

SVEUČILIŠTE U ZAGREBU  
Fakultet elektrotehnike i računarstva

Predmet: Teorija informacije (34315)  
Ak. godina: 2009./2010.

**Laboratorijske vježbe:**  
**I Z V J E Š T A J**

Grupa {7ip}:

1. {Martina Kocet, 0036440017}
2. {Vladimir Čipčić, 0036430157}
3. {Goran Kalić, 0036426360}
4. {Mirko Jambrošić, 00364285897}

## Zadatak

Potrebno je napraviti komunikacijski sustav koji se sastoji od izvora informacije, koda informacije, koda kanala, kanala na koji utječu smetnje, dekodera kanala, dekodera informacije i odredišta. Izvorište generira simbole,  $a, b, c, d, e$  sa zadanim vjerojatnostima pojavljivanja  $p(a) = 0,4$   $p(b) = 0,1$   $p(c) = 0,1$   $p(d) = 0,2$   $p(e) = 0,2$ . Te simbole potrebno je kodirati Huffmanovim kôdom. Nakon toga, novonastale slijedove simbola (tzv. kodiranu poruku) potrebno je kodirati *parnim paritetom* (dobivamo zaštitno kodiranu poruku). Zaštitno kodirana poruka prednosi se komunikacijskim kanalom u kojem imamo dva slučaja. U prvom slučaju poruka prolazi kroz kanal na kojem ne djeluju smetnje, dok u drugom slučaju struja bitova prolazi kroz kanal na kojem se mogu dogoditi smetnja i okretanje bitova. Vjerojatnost pojave greške  $p_g = 1/20$  odnosno 5%. Zatim se poruka prima, obrađuje, provjerava te se pokušava iz pristigle struje bitova generirati početna poruka.

## Rješenje

### Informacijska svojstva kanala

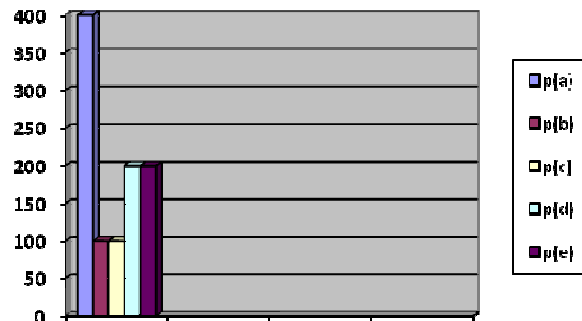
1. Izračunajte srednju vrijednost i standardnu devijaciju elemenata u nizu.

$$a=1 ; b=2 ; c=3 ; d=4 ; e=5 ;$$

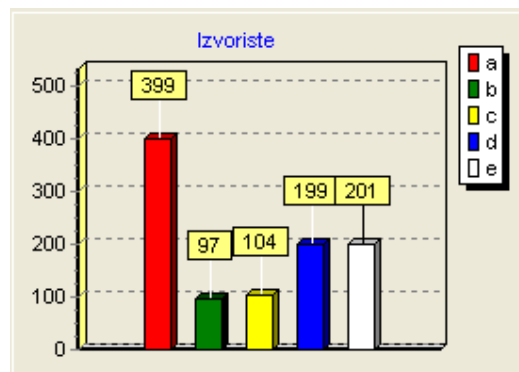
$$\mu = (1 \cdot 400 + 2 \cdot 100 + 3 \cdot 100 + 4 \cdot 200 + 5 \cdot 200) / 1000 = 2,7$$

$$\sigma_a^2 = 2,89 ; \sigma_b^2 = 0,49 ; \sigma_c^2 = 0,09 ; \sigma_d^2 = 1,69 ; \sigma_e^2 = 5,29 ;$$

2. Nacrtajte histogram generiranog slijeda.



Slika 1: Idealno izvorište



Slika 2: Realnije izvorište

Odredite vjerojatnosti pojavljivanja simbola u generiranom slijedu:

$$p(a) = 0,4 ; p(b) = 0,1 ; p(c) = 0,1 ; p(d) = 0,2 ; p(e) = 0,2$$

$$p(a) = 0,399 ; p(b) = 0,097 ; p(c) = 0,104 ; p(d) = 0,199 ; p(e) = 0,201$$

Vidimo da kod neidealnog izvora dolazi do sitnog odstupanja.

3. Izračunajte entropiju izvorišnog skupa simbola.

$$H(\text{izv.}) = \underline{2,122} \text{ [bit/simbol]}$$

4. Izračunajte entropiju odredišnog skupa simbola u slučaju da se koristi zaštitno kodiranje.

$$H(\text{odr.}) = \underline{2,11} \text{ [bit/simbol]}$$

5. Izračunajte transinformaciju u slučaju kad se koristi zaštitno kodiranje:

$$I(X;Y) = \underline{1,44} \text{ [bit/simbol]}$$

6. Izračunajte entropiju odredišnog skupa simbola u slučaju kad se NE koristi zaštitno kodiranje.

$$H(\text{odr.}) = \underline{2,386} \text{ [bit/simbol]}$$

7. Izračunajte transinformaciju u slučaju kad se NE koristi zaštitno kodiranje:

$$I(X;Y) = \underline{1,756} \text{ [bita/simbolu]}$$

8. Umetnite izvorni kôd funkcije u Matlabu kojom generirate simbole sa zadanim vjerojatnostima pojavljivanja. Izlaz iz funkcije mora biti vektor sa svim simbolima. Funkciju detaljno komentirajte.

Funkcija koja generira simbole naziva se *izvorište\_neidealno()*:

```
unit izvorište_neidealno;
```

```
interface
```

```
uses
```

```
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,  
Dialogs, glavni_unit, math, izvorište;
```

```
procedure pripremi_neidealno_izvorište();
```

```
var
```

```
na:integer;
```

```
nb:integer;
```

```
nc:integer;
```

```
nd:integer;
```

```
ne:integer;
```

```
implementation
```

```
procedure napravi_poruku();
```

```
var
```

```
a_g: integer;
```

```
b_g:integer;
```

```

c_g:integer;
d_g:integer;
e_g:integer;
i:integer;
r:integer; //random broj
begin
na:=0;
nb:=0;
nc:=0;
nd:=0;
ne:=0;
a_g:=trunc(100*StrToFloat(Form1.Edit1.Text));
b_g:=a_g+trunc(100*StrToFloat(Form1.Edit2.Text));
c_g:=b_g+trunc(100*StrToFloat(Form1.Edit3.Text));
d_g:=c_g+trunc(100*StrToFloat(Form1.Edit4.Text));
e_g:=100;
for i:=0 to 1000 do
begin
r:=RandomRange(0,100);
If (r<a_g) then
begin
poruka:=poruka+'a';
inc(na);
end
else if (r<b_g) then
begin
poruka:=poruka+'b';
inc(nb);
end
else if (r<c_g) then
begin
poruka:=poruka+'c';
inc(nc);
end
else if (r<d_g) then
begin
poruka:=poruka+'d';
inc(nd);
end
else if (r<e_g) then
begin
poruka:=poruka+'e';
inc(ne);
end;
end;
end;
end;

procedure pripremi_neidealno_izvoriste();
begin
poruka:="";
napravi_poruku();
form1.Memo1.Lines.Clear;
Form1.Memo1.Lines.Add(poruka);

```

end;

end.

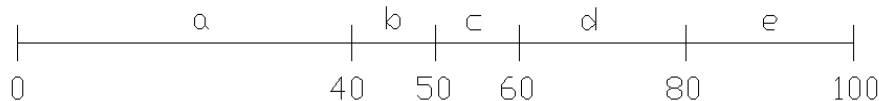
Ideja funkcije je da se odredi interval unutar kojeg se nalazi pojedini simbol, te se pokrene generator nasumičnih (random) brojeva. Ovisno o intervalu unutar kojeg se nalazi dobiveni generirani broj, odredi se koji je simbol na izlazu.

Dakle, prvo se određuje interval za svaki simbol. To je u kodu ostvareno na slijedeći način:

```
a_g:=trunc(100*StrToFloat(Form1.Edit1.Text));  
b_g:=a_g+trunc(100*StrToFloat(Form1.Edit2.Text));  
c_g:=b_g+trunc(100*StrToFloat(Form1.Edit3.Text));  
d_g:=c_g+trunc(100*StrToFloat(Form1.Edit4.Text));  
e_g:=100;
```

Korisnik unosi vjerojatnosti pojedinih simbola, te se one množe sa 100 i označavaju interval. Varijable a\_g, b\_g ... e\_g označavaju gornje granice intervala simbola.

U našem slučaju, gdje je  $p(a)=0.4$ ,  $p(b)=0.1$ ,  $p(c)=0.1$ ,  $p(d)=0.2$ ,  $p(e)=0.2$  intervali će biti podijeljeni kao na Slici 3. Vidi se kako simbol s većom vjerojatnosti ima i veći interval.



Slika 3: Intervali generiranja simbola

Zatim se pokreće generator nasumičnih brojeva:

```
for i:=0 to 1000 do  
begin  
r:=RandomRange(0,100);
```

Ovisno o intervalu u kojem se nalazi dobiveni broj, generirat će se određeni simbol.

Na primjer, ako se generatorom dobije broj 35 koji se nalazi u intervalu [0,40), na izlazu će biti simbol 'a'. Ako se dobije broj 78 koji je u intervalu [60,80), na izlazu će biti simbol 'd' (to je ostvareno u funkciji *napravi\_poruku()* nakon generiranja nasumičnog broja). Svi ti simboli spremaju se u jednu poruku od 1000 znakova i time je ostvaren generator simbola na izvoristu.

Izlaz funkcije:

```
aaeaadaababaeaaeadadedaaabaeabaeadebeeeaaacacadddcadedeadaabeccedaadedaedcedadaeac  
eaaaacbaadddaaadeadbdbaeaaadacaeeaddebacdbccedabbaacdeeeaddbdaaeaaeaeacaaecbeaa  
baedaeaeacadcabaadedddadaadbeaaaaabcbdcacdadeaddbaddeadadbeaaaeadeaeacbadaca  
ddeaebbaccadabadeceaaaeabcecdeddeeddaadeeaebeeadacdedeaabdebadedebaaceeeedeadaec  
bbdeacacdaadabaaaaeaeababdebcbdaabdaabcbbaeababadeeaaaaaadcaedecdaaddeabaacabec
```

aadeceaaeebeadeaaaaadadbeaedbcdabeacbabaccecadaeaaabccacdbdcccdeaeaeadaaaedaab  
caaaaaaaacaaaddabadaaaadeeeadbaebaedcabaadcaaaecdeaaaaaaeaaadaacedeaaecaccbdaea  
bebcadaaaecbaadeaaecacadeaebdaaaadadaadacadaaacbaadacaeeaaadeaeadebdadadbded  
edabeeadcdadeadaacaadedadaadcedacbeeacedaaadeaaddcabbcdaeaeaeceadbaedbcbddeeeadda  
aaeeedeaeabaceaecadadadcaaddaadaadcaaebdbabbcddeadadbdacaaaaaebbdaecbaaaedaa  
aaadeeabdaabaeeadceaaaaedaaaaeaaaceedbcdaeaeadeaaadadebbcadaceacdebadcaadaabeaab  
adeadaadddedadadeedccadadcbbdbacddcaeeebcedddaeaecebaabaededadaaaeeebadeaeabdad  
adcaeabbdeddaadd.

## Zaštitno kodiranje

**Napomena:** Sve niže definirane vrijednosti je generiranjem velikog broja ulaznih simbola potrebno statistički odrediti u Matlabu. Ako možete odrediti tu vjerojatnost računski, tada provjerite da li rezultat računa odgovara rezultatu dobivenom statističkom analizom.

1. Udaljenost za zadani zaštitni kôd dobivena provjeravanjem svih mogući kôdnih riječi iznosi:  $d(K) = 2$
2. Kôd može ispraviti 0 pogrešaka.
3. Kôd može detektirati 1 pogrešku.
4. Naš kod je kodiran paritetnim bitom, što znači da krajnji desni bit označava ispravnost/neispravnost u ovisnosti sa brojem jedinica. Pomoću tog zadnjeg bita realno možemo otkriti neispravnost pogreškom jednog bita u poruci.

Dok možemo primjetiti da paritetni bit neće odgovarati i kod greške sa bilo kojim neparnim brojem bitova. A kod greške sa parnim brojem bitova greška neće biti otkrivena.

Vjerojatnost ispravnog dekodiranja poruke iznosi:

$$P(K) = \sum_{i=0}^n \binom{n}{i} p_g^i (1 - p_g)^{n-i}, \quad t=1$$

npr. za slučaj duljine kodne riječi  $n=4$ :

$$P(K) = 0.95^4 + \binom{4}{1} 0.05^1 0.95^3 = 0.98598$$

5. Kodna brzina zaštitnog kôda iznosi (za duljinu kodne riječi  $n=4$ ,  $k=3$ ):

$$R(K) = \frac{k}{n} = \frac{3}{4} = 0.75$$

6. Navedite generirajuću matricu  $G$ :

$$K = \begin{Bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{Bmatrix}$$



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

$$G = [I_3 | A] \gg A = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

7. Navedite matricu provjere pariteta **H** za zadani kôd:

$$H = [A^T | I_1]$$

$$H = [1 \ 1 \ 1 \ 1]$$

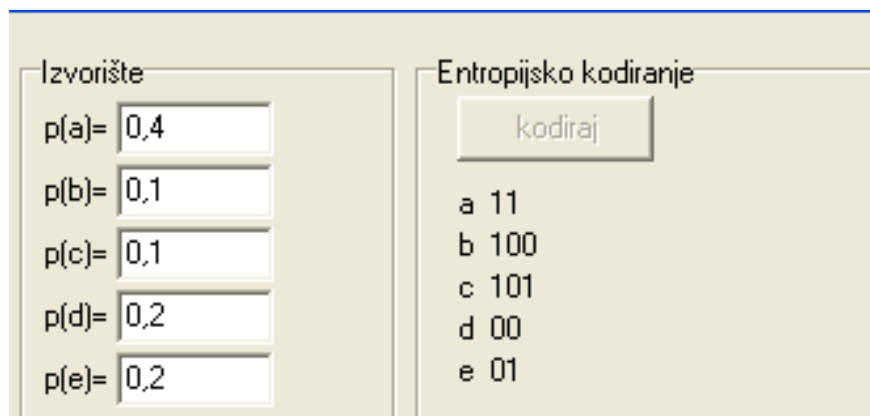
## Entropijsko kodiranje

### Prosječna duljina kodne riječi

Prosječna duljina kodne riječi računa se pomoću vjerojatnosti pojavljivanja simbola i njihovih duljina nakon kodiranja, formulom  $L = \sum_{i=1}^n p(x_i)l(x_i) = \sum_{i=1}^n p_i l_i$

U našem slučaju, **L= 2.2 bit/simbol**.

Vrijednosti potrebne za izračun prikazane su na Slici 4.



Izvorište	Entropijsko kodiranje
p(a)= 0,4	a 11
p(b)= 0,1	b 100
p(c)= 0,1	c 101
p(d)= 0,2	d 00
p(e)= 0,2	e 01

Slika 4: Vjerojatnosti simbola na izvorištu i pripadne kodne riječi

### Huffmanovo kodiranje

Entropijsko kodiranje simbola na izvorištu provedeno je na temelju Huffmanovog kodiranja.

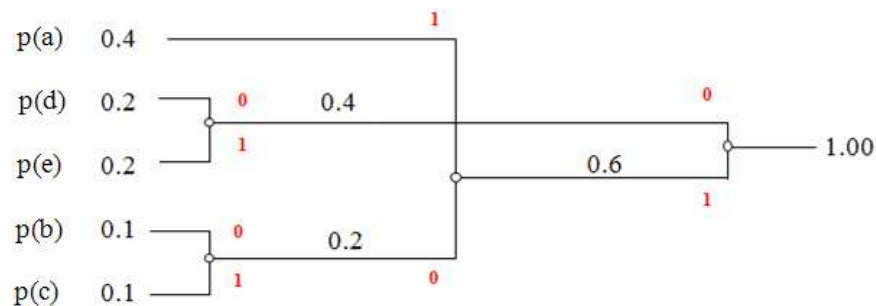
Algoritam započinje kreiranjem simbola. Svaki simbol je klasa s atributima koji ju definiraju:

```
TCvor = Class(TObject)
private
    vjerojatnost:double;
    znak:char;
    nasljednik:Char;
    iskoristen:Boolean;
    bit:Integer;
end;
```

Vrijednosti atributa *vjerojatnost* i *znak* intuitivno su jasni. *Nasljednik* je novi simbol koji nastaje zbrajanjem vjerojatnosti dvaju simbola. Ukoliko se neki simbol zbroji s drugim, on se više ne upotrebljava, što je označeno u atributu *iskoristen*. Pri zbrajanju simbola, svakom od njih se dodijeli bit 0 ili 1 ovisno o tome koji ima veću vjerojatnost (0 – za simbol manje vjerojatnosti, 1 – za simbol veće vjerojatnosti) i spremi se u atribut *bit*.

Nakon kreiranja simbola, započinje proces kodiranja. Prvo se traži simbol s najmanjom vjerojatnosti koji nije iskorišten i pridjeljuje mu se bit 0. Zatim se pronade slijedeći neiskorišten simbol s najmanjom vjerojatnosti i pridjeljuje mu se bit 1. Kada se pronadu dva simbola s najmanjim vjerojatnostima, stvori se novi simbol – nasljednik kojemu je vjerojatnost pojavljivanja jednaka zbroju vjerojatnosti prethodna dva.

Opisani se postupak odvija rekurzivno dok se ne obavi za sve simbole, odnosno dok ukupna vjerojatnost ne postane 1. Tada je izgrađeno stablo koje je za vjerojatnosti zadane zadatkom prikazano na Slici 5.



*Slika 5: Stablo dobiveno Huffmanovim kodiranjem*

Zadnji je korak „prošetati“ se stablom i okrenuti redoslijed bitova jer se u pravilu bitovi čitaju s desna na lijevo. Kao rezultat dobiju se kodne riječi prikazane na Slici 4 (a=11, b=100, c=101, d=00, e=01).

## **Dodatak**

Vlastiti komentari vezani uz laboratorijski zadatak. Ovdje, po potrebi, napišite zanimljivosti koje ste sami uočili prilikom izrade vježbe, priložite dodatne grafove i histograme za koje smatrate da dobro opisuju promatrane fenomene, komentirajte izvadak dijela koda za koji smatrate da ste ga jako dobro napisali itd.