je ciklični kod:

$$z_1 = 000, z_2 = 101, z_3 = 011, z_4 = 110.$$

Sledeći kod nije ciklični kod:

$$z_1 = 000, z_2 = 001, z_3 = 010, z_4 = 011,$$

Pošto ciklična permutacija kodne reči z_3 nije u skupu kodnih reči.

Ciklični kodovi su važna klasa linearnih kodova, pošto je u ovoj kasi moguće konstruisati efikasne procedure kodovanja i dekodovanja.

5.3.1. CIKLIČNI KODOVI I POLINOMI

Ciklični kodovi imaju dodatno ograničenje na kodne reči (svaka kodna reč je ciklična permutacija neke kodne reči), što ih u pogledu algebarske strukture čini kompletnijim od vektorskih prostora. Algebra nad polinomima je pogodan matematički aparat za njihovu reprezentaciju.

Neka je z_i kodna reč sa n simbola $z_i = z_{i,1}, z_{i,2}, \dots, z_{i,n}$ predstavljena polinomom

$$z_i(X) = z_{i,1} \cdot X^{n-1} + z_{i,2} \cdot X^{n-2} + \dots + z_{i,n},$$

Odnosno j ti simbol $z_{i,j}$ kodne reči z_i dužine n je koeficijent uz stepen X^{n-j} odgovarajućeg polinoma $z_i(X)$.

Šta je onda $X \cdot z_i(X)$ modulo $(X^n - 1)$?

Pokazaćemo da ova operacija odgovara cikličnom pomeranju u levo simbola kodne reči z_i .

DOKAZ

$$\begin{aligned} X \cdot z_i(X) &= X \cdot \left(\sum_{j=1}^n z_{i,j} \cdot X^{n-1} \right) = \sum_{j=1}^n z_{i,j} \cdot X^{n-j+1} = \sum_{k=0}^{n-1} z_{i,k+1} \cdot X^{n-k} \\ &= z_{i,1} \cdot X^n + \sum_{k=1}^{n-1} z_{i,j+k} \cdot X^{n-k} \end{aligned}$$

Modul $X^n - 1$, jednostavno znači da X^n odgovara 1, X^{n+1} odgovara X , i td. Stoga $z_{i,1} \cdot X^n$ modulo $X^n - 1$ odgovara $z_{i,1}$, tako da dalje imamo:

$$X \cdot z_i(X) = z_{i,1} + \sum_{k=1}^{n-1} z_{i,k+1} \cdot X^{n-1} = \sum_{k=1}^{n-1} z_{i,k+1} \cdot X^{n-k} + z_{i,1} \cdot X^{n-n}$$

Što zaista odgovara kodnoj reči $z_{i,2}, z_{i,3}, \dots, z_{i,n}, z_{i,1}$, odnosno cikličnom pomeranju u levo kodne reči z_i .

PRIMER 5.22 Modul $X^n - 1$ aritmetika.

Navedimo jedan primer ove aritmetike:

$$\begin{aligned}(X^2 + 1) \cdot (X + 1) &= X^3 + X^2 + X + 1 = 1 + X^2 + X + 1 \bmod(X^3 - 1) \\ &= X^2 + X + (1 + 1) \bmod(X^3 - 1) = X^2 + X \bmod(X^3 - 1)\end{aligned}$$

pošto je u binarnom slučaju $1 + 1 = 0$.

PRIMER 5.23 Polinomijalna prezentacija cikličnih kodova.

Podsetimo se binarnog cikličnog koda iz prethodnog primera:

$$z_1 = 000, z_2 = 101, z_3 = 011, z_4 = 110.$$

Polinomijalna prezentacija ovog koda je data sa:

$$z_1(X) = 0$$

$$z_2(X) = 1 \cdot X^2 + 0 \cdot X + 1 = X^2 + 1$$

$$z_3(X) = 0 \cdot X^2 + 1 \cdot X + 1 = X + 1$$

$$z_4(X) = 1 \cdot X^2 + 1 \cdot X + 0 = X^2 + X$$

Primetimo da je:

$$X \cdot z_2(X) = z_3(X) \bmod(X^3 - 1),$$

što znači da je z_3 ciklični pomeraj u levo kodne reči z_2 .

Pitanje:

Šta su polinomijalne prezentacije sledećih kodnih reči?

1. 00000000
2. 10001
3. 0000001
4. 1111

Odgovori:

1. 0
2. $X^4 + 1$
3. 1
4. $X^3 + X^2 + X + 1$

Pitanje:

Posmatrajmo dve kodne reči z_1 i z_2 jednog cikličnog koda.

Šta je onda $z_1 \cdot z_2$ u sledećim slučajevima?

1. $z_1 = 010110, z_2 = 000100$
2. $z_1 = 1010, z_2 = 0101$
3. $z_1 = 11001, z_2 = 01010$

Odgovori:

1. $(X^4 + X^2 + X) \cdot X^2 = X^4 + X^3 + 1 = 011001, \text{ mod}(X^6 - 1)$
2. $(X^3 + X)(X^2 + 1) = X^5 + X^3 + X^3 + X = X + X = 0, \text{ mod}(X^4 - 1)$
3. $(X^4 + X^3 + 1)(X^3 + X) = X^7 + X^6 + X^3 + X^5 + X^4 + X = X^2 + X + X^3 + 1 + X^4 + X = X^4 + X^3 + X^2 + 1 = 11101$

OSOBIKA 5.16

Ako je $z(X)$ polinom koji odgovara kodnoj reči z cikličnog koda dužine n , tada svaki polinom $p(X)$, $p(X) \cdot z(X) \text{ mod } (X^n - 1)$ odgovara polinomu čija odgovarajuća kodna reč pripada tom istom cikličnom kodu (ciklični pomeraj u levo plus linearne kombinacije).

TEOREMA 5.5

Za svaki ciklični (n, m) kod C postoji polinom $g_C(X)$ stepena $n - m$, takav da je:

$$C = \{g_C(X) \cdot p : p \in GF(X), \deg(p) < m\}$$

Odnosno svaki polinom kodne reči je umnožak $g_C(X)$ i obrnuto. To znači da je kod C generisan pomoću $g_C(X)$. Ovaj polinom se naziva generatorom koda C .

Kodovanje reči u pomoću cikličnog koda C se sastoji u slanju $z(X) = u(X) \cdot g_C(X)$.

Kao što znamo, uvek je poželjno imati sistematski kod u kome prvi simboli odgovaraju poruci. Za (n, m) ciklični kod, tada procedura kodovanja obuhvata sledeće korake:

1. Množenje polinoma poruke $u(X)$ sa X^{n-m} (ciklično pomeranje u levo za $n - m$ mesta). Primetimo da je $n-m$ stepen generatora.
2. Deljenje $X^{n-m} \cdot u(X)$ sa generatorom $g(X)$ i dobijanje ostatka $r(X)$.
3. Kodovanje $u(X)$ je tada $z(X) = X^{n-m}u(X) - r(X)$, čijih m najviših simbola odgovara m simbola poruke u .

PRIMER 5.24 Sistematsko kodovanje cikličnim kodovima

Posmatrajmo $(7,4)$ binarni ciklični kod:

$$\begin{aligned} z_1 &= 000000, z_2 = 0001011, z_3 = 0010110, z_4 = 0101100, \\ z_5 &= 1011000, z_6 = 0110001, z_7 = 1100010, z_8 = 1000101, \\ z_9 &= 1010011, z_{10} = 0100111, z_{11} = 1001110, z_{12} = 0011101, \\ z_{13} &= 0111010, z_{14} = 1110100, z_{15} = 1101001, z_{16} = 1111111, \end{aligned}$$

Ovaj kod ima generator $g(X) = z_2(X) = X^3 + X + 1$. Cilj nam je da prenesemo poruku $u = 1101$, odnosno $U(X) = X^3 + X^2 + 1$.

Podelimo prvo $X^3u(X) = X^6 + X^5 + X^3$ sa $g(X)$. Nakon deljenja dobijamo:

$$X^3u(X) = (X^3 + X^2 + X + 1)g(X) + 1.$$

Odnosno kodna reč je:

$$z(X) = X^3u(X) + 1 = X^6 + X^5 + X^3 + 1.$$

što odgovara 1101001. Da sumiramo: poruka 1101 se koduje u 1101001. Primetimo da su prva četiri bita jednaka originalnoj poruci u , što je i bio cilj sistematskog kodovanja.

$$\bullet \quad 1 \cdot X^6 + 1 \cdot X^5 + 0 \cdot X^4 + 1 \cdot X^3$$

$$\frac{X^3 + X + 1}{X^3}$$

- $\underline{1 \cdot X^6 + 0 \cdot X^5 + 1 \cdot X^4 + 1 \cdot X^3}$
- $0 \cdot X^6 + 1 \cdot X^5 + 1 \cdot X^4 + 0 \cdot X^3$ $\frac{X^3+X+1}{X^3+X^2}$
- $\underline{0 \cdot X^6 + 1 \cdot X^5 + 0 \cdot X^4 + 1 \cdot X^3 + 1 \cdot X^2}$
- $0 \cdot X^6 + 0 \cdot X^5 + 1 \cdot X^4 + 1 \cdot X^3 + 1 \cdot X^2$ $\frac{X^3+X+1}{X^3+X^2+X}$
- $\underline{0 \cdot X^6 + 0 \cdot X^5 + 1 \cdot X^4 + 0 \cdot X^3 + 1 \cdot X^2 + 1 \cdot X}$ $\frac{X^3+X+1}{X^3+X^2+X+1}$
- $0 \cdot X^6 + 0 \cdot X^5 + 0 \cdot X^4 + 1 \cdot X^3 + 0 \cdot X^2 + 1 \cdot X$
- $\underline{0 \cdot X^6 + 0 \cdot X^5 + 0 \cdot X^4 + 1 \cdot X^3 + 0 \cdot X^2 + 1 \cdot X + 1}$
- $0 \cdot X^6 + 0 \cdot X^5 + 0 \cdot X^4 + 0 \cdot X^3 + 0 \cdot X^2 + 0 \cdot X + 1$ -----> ostatak deljenja 1
- $X^3 u(X) = (X^3 + X^2 + X + 1)g(X) + 1$

TEOREMA 5.6

Generator cikličnog koda dužine n je faktor polinoma $X^n - 1$.

TEOREMA 5.7

Svaki faktor polinoma $X^n - 1$ stepena r je generator jednog $(n, n - r)$ cikličnog koda.

5.3.2. DEKODOVANJE CIKLIČNIH KODOVA

Postupak dekodovanja cikličnih kodova se sastoji u sledećem:

1. Prvo se izračunava sindrom na osnovu primljene reči (podsetimo se da sindrom zavisi samo od greške, a ne i od emitovane kodne reči i jednak je nuli ako je primljena reč jednaka kodnoj reči).
2. Izračunavanje korektora (koji je suprotan od greške).
3. Konačno, primenjuje se korektor na primljenu reč.

Konstrukcija sindroma $\hat{z}(X)$ se svodi na izračunavanje ostatka deljenja $\hat{z}(X)$ sa generatorom koda $g(X)$.

Zaista, mi znamo za ciklične kodove da je svaka kodna reč $z(X)$ multipl od generatora $g(X)$. Stoga je ostatak deljenja $z(X) + e(X)$ sa $g(X)$ isti kao ostatak deljenja samo sa $e(X)$.

$$z(X) = \alpha(X) \cdot g(X)$$

$e(X) = \beta(X) \cdot g(X) + s(X)$, odakle sledi da je:

$$z(X) + e(X) = [\alpha(X) + \beta(X)] \cdot g(X) + s(X),$$

pri čemu je $\deg(s(X)) < \deg(g(X))$.

Jasno je iz ove konstrukcije da je sindrom $s(X)$ nula ako i samo ako je $\hat{z}(X)$ kodna reč (odnosno multipl od $g(X)$). Korektori koji odgovaraju nenultim sindromima se mogu dobiti deljenjem sa $g(X)$. Imajmo u vidu sledeće:

- za jednu grešku X^i stepena $i < n - m$, sindrom je jednostavno X^i ,
- za jednu grešku X^{n-m} , sindrom je $X^{n-m} - g(X)$.

PRIMER 5.25 Dekodovanje cikličnih kodova.

Nastavimo sa prethodnim primerom: poruka 1101 je kodovana u 1101001 i poslata preko datog kanala. Pretpostavimo da je drugi bit promenjen, tako da je primljena reč 1001001.

Kako se odvija dekodovanje?

1001001 odgovara $\hat{z}(X) = X^6 + X^3 + 1$, čijim se deljenjem sa $g(X)$ dobija:

$$\begin{array}{r|l} X^6 + & X^3 + & 1 & X^3 + X + 1 \\ X^6 + X^4 + X^3 & & & \underline{X^3 + X} \\ & X^4 + & X^2 + X & \\ & & \underline{X^2 + X + 1} & \end{array}$$

Odnosno sindrom je $X^2 + X + 1$.

Sindrom	Korektor
1	1
X	X
X^2	X^2
$X + 1$	X^3
$X^2 + X$	X^4
$X^2 + X + 1$	X^5
$X^2 + 1$	X^6

Prvih 4 je dobijeno na osnovu iznetih zapažanja, dok su poslednje 3 dobijene deljenjem greške sa generatorom $g(X)$.

Nalazimo da korektor mora biti X^5 , dok je konačna dekodovana reč:

$$z(X) = \tilde{z}(X) + X^5 = X^6 + X^5 + X^3 + 1, \rightarrow 1101001.$$

Budući da je korišćen sistematični kod, prvih 4 simbola ove kodne reči su prva četiri simbola originalne poruke: 1101.

PITANJE

Razmotrimo (7,4) linearni kod, čiji je generator $g(X) = X^3 + X^2 + 1$.

Kako će se dekodovati sledeće poruke?

1. 1001011
2. 1011001
3. 0000001

ODGOVORI

1. 1001

$$1001011 = X^6 + X^3 + X + 1 = (X^3 + X^2 + X + 1)(X^3 + X^2 + 1)$$

Ovo znači da je sindrom jednak 0 (ostatak deljenja sa $g(X)$), i budući da se radi o sistematskom kodu, prva 4 bita primljene kodne reči su jednaka poruci.

2. 1010

$$1011001 = X^6 + X^4 + X^3 + 1 = (X^3 + X^2)g(X) + X^2 + 1$$

Sindrom	Korektor
1	1
X	X
X²	X²
X + 1	X³
X² + X	X⁴
X² + X + 1	X⁵
X² + 1	X⁶

Sindrom/korektor tabela za generator $g(X) = X^3 + X^2 + 1$.

Prema tome, izračunatom sindromu $X^2 + 1$, odgovara korektor X^3 . Dekodovana poruka se dobija oduzimanjem primljene kodne reči i greške, odnosno:

$$X^6 + X^4 + X^3 + 1 + X^3 = 1010001.$$

Vidi se zbog sistematičnosti koda da je poruka 1010.

3. 0000001=1

Odavde sledi da je sindrom jednostavno 1, korektor takođe 1, pa je dekodovana reč $1 + 1 = 0000000$, a poruka 0000.

5.4. KONVOLUCIONI KODOVI

Naučićemo:

Šta su konvolucioni kodovi?

Kako se vrši kodovanje konvolucionim kodovima?

Šta je stanje i dijagram stanja?

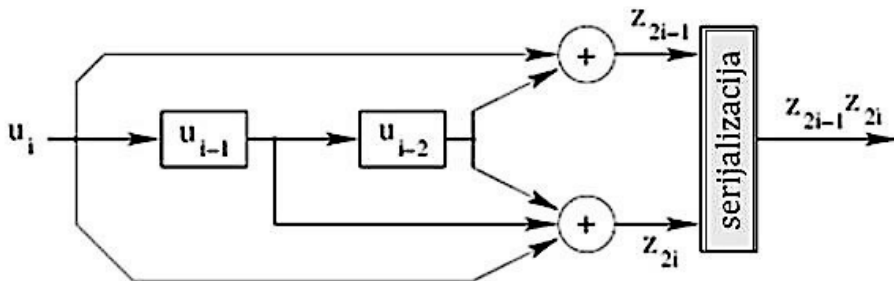
Šta je rešetkasta prezentacija vezana za predstavljanje ovih kodova?

Kako koristiti Viterbijev algoritam na rešetkama, u cilju dekodovanja minimalnim rastojanjem?

Kako izračunati minimalno rastojanje konvolucionih kodova?

Konvolucioni kodovi ne pripadaju klasi blok kodova. Mehanizam kodovanja pamti prethodno kodovane simbole. U izvesnom smislu se mogu tretirati kao neograničeni blok kodova. Proces kodovanja i dekodovanja se značajno razlikuje. Daleko su efikasniji od blok kodova.

Izlaganje o načinu kodovanja konvolucionim kodovima ćemo započeti jednim primerom kodovanja. Šema koda za odabrani primer je dat (Slika 5.1). Radi se o binarnom kodu.



Slika 5.1 Svaki simbol poruke u_i se koduje u dva simbola z_{2i-1} i z_{2i}

U svakom vremenskom trenutku i , jedan simbol poruke u_i ulazi u koder, dok dva simbola kodne reči z_{2i-1} i z_{2i} se emituju, odnosno $u = (u_1, u_2, \dots, u_i, \dots)$ se koduje u $z = (z_1, z_2, \dots, z_{2i-1}, z_{2i}, \dots)$. Stoga je brzina ovog koda $1/2$.

Proces kodovanja za primer sa (Slika 5.1) se može opisati sa:

$$z_{2i-1} = u_i + u_{i-2}$$

$$z_{2i} = u_i + u_{i-1} + u_{i-2}$$

$$u_i \rightarrow (u_{i-2} + u_i, u_{i-2} + u_{i-1} + u_i)$$

Ove jednačine se mogu posmatrati i kao diskretna konvolucija ulazne sekvence sa sekvencama 1, 0, 1, 0, 0... i 1, 1, 1, 0, 0....

Izračunati niz $y[n]$ generisan konvolucijom nizova $x[n]$ i $h[n]$ datih na donjim slikama:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

5.4.1. KODOVANJE KONVOLUCIONIM KODOVIMA

Ako se vratimo na prethodni primer zapazićemo da proces kodovanja nije u potpunosti određen samo konvolucijom, budući da nam nisu poznate prethodne vrednosti u_{i-2} i u_{i-1} . Odnosno, to je ekvivalentno pitanju šta su početna stanja sistema u trenutku $t=0$?

Pogodno je usvojiti nulta početna stanja, dakle $u_{-1} = 0$ i $u_0 = 0$.

Da bi se ovo obezbedilo i za svaku narednu kodnu reč, neophodno je da koder nakon kodovanja prethodne kodne reči pređe u ovo nulto početno stanje. To znači da svaka poruka koja ulazi u koder mora na kraju da sadrži dovoljno nula da bi se sve memorije sistema vratile u nulta početna stanja.

U prethodnom primeru to znači da se kodovanje svake reči završava kodovanjem još dve nule.

PRIMER 5.26 Kodovanje konvolucionim kodom.

Potrebno je kodovati poruku $u = 101$ pomoću konvolucionog kodera prikazanog na (Slika 5.1). Da bi smo sagledali proces kodovanja, izračunajmo sve relevantne veličine. One su date u donjoj tabeli.

i	u_i	Stanje ($u_{i-1}u_{i-2}$)	$z_{2i-1}z_{2i}$
1	1	00	11
2	0	10	01
3	1	01	00
4	(0)	10	01
5	(0)	01	11

Ovo su dodati nulti simboli u cilju vraćanja kodera u nulto stanje. Stoga je odgovarajuća kodna reč 1101000111.

Razmotrimo sada opšti postupak generisanja koda pomoću kodera sa (Slika 5.1).

Neka je poruka dužine 3 bita, odnosno neka je $u = (u_1, u_2, u_3)$, što znači da u postupku kodovanja stvarno učestvuje poruka $u = (u_1, u_2, u_3, 0, 0)$. Dužina odgovarajuće kodne reči je $2 \cdot 5 = 10$. U matričnoj formi, mogu se zapisati na sledeći način:

$$(z_{2i-1}, z_{2i}) = (u_{i-2}, u_{i-1}, u_i) \cdot \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

Kompletna kodna reč z se dobija množenjem $u = (u_1, u_2, u_3, 0, 0)$ sa matricom:

$$G_3 = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Ili jednostavnije, množenjem $u = (u_1, u_2, u_3)$ sa matricom:

$$G_3 = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Dakle, $z = u \cdot G_3$.

Za poruke dužine m , ova formula se može generalizovati na $z = u \cdot G_m$, gde je G_m $m \times (2m + 4)$ matrica napravljena tako da su redovi matrice složeni u jedan red, a zatim za svaki novi red pomereni u desno za dve kolone.

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \rightarrow 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0$$

$$G_3 = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

5.4.2. OPŠTE DEFINICIJE KONVOLUCIONIH KODOVA

DEFINICIJA 5.15

(n, k, r) D-arni konvolucionni kod je neograničeni linearni kod, čija generator matrica ima sledeću (beskonačnu) formu:

$$G_3 = \begin{bmatrix} F_0 & F_1 & F_2 & \dots & F_r & [0] & [0] & [0] & \dots \\ [0] & F_0 & F_1 & \dots & F_{r-1} & F_r & [0] & [0] & \dots \\ [0] & [0] & F_0 & \dots & F_{r-2} & F_{r-1} & F_r & [0] & \dots \end{bmatrix}.$$

Gde su F_i $k \times n$ matrice a $[0]$ $k \times n$ nula matrice. Svaki skup od k redova matrice G je isti kao prethodni, pomeren za n mesta u desno.

DEFINICIJA 5.15 Nastavak

Poruka u dužine m $u = (u_1, \dots, u_m)$ se koduje postupkom

$$z = \bar{u} \cdot G_m',$$

gde je $\bar{u}$ vektor dužine $m' = qk$, $q = \left\lceil \frac{m}{k} \right\rceil$,

$$\bar{u} = (u_1, \dots, u_m, 0, \dots, 0).$$

G_m' je gornja leva podmatrica matrice G dimenzije:

$$qk \times n(r + q).$$

Primetimo da je $\bar{u} = u$, odnosno $m' = m$, ako je m multipl od k (dakle ovo sigurno važi u posebnom slučaju $k = 1$)

U uvedenim notacijama u Definiciji 5.15, k odgovara broju simbola poruke koji ulaze u koder (k ulaznih linija), n odgovara broju simbola izlazne kodne reči po jednom ulazu (n izlaznih linija), dok je r maksimalni broj memorijskih elemenata (dužina registara), na jednoj ulaznoj liniji.

PRIMER 5.27

Kodera sa (Slika 5.1) implementira (2,1,2) konvolucioni kod:

$k = 1$ ulaznih linija, $r = 2$ memorija, dajući $n = 2$ bita kodnih reči za svaki ulazni bit. Kao što smo videli u prethodnom odeljku, za ulazne poruke od 3 simbola, generator matrica je 3×10 matrica.

$$G_3 = \begin{bmatrix} \boxed{1} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{1} \boxed{0} \boxed{0} \boxed{0} \boxed{0} \\ \boxed{0} \boxed{0} \boxed{1} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{1} \boxed{0} \boxed{0} \\ \boxed{0} \boxed{0} \boxed{0} \boxed{0} \boxed{1} \boxed{1} \boxed{0} \boxed{1} \boxed{1} \boxed{1} \end{bmatrix}.$$

U kojoj je zaista $k = 1$ red, a pomeranje redova u desno se vrši za 2 pozicije. U odnosu na opštu definiciju generator matrice, za matricu G_3 važi:

$F_0 = [1 \ 1]$, što odgovara koeficijentima za u_i

$F_1 = [0 \ 1]$, odgovara koeficijentima za u_{i-1}

$F_2 = [1 \ 1]$, odgovara koeficijentima u_{i-2} .

Primetimo da su konvolucioni kodovi linearni. Bilo koja linearna kombinacija kodnih reči je takodje kodna reč, uz konvenciju da se kraće kodne reči izjednačavaju prema dužoj, dodavanjem nula na kraju.

5.4.3. PREDSTAVA PREKO REŠETKI

Stanje jednog sistema je skup internih parametara (sadržaja registara, memorija i sl.) koji su neophodni za izračunavanje izlaza za zadate ulaze. Na primer, za koder sa (Slika 5.1), stanje u trenutku i je trenutni sadržaj dve memorije, odnosno $S = (u_{i-1}, u_{i-2})$.

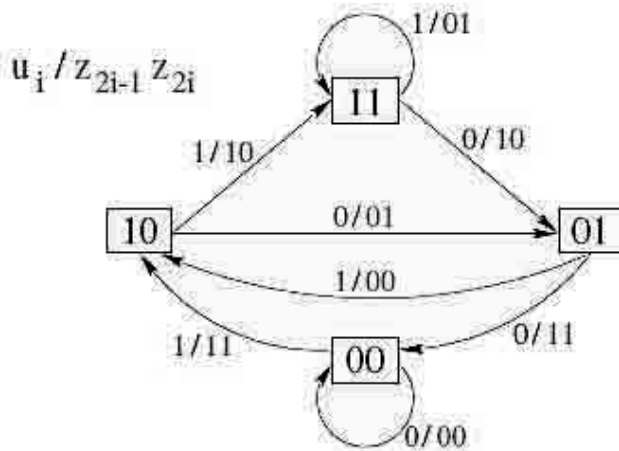
Rad koder je određen stanjem i ulazom. Konvolucionni koder je mašina stanja (engl. *State machine*). Rad takvih sistema je određen dijagramom stanja.

DEFINICIJA 5.16 Dijagram stanja.

Dijagram stanja koder je *gtaf* čiji su čvorovi sva moguća unutrašnja stanja koder. Grana između čvorova S_i i S_j u ovom grafu prezentuje činjenicu da postoji ulaz koji stanje S_i prevodi u S_j . Grane se po pravilu označavaju ulaznim simbolima i odgovarajućim izlaznim simbolima.

PRIMER 5.28 Dijagram stanja

Za koder sa (Slika 5.1) dijagram stanja je dat sa:



Slika 5.2 Dijagram stanja

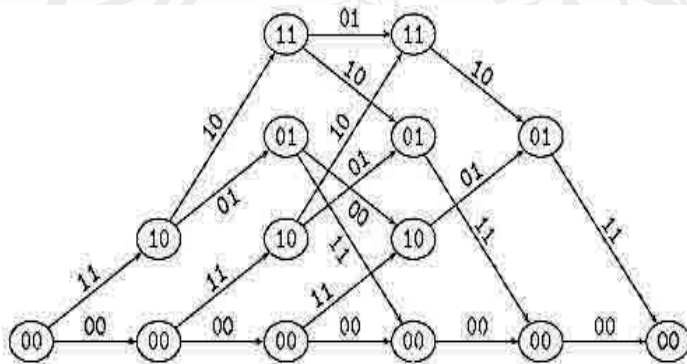
Plave oznake su ulazi koji menjaju stanja, dok su crvene oznake odgovarajući simboli izlaza. Na primer, ako se u stanju 01 primi kao ulaz simbol 1, stanje postaje 10, a na izlazu se pojavljuju dva simbola 00.

Skup kodnih reči konvolucionog koda (n,k,r) za sve moguće poruke dužine m bita odgovara svim mogućim putanjama dužine $n(r + \lceil \frac{m}{k} \rceil)$ u dijagramu stanja, koje startuju od nultog stanja i vraćaju se u to nulto stanje.

Sve putanje u dijagramu stanja iste dužine $n(r + \lceil \frac{m}{k} \rceil)$ razvijene u vremenu, se nazivaju rešetka dimenzije m konvolucionog koda (n,k,r) .

PRIMER 5.29 Rešetka

Za $(2,1,2)$ konvolucioni kod iz prethodnog primera, rešetka dimenzije 3, koja reprezentuje kodovanje svih poruka dužine 3 bita je data na sledećem dijagramu:

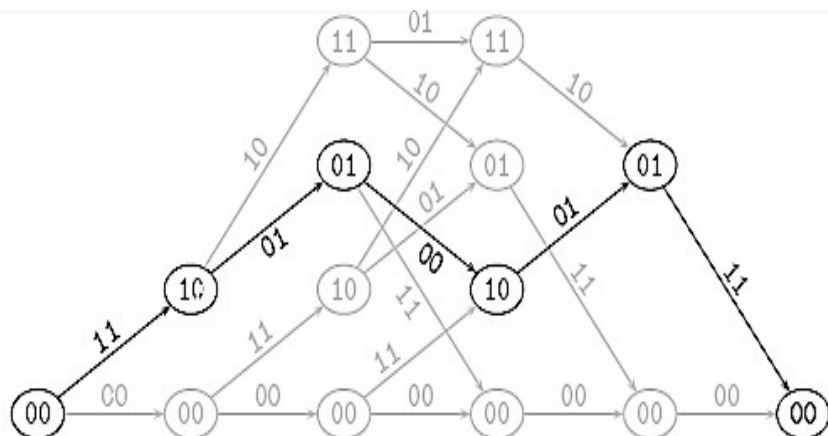


Slika 5.3 Predstava preko rešetki

Gornja grana, za svaki čvor, odgovara ulaznom bitu 1, a donja grana ulaznom bitu 0. Prve tri kolone grana odgovaraju kodovanju 3 bita poruke, a dve poslednje kolone grana odgovaraju završnim nulama, pa stoga imaju samo donje izlazne grane za odgovarajuće čvorove.

PRIMER 5.30 Kodovanje u rešetki.

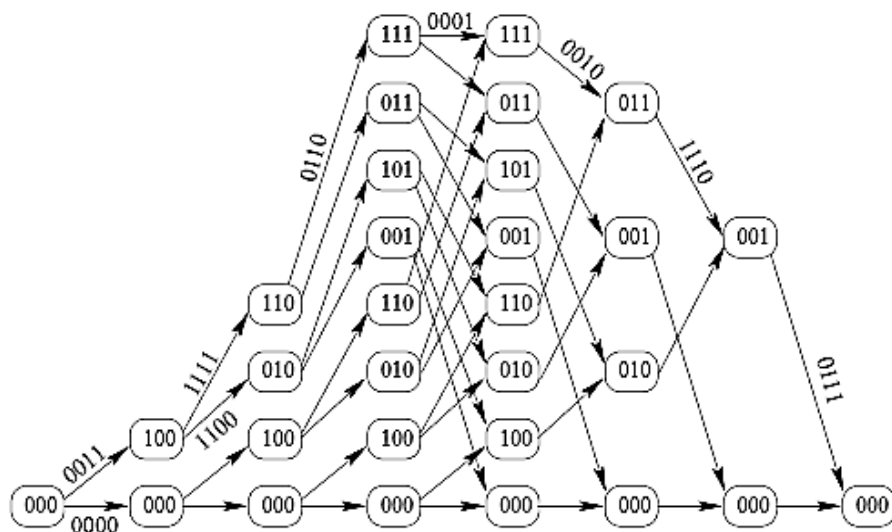
Kodovanje poruke $u = 101$, generiše sledeću putanju:



Slika 5.4 Kodovanje u rešetki

Odakle se vidi da je generisana kodna reč $z = 1101000111$.

Navedimo još jedan primer rešetke (u kojoj nisu sve grane označene):



Slika 5.5 Rešetka u kojoj sve grane nisu označene

U donjoj tabeli su date kodne reči emitovane u svakoj od mogućih situacija:

$S_i \backslash u_i$	0	1
000	0000	0011
001	0111	0100
010	1001	1010
011	1110	1101
100	1100	1111
101	1011	1000
110	0101	0110
111	0010	0001

5.4.4. DEKODOVANJE

Kao što smo videli kodnim rečima odgovaraju putanje od startnog čvora do krajnjeg čvora u rešetki. Dekodovanje se sastoji u nalaženju najpogodnije putanje koja odgovara primljenoj kodnoj reči. U terminima dekodovanja minimalnim rastojanjem, odnosno dekodovanjem kodne reči sa minimalnim brojem grešaka, najpogodnija putanja se definiše kao putanja sa minimalnim Hemingovim rastojanjem u odnosu na poruku koja se dekoduje.

Nalaženje te najbliže putanje se može obaviti dinamičkim programiranjem, odnosno Viterbijevim algoritmom.

Viterbijev algoritam dekoduje jedan za drugim blokove kodnih reči (odnosno m bita poruke nakon prethodnih blokova poruke), pamteći u svakom koraku samo lokalno optimalna rešenja. Ovo se svodi na to da se u svakom čvoru u koloni koja odgovara tekucem bloku dekodovanja, pamti najbolja putanju koja dolazi do tog čvora.

Na kraju, najbolja putanja pronađena za poslednji čvor je ujedno dekodovana poruka. Ono što je važno i korisno za Viterbijev algoritam, je svojstvo da je broj najboljih putanja za svaki vremenski korak, uvek manji ili jednak broju stanja kodera.

Stoga je Viterbijev algoritam linearne umesto eksponencijalne kompleksnosti naivnog algoritma dekodovanja koji bi se sastojao u poređenju svih putanja sa porukom koju treba dekodovati.

Pređimo na detaljniji opis Viterbijevog algoritma. Uvedimo prvo neke oznake.

$\gamma_i(s)$ – najbolje dekodovanje (sa minimalnim mogućim greškama) dužine i koje se završava u stanju s :

$$\gamma_i(s) = \min_{z_1, \dots, z_{2i} \text{ sa krajem u } s} d(z_1 \dots z_{2i}, \hat{z}_1 \dots \hat{z}_{2i})$$

Kompletno dekodovanje odgovara $\gamma_{|\hat{z}|}(00)$, $|\hat{z}|=m+2$, gde je $|\hat{z}|$ dužina poruke koja se dekoduje.

Lako je videti da za svaki par stanja (s, s') važi:

$$\gamma_i(s) = \min_{z_1, \dots, z_{2i} \text{ sa krajem u } s} (d(z_{2i-1}z_{2i}, \hat{z}_{2i-1}\hat{z}_{2i}) + \gamma_{i-1}(s'))$$

Ovo vodi ka Viterbijevom algoritmu:

$\gamma(00) = 0$

for i **from** 1 **to** $|\hat{z}|$ **do**

for all s **do**

$$\gamma_i(s) = \min_{z_1, \dots, z_{2i} \text{ sa krajem u } s} (d(z_{2i-1}z_{2i}, \hat{z}_{2i-1}\hat{z}_{2i}) + \gamma_{i-1}(s'))$$

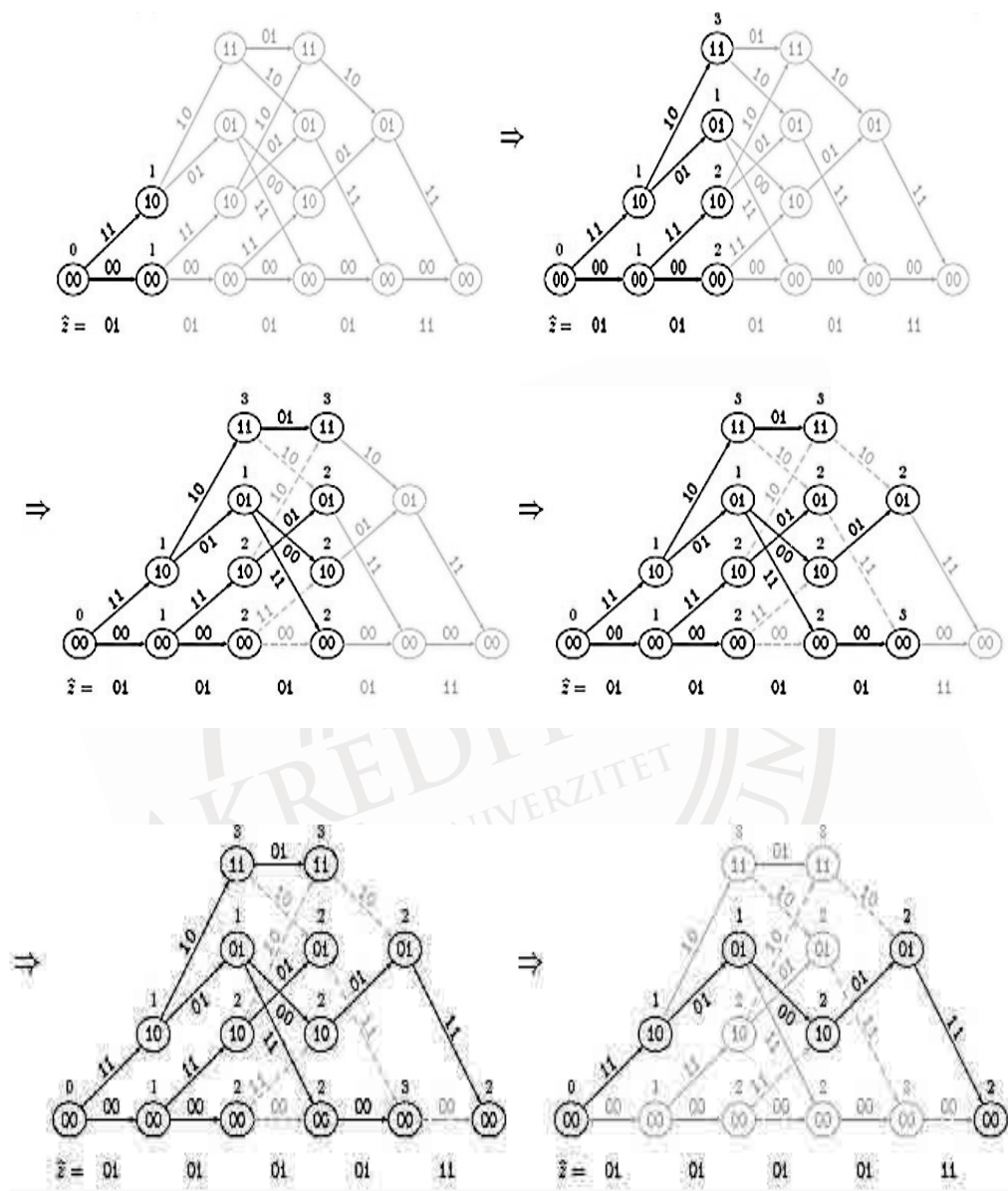
end for

end for

Rekonstrukcija optimalnog puta se dobija unazad, od krajnjeg do početnog stanja.

PRIMER 5.31

Pretpostavimo da je preko kanala sa šumom poslata kodna reč $z = 1101000111$, dok je primljena reč $\hat{z} = 0101010111$, odnosno desile su se dve greške. Tada će Viterbijev algoritam proći kroz sledeće korake.



Slika 5.6 Koraci u Viterbijevom algoritmu

U prvom koraku rada Viterbijevog algoritma, razmatraju se samo dva moguća puta:

- Razmatrani prvi kodni simbol je 1, što vodi od stanja 00 ka stanju 01. Tada se emituje 11 i pravi se jedna greška, budući da je primljeno 01.
- Razmatrani prvi simbol je 0, što vodi od stanja 00 ka stanju 00. Tada se emituje 00 i pravi se jedna greška, budući da je primljeno 01.

U drugom koraku, postoji samo jedan put za dostizanje svakog od stanja (i sada postoje 4 takva puta). Minimalan broj grešaka za svako od ova četiri stanja je jednak minimalnom broju grešaka prethodnih stanja plus broj grešaka odgovarajuće putanje (odnosno razlike između emitovanih i primljenih simbola).

Na primer, ako se razmatra put između stanja 10 i 11, imamo dodatne dve greške, budući da se tada emituje 10, a primljeno je 01. Ovo vodi ka minimalnom broju grešaka u stanju 11 u koraku 2 $\gamma_2(11) = 1 + 2 = 3$.

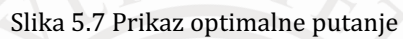
U koraku tri, imamo dve moguće putanje za svako stanje. Algoritam zadržava samo po jednu alternativu koja proizvodi manji broj grešaka. Ostale grane su označene isprekidanim bleđim linijama.

Algoritam radi na istovetnom principu sve do poslednjeg koraka, u kome se dostiže nulto stanje sa minimalnim brojem grešaka jednakom 2.

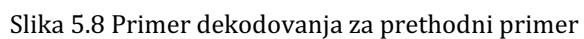
Poslednji korak je rekonstrukcija putanje unazad (plave linije). Ova putanja odgovara kodnoj reči 1101000111.

Nakon dobijanja kodne reči, neophodno je rekonstruisati originalnu poruku. Ovo možemo obaviti prateći oznake grana (0 ili 1) na nađenoj optimalnoj putanji, ili uzimanjem prvog bita svakog stanja kroz koja prolazi optimalna putanja u redosledu od kraja putanje ka početku.

Za ovaj primer i optimalnu putanju označenu plavom bojom, dobijamo niz 10100, i odbacivanjem dve zadnje nule koje nisu deo poruke, dobijamo konačno poruku 101, što i odgovara originalno poslatoj poruci.



Za koder iz prethodnog primera, dekodovati 101110101111101101001100010.



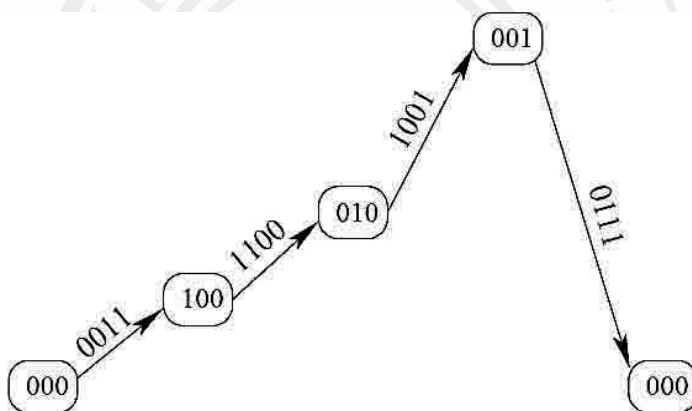
Primljena poruka ima 9 grešaka. Emitovana kodna reč je 0000 0011 1111 0101 1110 0111 0000, što odgovara kodovanoj poruci 0110000, odnosno originalnoj poruci 0110.

Budući da su konvolucionni kodovi linearni kodovi, primenljiva je Teorema 5.2, prema kojoj je minimalno rastojanje jednako minimalnoj težini.

OSOBINA 5.17 Minimalna težina konvolucionog koda.

Minimalna težina jednog konvolucionog koda jednaka je minimalnom broju nenultih simbola na putanji koja polazi i završava se u nultom stanju.

Za koder iz prethodnog primera nalazimo da je $d_{\min}(C) = 9$. Na donjoj slici je iz rešetke kodovanja izdvojena kodna reč koja ima težinu jednaku minimalnoj mogućoj težini 9 za ovaj konvolucionni kod.



Slika 5.9 Kodna reč minimalne težine

5.5. KODOVI NISKE GUSTINE – LDPC

Robert Gallager je 1962. godine definisao posebnu klasu kodova zasnovanu na matricama provere parnosti sa niskom gustinom ili takozvani LDPC kodovi (engl. *Low density parity check matrices*). Međutim, oni su vrlo brzo odbačeni kao nepraktični.

LDPC kodove su 1996. godine ponovo otkrili D. MacKay i R. Neal. Pokazanje je da ovi kodovi imaju veoma veliki kodni dobitak ako se dekoduju iterativno, po *soft* metodu dekodovanja. LDPC kodovi mogu dostići performanse turbo kodova uz znatno niže troškove implementacije.

LDPC kodovi niske gustine predstavljaju klasu linearnih blok kodova. Naziv potiče od njihove karakteristične matrice za proveru parnosti, koja sadrži samo nekoliko jedinica u odnosu na veliki broj nula.

Njihova glavna prednost je što pružaju performanse koje su blizu kapaciteta kanala za veliki broj različitih kanala i linearno vreme kompleksnih algoritama za dekodovanje.

LDPC kodovi se koriste u svim bitnijim bežičnim i optičkim sistemima, kao što su: WiMax (*Worldwide Interoperability for Microwave Access*), DVB (*Digital Video Broadcasting*) standardi druge generacije - najčešće u sprezi sa BCH kodovima, 10 Gbase-T Ethernet itd.

Svaki od ovih sistema koristi različit tip LDPC koda u zavisnosti od mogućnosti sistema i potrebne verovatnoće greške.

Turbo i LDPC kodovi postaju dominantne tehnike zaštitnog kodovanja. U slučaju kada su ove tehnike suviše kompleksne mogu se koristiti blok kodovi (ciklični, zbog jednostavne realizacije koda i dekodera) ili konvolucionni kodovi sa sekvencijalnim dekodovanjem.

U cilju postizanja vrlo visokih brzina prenosa u mobilnim bežičnim sistemima neophodno je kombinovati turbo kodove sa promenljivim kodnim količnikom i konstelacije višeg reda.

5.5.1. REPREZENTACIJA LDPC KODOVA

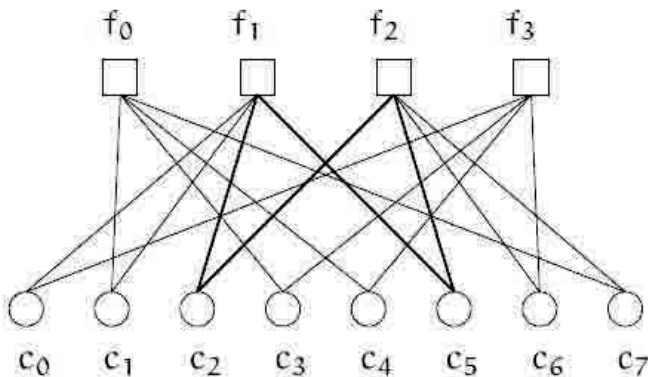
U osnovi postoje dve različite mogućnosti za reprezentaciju LDPC kodova. Kao i svi linearni blok kodovi mogu biti opisani matricama ili druga mogućnost je grafički prikaz.

Matrična reprezentacija

Pogledajmo primer matrice za proveru parnosti kod kodova niske gustine. Sledeća matrica predstavlja matricu za proveru parnosti sa dimenzijama $n \times m$ za kod (8,4).

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Sada možemo da definišemo dva broja koja opisuju ove matrice, w_r za jedinice u svakom redu i w_c za kolone u matrici. Da bi ova matrica postala matrica niske gustine, potrebno je da ispuni dva uslova $w_c \ll n$ i $w_r \ll m$. Da bi se to postiglo, matrica za proveru parnosti mora da bude veoma velika, tako da matrica iz primera ne može da se zove matrica niske gustine.



Slika 5.10 Prikaz matrice preko grafa

Na slici (Slika 5.10), prikazan je graf koji odgovara matrici za proveru parnosti. Označena putanja $c_2 \rightarrow f_1 \rightarrow c_5 \rightarrow f_2 \rightarrow c_2$ je primer za kratki ciklus. Ovakve putanje je bitno izbeći, jer one su loše za performanse pri dekodovanju.

Grafička reprezentacija

Taner je uveo efikasan grafički prikaz LDPC kodova. Pored kompletno predstavljenog koda, ovi grafovi pomažu da se opiše šema algoritma za dekodovanje, kao što će biti objašnjeno u ovom poglavlju.

Tanerovi grafovi su biparitetni grafovi, što znači da su čvorovi podeljeni u dve različite grupe gde grane povezuju samo dva različita tipa čvorova. Prvi tip čvorova u Tanerovom grafu su promenljivi čvorovi (engl. *variable nodes*), a drugi kontrolni čvorovi (engl. *check nodes*).

Slika 5.10 je primer kako preko Tanerovog grafa može da se predstavi matrica. Ovaj graf se sastoji od m kontrolnih čvorova (bitovi parnosti) i n promenljivih čvorova (bitovi u kodnoj reči). Kontrolni čvorovi f_i su povezani sa promenljivim čvorovima c_j ako je element h_{ij} od H jednak 1.

5.5.2. REGULARNI I IREGULARNI LDPC KODOVI

LDPC kodovi mogu biti regularni i nepravilni. Kod je regularan ako je w_c konstantno za svaku kolonu i ako je $w_r = w_c \cdot (n/m)$ takođe, konstantno za svaki red. Matrica iz prvog primera je regularna sa $w_c = 2$ i $w_r = 4$. Takođe, moguće je uvideti regularnost ovog koda gledajući grafičku reprezentaciju. Na grafu postoji isti broj dolazećih veza za promenljive čvorove i za sve kontrolne čvorove. Ukoliko je matrica H niske gustine, a broj jedinica u svakom redu i koloni nije konstantan, tada se kod naziva iregularni LDPC kod.

Tačnije, LDPC kod koji ne ispunjava zadate kriterijume regularnog koda je iregularni (nepravilni) LDPC kod.

Postoji nekoliko različitih algoritama za konstruisanje odgovarajućeg LDPC koda. Jedan takav algoritam je predstavljen od strane Galagera. Osim tog algoritma, Mekež je predložio jedan koji polunausumično generiše matricu niske gustine za proveru parnosti. Ovo je veoma interesantno jer ukazuje na dobre performanse

LDPC kodova, koji ne predstavljaju težak problem. Tačnije, kodovi izabrani na slučajan ili poluslučajan način su velikom verovatnoćom dobri kodovi. Problem koji se javlja ogleda se u visokoj kompleksnosti algoritma za kodovanje ovakvog koda.

5.5.3. PERFORMANSE I KOMPLEKSNOST LDPC KODOVA

Jedna posebna klasa LDPC kodova je samo 0.0045dB daleko od Shannon-ove granice. (Prva Šenonova teorema - nemoguće je izvršiti kompresiju a da prosečan broj bita po simbolu bude manji od entropije izvora datih simbola, ili će doći do gubitka informacije. Ipak, moguće je vršiti kompresiju gde će broj bita po simbolu biti približan entropiji izvora sa malom verovatnoćom gubitka informacije). Ipak, u pitanju je iregularni kod sa kodnim količnikom $1/2$ sa veličinom bloka od 107 bita. Potrebno je više od 1000 iteracija da bi se dostigao ovaj rezultat.

Blokovi velikih dužina rezultuju sa velikom verifikacionom i generator matricom. Kompleksnost množenja kodne reči sa matricom zavisi od broja jedinica u matrici. Ukoliko stavimo matricu niske gustine H u formi $[P^T \ I]$ preko Gausove eliminacije generator matrice G može se izračunati kao $G = [I \ P]$. Podmatrica P neće biti niske gustine, tako da će kompleksnost kodovanja biti prilično visoka.

Budući da kompleksnost raste $O(n^2)$, tada ni matrice niske gustine ne daju dobre performanse ako je dužina bloka velika. Za kodovanje i dekodovanje se koriste iterativni algoritmi. Ovi algoritmi obavljaju lokalne proračune i rezultat prosleđuju putem poruka. Ovaj korak se obično ponavlja nekoliko puta.

Termin „lokalno računanje“ ukazuje na strategiju podeli pa vladaj, koja razlaže kompleksan problem na više manjih problema i rešava ih. Verifikaciona matrica niske gustine pomaže ovim algoritmima na više načina. Prvo, sadrži lokalne proračune, smanjuje kompleksnost kombinovanjem više manjih problema, a time smanjuje i broj potrebnih poruka za razmenu svih informacija.

Takođe, primećeno je da iterativni algoritmi za dekodovanje kodova niske gustine postižu optimalne performanse kod dekodera.

5.5.4. LDPC ALGORITMI DEKODOVANJA

U ovom delu diskutovaćemo o dva algoritma za dekodovanje LDPC kodova. Naziv prvog algoritma je *hard-decision*, a drugog *soft-decision* algoritam dekodovanja. Zadržaćemo originalne nazive zbog istoimenih argumenata u funkcijama Octave programskog okruženja.

Hard-decision dekodovanje

Algoritam će biti objašnjen na osnovu primera koda koji je uveden na početku poglavlja o LDPC kodovima. Na primer, originalna kodna reč je $C = [1\ 0\ 0\ 1\ 0\ 1\ 0\ 1]$. Pretpostavimo da koristimo binarni simetrični kanal i da primljena kodna reč sadrži jednu grešku na poziciji c_1 gde je nula promenjena u jedinicu.

U prvom koraku svi promenljivi čvorovi c_i šalju poruku do njihovih (uvek dva u našem primeru) kontrolnih čvorova f_i , ta poruka sadrži jedan bit za koji veruju da je ispravan. U ovoj fazi postoji samo informacija o čvorovima c_i koja odgovara primljenom i -tom bitu od c , y_i . Na primer, to znači da c_0 šalje poruku koja sadrži 1, u čvorove f_1 i f_3 i tako dalje.

Tabela 5.1 Pregled poruka primljenih i poslatih od c čvorova u drugom koraku, poruke koje su prošle kroz algoritam.

c čvor	primljeno/poslato
f_0	Primljeno: $c_1 \rightarrow 1\ c_3 \rightarrow 1\ c_4 \rightarrow 0\ c_7 \rightarrow 1$ Poslato: $0 \rightarrow c_1\ 0 \rightarrow c_3\ 1 \rightarrow c_4\ 0 \rightarrow c_7$
f_1	Primljeno: $c_0 \rightarrow 1\ c_1 \rightarrow 1\ c_2 \rightarrow 0\ c_5 \rightarrow 1$ Poslato: $0 \rightarrow c_0\ 0 \rightarrow c_1\ 1 \rightarrow c_2\ 0 \rightarrow c_5$
f_2	Primljeno: $c_2 \rightarrow 0\ c_5 \rightarrow 1\ c_6 \rightarrow 0\ c_7 \rightarrow 1$ Poslato: $0 \rightarrow c_2\ 1 \rightarrow c_5\ 0 \rightarrow c_6\ 1 \rightarrow c_7$
f_3	Primljeno: $c_0 \rightarrow 1\ c_3 \rightarrow 1\ c_4 \rightarrow 0\ c_6 \rightarrow 0$ Poslato: $1 \rightarrow c_0\ 1 \rightarrow c_3\ 0 \rightarrow c_4\ 0 \rightarrow c_6$

U drugom koraku svaki kontrolni čvor f_j računa odgovor za svaki povezani promenljivi čvor. Poruka odgovor sadrži bit za koji f_j veruje da je ispravan za čvor c_i pretpostavljajući da su preostali promenljivi čvorovi povezani na f_j ispravni. Drugim rečima, ako pogledate primer, svaki kontrolni čvor f_j je povezan sa 4 promenljiva čvora. Dakle, kontrolni čvor na osnovu tri primljene poruke od

promenljivih čvorova izračunava bit četvrtog promenljivog čvora koji treba da ispuni jednačinu pariteta. U tabeli (Tabela 5.1) dat je pregled ovog koraka.

Važno je reći, da je ovo tačka u kojoj algoritam za dekodovanje završava. To će biti slučaj kada su ispunjene sve jednačine pariteta. Kasnije ćemo videti da ceo algoritam sadrži petlju.

U narednoj fazi promenljivi čvorovi dobijaju poruke od kontrolnih čvorova i koriste ovu dodatnu informaciju da odluče da li je originalno primljeni bit ispravan. Jednostavan način za to je većinsko glasanje. Ukoliko bolje pogledamo ovaj primer, vidimo da svaki promenljivi čvor ima tri izvora informacija u vezi sa bitom. Originalni primljeni bit i dve sugestije od kontrolnih čvorova. Tabela 5.2 ilustruje ovaj korak. Sada promenljivi čvorovi mogu poslati još jednu poruku sa svojim *hard-decision* metodom za ispravnu vrednost na kontrolnim čvorovima.

Sada se vratimo na drugi korak. U ovom primeru, drugo izvršavanje koraka 2 će prekinuti proces dekodovanja otkada je c_1 glasalo za 0 u poslednjem koraku. Ovo ispravlja grešku u prenosu i sve jednačine provere su zadovoljene.

Tabela 5.2 Promenljivi čvor koristi poruke kontrolnih čvorova po sistemu većinskog glasa za ispravan bit.

promenljivi čvor	y_i primljeno	poruke kontrolnih čvorova	odluka
c_0	1	$f_1 \rightarrow 0$ $f_3 \rightarrow 1$	1
c_1	1	$f_0 \rightarrow 0$ $f_1 \rightarrow 0$	0
c_2	0	$f_1 \rightarrow 1$ $f_2 \rightarrow 0$	0
c_3	1	$f_0 \rightarrow 0$ $f_3 \rightarrow 1$	1
c_4	0	$f_0 \rightarrow 1$ $f_3 \rightarrow 0$	0
c_5	1	$f_1 \rightarrow 0$ $f_2 \rightarrow 1$	1
c_6	0	$f_2 \rightarrow 0$ $f_3 \rightarrow 0$	0
c_7	1	$f_0 \rightarrow 1$ $f_2 \rightarrow 1$	1

Soft-decision dekodovanje

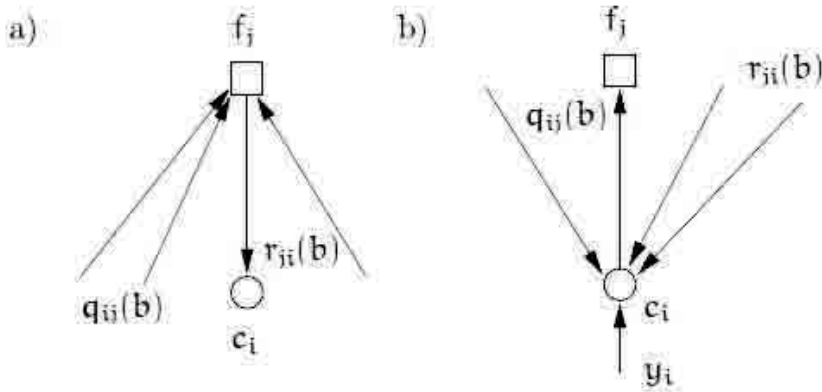
Opis metode *hard-decision* u fazi dekodovanja LDPC koda ima za svrhu da predstavi samu ideju o LDPC kodovima i može da posluži kao dobar školski primer. Metoda dekodovanja *soft-decision* predstavlja poseban dizajn LDPC dekodera koji primaju sekvencu verovatnoća, umesto binarne sekvence koja ukazuje verovatnoću pozicije (od 0 do 1). Ovaj metod doprinosi postizanju boljih performansi dekodovanja i iz tog razloga je više upotrebljavani metod za

dekodovanje LDPC koda. Osnovna ideja ovog dekodera je potpuno ista kao i kod *hard-decision* dekodera. Pre nego se upoznamo sa ovim algoritmom, predstavimo notaciju koju ćemo na dalje koristiti.

- $P_i = P_r(c_i = 1|y_i)$
- q_{ij} je poruka poslata od promenljivog čvora c_i do kontrolnog čvora f_j . Svaka poruka sadrži uvek par $q_{ij}(0)$ i $q_{ij}(1)$. Na osnovu ovih vrednosti određuje se vrednost y_i , da li je 1 ili 0.
- r_{ij} je poruka poslata od kontrolnog čvora f_i do promenljivog čvora c_i .

Broj koraka u fazi dekodovanja odgovara *hard-decision* algoritmu.

Svi promenljivi čvorovi šalju svoje q_{ij} poruke. Budući da nema dostupnih drugih informacija u ovom koraku, $q_{ij}(1) = P_i$ i $q_{ij}(0) = 1 - P_i$.



Slika 5.11 a) ilustruje računanje $r_{ji}(b)$ i b) $q_{ij}(b)$

Kontrolni čvorovi izračunavaju njihove odgovor poruke r_{ji} :

$$r_{ij}(0) = \frac{1}{2} + \frac{1}{2} \prod_{i' \in v_j} (1 - 2q_{i'j}(1))$$

i

$$r_{ij}(1) = 1 - r_{ij}(0)$$

Ova verovatnoća je jednaka verovatnoći $r_{ij}(0)$ da je $c_i = 0$. Ovaj korak i korišćena informacija za izračunavanje odgovora prikazan je na slici (Slika 5.11).

Promenljivi čvorovi ažuriraju njihove poruke (odgovore) prema kontrolnim čvorovima. Sve je urađeno preko sledećih jednačina,

$$q_{ij}(0) = K_{ij}(1 - P_i) \prod_{j' \in c_i \setminus j} r_{ji}(0)$$

$$q_{ij}(1) = K_{ij}P_i \prod_{j' \in c_i \setminus j} r_{ji}(1)$$

pri čemu je konstanta K_{ij} izabrana na način da se osigura $q_{ij}(0) + q_{ij}(1) = 1$. $C_i \setminus j$ sada znači sve kontrolne čvorove osim f_j . Na slici (Slika 5.11) prikazano je izračunavanje u ovom koraku.

U ovom trenutku, promenljivi čvorovi ažuriraju svoje trenutne procene $\hat{c}_i$ od njihove promenljive c_i . Ovo je urađeno izračunavanjem verovatnoća za 0 i 1 glasanjem za veću. Korišćene jednačine:

$$Q_i(0) = K_i(1 - P_i) \prod_{j \in c_i} r_{ji}(0)$$

i

$$Q_i(1) = K_iP_i \prod_{j \in c_i} r_{ji}(1)$$

su prilično slične onima za izračunavanje $q_{ij}(b)$ ali se sada koristi informacija iz svih kontrolnih čvorova.

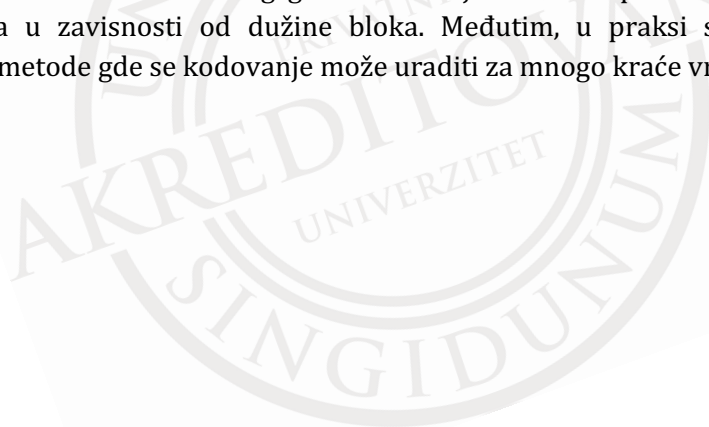
$$\hat{c}_i = \begin{cases} 1 & \text{ako je } Q_i(1) > Q_i(0), \\ 0 & \text{drugo} \end{cases}$$

Ukoliko trenutna kodna reč ispunjava jednačinu za proveru parnosti algoritam se završava. U suprotnom, završetak je obezbeđen preko maksimalnog broja iteracija.

Soft-decision algoritam dekodovanja LDPC koda je jednostavna varijanta dekodera pogodna za BSC kanal i postoji mogućnost za poboljšanje performansi. Pored problema sa performansama postoje numerički problemi stabilnosti zbog multipliciranja verovatnoća.

5.5.5. LDPC ALGORITAM KODOVANJA

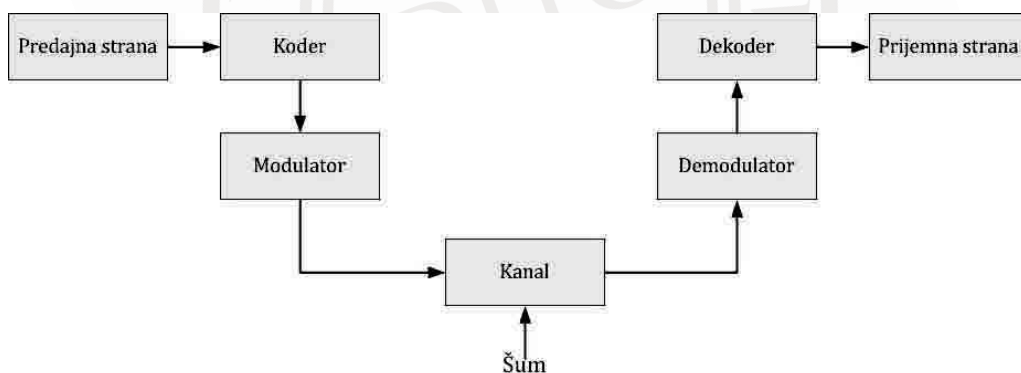
Posle objašnjenja algoritama dekodovanja, vidimo da algoritmi samo ispravljaju greške u kodnim rečima. To bi bilo dovoljno za tradicionalni sistemski blok kod, jer se kodna reč sastojala od bitova poruke i kontrolnih (paritetnih) bitova. Do sada nigde nismo pomenuli da je moguće da se direktno vidi originalna poruka (bitovi poruke) u kodnoj reči. Kodovanje kod LDPC koda je urađeno tako da se izaberu određeni promenljivi čvorovi na čije pozicije se postavljaju bitovi poruke. U drugom koraku se izračunavaju kontrolni bitovi koji se postavljaju na pozicije kontrolnih čvorova Tanerovog grafa. Na ovaj način kompleksnost će da bude kvadratna u zavisnosti od dužine bloka. Međutim, u praksi se koriste više pametne metode gde se kodovanje može uraditi za mnogo kraće vreme.



5.6. PRAKTIČNA PRIMENA ZAŠTITNIH KODOVA

Na slici (Slika 5.12) prikazana je apstraktna šema digitalnog komunikacionog sistema sa implementiranim komponentama za zaštitno kodovanje. Na strani pošiljaoca upotrebljen je koder, dok na strani primaoca nalazi se dekodek. Navešćemo sada neke popularne tehnike kodovanja.

1. ARQ (*Automatic Repeat Request*) – automatsko izdavanje zahteva za ponavljanjem poruke;
2. FEC (*Forward Error Correction*) – dekodovanje primljene sekvence kao najbliže kodne reči;
3. Hibridni ARQ;
4. Interliving;
5. Dekodovanje sa brisanjem;
6. Konkatencija kodova;



Slika 5.12 Digitalni komunikacioni sistem

5.6.1. ARQ

Zaštitno kodovanje se obavlja tako da kod sam po sebi ne ispravlja već samo detektuje greške. Kada se dogodi detekcija greške, šalje se zahtev za ponavljanjem poruke.

U pogledu greške dekodovanja, ARQ je bolji od FEC, budući da se primljene kodne reči prihvataju samo ako su prenete bez greške (dakle greška dekodovanja je nula), pod uslovom da detekcija grešaka se uvek obavlja ispravno. Ovo se ne dobija besplatno. Cena je plaćena snižavanjem brzine prenosa. Dodatna prednost ARQ je daleko jednostavnije dekodovanje u poređenju sa FEC.

ARQ zahteva dodatni povratni kanal, koji zavisno od primene i scenarija komuniciranja nije uvek na raspolaganju.

U pogledu ometanja, ARQ je osetljiv, budući da ometanje može biti tako dizajnirano da se uvek dešava po neka greška u svakoj kodnoj reči. Optimalna situacija sa tačke gledišta ometača je da neprekidno forsira jednu grešku u jednoj kodnoj reči, čime bi se ta kodna reč neprekidno reemitovala.

Postoje dva tipa ARQ:

1. Stani i čekaj
2. Neprekidni

Stani i čekaj ARQ. Predajnik šalje jednu kodnu reč i čeka pozitivnu potvrdu ispravnog prijama (ACK) ili negativnu potvrdu (NAK), pre nego što pošalje narednu kodnu reč. Dobra strana stani i čekaj ARQ je potreba za poludupleksnim kanalom. Negativna strana stani i čekaj ARQ je trošenje vremena na čekanje potvrde, što snižava ekvivalentnu brzinu prenosa.

Neprekidni ARQ zahteva dupleksni kanal, tako što se kodne reči po jednom kanalu neprekidno šalju sve dok se po drugom kanalu ne primi NAK. Postoje dva načina obrade NAK-a:

ARQ sa slanjem unazad - N kada se ponovo šalje kodna reč u kojoj je detektovana greška plus sve kodne reči unazad. Parametar N se određuje na osnovu ukupnog kašnjenja kanala u oba smera.

Na primer, kod geostacionarnih satelitskih kanala, kašnjenje može biti vrlo veliko, do 540 msec. Prijemnik mora da memoriše N kodnih reči primljenih u okviru ovog vremena, što zahteva veliku memoriju.

Selektivni ARQ - kod koga se prenosi ponovo samo kodna reč u kojoj je detektovana greška, čime se dobija veća brzina prenosa nego u slučaju ARQ sa slanjem unazad – N .

Oba tipa neprekidnog ARQ daje veću brzinu prenosa u odnosu na stani i čekaj ARQ, na račun većih memorijskih zahteva.

5.6.2. FEC

FEC je pogodan za primene u kojima korisnik mora dobiti tačnu poruku iz prvog slanja. Primer su jednosmerni i difuzioni (engl. *Broadcast*) kanali. Današnji kodovi za ispravljanje grešaka pripadaju dvema osnovnim kategorijama:

1. blok kodovima,
2. konvolucionim kodovima.

Najpopularniji blok kodovi su Rid Solomonovi kodovi - RS, koji rade na nivou m -bitnih simbola, a ne na nivou bita. Budući da RS kodovi ispravljaju greške na nivou simbola, a ne bita, pogodni su za ispravljanje koncentrisanih, paketnih (engl. *burst*) grešaka.

Na primer: 2 – simbolni RS kodovi sa 8 bitova po simbolu, mogu korigovati sve pakete grešaka od 16 bitova ili manje. Danas se koriste u JTIDS – novom standardu za duboki kosmos i za kompakt disk - CD plejere.

Konvolucionni kodovi uvode ulazni niz bita poruke u pomerački registar dužine r . Snaga ovih kodova zavisi od veličine r . Veće r daje efikasnije kodove, ali to znatno usložnjava dekodovanje.

Postoji više metoda dekodovanja konvolucionih kodova, ali je danas najrasprostranjeniji Viterbijev algoritam, otkriven 1967 godine od strane Andrew Viterbija. Danas je raspoloživ u jednom čipu izrađenom VLSI tehnologijom.

Viterbijev dekodler je neefikasan i nepraktičan za veliko K . Za veliko r , ($r < 9$) češće se koristi drugi algoritam dekodovanja koji se naziva sekvencijalni dekodler.

Treći postupak dekodovanja je dekodier sa povratnom spregom, pogodan za kanale sa paketnim greškama, dok je znatno slabijih performansi ako su greške na kanalu slučajno raspodeljene.

Generalno se može reći da konvolucionni kodovi daju bolje rezultate od blok kodova jednake kompleksnosti kodovanja i dekodovanja. Problem sa svim kodovima koje smo do sada razmatrali je potreba da se proširi propusni opseg za onoliko koliko zahtevaju dodati biti parnosti (redundantni biti) ako želimo da zadržimo brzinu prenosa nekodovanih poruka istom.

1976. godine Gotfrjd Ungerbok (*Gottfried Ungerboeck*) je otkrio klasu kodova koji integrišu funkciju kodovanja i modulisanja, pri čemu otpada zahtev za proširenjem frekventnog opsega prenosa. Ovi kodovi su nazvani Ungerbokovi kodovi ili *trelis* kodovana modulacija (TCM). Bukvalno svi modemi koji rade na telefonskim linijama na brzinama većim od 9.6 kb/sec danas koriste TCM.

5.6.3. HIBRIDNI ARQ

Hibridni ARQ kombinuju kodove za detekciju i kodove za ispravljanje grešaka, čime se povećava efikasnost korišćenja kanala. U prijemniku se prvo pokušavaju ispraviti sve greške u primljenoj kodnoj reči. Ukoliko se u tome ne uspe, koristi se jedna od pomenute tri ARQ tehnike za retransmisiju.

Hibridni ARG I vrste šalje sve neophodne bitove parnosti za detekciju i ispravljanje grešaka.

Hibridni ARQ II vrste šalju samo bitove parnosti neophodne za detekciju grešaka i čuvaju u rezervi bitove parnosti za ispravljanje grešaka. Kada prijemnik detektuje grešku, zahteva slanje bitova za korekciju grešaka i na osnovu njih pokušava da ispravi greške pre nego što pošalje zahtev za retransmisiju celokupne kodne reči. Hibridni ARQ II vrste je vrlo efikasan za kanale koji se karakterišu dobrim stanjima koja su većinska u odnosu na ređa loša stanja kanala.

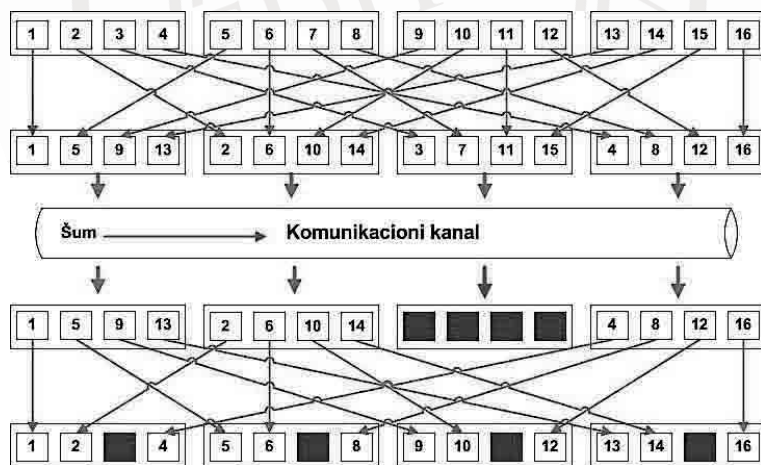
5.6.4. INTERLIVERI

Jedan od najpopularnijih načina da se koriguju paketske greške je korišćenje kodova koji dobro rade na slučajnim greškama, tako što se prethodno kroz postupak interlivinga paketske greške rasprše tako da izgledaju kao slučajno raspodeljene.

Postoje dva tipa interlivinga koji se danas najviše koriste:

1. Blok interliving.
2. Konvolucionni interliving.

Proces interlivinga možemo opisati na sledeći način. Na predajnoj strani posle funkcije kodovanja, nad kodnim rečima primenjujemo funkciju interlivinga. Zatim poruku šaljemo kroz komunikacioni kanal u kom postoji verovatnoća za nastanak paketskih grešaka. Na kraju, na strani primaoca primenjuje se inverzna funkcija za deinterliving i dobijena poruka se prosleđuje funkciji za dekodovanje koja može da ispravi sve slučajno raspoređene greške. Slučajna raspodela grešaka postiže se primenom interlivinga.



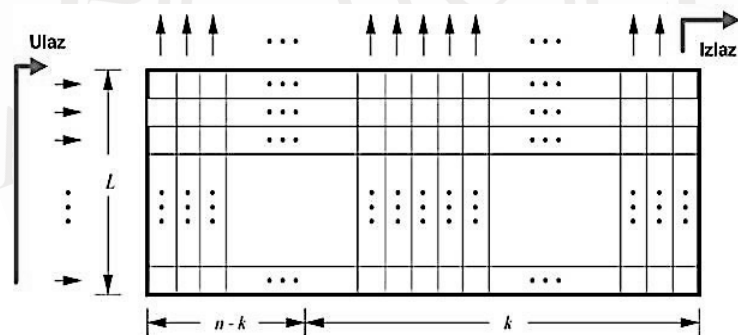
Slika 5.13 Jedan primer blokovskog interlivinga

Blokovski interliving - se popunjava sa L kodnih reči složenih po redovima. Svaka kodna reč je dužine n (Slika 5.14). Ovih L kodnih reči se zatim šalje po kolonama,

sve dok se celokupni interliver ne isprazni. Nakon toga se interliver ponovo puni i započinje novi ciklus.

Na prijemu se kodne reči deinterlivinguju pre dekodovanja. Paketne greške dužine L ili manje bita ne izazivaju više od 1 bita greške u bilo kojoj kodnoj reči. Dekoder koji dobro radi na slučajnim greškama će sa velikom verovatnoćom ispraviti po jednu grešku u svakoj kodnoj reči umesto celokupan paket grešaka.

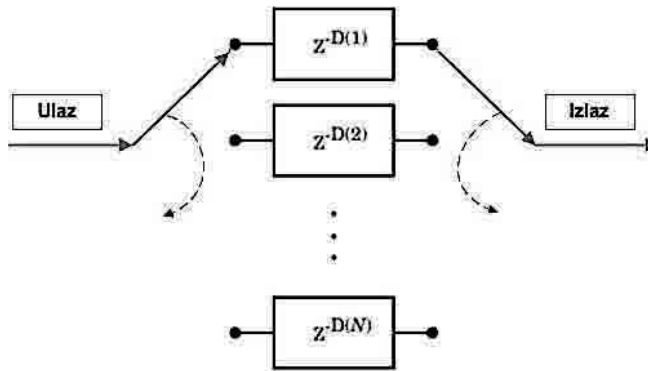
Parametar L se naziva stepen interlivera ili dubina interlivinga. L se bira prema najgorim uslovima prenosa datog kanala. Mora biti veći od najvećeg očekivanog paketa grešaka u datom kanalu. Osnovni nedostatak interlivinga je kašnjenje koje se unosi popunjavanjem redova. Konvolucionni interliving eliminiše ovaj problem, osim kašnjenja vezanog za početno punjenje. Ovim se redukuju memorijski zahtevi na polovinu u odnosu na blokovske interlivere. Zajednički nedostatak oba tipa interlivera je kašnjenje koje u nekim kanalima može iznositi i nekoliko sekundi, što može biti neprihvatljivo za neke aplikacije (recimo prenos govora).



Slika 5.14 Šema blokovskog interlivinga

Konvolucionni interliving - se sastoji od skupa pomeračkih registara, svaki sa ustaljenim kašnjenjem. Svaki novi simbol iz ulaznog vektora napaja susedni prenosni registar, a „najstariji” simbol u tom registru postaje deo izlaznog vektora. Konvolucionni interliver ima memoriju čija operacija ne zavisi samo od trenutnog simbola, već i od prethodnog. Šema (Slika 5.15), prikazuje strukturu konvolucionnog interlivera, pokazujući skup smena registra i njihove vrednosti

kašnjenja $D(1), D(2), \dots, D(N)$. KTH registar drži $D(k)$ simbol, gde je $k = 1, 2, \dots, N$.



Slika 5.15 Šema konvolucionog interlivera

Pseudoslučajni interliveri – u ovoj tehnici interlivinga ideja je da se eliminišu regularni obrasci ili određene statističke osobine iz određene sekvence podataka i to tokom veoma dugog perioda. Pseudoslučajni interliver se često koristi kod turbo kodova. Uniformni slučajni interliver je stvarna apstrakcija nad statističkim verovatnoćama za neki interliver sa ogromnom periodom L . Ovakvi interliveri pokušavaju da uvedu element slučajnosti u neki dizajn koda bez povećanja kompleksnosti, pretpostavljajući korišćenje iterativnog dekodovanja dva interlivovana koda. U praksi pseudoslučajni interliveri su predvidljivi i invertibilni. Ponašaju se kao PRNG generatori koji imaju određenu periodu za svako izabrano unutrašnje stanje generatora. Perioda generatora određena je sa polinomom. Ukoliko su podaci matrice sa više redova, funkcija procesira pojedinačno kolone matrice.

5.6.5. DEKODOVANJE SA BRISANJEM

Kada prijemnik radi u uslovima ometanja, fedinga ili nekih drugih tranzijentnih smetnji, može preći u režim rada po kome može doneti odluku da je neki bit ili simbol obrisao.

Na primer, prijemnik može da obriše simbole A i L iz poruke „SIGNAL“, čime se dobija „SIGN--“. Ovo nije isto kao „SIGN“, u kome su prosto odbačena dva

poslednja simbola. Pošto se lokacija izbrisanih bita zna, po pravilu dekodovanje sa brisanjem zahteva manje bita za proveru parnosti od kodova za ispravljanje grešaka.

Kod čije je minimalno rastojanje d_{min} , može da koriguje e brisanja, ako je:

$$d_{min} = e + 1.$$

Podsetimo se da kodovi za ispravljanje grešaka mogu da isprave t grešaka i e brisanja, dogod važi:

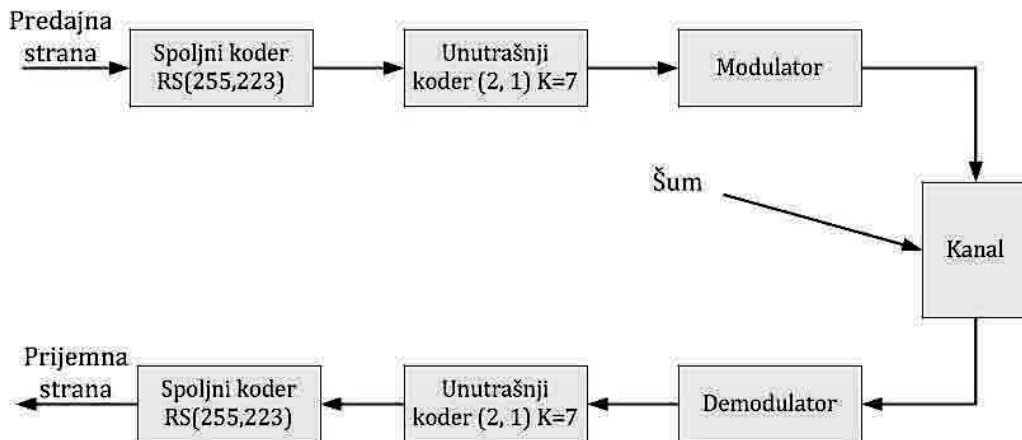
$$d_{min} \geq 2t + e + 1.$$

Na primer, jedan kod za ispravljanje grešaka koji ima minimalno rastojanje 7, može da koriguje 2 greške i 2 brisanja.

5.6.6. KONKATENACIJA

Konkatenacija je kombinovanje dva koda u serijskoj vezi. Po pravilu se koristi na onim kanalima na kojima jedan kod ne može da koriguje sve greške koje se mogu pojaviti.

Ilustrujmo to jednim primerom, videti (Slika 5.16). Unutrašnji kod brzine $\frac{1}{2}$, $K = 7$, je konvolucionni kod sa Viterbijevim dekodovanjem, kojim se koriguje većina slučajnih grešaka. Spoljašnji kod je Rid Solomonov kod sa 16 simbola, kojim se ispravljaju sve paketske greške koje je propustio Viterbijev dekodier.



Slika 5.16 Konkatenacija kodova

RS kod radi sa 8 bitnim simbolima i stoga je veoma efikasan u ispravljanju paketskih grešaka. Ukupna brzina konkateniranog koda jednaka je proizvodu brzina sastavnih kodova, odnosno u ovom primeru $(1/2)(223/255) \cong 0.44$.

Kodni dobitak konkateniranih kodova je veoma velik. Neki od njih rade bukvalno na nivou termalnih šumova i tada se pojavljuje problem sinhronizacije.

I pored ovih dobrih svojstava, konkatenirani kodovi se teško implementiraju zbog njihove složenosti i male brzine, što je veliki problem u kanalima sa ograničenim propusnim opsegom.

5.6.7. JEDAN PRIMER KODOVANJA U CD

CD medijum je 120 mm plastični disk, na kome se digitalni audio podaci memorišu pomoću jamica (pits) koje očitava laserki snop. AD i DA konverzija se obavlja na 44.1 Ksample/sec, čime se obezbeđuje audio opseg od 20 kHz. Svaki audio odbirak se koduje uniformno sa 2^{16} nivoa kvantizacije (običan PCM ima 2^8 nivoa). Rezultujuća dinamika je 96 dB, a ukupne harmonijske distorzije su 0.005%.

Greške nastaju iz dva izvora:

1. Prašina, mehurići vazduha, nesavršenosti utiskivanja jamica.

2. Otisci prstiju, ogrebotine, nataložena prašina.

Sony je bio idealan partner, budući da je već imao iskustvo sa kodovima za ispravljanje grešaka. Filipsovi stručnjaci su pokušali da koriste uobičajene kodove, ali rezultat je bio još gori, budući da su greške pri očitavanju CD-a po svojoj prirodi paketske.

Nakon godinu dana saradnje ustanovljena je konačna šema kodovanja koja se sastoji od višenivovske primene interlivinga i Rid Solomonovih kodova. Ova šema je poznata pod nazivom CIRC - *Cross Interleave Reed Solomon Code*. Sastoji se od 4 nivoa kontrole grešaka:

1. Dekoder obezbeđuje nivo FEC.
2. Ukoliko paketne greške prevazilaze mogućnosti FEC, dekodier postavlja nivo ispravljanja sa brisanjem.
3. Ako greške prevazilaze nivo ispravljanja sa brisanjem, dekodier sakriva dati odbirak i zamenjuje ga interpolirani odbirkom između pouzdanih susednih odbiraka.
4. Ukoliko greške prevazilaze mogućnosti interpolacione korekcije, dekodier se utišava ili daje posebnu signalizaciju.

Rezultantna performansa CIRC je data u sledećoj tabeli.

Performanse kontrole greške CIRC-a	
Maksimalno greške	ispravljanje ~ 4 000 bitova (2.5 mm na disku)
Maksimalna greške	interpolacija ~ 12.000 bitova (8 mm)
Nedetektovani uzorci greške	Manje od jedne na svakih 750 časova (10^{-3})
Novi diskovi imaju tipičan stopu greške (engl. BER)	$\sim 10^{-4}$

CIRC koder koristi tri nivoa interlivinga i dva konkatenerana Rid Solomonova koda. Unutrašnji RS je $RS(32,28)$ kod, dok je spoljašnji $RS(28,24)$ kod. Oba koda koriste 8 – bitne simbole, a ukupna brzina koda je $\frac{3}{4}$.

Unutrašnji kod ima minimalno rastojanje $d_{\min} = 5$, tako da može da koriguje najviše dva pogrešna simbola. Spoljašnji kod ima takodje minimalno rastojanje $d_{\min} = 5$, i služi ili za ispravljanje grešaka ili ispravljanje brisanja, zavisno od toga koliko je unutrašnji kod bio uspešan u korigovanju paketnih grešaka.

CIRC šema kodovanja je postala ekstremno uspešna. Ona je konvertovala kanal koji nije tolerisao trunku prašine u vrlo robustan praktičan sistem CD audio nosača koji je osvojio svet. Današnji CD može da toleriše 8 mm velike rupe probušene kroz disk, bez audio perceptivnih efekata.



5.7. LABORATORIJSKI RAD

VEŽBA 7.

Hemingov zaštitni kod (12,8), uvodna napomena:

Hemingov kod se najčešće koristi za ispravljanje grešaka u RAM memorijama. Poruka koja se prenosi je dužine 12 bita pri čemu je 8 bita rezervisano za podatke a preostala 4 bita su kontrolna i izračunavaju se na osnovu bitova podataka. Kontrolni bitovi se nalaze na prvoj, drugoj, četvrtoj i osmoj poziciji.

Hemingov kod za ispravljanje 1 bita greške implementira se na sledeći način:

1. Biti poruke su pobrojani sa leva na desno počevši od 1.
2. Svaki bit koji predstavlja stepen 2 je bit provere (1, 2, 4, 8).
3. Ostali bitovi (3, 5, 6, 7, 9) su biti podatka.

Svaki bit provere uspostavlja neparnu parnost sa grupom bita podatka. Prvi bit provere selektuje binarne vrednosti sa bitom 1 na prvom mestu. Drugi bit provere selektuje binarne vrednosti podataka sa bitom 1 na drugom mestu itd.

Pozicija bita	Broj pozicije	Bit za proveru	Bit podatka
12	1 1 0 0	M8	
11	1 0 1 1	M7	
10	1 0 1 0	M6	
9	1 0 0 1	M5	
8	1 0 0 0	C4	
7	0 1 1 1	M4	
6	0 1 1 0	M3	
5	0 1 0 1	M2	
4	0 1 0 0	C3	
3	0 0 1 1	M1	
2	0 0 1 0	C2	
1	0 0 0 1	C1	

Slika 5.17 Bitovi podatka i kontrolni bitovi

$$C1 = M1 \oplus M2 \oplus M4 \oplus M5 \oplus M7$$

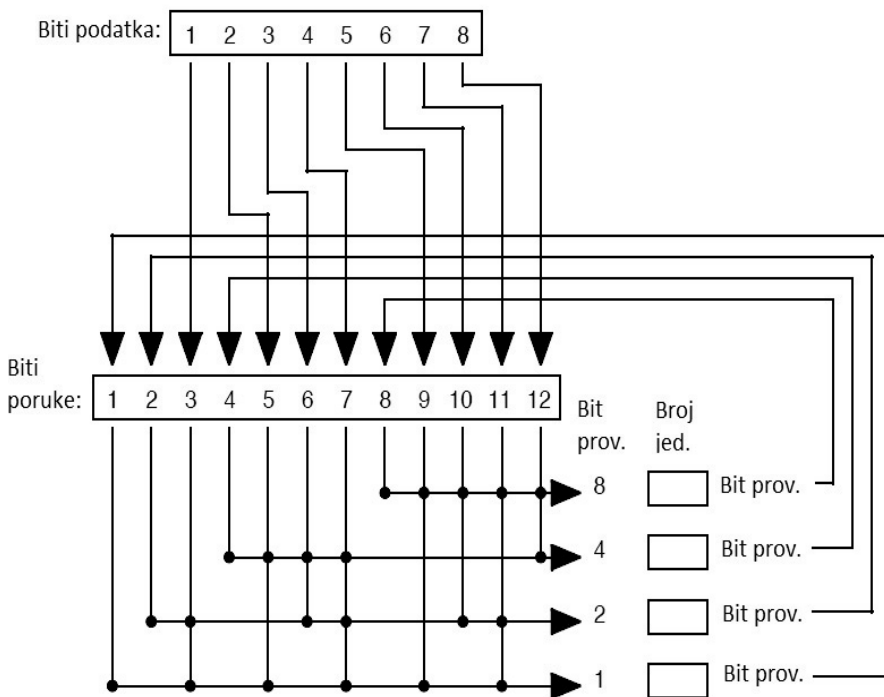
$$C2 = M1 \oplus M3 \oplus M4 \oplus M6 \oplus M7$$

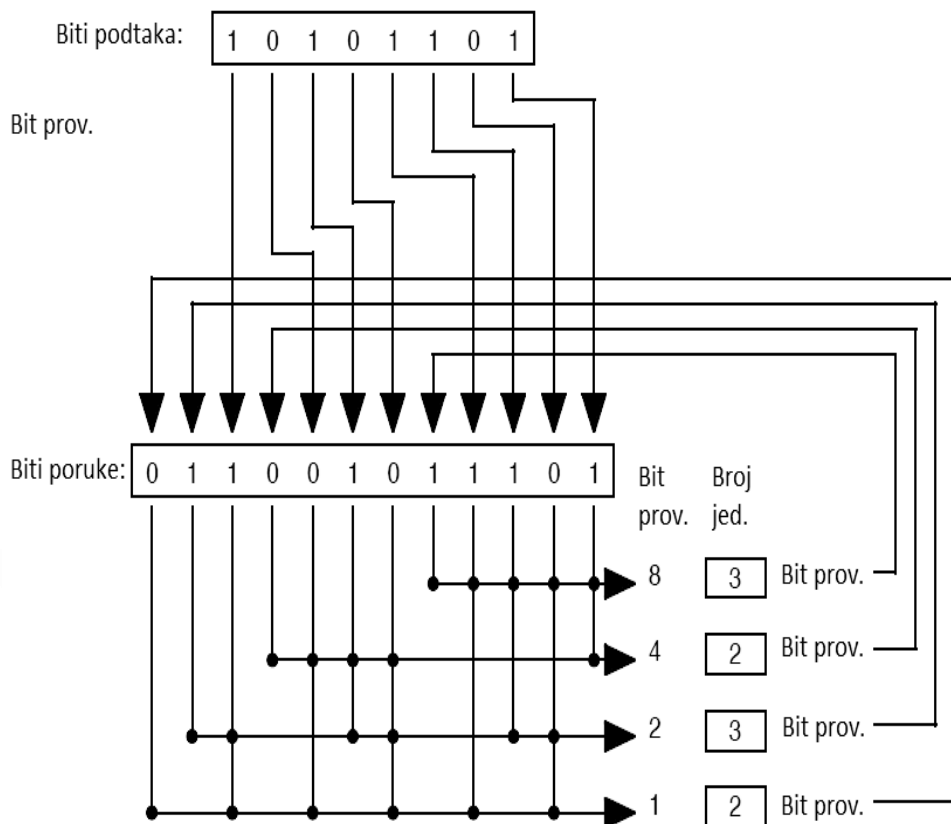
$$C3 = M2 \oplus M3 \oplus M4 \oplus M8$$

$$C4 = M5 \oplus M6 \oplus M7 \oplus M8$$

Koder (12, 8):

Pre prenosa podataka neparna parnost je uspostavljena nad svakim bitom provere i njegovom asocijativnom vezom sa podacima.



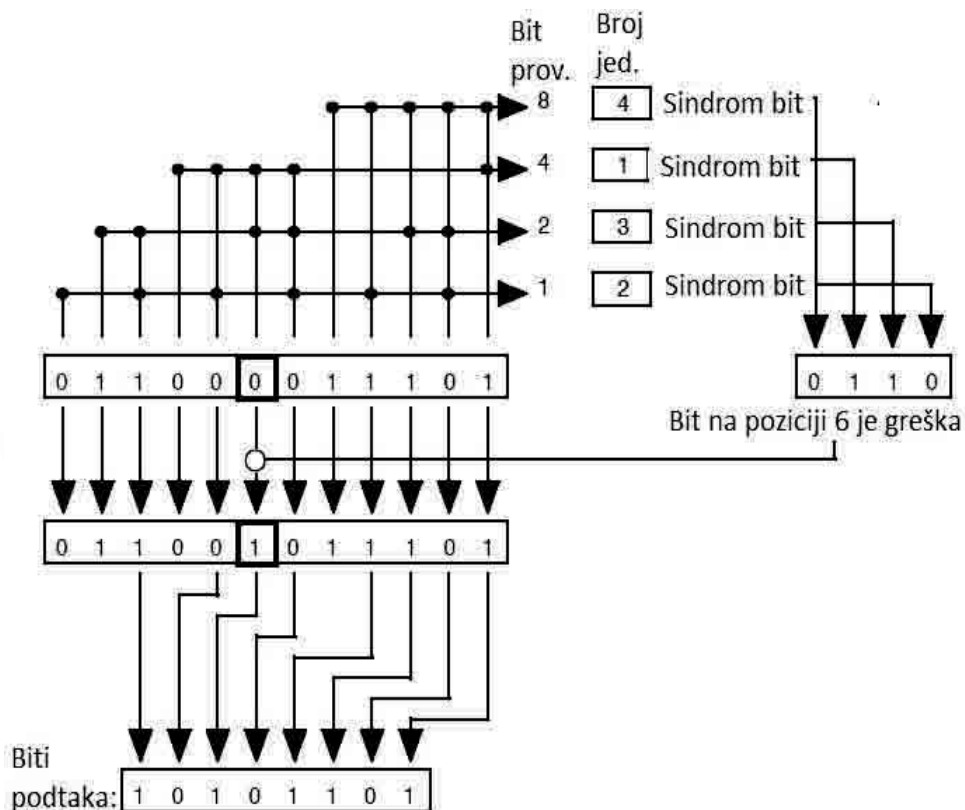


Slika 5.18 Šema koda za Hemingov kod (12,8)

Dekoder (12, 8):

Nakon prenosa poruke izračunavamo bit parnost (engl. *parity bit*) za svaki skup podataka po pripadnosti prema bitu provere (engl. *check bit*). Ukoliko je sindrom vrednosti 0, nije bilo grešaka a ukoliko je 1, postoji greška.

U ovom primeru promenićemo vrednost bita na poziciji 6.



Slika 5.19 Šema dekodera za Hemingov kod (12,8)

Proširenja i ograničenja Hemingovog koda (12, 8)

Sindrom vrednosti	Parnost poruke	Rezultat
nulta	neparno	nema greške
nenulta	parno	korekcija 1 bit
nenulta	neparno	Nema korekcije

VEŽBA 8.

Hemingov binarni zaštitni kod (7,4), uvodna napomena:

Hemingov kod se označava sa $(n\ k)$, pri čemu broj k predstavlja informacione bitove, a n ukupan broj bitova u kodnoj reči. U izrazu $n = m + k$, broj m označava broj zaštitnih bita. Ukoliko je potrebno ispraviti u kanalu pojedinačne greške, broj zaštitnih bitova se računa po formuli:

$$2^m \geq k + m + 1.$$

Svakako, za mogućnost ispravljanja većeg broja bitova, broj zaštitnih bitova mora da se povećava. U nastavku razradićemo funkciju Hemingovog koda (7,4) za kodovanje i dekodovanje. Broj zaštitnih bita, $m = 3$. Ukoliko se proveri navedena nejednakost:

$$2^3 \geq 4 + 3 + 1,$$

vidi se da je ispunjen uslov za ispravljanje samo jedne greške.

Hemingov kod (7,4), na reč dužine 4 bita, dodaje 3 bita i formira kodnu reč dužine 7 bitova. Zaštitna tri bita će prilikom dekodovanja odrediti položaj jedne greške preko sindroma, na taj način dekodler Hemingovog koda ispraviće grešku i odrediti poslatu sekvencu.

U prvom koraku generisaćemo verifikacionu matricu H . Matricu genrišemo tako što po kolonama upisujemo binarne vrednosti brojeva iz opsega od 1 do 7. Na primer, u prvu kolonu je upisana binaran vrednost 001 koja decimalno predstavlja broj 1.

H	0	0	0	1	1	1	1
	0	1	1	0	0	1	1
	1	0	1	0	1	0	1

U drugom koraku prelazimo na funkciju kodovanja. Proces kodovanja predstavljamo sa izrazom $Z = U \cdot G$, gde U predstavlja poruku, G generator

matricu, a Z kodovanu poruku. Za funkciju kodovanja neophodno je generisati generator maticu G .

Generator matica prikazana je u tabeli ispod. Osenčeni deo tabele dobijen je tako što smo invertovali vrednosti verifikacione matrice H . Zatim poslednji red generator matrice dobijen je preko funkcije za neparnu par ($001 = 1, 010 = 1, 011 = 0, 100 = 1, 101 = 0, 110 = 0, 111 = 1$)nost za svaku kolonu verifikacione matrice.

Poruka koju sada treba da kodujemo je na primer poruka $U = 1000$. Primer množenja matrice i vektora dat je u tabeli koja sledi.

G							$\times$	U	$=$								$=$	Z
1	1	1	0	0	0	0		1		1	1	1	0	0	0	0		1110000
1	0	0	1	1	0	0		0		0	0	0	0	0				
0	1	0	1	0	1	0		0		0	0	0	0	0				
1	1	0	1	0	0	1		0		0	0	0	0	0				

U trećem koraku na redu je funkcija dekodovanja ili strana dekodera. Dekodovanje se obavlja jednostavnu, po principu određivanja sindrom reči na osnovu koje se utvrđuje da li postoji greška ili ne postoji. Ukoliko je rezultat množenja $S = Z \cdot H$ nula kodna reč, tada je kodovana poruka prenesena bez greške ili ukoliko je različita od 0 npr. 3, to će značiti da je greška na trećem bitu. Nakon promene vrednosti bita na trećoj poziciji prelazimo u fazu dekodovanja.

H							$=$								$=$	S		
0	0	0	1	1	1	1		0	0	0	0	0	0	0		0	0	
0	1	1	0	0	1	1		0	1	1	0	0	0	0		2	=	0
1	0	1	0	1	0	1		1	0	1	0	0	0	0		2		0
$\times$																$S = 0$, ne postoji greška		
Z																		
1	1	1	0	0	0	0												

Brojevi (0,2,2) u tabeli predstavljaju broj jedinica po redovima matrice koja predstavlja proizvod $H \cdot Z$. Na osnovu funkcije za neparnu parnost (XOR) određujemo sindrom reč.

U nastavku ovog primera uvodimo grešku na trećem bitu kodovane poruke ili kodne reči. Nakon prenosa podataka preko komunikacionog kanala sa greškom dogodila se greška na trećoj poziciji.

Ponovo prelazimo u fazu određivanja sindrom reči, sličnu kao u prethodnom slučaju.

H							=								=	S		
0	0	0	1	1	1	1		0	0	0	0	0	0	0		0	0	
0	1	1	0	0	1	1		0	1	0	0	0	0	0		1	1	
1	0	1	0	1	0	1		1	0	0	0	0	0	0		1	1	
$\times$																$S = 3$		
Z															greška			
1	1	0	0	0	0	0									postoji			

Kada smo odredili sindrom reč $S = (011)$, možemo da zaključimo da se greška nalazi na trećoj poziciji. Sada prelazimo u poslednju fazu dekodovanja kada je postojala greška u prenosu. Dekodovana poruka U dobija se iz ozraza $U = Z' \oplus S$, gde Z' označava kodovanu poruku prenešenu na drugu stranu kanala koja potencijalno sadrži grešku.

Z'								Pozicije označene sa P predstavljaju kontrolne bitove, a pozicije označene sa I predstavljaju informacione bitove										
1	1	0	0	0	0	0												
XOR																		
S								P_1	P_2	I_2	P_3	I_2	I_3	I_4	U			
0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	I_1	I_2	I_3	I_4
Z															1	0	0	0
1	1	1	0	0	0	0	0								Kraj.			

U nastavku ovog primera sledi rešenje zadatka:

Sledećim naredbama definisaćemo Hemingov kod (7,4). n - označava dužinu kodne reči, r - označava broj kontrolnih bitova, a k - broj informacionih bitova.

Broj redundantnih bitova ili kontrolnih bitova:

$$r = \log_2(n+1)$$

$$r = 3$$

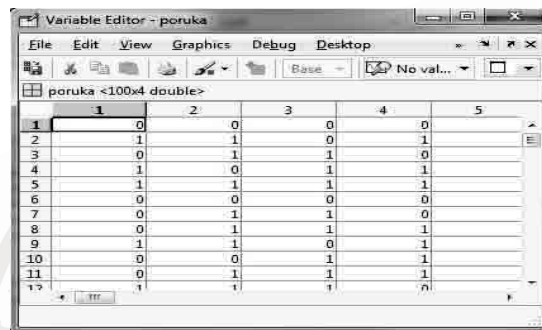
Broj informacionih bitova:

$$>> k = n - \log_2(n+1)$$

$$k = 4$$

Generisanje neke slučajne poruke:

```
>> poruka = vec2mat(randint(100*4,1,[0,1]),k);
```

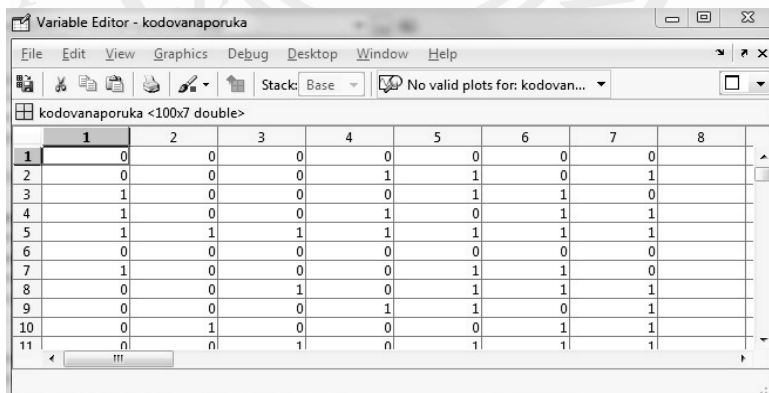


	1	2	3	4	5
1	0	0	0	0	
2	1	1	0	1	
3	0	1	1	0	
4	1	0	1	1	
5	1	1	1	1	
6	0	0	0	0	
7	0	1	1	0	
8	0	1	1	1	
9	1	1	0	1	
10	0	0	1	1	
11	0	1	1	1	
12	1	1	1	0	

Slika 5.20 Generisana poruka

Kodovanje sa algoritmom *hamming/binary*:

```
>> kodovanaporuka = encode(poruka,n,k,'hamming/binary');
```



	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	
2	0	0	0	1	1	0	1	
3	1	0	0	0	1	1	0	
4	1	0	0	1	0	1	1	
5	1	1	1	1	1	1	1	
6	0	0	0	0	0	0	0	
7	1	0	0	0	1	1	0	
8	0	0	1	0	1	1	1	
9	0	0	0	1	1	0	1	
10	0	1	0	0	0	1	1	
11	0	0	1	0	1	1	1	

Slika 5.21 Kodovana poruka

Sada simuliramo prenos poruke preko šumnog kanala, tako što ugrađujemo pogrešne bitove u kodovanu poruku:

```
>> kodovanaporukasadumom = rem(kodovanaporuka + randerr(100,n,[0 1; .25  
.75]), 2);
```

Nakon prenosa preko šumnog kanala, utvrdićemo broj pogrešnih bitova preko XOR operacije:

```
>> [brojGresaka, greska] = biterr(kodovanaporuka, kodovanaporukasadumom)

brojGresaka = 70
greska = 0.1000
```

Prelazimo na fazu detekcije i ispravljanja grešaka:

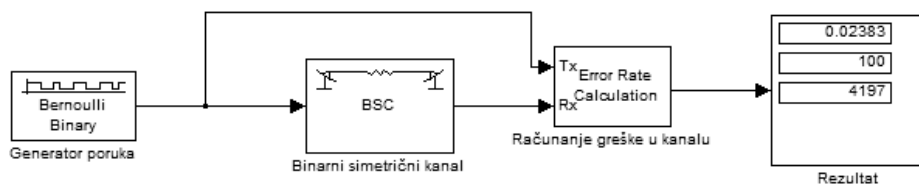
```
>> dekodovanaporuka = decode(kodovanaporukasadumom, n, k,  
'hamming/binary');
```

Na kraju proveravamo performanse Hemingovog koda (7,4) i utvrđujemo da je ovaj kod ispravio sve greške (10% pogrešnih bitova):

```
>> [brojGresaka, greska] = biterr(dekodovanaporuka, poruka)

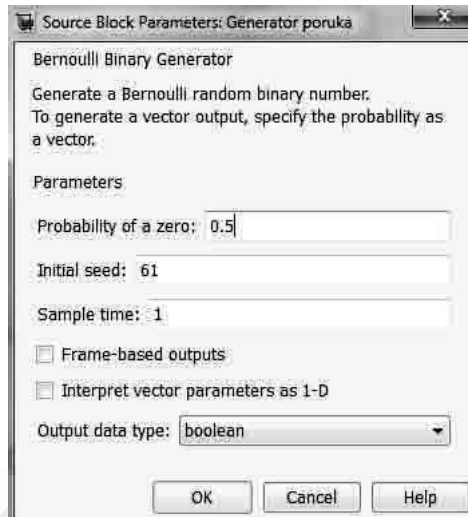
brojGresaka = 0
greska = 0
```

Nastavak rešenja zadatka u simulink-u, sledi simulacija BSC kanala sa šumom:



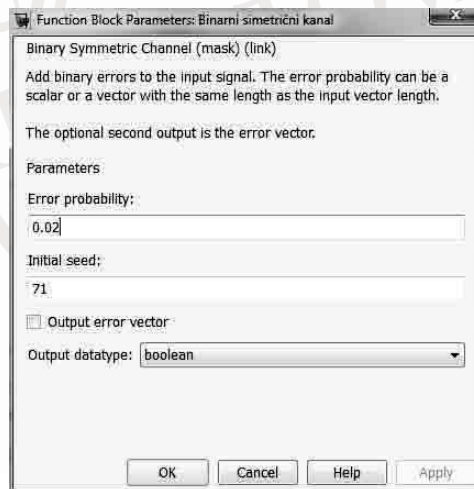
Slika 5.22 BSC sa verovatnoćom greške 0,02

Podešavanje generatora poruka:



Slika 5.23 Bernulijev generator poruka sa verovatnoćom 0,5

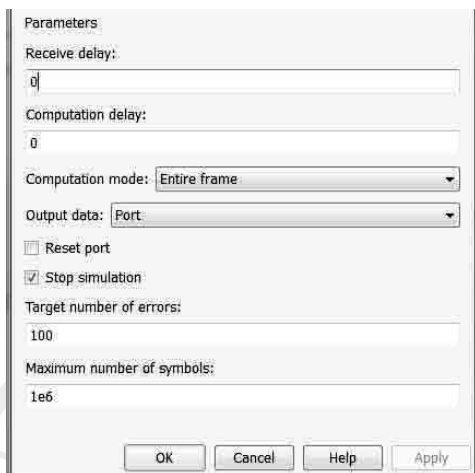
Podešavanje binarnog simetričnog komunikacionog kanala:



Slika 5.24 Simulacija BSC kanala sa šumom, verovatnoća greške 0,02

Verovatnoća greške u kanalu je 0,02, a inicijalno stanje 71. Inicijalni *sid* ili stanje generatora predstavlja neko slučajno izabrano početno stanje pseudo generatora za generisanje greške.

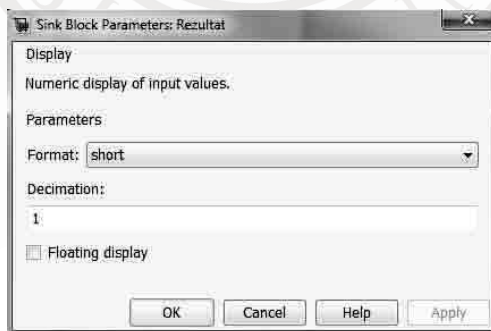
Podešavanje bloka za računanje greške:



Slika 5.25 Blok za računanje greške u kanalu

Parametri bloka sa slike (Slika 5.25) kontrolišu vreme izvršavanja kompletne simulacije. Nakon 100 nastalih grešaka simulacija će biti zaustavljena i rezultat simulacije biće prosleđen narednom bloku za prikaz rezultata.

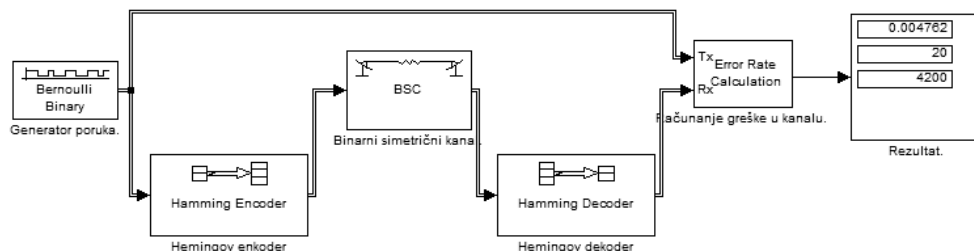
Podešavanje bloka za prikaz rezultata:



Slika 5.26 Blok za prikaz rezultata simulacije

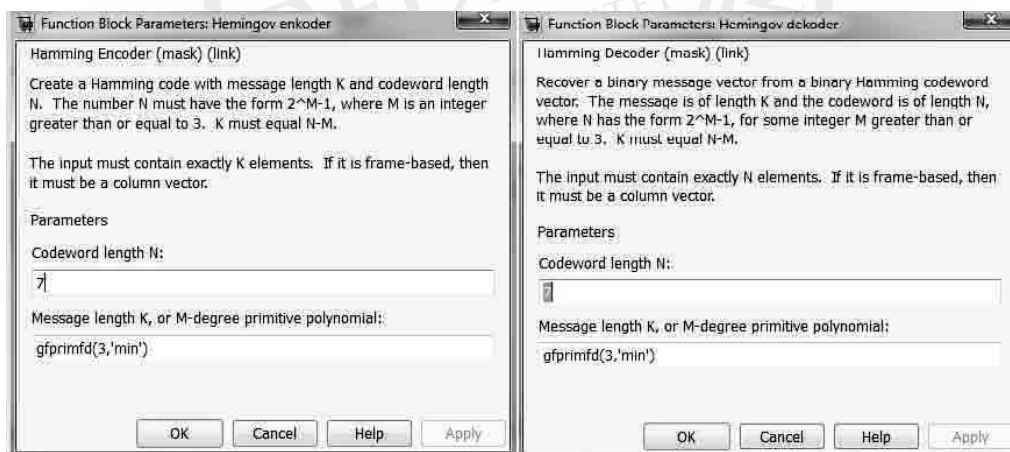
Parametri ovog bloka u datoj simulaciji zadržavaju predefinisane vrednosti. Na rezultatu simulacije možemo da vidimo da je na 4197 bitova prenesene poruke bilo ukupno 100 bita greške.

U nastavku proširujemo simulaciju u simulink-u sa još jednim identičnim kanalom, tim što u drugom kanalu koristimo zaštitni kod koji se implementira preko enkodera na predajnoj strani i dekodera na prijemnoj strani.



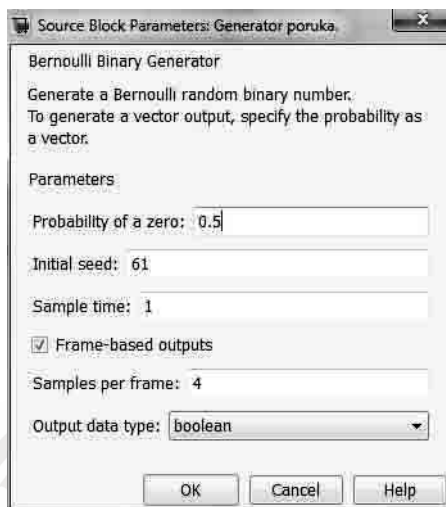
Slika 5.27 BSC sa verovatnoćom greške 0,02 i zaštitnim kodom 7,4

U prethodnoj simulaciji na slici (Slika 5.24) koristili smo Hemingov zaštitni kod (7,4). Na sledećoj slici prikazano je podešavanje enkodera i dekodera:



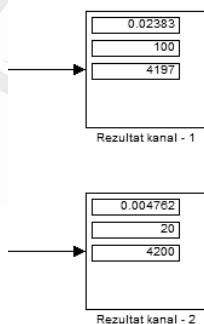
Slika 5.28 Hemingov enkoder i dekoder za kod (7,4)

U nastavku sledi podešavanje generatora poruka:



Slika 5.29 Generator poruka sa verovatnoćom 0,5

Za razliku od bloka za generisanje poruka u prvom kanalu, sada je potrebno definisati takt generatora i broj isporučenih bitova u svakom taktu. Aktiviraćemo opciju „*Frame-based outputs*” i za vrednost parametra „*Samples per frame*” postaviti vrednost 4. Vrednost 4 predstavlja broj k ili informacione bitove kod Hemingovog zaštitnog koda.



Slika 5.30 Poređenje grešaka između kanala bez ECC i sa ECC

Na osnovu rezultata simulacije prikazanog na slici (Slika 5.30), za isti broj prenesenih poruka greška je smanjena za 80 bitova.

VEŽBA 9.

Interliving, uvodna napomena:

Jedan od najpopularnijih načina da se koriguju paketske greške je korišćenje kodova koji dobro rade na slučajnim greškama, tako što se prethodno kroz postupak interlivinga paketske greške rasprše tako da izgledaju kao slučajno raspodeljene.

Postoje dva tipa interlivera koji se danas najviše koriste, to su blokovski i konvolucioni interliveri.

U ovom primeru prikazaćemo pseudoslučajni interliver. Pseudoslučajni interliveri eliminišu regularne obrasce ili određene statističke osobine iz određene sekvence podataka i to tokom veoma dugog perioda. Pseudoslučajni interliver se često koristi kod turbo kodova.

Sledeći primer ilustruje kako interliver poboljšava performanse kodova za ispravljanje grešaka u komunikacionim kanalima sa rafalnim greškama. U ovom modelu koristimo pseudoslučajni interliver i Hemingov binarni zaštitni kod 7,4.

Postavka parametara modela sa i bez primene interlivera.

Definišemo početna stanja generatora:

```
>> st1 = 27221; st2 = 4831;
```

Određujemo parametre za Hemingov kod:

```
>> n = 7; k = 4;
```

Generišemo neku slučajnu poruku za kodovanje:

```
>> poruka = randint(k*500,1,2,st1);
```

Kodujemo poruku Hemingovim kodom:

```
>> kodovana_poruka = encode(poruka,n,k,'hamming/binary');
```

Generišemo slučajnu paketsku grešku nad kodovanom porukom:

```
>> greska = zeros(size(kodovana_poruka)); greska(n-2:n+3) = [1 1 1 1 1];
```

Rezultat sa primenom interlivera.

Pozivamo funkciju za interliving:

```
>> interliver = randintrlv(kodovana_poruka,st2);
```

Uvodimo rafalnu grešku preko XOR operacije:

```
>> interliver_greska = bitxor(interliver,greska);
```

Pozivamo funkciju za deinterliving:

```
>> deinterliver = randdeintrlv(interliver_greska,st2);
```

Dekodujemo poruku:

```
>> dekodovana_poruka = decode(deinterliver,n,k,'hamming/binary');
```

Prikazujemo statističke podatke za grešku:

```
>> disp('Broj greska i stopa greske, sa primenom interlivera:');
```

```
>> [broj_gresaka , stopa_greske] = biterr(poruka, dekodovana_poruka)
```

```
broj_greska =    0
```

```
stopa_greske =    0
```

Rezultat bez primene interlivera.

Uvodimo rafalnu grešku preko XOR operacije:

```
>> kodovana_poruka_greska = bitxor(kodovana_poruka,greska);
```

Dekodujemo poruku:

```
>> dekodovana_poruka =
```

```
decode(kodovana_poruka_greska,n,k,'hamming/binary');
```

Prikazujemo statističke podatke za grešku:

```
>> disp('Broj greska i stopa greske, bez primene interlivera:');  
>> [broj_gresaka , stopa_greske] = biterr(poruka,dekodovana_poruka)  
  
      broj_gresaka =    4  
      stopa_greske = 0.0020
```



6. KRIPTOGRAFIJA – INFORMACIONO TEORIJSKI UVOD

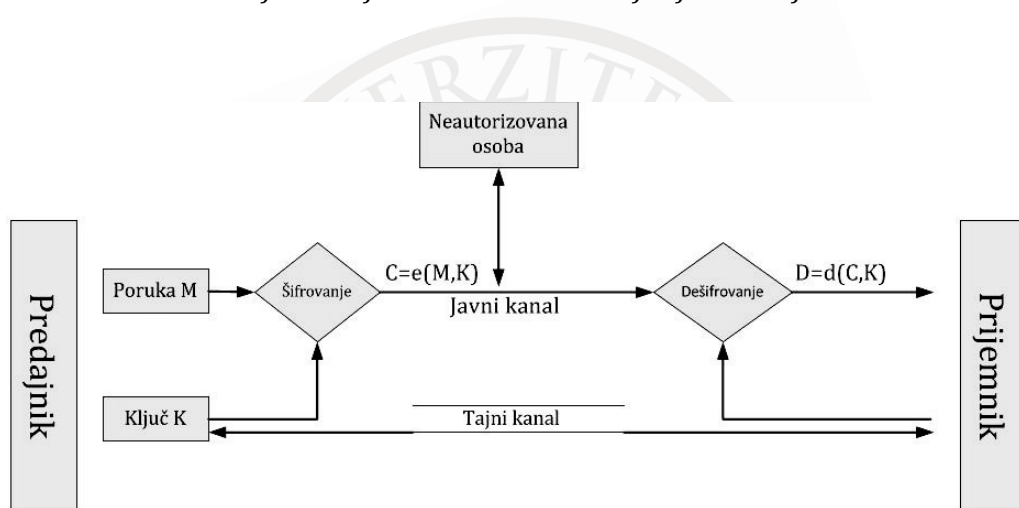
Ciljevi ovog poglavlja su da damo odgovore na sledeća pitanja:

Šta je predmet i cilj kriptografije?

Teorijska postavka problema kriptografije u okviru teorije informacija.

Šta je perfektan tajnost i kako se postiže?

Šta je tačka jedinstvenosti i kakav je njen značaj?



Slika 6.1 Kodovanje poruka u cilju obezbeđenja tajnosti i autentičnosti.

Sledi funkcija šifrovanja sa (Slika 6.1):

$$C = e(M, K)$$

C – šifrat, M – poruka, K – tajni ključ, $H(C|M, K) = 0$,

Funkcija šifrovanja je deterministička, odnosno C je jednoznačno dređeno ako se zna M i K .

Funkcija dešifrovanja:

$$d(e(M, K), K) = M,$$

dešifrovanje istim tajnim ključem daje originalnu poruku. Dešifrovanje je takođe determinističko, odnosno:

$$H(M|C, K) = 0$$

Primetimo važnu činjenicu da $H(C|M, K) = 0$, i $H(M|C, K) = 0$ ne impliciraju $H(K|M, C) = 0$. Odnosno, principijelno je moguće više ključeva za fiksirano M i C .

Međutim, u kriptografskoj praksi se to po pravilu ne dešava (sa razlogom), tako da se praktično uzima da i za ključ važi $H(K|M, C) = 0$.

Cilj kriptografije je zaštita poruke od:

1. Pogrešnog primaoca (tajnost), nemoguće je dobiti M na osnovu C , bez poznavanja K .
2. Pogrešnog emitovanje (autentifikacija), nemoguće je zameniti poruku modifikacijom šifrata C u C' , bez poznavanja K .

Cilj kriptanalitičara je razbijanje sistema tajnosti/autentifikacije, odnosno dolaženje do M ili K na osnovu poznavanja C . Pretpostavke koje se pri tome čine su:

1. Postupak šifrovanja i dešifrovanja je svima poznat (Kerkhofov princip) kao i statistička svojstva poruka.
2. Neautorizovani korisnici ne znaju ključ K .
3. C je svima dostupno (ali ne i K i M).
4. Sva tajnost se svodi na to da napadač sistema ne poznaje tajni ključ.

6.1. PERFEKTNA TAJNOST

DEFINICIJA 6.1 Perfektna tajnost

Šifrski sistem je perfektan, ako i samo ako uzajamna informacija između otvorene poruke M i odgovarajućeg šifrata C je jednaka nuli, odnosno:

$$I(C; M) = 0.$$

TEOREMA 6.1

U perfektnom šifarskom sistemu mogućih ključeva mora biti barem koliko i mogućih poruka.

TEOREMA 6.2

U perfektnom šifarskom sistemu neodređenost ključa $H(K)$ je najmanje velika koliko i neodređenost poruka $H(M)$ $H(K) \geq H(M)$.

Posledice ove dve teoreme su zaključak da u perfektnim šifarskim sistemima ključevi moraju biti dovoljne kompleksnosti ili najmanje onoliko koliko je kompleksnost poruka.

PRIMER 6.1 One-Time-Pad.

Ovo je primer dobro poznatog perfektnog šifarskog sistema. U ovom sistemu, ključ je jednokratni čisto slučajan binarni niz $K = K_1, K_2 \dots K_n$ takav da je:

$$(K_i = 0) = p(K_i = 0 | K_1, \dots, K_{i-1}) = 0.5$$

gde je n veličina najveće poruke koju treba preneti. Šifrovanje se obavlja na osnovu sabiranja na nivou bita:

$$C_i = M_i + K_i.$$

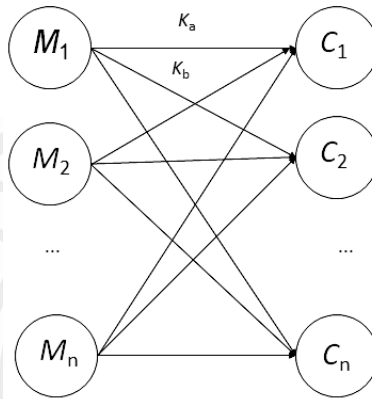
Ključ se sme koristiti samo jednokratno. Upotreba jednog ključa u dva šifrovanja sa dve različite poruke, kompromituje ovaj sistem.

6.1.1. PERFEKTNE ŠIFRE: DEFINICIJA

Kod perfektne šifre postoji ključ koji preslikava svaku poruku u svaki šifrat sa jednakom verovatnoćom.

Za svako i, j :

$$p(M_i | C_j) = p(M_i).$$



Slika 6.2 Grafička prezentacija perfektne šifre

Definicija:

$$\forall i, j: P(M_i | C_j) = P(M_i)$$

Šifra je perfektna ako:

$$\forall M, C \ P(C | M) = P(C)$$

ili, ekvivalentno:

$$\forall M, C \ P(M | C) = P(M),$$

bez poznavanja ključa, svaki šifrat podjednako verovatno može odgovarati bilo kom otvorenom tekstu.

PRIMER 6.2 One-Time pad šifra.

$$k = 0110010110001$$

$$m = 1100110101010$$

$$C = 1010100011011$$

TEOREMA 6.3

One-Time pad je perfektna šifra.

DOKAZ

Za $1 \leq i \leq n$:

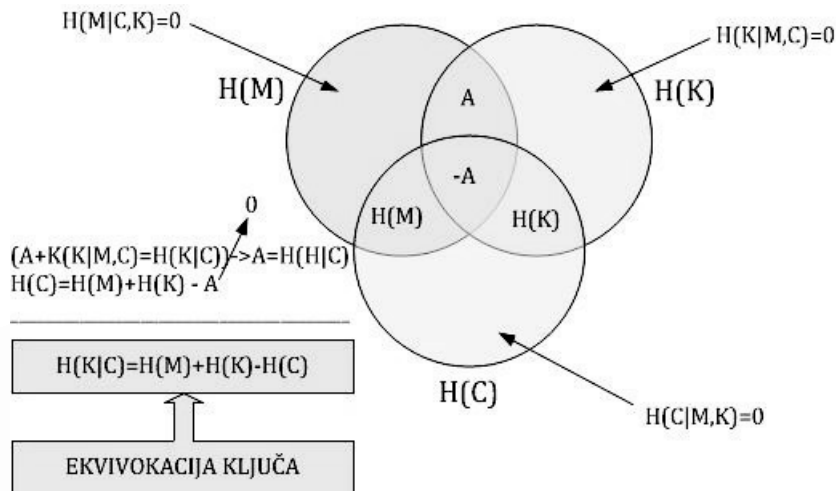
$$\begin{aligned} p(C_i = 0 | C_1, \dots, C_{i-1}, M_1, \dots, M_n) \\ &= p(M_i = 0 | C_1, \dots, C_{i-1}, M_1, \dots, M_n) \cdot p(K_i = 0) \\ &\quad + p(M_i = 1 | C_1, \dots, C_{i-1}, M_1, \dots, M_n) \cdot p(K_i = 1) \\ &= 0.5[p(M_i = 0 | C_1, \dots, C_{i-1}, M_1, \dots, M_n) \\ &\quad + p(M_i = 1 | C_1, \dots, C_{i-1}, M_1, \dots, M_n)] = 0.5 \end{aligned}$$

Slično:

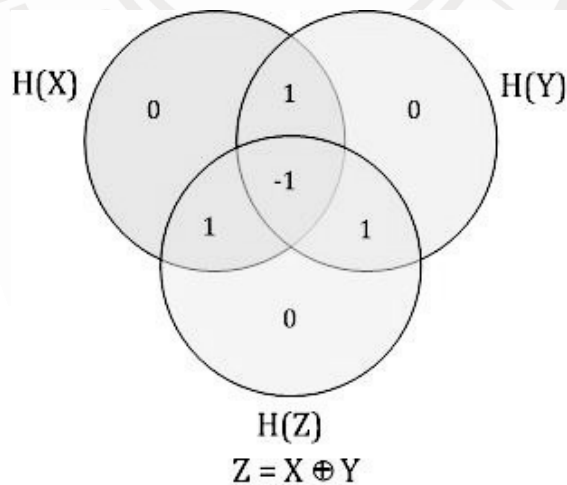
$p(C_1 | M_1, \dots, M_n) = 0.5$, i $p(C_i | C_1, \dots, C_{i-1}) = 0.5$ za $i, 1 \leq i \leq n$, pa je $P(C|M) = P(C)$, tj. $I(C; M) = 0$.

6.1.2. GRAFIČKA PREZENTACIJA PERFEKTNE ŠIFRE

Gore navedeni stav je moguće dokazati i grafički na osnovu Jengovih dijagrama.

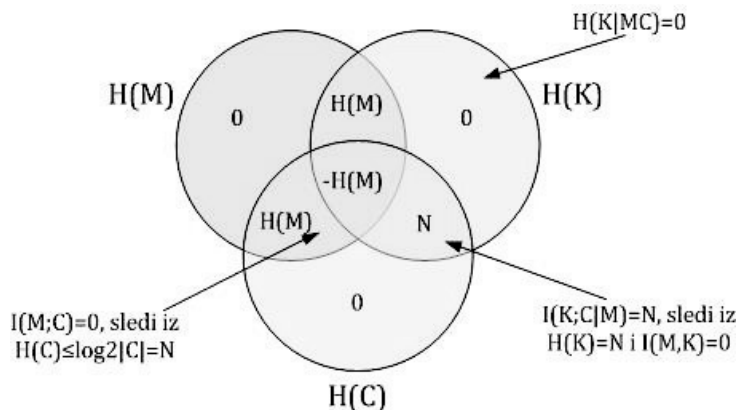


Međuzavisnost informacionih mera u relaciji $Z = X \oplus Y$.



Slika 6.3 Međuzavisnost informacioni mera

6.1.3. DOKAZ ŠENONOVE TEOREME O PERFEKTNOJ TAJNOSTI - OTP



Slika 6.4 Perfektna tajnost One-Time pad-a

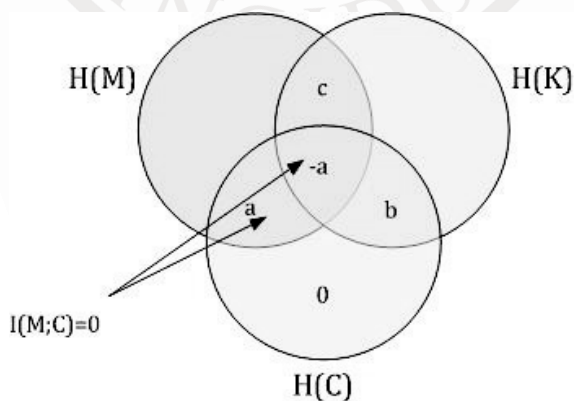
Za svaki perfektan kriptosistem sa jedinstvenim dekodovanjem važi $H(K) \geq H(M)$.

Jedinstvenost dekodovanja:

$$H(M | C, K) = 0.$$

Iz $I(C; K) \geq 0 \rightarrow b \geq a$, a odavde:

$$H(K) \geq b - a + c \geq a - a + c = H(M).$$



Slika 6.5 Dokaz Šenonove teoreme

6.2. NEPERFEKTNA TAJNOST I TAČKA JEDINSTVENOSTI

U praksi se često odstupa od perfektnih šifara zbog kompromisa između nivoa potrebne tajnosti i cene uvođenja jednog šifarskog sistema. Široko je rasprostranjena klasa neperfektnih šifarskih sistema, čiji je ključ znatno manji od poruka i koristi se veliki broj puta.

Šta možemo reći o praktično upotrebljivijim, ali neperfektnim šifarskim sistemima?

Šenon je uveo tzv. ekvivokacionu funkciju, definisanu na skupu prirodnih brojeva:

$$\alpha(n) = H(K|C_1 \dots C_n).$$

Intuitivno je jasno da što više šifrata C imamo na raspolaganju, sve će se više smanjivati neodređenost ključa K . Formalno, to bi značilo:

$$\lim_{n \rightarrow \infty} \alpha(n) = 0$$

Tačka jedinstvenosti se definiše kao najmanje n za koje je $\alpha(n) \cong 0$.

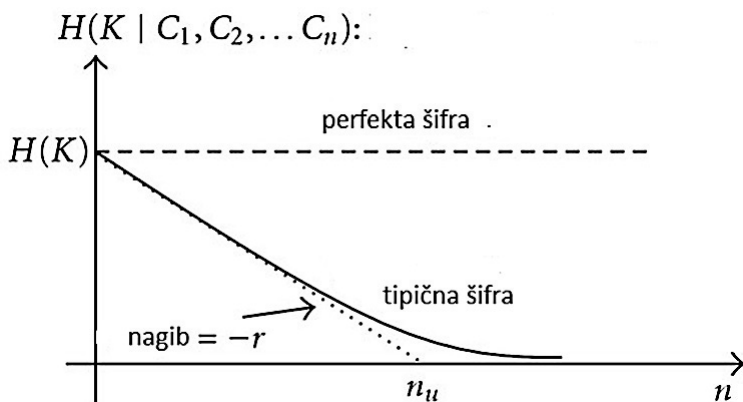
DEFINICIJA 6.2 Tačka jedinstvenosti.

Tačka jedinstvenosti datog kriptosistema je najmanje n za koje važi:

$$H(K|C_1 \dots C_n) \cong 0.$$

6.2.1. EKVIVOKACIJA KLJUČA I TAČKA JEDINSTVENOSTI

Po definiciji, tačka jedinstvenosti je najmanja količina šifrata kojom mora raspolagati kriptanalitičar, da bi mogao da odredi ključ K jednoznačno. Drugim rečima to je najmanja količina šifrata koja omogućava da sistem bude razbijen.



Slika 6.6 Tačka jedinstvenosti – perfektne i neperfektne šifre

TEOREMA 6.4

Ako važi:

M i C su iste dužine n i pripadaju istom alfabetu. Šifrovane poruke imaju približno maksimalnu neodređenost:

$$H(C_n) \cong n \log D,$$

što je kriterijum kome teže svi dizajneri kriptosistema.

Ključevi i poruke su nezavisni

$$H(M_n, K) = H(M_n) + H(K),$$

što je uobičajeno i prirodno. tada je tačka jedinstvenosti aproksimativno jednaka

$$u \approx \frac{H(K)}{R(M) \cdot \log D},$$

gde je $R(M)$ redundansa poruka M koje se šifruju, definisana sa:

$$R(M) = 1 - \frac{H_\infty(M)}{\log D}.$$

DOKAZ:

Pretpostavimo da je n dovoljno veliko, tako da važi:

$$H(M_n) \approx n \cdot H_\infty(M),$$

$$\begin{aligned} H(K|C_n) &= \overbrace{H(M_n K C_n) - H(M_n | K C_n)}^{H(K C_n)} - H(C_n) = H(M_n K C_n) - H(C_n) \\ &= H(M_n K) - H(C_n) = H(M_n) + H(K) - H(C_n) \\ &= n \cdot H_\infty(M) + H(K) - n \cdot \log D \end{aligned}$$

Tačka jedinstvenosti je definisana sa:

$$H(K|C_u) = 0,$$

odnosno:

$$u(H_\infty(M) - \log D) + H(K) = 0,$$

odakle sledi:

$$u = \frac{H(K)}{\log D - H_\infty(M)} = \frac{H(K)}{R(M) \cdot \log D}.$$

PRIMER 6.5 Tačka jedinstvenosti.

Posmatrajmo poruke pisane na engleskom jeziku sa 27 znakova uključujući razmake između reči i šifrovane ključem koji se sastoji od 20 nezavisnih karaktera engleskog alfabeta.

Tada je $H(K) = 20 \log 27$.

Ako za engleski jezik usvojimo entropiju od 2 bita po znaku, redundansa poruka je data sa $R(M) = 1 - 2/\log 27 \approx 0.58$. Tačka jedinstvenosti ovog šifarskog sistema je data sa:

$$u = \frac{H(K)}{R \log D} = \frac{20 \cdot \log(27)}{\log(27) - 2} \cong 35$$

Ovaj rezultat znači da sa poznavanjem šifrata dužine oko 35 karaktera, ključ može odrediti jednoznačno.

Primetimo da je tačka jedinstvenosti u , principijelno, količina šifrata dovoljna za jednoznačno određivanje ključa. Međutim, o samom konkretnom načinu i složenosti određivanja ključa K na osnovu poznavanja šifrata nema nijedne reči u Šenonovim radovima.

6.2.2. POVEĆANJE TAČKE JEDINSTVENOSTI

Iz same definicije tačke jedinstvenosti vidimo da je jedan mogući način za njeno povećavanje, smanjivanje redundanse poruka koje se šifruju, odnosno povećanje njihove entropije.

Stoga je na primer dobra ideja komprimovati poruke pre šifrovanja.

I zaista za najbolje moguće komprimovanje:

$$H(M_n) \cong n \log D, \quad R(M) \cong 0, \quad u \rightarrow \infty.$$

PRIMER 6.6

Tačka jedinstvenosti kriptosistema kojim se šifruju poruke iz alfabeta od 96 znakova, pri čemu je entropija poruka 3 bita/po znaku, dok je entropija ključa 33 bita je:

$$u = \frac{33}{\log 96 - 3} = 9.2,$$

$$R = 1 - \frac{3}{\log 96} = 54.4\%.$$

PRIMER 6.7

Tačka jedinstvenosti kriptosistema kojim se šifruju binarne poruke, redundanse 25%, dok su ključevi dužine 16 bita i uniformno su raspodeljeni je:

$$u = \frac{16}{0.25 \cdot 1} = 64.$$

6.3. AUTENTIFIKACIJA

Naučićemo:

Zašto su autentifikacija i tajnost teoretski nekompatibilni?

Kako obezbediti autentifikaciju u praksi?

Autentifikacija obezbeđuje mehanizam kojim se možemo uveriti da je primljena poruka poslata od autorizovane osobe.

Drugim rečima kada primimo kriptogram C , moramo imati mehanizam kojim se uveravamo da je kriptogram poslat od strane autorizovane osobe i da nije rezultat prevare u kojoj iza kriptograma stoji neka druga neautorizovana osoba.

Relativno je novi rezultat u kriptologiji koji ustanovljava da su tajnost i autentičnost dva različita koncepta sa nekompatibilnim ciljevima.

Ovaj stav sledi iz sledeće teoreme.

TEOREMA 6.7

Verovatnoća P_I da je kriptogram falsifikovan, odnosno da ga je poslala neautorizovana osoba, je ograničena sa:

$$P_I \geq 2^{-I(C;K)}$$

gde su C i K slučajne veličine koje prezentuje moguće kriptograme odnosno ključeve, respektivno. Ova granica se dostiže u posebnim slučajevima.

Odavde sledi da garantovanje autentifikacije povlači za sobom da uzajamna informacija između kriptograma i ključa mora biti velika. S druge strane da bi smo obezbedili perfektnu tajnost, ta ista uzajamna informacija mora biti jednaka nuli.

Sa stanovišta tajnosti poruke i autentifikacije imamo protivrečnu situaciju, naime:

$$I(C; K) = I(C; K) + H(K) - H(M) - H(K|M, C),$$

ako stavimo da je:

$$I(C; M) = 0 \text{ i } H(K|M, C) = 0,$$

dobijamo:

$$I(C; K) = H(K) - H(M),$$

što povlači:

$$P_I \geq 2^{H(M)-H(K)} > 0,$$

odnosno:

$$(H(K) \gg H(M)),$$

što znači da ključ mora biti vrlo kompleksan.

Jedno moguće rešenje je da se obezbedi autentifikacija zadovoljavanjem uslova:

$$I(C; K) \gg 0,$$

a da se pri tome tajnost obezbedi na osnovu praktične tajnosti zasnovane na algoritamskoj kompleksnosti. Ovo je jedan od razloga zašto su kriptografski sistemi zasnovani na algoritamskoj kompleksnosti popularni za autentifikaciju.

BIBLIOGRAFIJA

A. J. Viterbi and J. K. Omura. Principles of Digital Communication and Coding. [Knjiga]. - New York : McGraw-Hill, 1979.

Aiden A. Bruen, Mario A. Forcinito Cryptography, information tehory, and error-correction [Knjiga]. - New Jersey : A John Wiley & Sons, 2005.

Blahut, Richard E. Theory and Practice of Error Control Codes [Knjiga]. - [s.l.] : Addison-Wesley, 1983.

Dušan B. Drajić, Predrag N. Ivaniš Uvod u teoriju informacija i kodovanje [Knjiga]. - Beograd : Akademska misao Beograd, 2009.

F. Bavaud, J.-C. Chappelier, J. Kohlas An Introduction to Information Theory and Applications [Knjiga]. - 2005 : [s.n.].

Lang, Serge. Algebra [Knjiga]. - [s.l.] : Addison-Wesley, 1993. - T. Third Edition.

Lin, Shu and Daniel J. Costello, Jr. Error Control Coding: Fundamentals and Applications. Englewood Cliffs [Knjiga]. - [s.l.] : Prentice-Hall, 1983.

N. M. Abramson. Information Theory and Coding. [Knjiga]. - New York : McGraw-Hill, 1963.

R. Adler. Ergodic and mixing properties of infinite memory channels [Radovi]. - [s.l.] : Proc. Amer. Math. Soc, 1961. - 12:924-930.

Raymond W Yueng A new Outlook on Shannon Information Measures [Radovi] // IEEE Transactions on IT. - 1991. - T. 37.

Shannon C. E. A mathematical theory of communication [Časopis] // Bell System Tehnical Journal. - 1948. - str. 379-423.

Van Lint, J. H. Introduction to Coding Theory [Knjiga]. - [s.l.] : Springer-Verlag, 1982.

Bernhard M.J. Leiner, LDPC Codes – brief Tutorial, [Stud. ID]: 53418L, April 8, 2005.

DODATAK A – UVOD U EKSPERIMENTALNO OKRUŽENJE

Za programsko-razvojno okruženje u ovom udžbeniku izabrali smo GNU Octave programski paket. GNU Octave je interpreterski jezik visokog nivoa, prvenstveno namenjen za numeričke proračune. Koristi se za rešavanje numeričkih linearnih i nelinearnih problema, kao i za obavljanje drugih numeričkih eksperimenata.

Octave obezbeđuje odlično radno okruženje sa alatima za obradu podataka, vizualizaciju podataka i standardizovane algoritme koji se primenjuju u svim visoko-tehnološkim poljima. Pored mnogobrojnih algoritama nudi programiranje na visokom nivou. Svi primeri u ovom udžbeniku razvijeni su upravo u ovom softveru. Octave jezik je veoma sličan Matlab-u, tako da je većina programa kompatibilna između ova dva programa.

Slede osnovni programski kodovi koji se odnose na elementarno programiranje u Octave.

Deklarisanje vektora:

Generisanje niza X u jednom redu:

```
>> x = [1 3 5 7 9 11]
```

Brži način za generisanje niza X iste veličine:

```
>> x = 1:2:11
```

```
x = 1 3 5 7 9 11
```

Transponovanje niza X u jednu kolonu:

```
>> y = x'
```

```
y =
```

```
1
```

```
3
```

```
5
```

7

9

11

Generisanje niza Z u opsegu vrednosti od 5 do 30 sa korakom 5.

```
>> Z = 5:5:30
```

```
z = 5 10 15 20 25 30
```

Niz Z je generisan i nije prikazan u *command window*. Možemo saznati vrednost trećeg elementa ili proslediti vrednost trećeg elementa tako što ćemo uneti naredbu:

```
>> z(3)
```

```
ans = 15
```

Na ovaj način dobili smo novu promenljivu **ans** koja je automatski deklarisan. Vrednost kojoj unapred nije određen naziv, automatski dobija naziv **ans**.

Naredba za pregled svih promenljivih koje su deklarisan u jednoj instanci:

```
>> who
```

```
Your variables are:
```

```
ans x y z
```

Naredba za uvid u detalje neke promenljive, npr. matrica podataka:

```
>> whos
```

Name	Size	Bytes	Class	Attributes
ans	1x1	8	double	
x	1x6	48	double	
y	6x1	48	double	

z 1x6 48 double

Ili ako želimo da saznamo dimenziju matrice, unosimo sledeću naredbu:

```
>> [m,n]=size(x)
```

```
m = 1
```

```
n = 6
```

ili

```
>> size(x)
```

```
ans = 1 6
```

Ukoliko želimo trajno da sačuvamo vrednosti svih promenljivih iz jedne instance unosimo sledeću naredbu:

```
>> save nazivFajla
```

, ako želimo čuvati vrednosti samo određenih promenljivih:

```
>> save nazivFajla x
```

, ako želimo izbrisati sve promenljive koristićemo naredbu:

```
>> clear all
```

, ako želimo koristiti promenljive koje smo sačuvali u datoteku, koristimo naredbu:

```
>> load nazivFajla
```

Deklarisanje i inicijalizovanje matrica:

```
>> A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
```

```
A =
```

```
16  3  2 13
```

```
5  10 11  8
```

```
9   6  7 12
```

```
4  15 14  1
```

Izračunavanje sume po kolonama matrice A:

```
>> sum(A)
```

```
ans = 34 34 34 34
```

Transponovanje matrice A:

```
>> A'
```

```
ans =
```

```
16  5  9  4
```

```
3  10  6 15
```

```
2  11  7 14
```

```
13  8 12  1
```

Prikaz i izračunavanje sume dijagonalnih elemenata:

```
>> diag(A)
```

```
ans = 16
```

```
10
```

```
7
```

```
1
```

```
>> sum(diag(A))
```

```
ans = 34
```

Prikaz vrednosti elementa sa tražene pozicije:

```
>> A(2,4)
```

```
ans = 8
```

Brzo generisanje matrice:

```
>> Z = zeros(2,3)
```

```
Z = 0 0 0
```

```
0 0 0
```

```
>> F = 5*ones(2,3)
```

```
F = 5 5 5
```

```
5 5 5
```

```
>> N = 10*rand(1,8)
```

```
N = 6.5574 0.3571 8.4913 9.3399 6.7874 7.5774 7.4313 3.9223
```

Rad sa M – fajlovima. Prvo kreiramo fajl Matrica.m. Unećemo sledeću deklaraciju matrice A i sadržaj sačuvati u fajlu:

```
A = [16.0 3.0 2.0 13.0
```

```
5.0 10.0 11.0 8.0
```

```
9.0 6.0 7.0 12.0
```

```
4.0 15.0 14.0 1.0];
```

Sada, ako želimo da učitamo promenljivu A iz fajla M, unećemo sledeću komandu:

```
>>Matrica
```

Zatim unesemo naziv promenljive A:

```
>> A
A =    16    3    2    13
      5   10   11    8
      9    6    7   12
      4   15   14    1
```

Konkatenacija matrica je proces vezivanja ili spajanja dve manje matrice u jednu veću. Primer za matricu A:

```
>> B = [A A+32;A+48 A+16]
B =    16    3    2    13   48   35   34   45
      5   10   11    8   37   42   43   40
      9    6    7   12   41   38   39   44
      4   15   14    1   36   47   46   33
      64   51   50   61   32   19   18   29
      53   58   59   56   21   26   27   24
      57   54   55   60   25   22   23   28
      52   63   62   49   20   31   30   17
```

Primena logaritamske funkcije nad nizom vrednosti

```
>> X = (1:2:16)';
```

```
>> X
```

```
X =    1
```

```
      3
```

5

7

9

11

13

15

>> Log = log2(X)

Log = 0

1.5850

2.3219

2.8074

3.1699

3.4594

3.7004

3.9069

Sledi uvoženje podataka iz Excel datoteke. Napraviti *excel* datoteku podaci.xlsx. U datoteci definisati proizvoljnu matricu [4 x 4]. Promeniti naziv „Sheet1“ u naziv „Matrica“. Unošenjem naredbe *xlsinfo*(nazivDatoteke) dobijamo detaljan uvid u *excel* datoteku.

>> [type, sheets]=xlsinfo('podaci.xlsx');

>> type

type = Microsoft Excel Spreadsheet

>> sheets

```
sheets = 'Matrica' 'Sheet2' 'Sheet3'
```

Ako želimo da uvezemo podatke sa lista Matrica, unosimo naredbu:

```
>> A = xlsread('podaci.xlsx','matrica');
```

```
>> A
```

```
A = 1 2 3 4
```

```
2 3 4 5
```

```
3 4 5 6
```

```
4 5 6 7
```

Zatim primenimo proces konkatencije nad matricom A i upišemo podatke u *excel* datoteku.

```
>> B = [A A+12;A+5 A+8];
```

```
>> B
```

```
B = 1 2 3 4 13 14 15 16
```

```
2 3 4 5 14 15 16 17
```

```
3 4 5 6 15 16 17 18
```

```
4 5 6 7 16 17 18 19
```

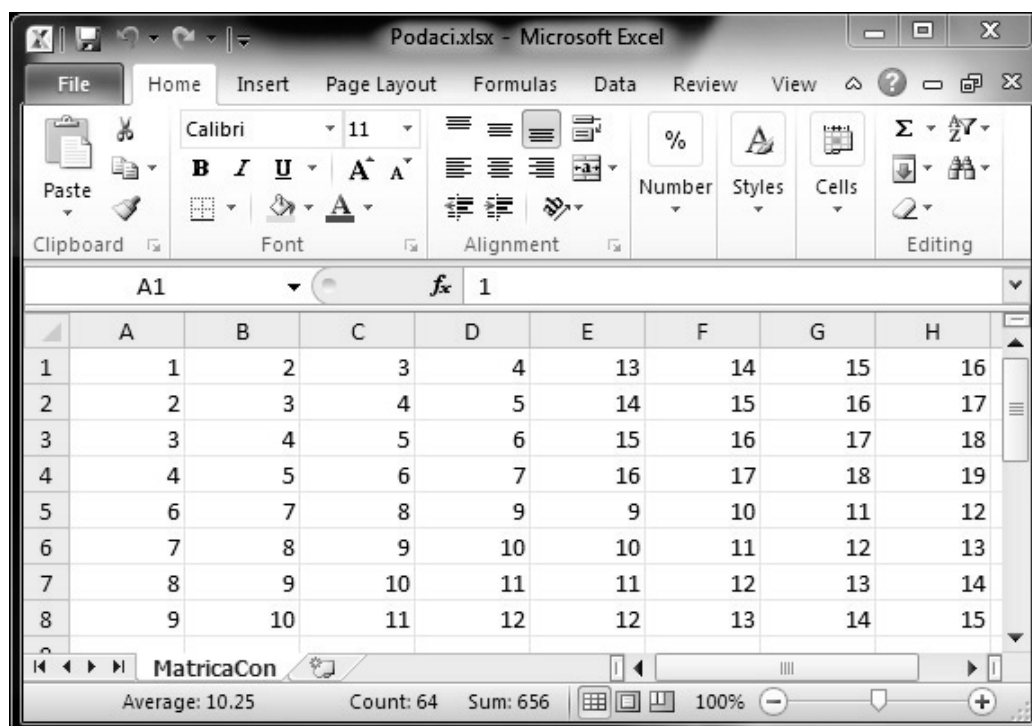
```
6 7 8 9 9 10 11 12
```

```
7 8 9 10 10 11 12 13
```

```
8 9 10 11 11 12 13 14
```

```
9 10 11 12 12 13 14 15
```

```
>> xlswrite('Podaci.xlsx',B,'MatricaCon');
```

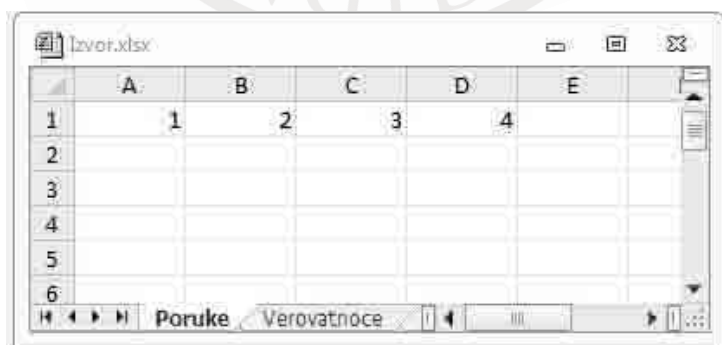


The screenshot shows the Microsoft Excel interface with the file 'Podaci.xlsx'. The active worksheet is 'MatricaCon'. The data is organized in a table with columns A through H and rows 1 through 8. The values in the table are as follows:

	A	B	C	D	E	F	G	H
1	1	2	3	4	13	14	15	16
2	2	3	4	5	14	15	16	17
3	3	4	5	6	15	16	17	18
4	4	5	6	7	16	17	18	19
5	6	7	8	9	9	10	11	12
6	7	8	9	10	10	11	12	13
7	8	9	10	11	11	12	13	14
8	9	10	11	12	12	13	14	15

The status bar at the bottom shows: Average: 10.25, Count: 64, Sum: 656, and a zoom level of 100%.

Generisanje Hafmanovog rečnika i Hafmanovih kodova. Napravićemo *excel* datoteku sa sledećim sadržajem:



The screenshot shows an Excel worksheet named 'Izvori.xlsx'. The data is organized in a table with columns A through E and rows 1 through 6. The values in the table are as follows:

	A	B	C	D	E
1	1	2	3	4	
2					
3					
4					
5					
6					

The status bar at the bottom shows: Poruke, Verovatnoće, and a zoom level of 100%.

	A	B	C	D	E
1	0.5	0.35	0.12	0.03	
2					
3					
4					
5					
6					

Unosimo sledeće naredbe za uvoz podataka u Octave okruženje:

```
>> X = xlsread('Izvor.xlsx','Poruke');
```

```
>> p = xlsread('Izvor.xlsx','Verovatnoce');
```

```
>> X
```

```
X = 1 2 3 4
```

```
>> p
```

```
p = 0.5000 0.3500 0.1200 0.0300
```

Uneti sledeće naredbe za pravljenje Hafmanovog rečnika:

```
>> R = huffmandict(X,p);
```

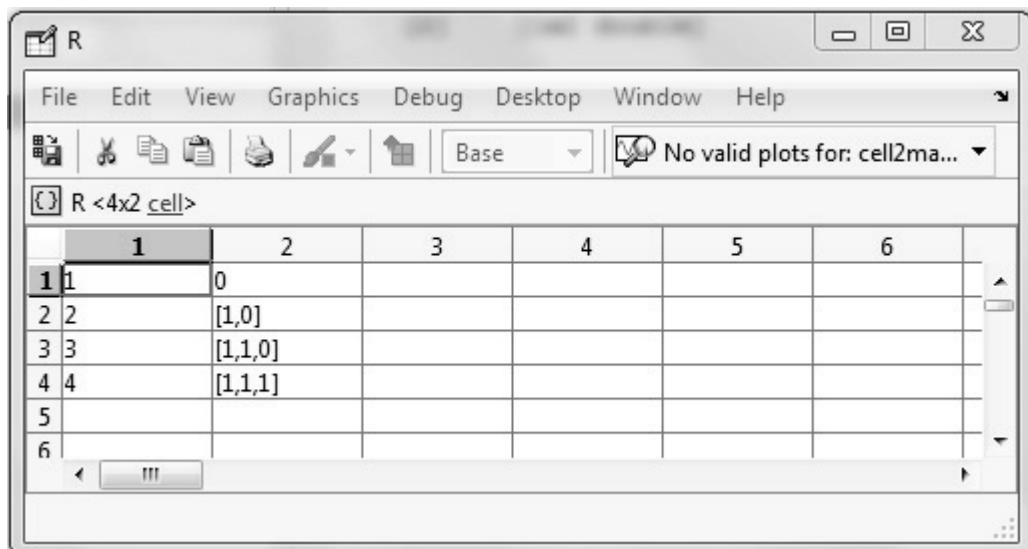
```
>> R
```

```
R = [1] [ 0 ]
```

```
[2] [1x2 double]
```

```
[3] [1x3 double]
```

```
[4] [1x3 double]
```



	1	2	3	4	5	6
1	1	0				
2	2	[1,0]				
3	3	[1,1,0]				
4	4	[1,1,1]				
5						
6						

Sledi Hafmanovo kodovanje.

Sledećom komandom generišemo 50 istih blokova poruke.

```
>> poruke = repmat([3 3 1 3 3 3 3 2 3],1,50);
```

Konačni skup simbola u poruci:

```
>>simboli = [1 2 3];
```

Nasumično određene verovatnoće za 3 simbola:

```
>>verovatnoce = [0.1 0.1 0.8];
```

Rečnik za Hafmanov kod:

```
>>recnik = huffmandict(simboli,verovatnoce);
```

Generisanje kodne sekvence preko Hafmanovog kodera:

```
>>kodnaSekvenca = huffmanenco(poruke,recnik);
```

Generisanje dekodovane sekvence preko Hafmanovog dekodera:

```
>>dekodovanaSekvenca = huffmandeco(kodnaSekvenca,recnik);
```

poruke

File Edit View Graphics

poruke <1x500 double>

	1	2	3
1	3	3	1
2			
3			

verovatnoce

File Edit View Graphics Debug

verovatnoce <1x3 double>

	1	2	3
1	0.1000	0.1000	0.8000
2			
3			

kodnaSekvenca

File Edit View Graphics

kodnaSekvenca <1x600 double>

	1	2	3
1	0	0	1
2			
3			

recnik

File Edit View Graphics Debug

recnik <3x2 cell>

	1	2	3
1	1	[1,1]	
2	2	[1,0]	
3	3	0	

dekodovanaSekvenca

File Edit View Graphics

dekodovanaSekvenca <1x500 double>

	1	2	3
1	3	3	1
2			
3			

simboli

File Edit View Graphics Debug

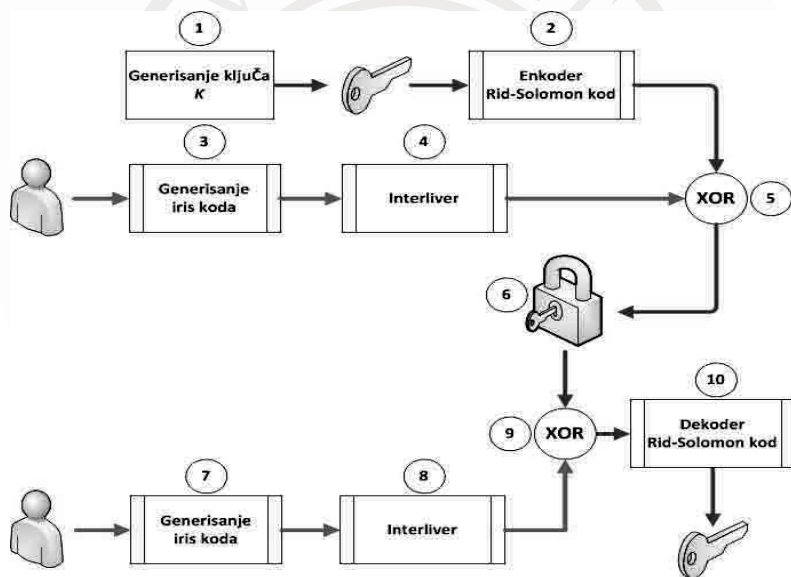
simboli <1x3 double>

	1	2	3
1	1	2	3
2			
3			

DODATAK B - JEDNA PRIMENA KODOVA ZA ISPRAVLJANJE GREŠAKA U KRIPTOGRAFIJI

U dodatku B prikazaćemo jednu primenu kodova za ispravljanje grešaka u sintezi sistema za generisanje kriptoloških ključeva na osnovu biometrijskih podataka. U ovom delu udžbenika data su uvodna razmatranja vezana za različite aspekte razvoja novog sistema za generisanje kriptoloških ključeva na osnovu biometrije irisa, kao i kompletan proces njegovog razvoja i integracije u standardne biometrijske sisteme.

Sledi generička šema za sintezu jednog kompleksnog sistema sa detaljnim opisima svih komponenti koje na direktan ili indirektan način učestvuju u svojstvu gradivnih komponenti ovog rešenja.



Generička šema sistema

Na slici je predstavljena generička šema predloženog rešenja sa svim svojim pod sistemima. Kompletan sistem sastoji se od dva pod sistema: pod sistema za kodovanje iris biometrijskog izvora koji je zasnovan na algoritmima tradicionalnih biometrijskih sistema i pod sistemu za prenos poruka preko biometrijskog kanala sa šumom. U nastavku sledi detaljno objašnjenje

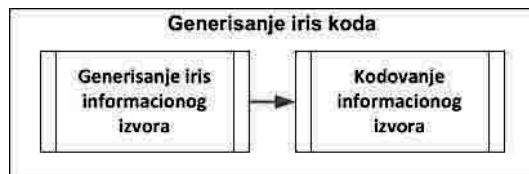
pojedinačnih uloga svih navedenih komponenti koje su sadržane u ovom dijagramu procesa. Komponente su numerisane brojevima od 1 do 10.

Komponenta 1 - predstavlja proces u kom se generiše kriptološki ključ zadate veličine. Preporučeno je da ključ bude generisan na slučajan način preko informacionih izvora koje je moguće pronaći u prirodnim okruženjima. U drugom slučaju koji nije toliko popularan, moguće je ključ generisati i nekim pseudoslučajnim generatorom. Ovakav način generisanja ključa može da naruši bezbednost jednog simetričnog kriptosistema.

Komponenta 2 - predstavlja proces zaštitnog kodovanja za korekciju grešaka u kome se koriste Rid-Solomon kodovi. U ovom procesu kriptološki ključ K je kodovan Rid-Solomon koderom. Uloga ovog procesa je od suštinske važnosti za sistem, zapravo tehnika preuzeta iz polja komunikacija, na osnovu koje je nastala i čitava ideja o potpuno jednom novom naučnom istraživačkom polju pod nazivom „biometrijska kriptografija”.

Komponenta 3 – predstavlja proces generisanja iris koda koji pripada podsistemu za kodovanje iris biometrijskog izvora. U našem slučaju specifičan proces koji u sebi sadrži dva podprocesa. Prvi podproces se odnosi na standardnu proceduru za generisanje biometrijskog obrasca koji je opisan u poglavlju 3. Drugim rečima, možemo da kažemo da je ovo jedan složen proces za formiranje iris informacionog izvora na osnovu koga je kasnije urađena sinteza kompletnog rešenja o kome je reč. Drugi podproces predstavlja još jedan složen proces koji uvodi informacione mere i na osnovu rezultata dobijenih preko teorijsko-informacione analize koduje informacioni izvor u cilju izdvajanja najkonzistentnijih informacija.

Napomenućemo da je ovo kritična tačka u ovom sistemu, gde se dobijen loš rezultat prenosi domino efektom na preostale komponente. Zapravo, performanse čitavog sistema zavise od kvaliteta procesa kodovanja iris informacionog izvora. Iz tog razloga sprovedi smo detaljnu i rigoroznu teorijsko-informacionu analizu o kojoj ćemo kasnije govoriti u ovom poglavlju.



Proces generisanja iris koda

Komponenta 4 – predstavlja još jednu tehniku preuzetu iz teorije komunikacija. Interliver ima značajnu uloga u sistemu. Koristi se za raspršivanje paketskih grešaka i smanjenje uzajamne informacije između različitih iris. U ovom rešenju koristi se slučajni interliver (engl. *Random interliver*). Tehniku interlivinga primenićemo nad kodovanim iris kodom.

Komponenta 5 - predstavlja logičku operaciju "XOR" u kojoj se bodovan kriptološki ključ čvrsto vezuje za originalni iris kod nad kojim je primenjena tehnika za interliving. Imajući u vidu da su oba podatka slučajna, dobijamo sistem koji po svim svojim osobinama, dosta podseća na „One-time pad” perfektni tajni kripto sistem.

Komponenta 6 - predstavlja podatak koji je rezultat „XOR” operacije. Dobijeni podatak u sebi čuva zaključanu originalnu biometriju i slučajno generisani ključ. Na osnovu ovog podatka nije moguće otkriti originalnu biometriju ili ključ. Ovaj podatak zadovoljava formu podatka za skladištenje sa aspekta zaštite privatnosti biometrijskih podataka.

Komponenta 7 - predstavlja proces u kome se započinje sa procedurom u kojoj regularni korisnik želi da na osnovu originalne iris biometrije regeneriše ključ koji je zaključan čvrstom vezom originalne iris biometrije i ključa. Ovaj proces je identičan procesima u komponenti 3.

Komponenta 8 - predstavlja proces koji je identičan procesu u komponenti 4. Tehnika interlivinga je primenjena i nad originalnom iris biometrijom koja služi za regenerisanje ključa K . Napominjemo da u oba slučaja interliver radi samo u jednom smeru bez inverznog režima. Tačnije, u ovom ekvivalentnom kanalu ne postoji potreba za invertibilnosti.

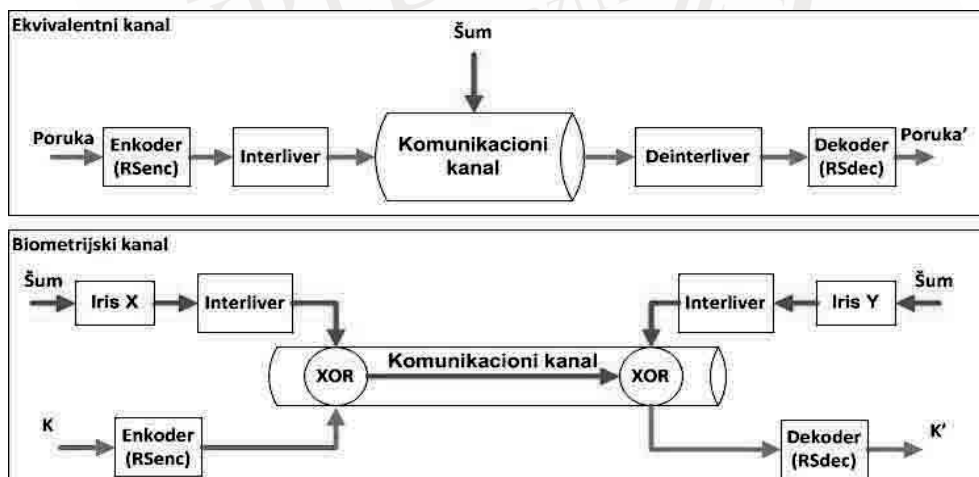
Komponente 9 i 10 - predstavljaju dva poslednja procesa u dijagramu procesa, preko kojih se regeneriše ključ koji je zaključan čvrstom vezom u komponenti 5

na dijagramu. Zatim, ponovo se primenjuje logička operacija „XOR” nad kodovanim iris kodom i podatkom koji u sebi čuva ključ. Zatim se primenjuje Rid-Solomon dekodler koji koriguje sve nastale greške. Ukoliko je dobijeni rezultat posle dekodera jednak ključu koji je generisan u procesu (1), to znači da je kompletna procedura realizovana uspešno i da smo napravili sistem koji ima mogućnost da na osnovu istog iris biometrijskog podataka generiše kriptološke ključeve.

Imajući u vidu da projektovani sistem pripada biometrijskoj kriptografiji, glavni izazov je integritet podataka.

Analiza ekvivalentnog kanala

Analiza ekvivalentnog kanala podrazumeva pronalaženje nekog klasičnog komunikacionog kanala iz teorije komunikacija koji utvrđuje uslove pod kojima je moguć pouzdan prenos informacija, a odgovara po svim svojim karakteristikama našem kanalu, ali sa različitim sadržajem. Potrebno je utvrditi maksimalnu brzinu pouzdanog prenosa po kanalu sa šumom uz upotrebu kodova za ispravljanje grešaka i maksimalnu količinu grešaka koja nastaje kada se informacija prenese većom brzinom od maksimalne.



Analiza ekvivalentnog kanala

Sada je jasno da ekvivalentni kanal predstavlja isti kanal koji je opisan na dva različita načina. Sve ovo je prikazano na slici iznad, koja sadrži analizu dva

ekvivalentna kanala. Gornji kanal na slici predstavlja ekvivalentni kanal koji se često može sresti u klasičnim komunikacionim kanalima. Šema kanala sadrži enkoder, dekoder, interliver, deinterliver i kanal sa šumom. Poruka koja se prenosi ovim kanalom, izložena je uticaju šuma koji se ispoljava u vidu paketskih grešaka u sadržaju poruke. Obzirom da se radi o većim paketskim greškama na predajnoj strani, primenjen je Rid-Solomonov kod i interliver čiji je zadatak da rasprši simbole kodovane poruke, a zatim se poruka šalje preko kanala. U toku prenosa poruka je izložena uticaju šuma koji utiskuje u sadržaj poruke paketske greške. Kada poruka stigne na prijemnu stranu, primenjuje se deinterliver koji simbole kodovane poruke vraća na iste indeksne pozicije koje su bile nakon kodovanja na predajnoj strani. Paketske greške koje su nastale sada su razbijene i ujednačeno raspoređene preko celog sadržaja poruke. U poslednjoj fazi, primenom Rid-Solomon dekodera koriguju se sve nastale greške.

Donji kanal na slici predstavlja biometrijski kanal sa prisustvom šuma. Šema kanala sadrži enkoder, dekoder, interliver i komunikacioni kanal. U ovom kanalu poruka je ključ K koji se prenosi preko šumnog kanala koji je indukovao sa dva irisa. Šum koji je prisutan u kanalu, ispoljava se preko paketskih grešaka. U cilju postizanja visokih performansi na strani dekodera, primenjujemo interliver nad oba irisa (jedan na predajnoj, a drugi na prijemnoj strani) koja zajedno indukuju komunikacioni kanal sa šumom. Zatim, ključ K kodujemo sa Rid-Solomonovim enkoderom i šaljemo kroz takav komunikacioni kanal nad kojim sada imamo kontrolu za određen opseg vrednosti paketskih grešaka. Poruka na prijemu ili ključ K se dekoduje Rid-Solomonovim dekoderom i dobijena poruka predstavlja ključ K' koji je jednak ključu K na predajnoj strani.

Možemo sad da zaključimo da smo analizom ekvivalentnog kanala sami modelovali ovaj novi kanal, preko koga ostvarujemo prenos poruka sa maksimalnim kapacitetom. Ovaj kanal ima osobine binarnog simetričnog kanala sa šumom i na ovaj način smo potvrdili Šenonovu teoremu, koja je bila veliko iznenađenje u njegovom najpoznatijem radu iz 1948. godine, u kome se tvrdi da ako izaberemo odgovarajući način kodovanja informacije za zadati kanal, može se postići po želji mala greška dekodovanja, odnosno prenosa informacija.

Uloga zaštitnog koda i interlivera

Greške koje nastaju nije moguće u potpunosti eliminisati, ali ih je moguće znatno smanjiti. U zavisnosti od stepena integracije, verovatnoća nastanka greške je veća ili manja. Sa aspekta projektovanja sistema, odnosno za postizanje visoke pouzdanosti u radu sistema koriste se tri osnovna pristupa:

- izbegavanje grešaka;
- ispravljanje grešaka;
- smanjenje grešaka;

Izbegavanje grešaka podrazumeva sprečavanje nastanka grešaka nastalih na hardverskoj ili softverskoj platformi. Ovakve tehnike se odnose na izbor kvalitetnog hardvera (iris kamera) koji učestvuju u procesu akvizicije slike irisa, kao i adekvatan izbor algoritama koji učestvuju u obradi signala (kodovanje irisa). Izbegavanje grešaka podrazumeva i faze u kojima se vrše testiranja gde se preispituju pojedine komponente sistema.

Ispravljanje grešaka je tehnika koja ne sprečava nastanak greške. Ova tehnika uspešno detektuje i ispravlja nastale greške. Možemo da kažemo da ova tehnika ima mogućnost transformacione invertibilnosti nad podatkom sa greškom, u cilju dobijanja istog podatka bez greške. Jedna od važnijih komponenti ovog sistema su zaštitni kodovi koje primenjujemo nad podatkom (ključ K), koji je u toku prenosa izložen greškama. Na ovaj način je omogućen prenos u kritičnim situacijama za zadati stepen tolerancije.

U projektovanju sistema korišćeni je Rid-Solomon kod. Ovaj kod ispravlja greške na nivou simbola, a ne na nivou bitova, pogodni su za ispravljanje koncentrisanih paketskih (engl. *burst*) grešaka. Oni imaju široku primenu u sistemima digitalnih komunikacija i sistemima za čuvanje podataka. Rid-Solomon kodovi su podgrupa BCH kodova i klasa linearnih, nebinarnih cikličnih kodova. Rid-Solomon kod je blok kod koji se predstavlja kao $RS(n, k, t)$ preko polja Galoa $GF(2^n)$.

Rid-Solomon kod uvodi veliku redundantnu informaciju (preko polinomske strukture) i tako postiže visoke performanse za detekciju i ispravljanje grešaka. Da bi se ova cena platila neophodno je koristiti samo konzistentnu informaciju (jedinstvene karakteristike). Iz ovog razloga su urađene rigorozne informacione

analize nad teksturom irisa, sa ciljem izbora upravo takvih segmenata koji će da sadrže većinom konzistentnu informaciju.

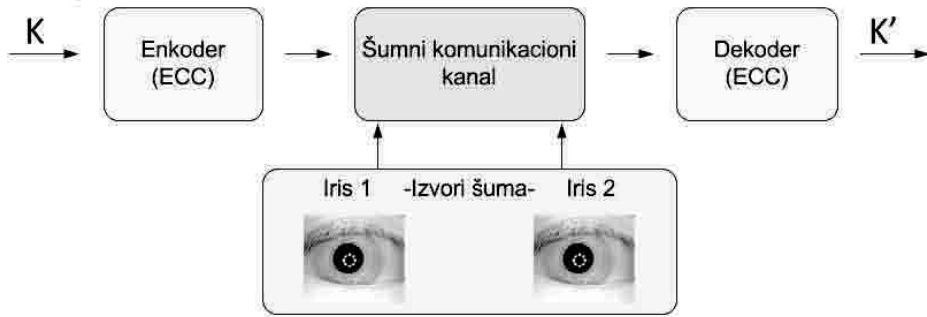
Smanjenje grešaka je uglavnom jedna od tehnika koja utiče na smanjenje nastalih grešaka koje nije moguće ispraviti. U našem slučajnu izabrali smo za ovu vrstu tehnike interliver. Interliveri koriste metodu preplitanja vrednosti u iris kodu. Metod za preplitanje može da bude raznovrsan. Mi koristimo slučaju permutaciju koja je zadata sa nekom inicijalnom vrednošću koja predstavlja početno stanje pesudogeneratora. Ovakvi interliveri uvode element slučajnosti u neki dizajn koda bez povećanja kompleksnosti, pretpostavljajući korišćenje iterativnog dekodovanja dva interlivovana koda.

Motiv za uvođenje zaštitnih kodova i interlivera

Primena zaštitnih kodova i interlivera nad irisom je stavila biometriju u paralelu sa klasičnim komunikacionim kanalima. Cilj je bio pronaći ekvivalentni kanal koji ima zajedničke osobine sa biometrijskim komunikacionim kanalom. Pronalazak ekvivalentnog kanala, davao bi garantovano dobre performanse celokupnog sistema što je krajnji cilj.

Na slici ispod biometrija je poistovećena sa klasičnim komunikacionim kanalom sa prisustvom šuma. U ovom primeru dva irisa indukuju biometrijski komunikacioni kanal preko koga imamo cilj da prenesemo ključ K . Vrednosti greške u ovom kanalu variraju u zavisnosti od toga da li dva irisa pripadaju jednoj osobi ili su od različitih osoba. U ovom apstraktnom modelu biometrijskog kanala, naš zadatak je da ključ K prenesemo preko kanala bez grešaka. Tačnije, cilj je projektovati komunikacioni kanal čiji je maksimalni kapacitet definisan izrazom:

$$C = \max I(K; K') = 1$$



Biometrijski komunikacioni kanal sa šumom

Razlozi za nastanak ovakvog motiva obrazložen je kroz sledeće činjenice:

Osnovni problem kod biometrijskih podataka je varijabilnost koja je prisutna svaki put u podacima nakon akvizicije biometrijskih podataka. Drugim rečima, svaki novi biometrijski podatak generisan na istom biometrijskom izvoru se uvek razlikuje 10% - 20% od svih prethodnih, dok se biometrijski podaci generisani na različitim biometrijskim izvorima razlikuju od 40% do 60%.

Kao odgovor na pomenuti problem varijabilnosti u biometriji irisa, nastala je ideja o ekvivalenciji sa drugim komunikacionim kanalima. Ova ideja je uvela primenu zaštitnih kodova za ispravljanje grešaka (engl. *error correcting code*), a tehnika je preuzeta iz oblasti komunikacija.

Kapacitet koda za ispravljanje grešaka u ovom slučaju performanse dekodera, moraju biti takve da odvajaju originalnog korisnika od lažnog. Na primer, biometrijski kanal indukovan sa dva iris koda sa istog biometrijskog izvora, mogu biti u stanju da prenesu poruku bez grešaka u našem slučaju ključ K , dok kanal indukovan sa dva iris kod sa različitih biometrijskih izvora, ne sme da prenese ključ K , čak šta više poželjno je da ključ K bude uništen, tako da je uzajamna informacija $I(K; K') = 0$.

Postizanjem pomenutih performansi, uspeh bi predstavili sa rezultatom $K == K'$, dok se K' koristi kao kriptološki ključ.

Pravljenjem kompromisa između ukupne količine informacije H iris koda i njihove uzajamne informacije I između istih i različitih kodova irisa, dobijamo okvire za maksimalnu veličinu ključa K .

Interliver je tehnika koja se koristi za raspršivanje paketskih grešaka. Analizom je utvrđeno da iris kod sadrži paketske greške. Ova tehnika se kod klasičnih komunikacionih kanala koristi za podizanje performansi zaštitnih kodova ili kodova za ispravljanje grešaka.

Sada možemo da zaključimo da sve ove činjenice nezaustavno utiču na stvaranje jedne nove oblasti, koja opravdano nosi ime biometrijska kriptografija, a sve to zahvaljujući postojanju teorije informacija i kodovanja.



REČNIK POJMOVA

A

Algoritam - opis za rešavanje nekog problema.

Arnost – veličina alfabeta. Za n -arno stablo: n , broj dece u svakom čvoru.

B

BSC – binarni simetrični kanal.

BSS – binarni simetrični izvor.

Blok kod – neprazan skup reči koje su iste dužine, predstavlja strukturu red.

G

Generator matrica – matrica koja se koristi za kodovanje poruke, građenje kodnih reči.

D

Dekodovanje – transformacija kodne reči u simbole poruke.

E

Entropija – mera za količinu neodređenosti.

I

Informacioni izvor – generator sekvence simbola koji predstavljaju poruke.

K

Kapacitet kanala - maksimalna prosečna količina informacija na izlazu kanala koja je jednaka količini informacije na ulazu.

Kodna reč - binarna sekvenca kojom je predstavljen simbol izvorne poruke.

Kriptoanaliza - umetnost i nauka o razbijanju šifrovanih poruka.

Komunikacioni kanal - formalizacija šta se dešava u prenetoj poruci između predajne i prijemne strane.

Kompletni kod - prefiksni kod čije kodno stablo ne sadrži nekorišćene listove.

Kodovanje - transformacija simbola poruke u kodne reči.

Kriptološki ključ - binarna sekvenca veće dužine, poseduje osobine slučajnih nizova. Koristi se kao početno stanje algoritma u fazi šifrovanja i dešifrovanja.

Kriptograf - lice koje dizajnira šifre.

Kriptoanalitičar - lice koje dizajnira metode za razbijanje šifara.

Kriptografija - nauka o pravljenju algoritama za šifrovanje.

Kriptoanaliza - nauka o razbijanju algoritama za šifrovanje.

L

Linearni kod - blokovski kod čija je struktura predstavljena sa vektorskim prostorom.

M

MAC - kod za proveru autentičnosti poruke.

Minimalna distanca - predstavlja minimalnu Hemingovu distancu (ne nula) između bilo koje dve različite kodne reči.

Minimalna distanca dekodovanja - okvir za dekodovanje u kojem se svaka primljena reč dekoduje u najbližu kodnu reč.

N

Nula kodna reč – kodna reč sačinjena od nula

O

One-time pad - perfektna šifra koja pripada grupi šifara sa simetričnim ključem. Ključ je generisan na slučajan način i ima jednaku dužinu kao i poruka. Ključ ne sme nikad da se ponavlja.

Očekivana dužina koda - očekivana vrednost za dužinu kodnih reči.

Jednosmerna funkcija - funkcija koja se lako izračunava u jednom smeru ali računanje u nazad je teško i nemoguće.

P

Poruka - podatak koji se razmenjuje preko komunikacionog kanala između predajne i prijemne strane. Može da bude kodovana i u šifrovanoj formi.

Perfektna tajnost – teorijski dokazan kriptografski mehanizam koji ne zavisi od računarske snage.

PKI - infrastruktura sa javnim ključevima. Sadrži bazu javnih ključeva preko kojih obezbeđuje autentifikaciju u srednjim i velikim mrežama.

PRNG - generator za generisanje pseudoslučajnih brojeva na osnovu inicijalnog stanja koje je određeno na slučajan način.

R

RSA - kriptografski sistem sa javnim ključem, patentiran od strane Rivesta, Šamira i Adlemana 1976. Sigurnost RSA zasniva se na složenosti faktORIZACIJE velikih brojeva. Ima široku primenu, implementiran je u PGP i SSL-u.

S

Stacionarni izvor – informacioni izvor čija statistika ne zavisi od vremena.

Sindrom – proizvod primljene reči i verifikacione matrice.

T

TRNG - generator slučajnih brojeva. Ovakve generatore je moguće konstruisati u prirodnim informacionim izvorima.

Težina reči – broj jedinica u binarnoj sekvenci.

F

FaktORIZACIJA - rastavljanje velikih prostih brojeva na činioce iz skupa prostih brojeva.

H

Hemingovo rastojanje - koristi se za određivanje razlike između dva niza jednake dužine.

Hafmanov kod – efikasan način kodovanja informacionog sadržaja u porukama, na osnovu poznatih verovatnoća simbola poruke.

C

CRC – (engl. *Cyclic Redudancy Check*). Kod široke primene, koristi se za detektovanje grešaka. Obično pre dešifrovanja, CRC se proverava.

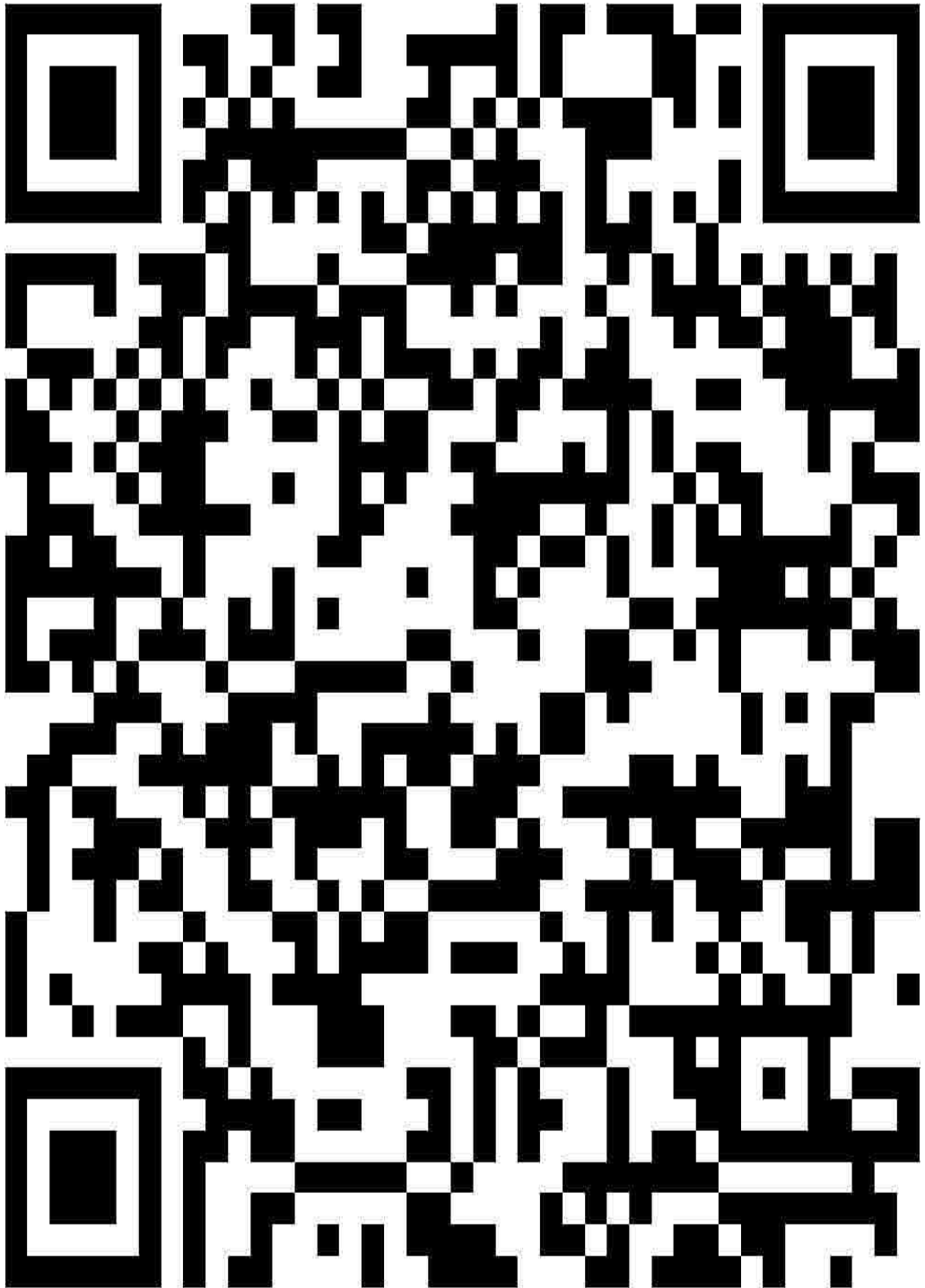
Ciklični kod - linerani kod, u kome svaki pomeraj kodne reči predstavlja takođe kodnu reč.

Š

Šifra - algoritam za šifrovanje. Mogu da postoje različiti tipovi šifara. Osnovne podele su na klasične i moderne, simetrične i asimetrične, apsolutno i praktično tajne.

Šifrat - rezultat funkcije za šifrovanje ili transformacija otvorene poruke koja je inverzna uz posedovanje pravog ključa.

Šifrovanje - proces transformacije otvorenog teksta u šifrat.



CIP - Каталогизација у публикацији -
Народна библиотека Србије, Београд

519.72(075.8)

004.22(075.8)

МИЛОСАВЉЕВИЋ, Милан, 1952-

Osnovi teorije informacija i kodovanja / Milan Milosavljević, Saša Adamović. - 3. izd. - Beograd : Univerzitet Singidunum, 2017 (Loznica : Mobid). - 238 str. : ilustr. ; 24 cm

Na vrhu nasl. str.: Fakultet za informatiku i računarstvo. - Tiraž 500. -
Rečnik pojmova: str. 233-237. - Bibliografija: str. 211.

ISBN 978-86-7912-610-8

1. Адамовић, Саша, 1985- [аутор]

а) Теорија информација б) Кодирање

COBISS.SR-ID 239783948

© 2017.

Sva prava zadržana. Nijedan deo ove publikacije ne može biti reprodukovан u bilo kom vidu i putem bilo kog medija, u delovima ili celini bez prethodne pismene saglasnosti izdavača.