Sveučilište u Splitu

Fakultet Elektrotehnike,
Strojarstva i Brodogradnje

TEORIJA INFORMACIJA I KODIRANJE

Laboratorijske vježbe – upute za rad i zadaci

Petar Šolić, Joško Radić

Sadržaj

Popis slika

Popis tablica

Uvod

Vježbe se održavaju svako drugi tjedan u sljedećim terminima:

- ▷ Ponedjeljak, grupa A u 08:00 (lab. B526)
- ▷ Srijeda, grupa B u 14:45 (lab. B525)

Prva vježba grupe A će se održati u ponedjeljak 13.10., a grupa B u srijedu 15.10.

Važne napomene:

- ▷ Prisustvovanje vježbama je obavezno
- ▷ Na početku svake vježbe održati će se kratki kolokvij.
- ▷ Tijekom odrađivanja vježbe piše se izvještaj koji se na kraju vježbe predaje. Predložak za izvještaj je potrebno donijeti na vježbu, a dostupan je na kraju teksta svake vježbe.

Vježba 1

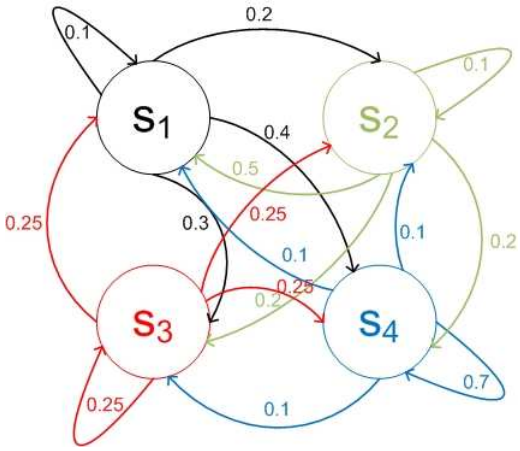
Markovljev model izvora informacije

Markovljev modela izvora informacije se sastoji od skupa stanja $\mathbb{S} = \{s_1, s_2, ..., s_M\}$, te matrice prijelaznih vjerojatnosti $P = \{p_{ij}\} = Pr\{S_n = s_j | S_{n-1} = s_i\}$ koja opisuje vjerojatnost prelaska iz nekog stanja s_i u neko drugo s_j ili isto stanje s_i. Kod Markovljevog modela izvora informacije, podrazumjeva se da je informacija nastala kao posljedica prijelaza modela iz jednog stanja u drugo (ili isto) stanje. Primjer matrice prijelaza koja ima 4 stanja:

$$P = \{p_{ij}\} = \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 \\ 0.5 & 0.1 & 0.2 & 0.2 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{bmatrix} \tag{1.1}$$

Matrica prijelaza može biti matrica prijelaza samo ako zadovoljava uvjet da je $\sum_j p_{ij} = 1$, jer bi u suprotnom značilo da je moguć prijelaz u neko drugo stanje koje nije definirano. Drugi način za opis Markovljevog modela izvora informacije je dijagram stanja, gdje se stanja standardno označavaju kružićima, a veze se ostvaruju strelicama. Iznad strelice koja povezuje stanja, standardno se piše vjerojatnost prijelaska između povezanih stanja. Primjer dijagrama stanja za matricu prijelaza (1.1) je prikazan na slici 3.1.

Tipičan primjer Markovljevog modela izvora informacije je tipkalo na telefonskom aparatu, tipkovnica računala, Moresov izvor informacije, itd. Kod navednih izvora informacije, zajedničko je da simbol koji taj izvor generira ovisi o simbolima koji su generirani prije njega. Takav tip izvora nazivaju se izvori informacije s memorijom. Ovakav model izvora informacije je zanimljiv



Slika 1.1: Dijagram stanja Markovljevog Modela temljen na matrici prijelaza (1.1). Različitim bojama su označeni prijelazi iz tih stanja.

zbog mogućnosti predviđanja, t.j. u situaciji kada je poznata matrica početnih vjerojatnosti (početna distribucija stanja) P_0, te matrica prijelaznih vjerojatnosti P, tada je moguće izračunati distribuciju stanja u idućem koraku promatranja, $P_1 = P_0P$, kao i distribuciju stanja u bilo kojem idućem koraku $P_n = P_{n-1}P = P_0P^n$.

Markovljevi modeli izvora informacije se mogu podijeliti na homogene i nehomogene, ergodične i neergodične, modele prvog ili višeg reda.

Homogen Markovljev model (stacionaran ili vremenski invarijatan) je u potpunosti zadan matricom P i matricom početnih vjerojatnosti P_0. Kod homogenog Markovljevog modela, distribucija stanja u n-tom trenutku ne ovisi o P_0, već samo o matrici P. To omogućava izračun stacionarnog stanja Markovljevog modela $P_s = P_sP$, odakle i dolazi naziv stacionarni ili vremenski invarijatan Markovljev model. Stacionarno stanje opisuje stanje Markovljevog modela izvora informacije, kod kojeg svaki idući prijelaz će rezultirati istom distibucijom stanja P_s.

Kod ergodičnog Markovljevog modela, matrica P je puna, što u bilo kojem trenutku omogućava prijelaz iz stanja s_i u neko drugo s_j ili isto stanje s_i.

Za Markovljev izvor čija je matrica prijelaznih vjerojatnosti zadana tako da stanje u idućem trenutku ovisi samo o trenutnom stanju tj. $p(s_j|s_{j-1}, s_{j-2}, ..., s_1) = p(s_j|s_{j-1})$, naziva se Markovljev model prvog reda. Tako je onda moguće definirati i Markovljev model kod kojih je trenutno stanje ovisno o nizu prethodnih stanja što upućuje na modele višeg reda.

Ukoliko postoji definirana matrica prijelaza P, te matrica početnih vjerojatnosti P_0, tada je moguće simulirati *šetnju* kroz Markovljev model. Naime, ukoliko se *šetnja* (slučajan dolazak u neko stanje) ponovi više puta, a rezultat uprosječi, tada je moguče ispratiti dinamiku konvergencije Markovljevog modela

u stacionarno stanje. Takav postupak velikog broja ponavljanja eksperimenata u skladu s matricom prijelaza se naziva Monte Carlo simulacija Markovljevog modela.

1.1 Zadaci

Prodaja čokolade - zadatak, modelirati Markovljevim modelom opisani proces koristći programski paket matlab

Neka tvrtka A proizvodi čokoladu i posjeduje 30% cjelokupnog tržišta čokolade. Isto tržište čokolade broji 500 000 kupaca. S ciljem povećanja prodaje, tvrtka A odluči zaposliti tvrtku za istraživanje tržišta kako bi predvidili efekt agresivne kampanje reklama.

Nakon istraživanja, tvrtka za istraživanje tržišta donese sljedeće rezultate:

- U svakom tjednu reklamiranja, tko ne kupuje čokoladu tvrtke A, krenuti će s njenom kupnjom s vjerojatnošću 0.6
- U svakom tjednu reklamiranja, tko kupuje čokoladu tvrtke A, prestati će je kupovati s vjerojatnošću 0.1.

Zadatak 1.1. Koliko je vjerojatnost da će netko tko kupuje čokoladu tvrtke A početi kupovati čokoladu tvrtke A nakon prvog tjedna reklamiranja, te prestati kupovati nakon drugog tjedna reklamiranja?

Zadatak 1.2. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon 3 tjedna reklamiranja?

Zadatak 1.3. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon 100 tjedana reklamiranja?

Zadatak 1.4. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon n tjedana reklamiranja? Izračunati stacionarno stanje Markovljevog modela.

Zadatak 1.5. Koliko će najviše kupaca kupovati čokoladu tvrtke A?

1.2 Izvještaj

Ime i prezime: _____

Broj indeksa: _____

Datum: _____

1. Koliko je vjerojatnost da će netko tko kupuje čokoladu tvrtke A početi kupovati čokoladu tvrtke A nakon prvog tjedna reklamiranja, te prestati kupovati nakon drugog tjedna reklamiranja?

2. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon 3 tjedna reklamiranja?

3. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon 100 tjedana reklamiranja?

4. Koliko će kupaca čokolade tvrtke A kupovati čokoladu tvrtke A nakon n tjedana reklamiranja? Izračunati stacionarno stanje Markovljevog modela.

5. Koliko će najviše kupaca kupovati čokoladu tvrtke A?

Vježba 2

Sadržaj informacije

Sadržaj informacije nekog simbola usko je povezan s neodređenošću, odnosno vjerojatnošću njegovog pojavljivanja. Veća vjerojatnost pojavljivanja znači manju neodređenost njegova pojavljivanja ali i manji sadržaj informacije. Stoga vrijedi da je vlastiti sadržaj informacije

$$I(x_i) = -\mathrm{ld}(p(x_i)) \quad bita \tag{2.1}$$

ovisan isključivo o vjerojatnosti njegova pojavljivanja, tako da je $I(x_j) > I(x_i)$, ako je $p(x_j) < p(x_i)$.

No, vlastiti sadržaj informacije nije mjera ukupne "informiranosti izvora". Da bi promatrali "ukupnu informiranost" nekog izvora koji može generirati ukupno n simbola, koristi se mjera srednjeg sadržaja informacije

$$H(X) = -\sum_{i}^{n} p(x_i)\mathrm{ld}(p(x_i)) \quad bita/simbolu \tag{2.2}$$

Posebice je zanimljivo promatrati ovisne izvore informacija. Najčešća primjena ovisnih izvora informacije je u promatranju izvora informacije kao prijemnika koji je ovisan o izvoru informacije odašiljača. Ako je izvor informacije odašiljača X, a prijemnika Y, možemo promatrati:

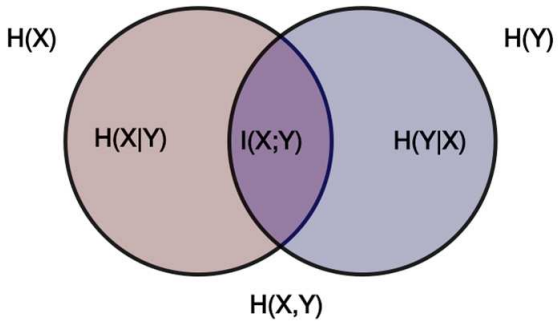
- ▷ Uvjetne sadržaje informacije

$$H(X|Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\mathrm{ld}(p(x|y)) \quad bita/simbolu \tag{2.3}$$

$$H(Y|X) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\mathrm{ld}(p(y|x)) \quad bita/simbolu \tag{2.4}$$

- ▷ Združeni sadržaj informacije

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\mathrm{ld}(p(x,y)) \quad bita/simbolu \tag{2.5}$$



Slika 2.1: Vennov dijagram prikaza odnosa između sadržaja informacije $H(X)$, $H(Y)$

- Uzajamni sadržaj informacije I(X;Y), koji se prema Slici 3.1, može izračunati kao:

$$I(X;Y) = H(X)+H(Y)-H(X,Y) = H(X)-H(X|Y) = H(Y)-H(Y|X) \tag{2.6}$$

2.0.1 Primjer: Bacanje novčića

Neka kutija X (izvor informacije-predajnik) sadrži 2 novčića, od kojih je jedan pravilan (ima i glavu i pismo), a drugi nepravilan (ima 2 glave). Predajnik nasumično uzima novčić i baca ga 2 puta, a prijemnik bilježi broj postignutih glava. Koliki je preneseni sadržaj informacije dobiven za dane eksperimente?

Rješenje

Združeni sadržaj informacije se može dobiti kao $I(X;Y) = H(X)-H(X|Y)$, gdje sa X označavamo izvor informacije-predajnik, dok sa Y označavamo izvor informacije-prijemnik.

Kako se izvor-odašiljač X sastoji od 2 simbola t.j. novčića, vjerojatnost da će neki od njih biti izvučen iznosi 0.5, što omogučava izračun $H(X)$

$$H(X) = -\sum_{i=1}^{2} p(x_i)\mathrm{ld}(p(x_i)) = -(-0.5-0.5) = 1\ bit/simbolu \tag{2.7}$$

gdje recimo sa x_1 označimo pravilan, a x_2 nepravilan novčić.

Y, t.j. prijemnik može iz 2 bacanja novčića zabilježiti 0, 1 ili 2 "glave". Tako da ćemo sa $Y = 0$, $Y = 1$ i $Y = 2$ označiti 0, 1 ili 2 postignute "glave".

Da bi dobili $H(X|Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\text{ld}p(x|y)$, potrebno je izračunati vjerojatnosti $p(x,y)$ i $p(x|y)$.

Iz Bayesovog teorema vrijedi da je $p(x|y) = p(x,y)/p(y)$. Temeljem odnosa X i Y iz zadatka, možemo pisati da je: $p(x,y) = \begin{bmatrix} \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ 0 & 0 & \frac{1}{2} \end{bmatrix}$

gdje je $\sum_x p(x,y) = p(y)$, t.j. $p(Y=0) = \frac{1}{8}$, $p(Y=1) = \frac{1}{4}$ i $p(Y=2) = \frac{5}{8}$. Tada možemo izračunati $p(x|y) = \begin{bmatrix} 1 & 1 & \frac{1}{5} \\ 0 & 0 & \frac{4}{5} \end{bmatrix}$ i vrijedi da je

$$H(X|Y) = -(\frac{1}{8}\text{ld }1+\frac{1}{4}\text{ld }1+\frac{1}{8}\text{ld }\frac{1}{5}+0\text{ld }0+0\text{ld }0+\frac{1}{2}\text{ld }\frac{4}{5}) = 0.4512 \quad bita/simbolu \tag{2.8}$$

tako da je:

$$I(X;Y) = H(X) - H(X|Y) = 1 - 0.4512 = 0.5488 \quad bita/simbolu \tag{2.9}$$

2.1 Zadaci: Prijenos abecede

Zadatak 2.1. Komunikacijskim sustavom prenose se poruke pomoću abecede koja se sastoji od 8 suglasnika i 8 samoglasnika. Svi su znakovi jednakovjerojatni i nema statističke veze između njih. Suglasnici se prenose pravilno, a samoglasnici samo u polovini slučajeva. U drugoj polovini slučajeva nastaju pogreške, pri kojima svaki samoglasnik može prijeći u bilo koji drugi s jednakom vjerojatnosšću.

- ▷ Nacrtaj Vennov dijagram za dani komunikacijski model i objasni elemente.
- ▷ Korištenjem programskog paketa Matlab izračunaj transinformaciju (preneseni sadržaj informacije), te skiciraj postupak rješavanja zadatka.
- ▷ Što bi morali napraviti da bi u danom sustavu povećali preneseni sadržaj informacije?

2.2 Izvještaj

Ime i prezime: _____

Broj indeksa: _____

Datum: _____

1. Nacrtaj Vennov dijagram za dani komunikacijski model i objasni elemente.

2. Korištenjem programskog paketa Matlab izračunaj transinformaciju (preneseni sadržaj informacije), te skiciraj postupak rješavanja zadatka.

3. Što bi morali napraviti da bi u danom sustavu povećali preneseni sadržaj informacije?

Vježba 3

Zaštitno kodiranje - Tajni ključ

Kod zaštite podataka temeljene na tajnom ključu za ispravno šifriranje i deširiranje potrebno je da predajna strana i prijemna strana posjeduju dijeljenu tajnu (ključ). Primjeri takvih sustava zaštita su:

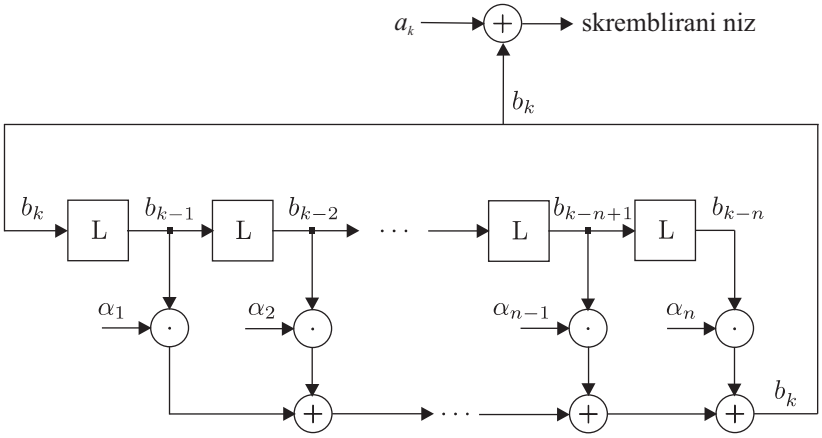
- ▷ Cezarova šifra
- ▷ Vignéreova šifra
- ▷ Šifiriranje permutiranjem i substitucijom
- ▷ Šifriranje korištenjem više abeceda (primjer: *Enigma* stroj)

Najveći nedostatak takvih sustava zaštite su činjenice da je ključ potrebno izmjeniti s prijemnikom i to putem nekog sigurnog kanala, te da je nešifrirani tekst na neki način koreliran sa šifriranim tekstom, što ga čini ranjivim na pokušaje dešifriranja.

U svrhu zaštite informacije, zbog jednostavnosti izrade elektroničkog sklopa, dugog perioda ponavljanja i uniformnosti izlaznog niza posebice je zanimljivo korištenje Linearnog posmičnog registra s povratnim vezama (*eng. Linear Feedback Shift Register - LFSR*). Postupak šifriranja korištenjem LFSR-a je jedan od načina skrembliranja ulaznog niza podataka. Osnovni elementi za izgradnju LFSR-a su registri, te operacije množenja i zbrajanja po modulu 2. Svaki LFSR je opisan prijenosnom funkcijom, čiji stupanj govori o broju registara koji se koriste. Generički primjer takovog sustava za zaštitu informacije je prikazan na slici 3.1, a prijenosna funkcija n-tog reda koja opisuje takav LFSR je

$$h(L) = 1 + \alpha_1 L^1 + \alpha_2 L^2 + ... + \alpha_n L^n \tag{3.1}$$

gdje vrijedi da $\alpha_i \in \{0, 1\}$ označava postojanje veze koja koja zbraja vrijednosti u registrima, a članovi polinoma L^r označavaju registre, tj. elemente za



Slika 3.1: Primjer korištenja LFSR-a za zaštitu informacije

kašnjenje - r označava indeks vremenskog trenutka za koji je ulazna vrijednost kašnjena. LFSR generira niz na način da je umnožak $h(L)$ i izlaza skremblera b_k jednak nuli, tako da je

$$b_k + \alpha_1 L^1 b_k + \alpha_2 L^2 b_k + ... + \alpha_n L^n b_k = 0 \tag{3.2}$$

kako vrijedi da L^r opisuje za koju vrijednost indeksa r je ulazna vrijednost kašnjena, tada je

$$b_k = \alpha_1 b_{k-1} + \alpha_2 b_{k-2} + ... + \alpha_n b_{k-n} \tag{3.3}$$

gdje su b_{k-i} vrijednosti u registrima skremblera.

Šifrirani izlaz biti će opisan relacijom:

$$s_k = a_k \oplus b_k \tag{3.4}$$

gdje je s_k šifrirani izlaz, a_k je ulaz (nešifrirani podaci) dok je b_k izlaz iz LFSR-a.

Zbog konačnog broja registara i veza koji ga definiraju, izlazni niz skremblera je periodičan, a period ovisi o skrativosti polinoma $h(L)$. Ukoliko je polinom $h(L)$ neskrativ, tada će period tog skremblera biti maksimalan i iznosti će $2^M - 1$, gdje je M stupanj polinoma $h(L)$. U slučaju polinoma $h(L)$ koji je skrativ, period će biti manji od $2^M - 1$.

Poznati sustavi koji koriste LFSR za zaštitu su A5/1 i A5/2 algoritmi kod GSM mobilnih telefona, E0 algoritam kod Bluetooth tehnologije.

Naprednija verzije zaštita su nelinearni posmični registar sa povratnim vezana (NLFSR) (niz kaskadno povezanih LFSRova), IBM-ov Data Encryption Standard (DES) sustav zaštite, kojima se postiže veća periodičnost izlaznog niza, a time i veća otpornost na napade.

3.1 Zadatak: Deskrembliranje ulaznog niza

Zadatak 3.1. Ako je zadan poptun skremblirani ulazni niz čiji su bitovi ASCII 7-bitno kodirani (upisani u Matlab datoteci *descr_ in.mat*), te 5 bitova ulaznog nešifriranog niza 11101, izračunati prijenosnu funkciju skremblera. Poznato je da je zadani LFSR opisan polinomom 5-og stupnja.

▷ Skicirati deskrembler i dešifrirati ulazni niz.

3.2 Izvještaj

Ime i prezime: _____

Broj indeksa: _____

Datum: _____

1. Skicirati skrembler i dešifrirati ulazni niz.

Vježba 4

Zaštitno kodiranje - Javni ključ

Zaštita podataka temeljena na javnom ključu podrazumjeva korištenje 2 odvojena ključa, od kojih je jedan tajan, a drugi javan i dostupan svima. Osnovni koncept takve zaštite je u tome da je jedan ključ komplement drugog, tj. ako se jednim ključem poruka zaključa, ta se šifrirana poruka može otključati drugim ključem. Korištenje takvih ključeva omogućava šifriranje korištenjem javnog ključa primatelja, dok će primatelj dešifrirati poruku korištenjem svog tajnog ključa. U tome slučaju je potrebno izmjeniti javne ključeve entiteta (osoba) koje sudjeluju u komunikaciji.

U takvom konceptu izmjene javnih ključeva moguće je izvesti *eng. Man In The Middle (MITM)* napad, kod kojeg postoji osoba koja presluškuje komunikaciju. Napadač se nalazi između pošiljatelja i primatelja i temeljem svojeg javnog i tajnog ključa dolazi do poruke, tako da se pošiljatelju predstavi kao primatelj, a primatelju kao pošiljatelj, tj. presretne njihove javne ključeve i pošalje svoj. Tada pošiljatelj i primatelj raspolažu s javnim ključem napadača koji kontrolira njihovu komunikaciju.

Koristeći isti koncept javnog i tajnog ključa, moguće je provesti digitalne potpise kojima se osigurava autentičnost pošiljatelja poruke. Ukoliko pošiljatelj "potpiše" poruku svojim tajnim ključem, tu poruku može bilo tko pročitati, no poruku je mogao poslati samo onaj tko posjeduje pravi tajni ključ.

Najpoznatiji algoritam koji koristi javni i tajni ključ je RSA. Kod RSA algoritma, za šifriranje i dešifiranje, potrebno je poštivati korake algoritma izrade javnog i tajnog ključa:

1. odaberi 2 velika prosta broja p i q
2. izračunaj $n = pq$
3. odredi iznos Eulerove fukcije $\varphi(n) = (p-1)(q-1)$

4. odaberi javni ključ e, takav da: $1 < e < \varphi(n)$ i $\text{nzd}(e, \varphi(n))$

5. odredi tajni ključ d, takav da: $ed = 1 \text{mod } \varphi(n)$

Šifriranje poruke m je tada

$$c = m^e \text{mod } n \tag{4.1}$$

dok se dešifrirana poruka dobije kao

$$m = c^d \text{mod } n \tag{4.2}$$

Iako su navedeni ključi (e i d) matematički ovisni, proračun jednog ključa iz drugog ukoliko su veliki (npr. 2048 bit-ni), predstavlja težak posao za računala. Naime problem određivanja ključa d se svodi na faktorizaciju broja n (na p i q), za koji ne postoji efikasan algoritam.

4.0.1 Gnu Privacy Assistant (GPA)

GPA je software za administraciju i izradu RSA (2048 bit-nih) ključeva te zaštitu i potpis datoteka korištenjem napravljenih ključeva. GPA je povezan sa PKS (*Public Key Server*) servima (npr. *http://zimmermann.mayfirst.org/* ili *keyserver.ubuntu.com* kojem je moguće poslati javni ključ i tako se dogovoriti o sigurnoj komunikaciji između dvije ili više osoba.

4.1 Zadatak: Šifriranje i potpis koristeći RSA algoritam

Zadatak 4.1. Za zadane $p = 7$ i $q = 11$:

- ▷ odredi par ključeva e i d
- ▷ Šifriraj poruku $m = 4$, te dešifiranjem potvrdi ispravnost algoritma.
- ▷ Koristeći software Gnu Privacy Assistant (GPA), izmjeniti ključeve s kolegom, te izmjenom šifiranih datoteka provjeriti potpis i dešifrirati primljenu poruku.

4.2 Izvještaj

Ime i prezime: _____
Broj indeksa: _____
Datum: _____

Za zadane $p = 7$ i $q = 11$:

1. Odredi par ključeva e i d

2. Šifriraj poruku $m = 4$, te dešifiranjem potvrdi ispravnost algoritma.

3. Koristeći software Gnu Privacy Assistant (GPA), izmjeniti ključeve s kolegom, te izmjenom šifiranih datoteka provjeriti potpis i dešifrirati primljenu poruku.

Vježba 5

Blok kodovi - Hammingov kod

Jedno od temeljnih pitanja u uspostavi komunikacijskog sustava jest kvaliteta usluge s čime je direktno povezana pouzdanost prijenosa, odnosno vjerojatnost nastanka pogreške. Shannon je pokazao (*Shannonov teorem*) da postoji kôd kojim se može postići prijenos sa po volji malom vjerojatnošću pogreške preko komunikacijskog kanala sa šumom ako je brzina prijenosa R manja od kapaciteta komunikacijskog kanala C. Međutim, teorem samo kaže da postoji kôd, međutim ne govori ništa o načinu konstrukcije koda. Dakle jasno je da je jedini način kojim se sustav može približiti Shannonovoj granici upotreba redundantnog kodiranja. Postoje tri osnovne grupe kodova:

- ▷ Blok kodovi
- ▷ Konvolucijski kodovi
- ▷ Trellis kodovi

U ovoj vježbi biti će ukratko opisani *Hammingovi kodovi* koji spadaju u grupu blok kodova. Osnovna ideja kodiranja je dodavanje redundantnih simbola (zalihosnih) osnovnoj puruci, te na taj način prekodirati poruku da je u prijamniku moguće detektirati grešku i korigirati je. Blok kodovima i konvolucijskim kodovima se na taj način ruši učinkovitost sustava jer se osim poruke prenose i redundantni simboli koji ne nose informaciju. Trellis kodiranjem se ne ruši učinkovitost jer se kodiranje obavlja u prostoru signala. Mogućnost korekcije ovisi o broju kontrolnih simbola, ali i o kodu, odnosno o načinu na koji su redundantni simboli iskorišteni.

Blok kodovi se generiraju tako da se svakoj ulaznoj poruci od k informacijskih simbola (najčešće bitova) doda c kontrolnih simbola (bitova). Na taj način se konstruira (n, k) blok kod gdje je $n = k + c$. Način na koji kôd ima

mogućnost ispravljanja grešaka je najjednostavnije interpretirati geometrijskim putem. Prije toga je potrebno uvesti pojam *Hammingove težine* i *Hammingove distance*. Hammingova težina kodne riječi $w(s_i)$ je definirana kao broj jedinica u toj kodnoj riječi. Hammingova distanca između kodnih riječi $d_H(s_i, s_j)$ je definirana kao broj mjesta u kojem se te dvije kodne riječi razlikuju. Jednostavno je pokazati da se Hammingova distanca može izračunati poću Hammingovih težina:

$$d_H(s_i, s_j) = w(s_i \oplus s_j) \tag{5.1}$$

gdje je $\oplus$ oznaka za zbrajanje po modulu 2.

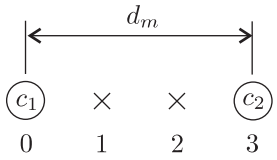
Primjer 5.1. Neka su zadane kodna riječ $s_1 = 10110010$ i $s_2 = 11010101$. Treba izračunati Hammingove težine i Hammingovu distancu.

Rješenje. Hammingove težine su: $w(s_1) = 4$, odnosno $w(s_2) = 5$, a Hammingova distanca $d_H(s_1, s_2) = w(s_1 \oplus s_2) = w(10110010 \oplus 11010101) = w(01100111) = 5$.

Na sl. 5.1 je shematski prikaz dviju kodnih riječi (proizvoljne duljine) s Hammingovom distancom $d_H(c_1, c_2) = 3$. Ako na kodnoj riječi c_1 nastane jedna greška njezin novi položaj će biti namjestu 1. Dakle u smislu Hammingove distance ta novonastala kodna riječ je još uvijek bliža izvornoj kodnoj riječi c_1 nego kodnoj riječi c_2. Dakle da bi se ispravila greška, kodnu riječ na kojoj je detektirana greška treba dekodirati kao kodnu riječ koja joj je najbliža u smislu Hammingove distance. U praksi kod ima mnogo više od dvije kodne riječi, međutim princip ispravljanja grešaka je isti. Dakle kod koji ima minimalnu Hammingovu distancu (u daljnjem tekstu samo distanca) jednaku d_m može korigirati $n_c = \frac{1}{2}(d_m - 1)$ bitova. Minimalna Hammingova distanca je najmanji broj koji se dobije ako se ispitaju Hammingove distance svake kodne riječi kôda sa svakom.

Hamming-ovi blok kodovi

Specijalni slučaj blok kodova su $(2^{n-k}, k)$ tzv. Hamming-ovi kodovi. To su linearni blok kodovi s distancom $d_m = 3$ koji mogu korigirati jednu grešku. Broj korekcija (n_c) jednak je $n_c = \frac{1}{2}(3 - 1) = 1$. Hammingova distanca za neki kod



Slika 5.1: Geometijsko predstavljanje dviju kodnih riječi Hammingove distance $d_H = 3$

općenito se svodi, kao što je već rečeno, na određivanje minimalne udaljenosti između svih izlaznih n-torki što često nije jednostavan zadatak. Jednostavnije rješenje za linearni (n, k) kod je preko minimalne Hammingove težine. Tako se određivanje Hammingove distance, a time i broja mogućih korekcija pogrešaka, svodi na traženje kodne riječi s najmanje jedinica (osim kodne riječi koja sadrži sve nule) među svim kodnim riječima kôda. Hammingov kod se može generirati pomoću generator matrice G koja za $(7, 4)$ kod može izgledati:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \tag{5.2}$$

Ovo je generator matrica $\mathbf{G}$ tzv. standardnog oblika (prvih k stupca čine jediničnu matricu) pomoću koje se generira sistematski linearni kod. Taj kod ima pri postupku dekodiranja korisno svojstvo: informacijski simboli se nalaze na prvih k mjesta u kodnoj riječi.

Poruka $\mathbf{v}_i$ (matrica dimenzije $k \times 1$) se kodira u kodnu riječ $\mathbf{x}_i$ (dimenzije $n \times 1$) množenjem transponiranog vektora poruke s generator matricom:

$$\mathbf{x}_i^T = \mathbf{v}_i^T \mathbf{G}, \qquad (mod\ 2) \tag{5.3}$$

Ovdje treba napomenuti da se kod zbrajanja elemenata u matričnom množenju zbraja po modulu 2. Za sistematski kod lako je izračunati matricu pariteta H na račun strukture matrice $\mathbf{G}$:

$$\mathbf{G} = \left[\mathbf{I} \vdots \mathbf{D}\right] \qquad \mathbf{H} = \left[-\mathbf{D}^T \vdots \mathbf{I}\right] \quad \Rightarrow \quad \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

Dekodiranje se vrši izračunavanjem vektora $\mathbf{s} = \mathbf{Hy}$ koji se naziva sindrom ili korektor, a $\mathbf{y}$ je primljeni vektor. U slučaju kada je $\mathbf{s} = \mathbf{0}$, odnosno nul vektor, tijekom prijenosa nije nastupila jednostruka greška. Ako tijekom prijenosa nastupi jedna greška, pomoću sindroma, koji je sad različit od nule, može se jednoznačno odrediti i ispraviti. Važno je napomenuti da uz broja grešaka veći od jedan dolazi do naglog pada kvalitete prijenosa pa je bolje u tom slučaju ne koristiti ovakvu kontrolu grešaka.

5.1 Zadaci

Zadatak 5.1.

Za kod definiran generator matricom:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{5.5}$$

ispisati sve kodne riječi, i odrediti hammingovu distancu.

Zadatak 5.2. Odabrati proizvoljnu kodnu riječ duljine četiri bita, i kodirati generator matricom definiranom u prvom zadatku.

Zadatak 5.3. Izračunati matricu pariteta **H**, i dekodirati kodiranu riječ iz prvog zadatka.

Zadatak 5.4. Proizvoljno invertirati bilo koji bit (simulacija greške u prijenosu) kodirane riječi (iz prvog zadatka), i takvu kodnu riječ dekodirati.

Zadatak 5.5. Koji stupac matrice pariteta je isti kao i sindrom.

5.2 Izvještaj

Ime i prezime: _____

Broj indeksa: _____

Datum: _____

1. Za kod definiran generator matricom:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{5.6}$$

   ispisati sve kodne riječi, i odrediti hammingovu distancu.

2. Odabrati proizvoljnu kodnu riječ duljine četiri bita, i kodirati generator matricom definiranom u prvom zadatku.

3. Izračunati matricu pariteta **H**, i dekodirati kodiranu riječ iz prvog zadatka.

4. Proizvoljno invertirati bilo koji bit (simulacija greške u prijenosu) kodirane riječi (iz prvog zadatka), i takvu kodnu riječ dekodirati.

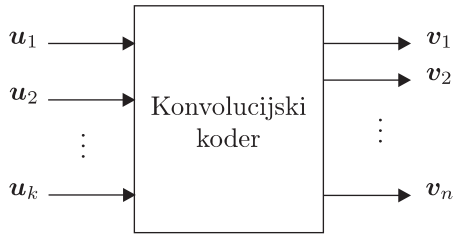
5. Koji stupac matrice pariteta je isti kao i sindrom.

Vježba 6

Konvolucijski kodovi

Za razliku od blok kodova koji se realiziraju pomoću sustava bez memorije, konvolucijski kodovi (definirani polinomima $\boldsymbol{g}_j^{(i)}$) se generiraju pomoću sustava s memorijom. Kodna riječ (n-torka) na izlazu konvolucijskog kodera ne ovisi samo o trenutnoj ulaznoj poruci (k-torci) već i o prethodnim porukama koje su pohranjene u memoriji kodera. Standardna oznaka za binarni konvolucijski kôd je (n, k, m) gdje je n broj bitova (n-torke) koji se generiraju na izlazu kodera ako se na ulaz dovede k bitova, a m je *veličina memorije* odnosno broj k-torki prethodno pohranjenih u memoriji. Prednost konvolucijskih kodova je u tome što je za male iznose n i k (do 4) implementacija ovakvih kodova jednostavna i prostorno vrlo mala što ove kodove čini pogodnim za primjenu u satelitskim komunikacijama. U primjeni su najznačajniji binarni konvolucijski kodovi, što se u daljnjem tekstu neće posebno naglašavati.

Konvolucijski koder se može predstaviti pomoću *linearnog, vremenski invarijantnog sustava* (LTI – *linear time-invariant system*). Invarijantnost se odnosi na nepromjenjivost prijenosne funkcije koje opisuju ovisnost izlaza o ulazu sl. 6.1. Ulaz $\boldsymbol{u}_i$ i izlaz $\boldsymbol{v}_j$ su povezani s prijenosnom funkcijom $\boldsymbol{g}_j^{(i)}$ operacijom

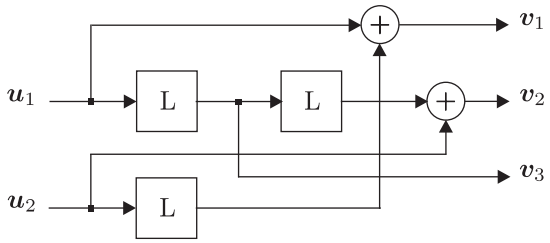


Slika 6.1: Model (n, k) konvolucijskog kodera

konvolucije. Pošto svaki izlaz može ovisiti o svim ulazima, općenito vrijedi:

$$\boldsymbol{v}_j = \boldsymbol{u}_1 * \boldsymbol{g}_j^{(1)} + \boldsymbol{u}_2 * \boldsymbol{g}_j^{(2)} + \cdots + \boldsymbol{u}_k * \boldsymbol{g}_j^{(k)} = \sum_{i=1}^{k} \boldsymbol{u}_i * \boldsymbol{g}_j^{(i)} \tag{6.1}$$

gdje $*$ označava operaciju konvolucije, a $\boldsymbol{g}_j^{(i)}$ impulsni odziv i-tog ulaza na j-ti izlaz. Odziv $\boldsymbol{g}_j^{(i)}$ se može odrediti na način da se na i-ti ulaz kodera dovede diskretni impuls $(1, 0, 0, \dots)$ i promatra odziv na j-tom izlazu a potom na svim ostalim ulazima treba biti nul-sekvenca $(0, 0, 0, \dots)$. Impulsni odzivi se nazivaju *generator sekvence* konvolucijskog kodera. Na sl. 6.2 prikazan je koder $(3, 2, 2)$ konvolucijskog koda. Generator polinomi su sljedeći:



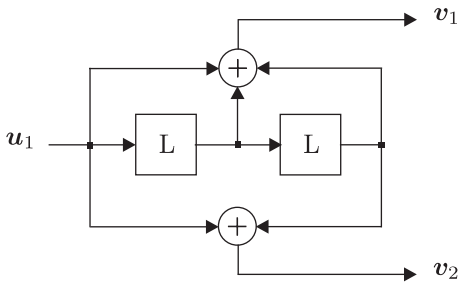
Slika 6.2: Konvolucijski koder $(3, 2, 2)$

$$\begin{array}{lll} \boldsymbol{g}_1^{(1)} = (1,0,0) & \boldsymbol{g}_2^{(1)} = (0,0,1) & \boldsymbol{g}_3^{(1)} = (0,1,0) \\ \boldsymbol{g}_1^{(2)} = (0,1,0) & \boldsymbol{g}_2^{(2)} = (1,0,0) & \boldsymbol{g}_3^{(2)} = (0,0,0) \end{array} \tag{6.2}$$

Izlazni tok podataka se formira na način da se u seriji uzimaju podaci s izlaza, odnosno u ovom slučaju vrijedi: $\boldsymbol{v} = (v_{1,0}, v_{2,0}, v_{3,0}, v_{1,1}, v_{2,1}, v_{3,1}, \dots, v_{1,L-1}, v_{2,L-1}, v_{3,L-1})$ gdje prvi indeks označava izlaz, a drugi diskretni vremenski interval. Tako npr. $v_{2,5}$ označava vrijednost na drugom izlazu ($\boldsymbol{v}_2$) u 5-tom vremenskom intervalu. S L je označena duljina sekvence.

Osim matematičkih relacija kojima je opisana funkcijska ovisnost izlaza o ulazu, moguća je i grafička predodžba rada konvolucijskog kodera pomoću *dijagrama stanja* a koja je značajno preglednija. Koder se može promatrati kao automat koji ima ulaze, izlaze i određeni broj stanja čiji broj je definiran brojem memorijskih elemenata i unutrašnjom strukturom kodera, odnosno realiziranim vezama. Ideja se sastoji u tome da se za svako stanje definira funkcijska ovisnost *prijelaza* i *izlaza* za svaku moguću vrijednost na ulazu. Funkcijom prijelaza je za svako stanje definirano u koje stanje koder prelazi za bilo koji ulaz, a funkcijom izlaza je definirana vrijednost na izlazu za svako stanje i proizvoljan ulaz. Prikaz rada konvolucijskog kodera pomoću dijagrama stanja je ilustrirana za koder prikazan na sl. 6.3.

Koder prikazan na sl. 6.3 ima $S = 4$ stanja. U dva memorijska elementa mogu se pojaviti četiri kombinacije pri čemu svaki memorijski element može biti

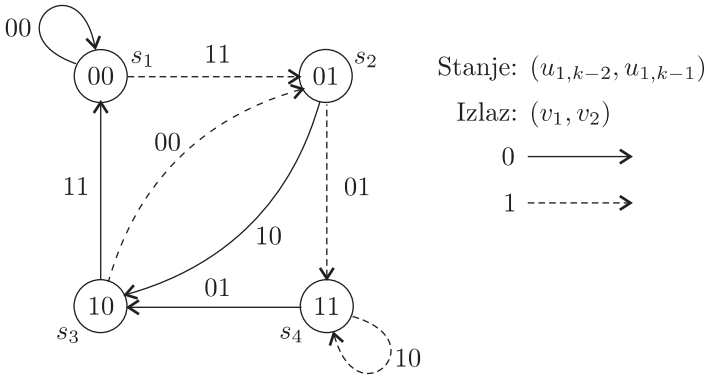


Slika 6.3: Konvolucijski koder $(2, 1, 2)$

u stanju 0 ili 1. U tablici 6.1 prikazana su moguća stanja navedenog kodera. Pomoću generator sekvenci, promatrajući stanja memorijskih elemenata, ulaze i izlaze može se konstruirati dijagram stanja (sl. 6.4).

Tablica 6.1: Tablica stanja

i	$s_i(u_{1,k-2}, u_{1,k-1})$
1	(0, 0)
2	(0, 1)
3	(1, 0)
4	(1, 1)



Slika 6.4: Dijagram stanja konvolucijskog kodera $(2, 1, 2)$

Osim dijagrama stanja, prikladan grafički prikaz konvolucijskog koda je pomoću *kodne rešetke* koja je posebno prikladna za prikazivanje postupka kodiranja i dekodiranja.

Na sl. 6.5 je prikazana kodna rešetka za kod $(2, 1, 2)$. Postoji više pristupa problemu dekodiranja konvolucijskih kodova kao što je dekodiranje na osnovi *logike većine*, dekodiranje na osnovu *Viterbijevog algoritma* i *sekvencijalno dekodiranje*. Dekodiranje konvolucijskih kodova s Viterbijevim algoritmom zasniva

se na pronalaženju puta minimalne težine prateći u svakom koraku onoliko puteva koliko ima stanja (u našem primjeru 4). Za svaki primljeni par simbola Viterbijev algoritam pridjeljuje težinu putu između dva stanja u skladu s pripadnom Hammingovom distancom. Na sl. 6.6 je prikazano dekodiranje sekvence $\boldsymbol{y} = (11\ 01\ 11)$. Vrijednost pripadne distance je označena ispod, a pripadni par digita iznad linije. Puna linija označava primljenu nulu, a isprekidana jedinicu. Nakon 3 koraka svako od stanja dosegnuto je sa po 2 puta od kojih svaki ima ukupnu težinu označenu masnijom brojkom. Za daljnji rad, Viterbijev dekoder eliminira po jedan od dolaznih puteva u svako od stanja i to onaj veće težine tako da nastavlja dalje pratiti 4 puta.

6.1 Zadataci

Zadatak 6.1. Odrediti generator polinome za koder prikazan na sl. 6.3

Zadatak 6.2. Odabrati proizvoljnu binarnu sekvencu $\boldsymbol{u}_i$ duljine 6 bitova, i kodirati je koristeći dijagram stanja prikazan na sl. 6.4. Kodiranje započeti iz stanja 1. Dodati ulaznoj sekvenci minimalno potreban broj bitova tako da dijagram stanja prijeđe u stanje 1.

Zadatak 6.3. Koliko bitova je potrebno za prijelaz dijagrama stanja u stanje 1, nakon što se kodira ulazna sekvenca?

Zadatak 6.4. Kodiranu sekvencu $\boldsymbol{v}$ dekodirati pomoću kodne rešetke (sl. 6.5 i 6.6) na način da se invertiraju proizvoljna 2 bita, što je ekvivalentno nastanku dvije greške u komunikacijskom kanalu.

Zadatak 6.5. Usporediti dekodiranu sekvencu $\hat{\boldsymbol{u}}$ sa sekvencom $\boldsymbol{u}$ i komentirati dobiveni rezultat.

6.2 Izvještaj

Ime i prezime: _____

Broj indeksa: _____

Datum: _____

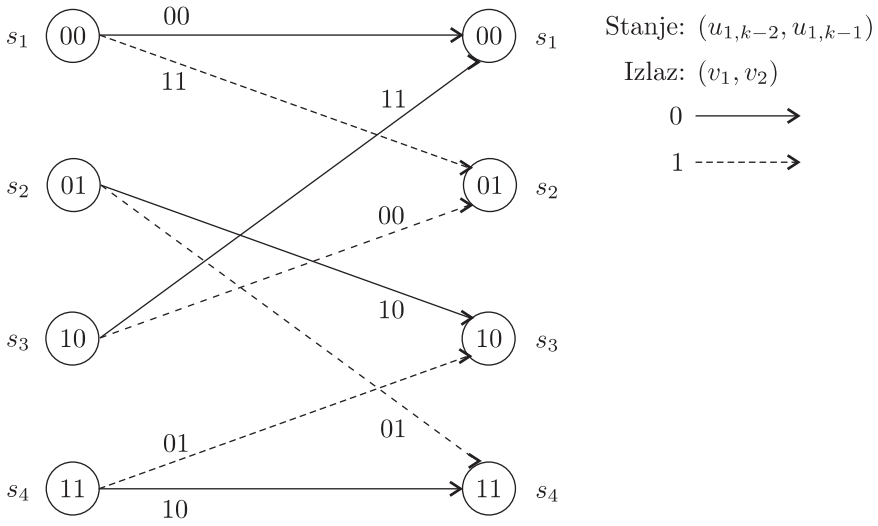
1. Odrediti generator polinome za koder prikazan na sl. 6.3

2. Odabrati proizvoljnu binarnu sekvencu $\boldsymbol{u}_i$ duljine 6 bitova, i kodirati je koristeći dijagram stanja prikazan na sl. 6.4. Kodiranje započeti iz stanja 1. Dodati ulaznoj sekvenci minimalno potreban broj bitova tako da dijagram stanja prijeđe u stanje 1.

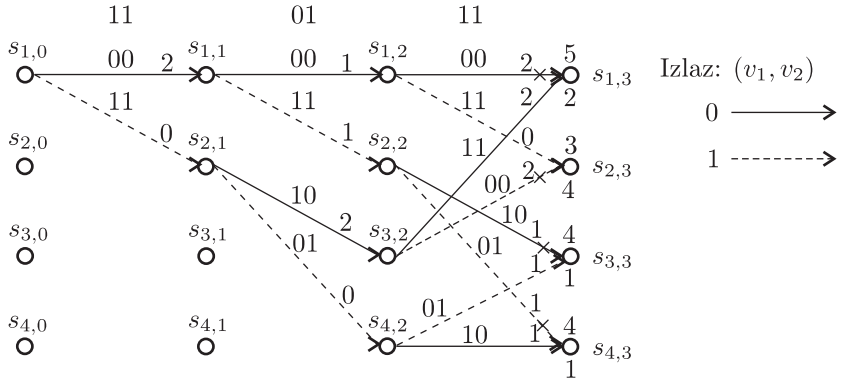
3. Koliko bitova je potrebno za prijelaz dijagrama stanja u stanje 1, nakon što se kodira ulazna sekvenca?

4. Kodiranu sekvencu $\boldsymbol{v}$ dekodirati pomoću kodne rešetke (sl. 6.5 i 6.6) na način da se invertiraju proizvoljna 2 bita, što je ekvivalentno nastanku dvije greške u komunikacijskom kanalu.

5. Usporediti dekodiranu sekvencu $\hat{\boldsymbol{u}}$ sa sekvencom $\boldsymbol{u}$ i komentirati dobiveni rezultat.



Slika 6.5: Kodna rešetka konvolucijskog kodera $(2, 1, 2)$



Slika 6.6: Dekodiranje konvolucijskog kodera $(2, 1, 2)$ pomoću kodne rešetke