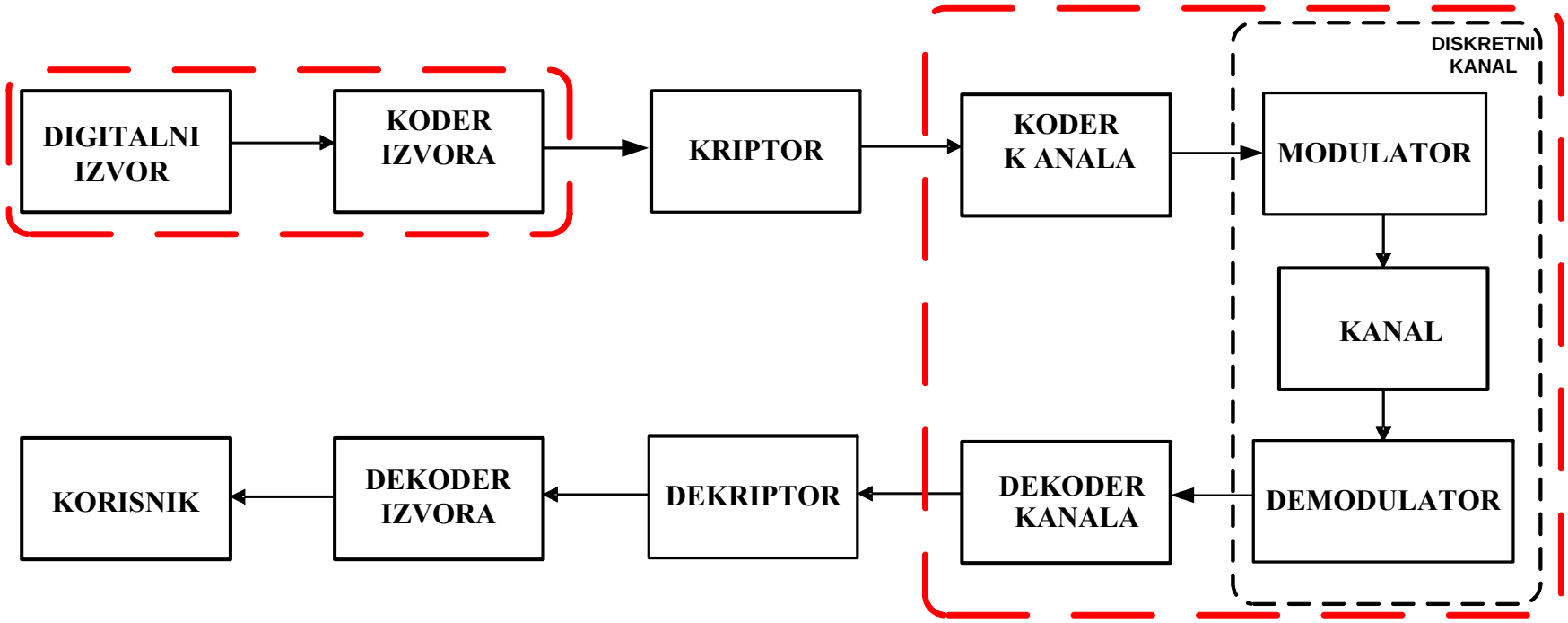
PRINCIPI MODERNIH TELEKOMUNIKACIJA
(SI2PMT)

Elektrotehnički fakultet
Katedra za Telekomunikacije
Beograd, 2012/2013.

-II-
Statistički i zaštitni kodovi

Blok šema sistema sa stanovišta teorije informacija

* Smatra se da izvor emituje nekakve simbole –> *q*-nivoski digitalni signal može se opisati sa *q* mogućih amplituda.
* Ciljevi sistema – *efikasan*, *siguran* i *pouzdan* prenos podataka.



Koder izvora, šifrator, zaštitni koder

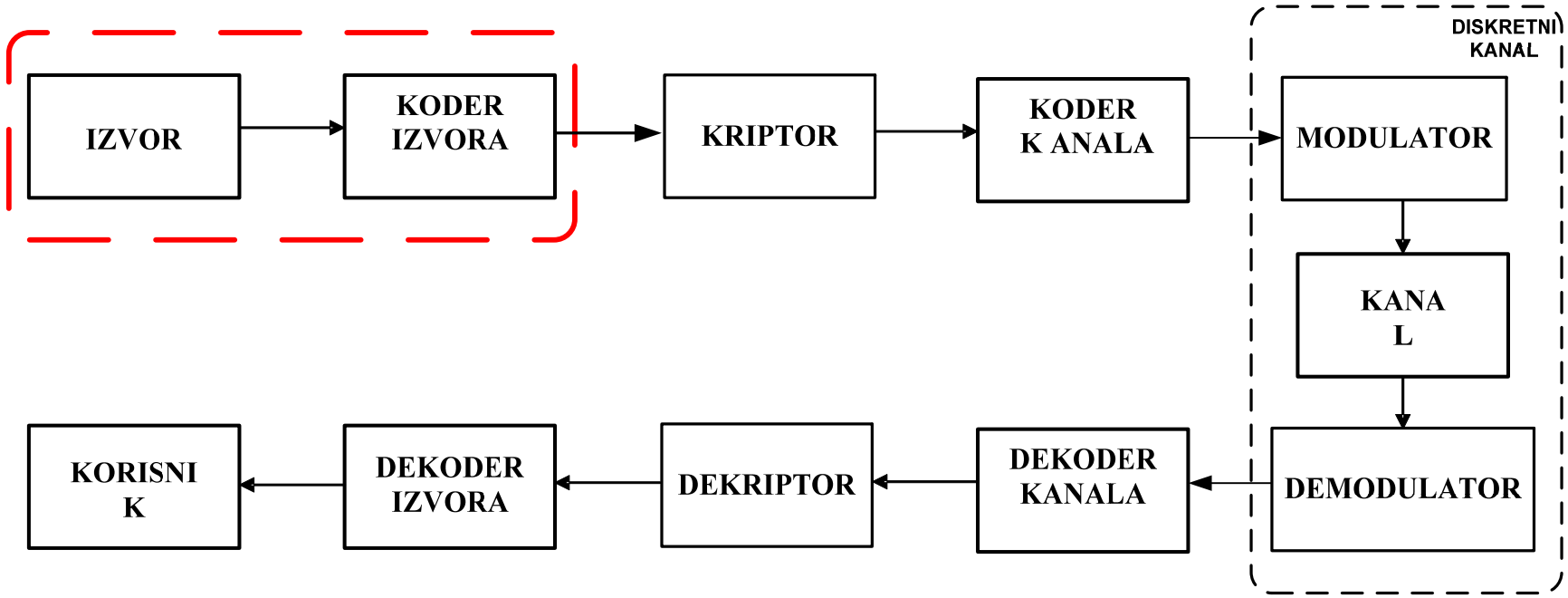
* ***Koder izvora*** **ovako digitalizovanu poruku pretvara u binarni oblik i ispuni neke dodatne zahteve:**
  - Cilj je svaku poruku predstaviti *minimalnim brojem bita* a da informacija bude preneta. Koliko *informacija* zaista emituje izvor?
  - Dekoder izvora u idealnom slučaju obavlja inverznu funkciju.

* **Ovako dobijeni binarni niz se u sledećem bloku (*šifrator*) šifruje, što ima za cilj očuvanje tajnosti pri prenosu podataka.**

* ***Zaštitni koder*** **– ima cilj da što je moguće više smanji verovatnoću greške pri prenosu pojedinih bita poruke. Na ulazu i izlazu kodera pojavljuju se biti, dok transformaciju bita u signale vrši modulator.**
  - Nakon zaštitnog kodera biti poruke su “oklopljeni” zaštitnim bitima.

Blok šema telekomunikacionog sistema

* **Blok šema sa stanovišta teorije informacija**
  - **Zanima nas prenos informacija kroz telekomunikacioni kanal**
  - **Koliko prenos može biti efikasan, siguran i pouzdan?**
* **Posmatra se prenos na nivou bita**
  - **Iz izvora izlaze biti**
  - **U diskretni kanal ulaze biti**



Diskretni izvor bez memorije

* **Opisuju se:**
  - Skupom mogućih poruka

$$S=\{s_1, s_2, ..., s_N\}$$

  - Verovatnoćama pojavljivanja pojedinih simbola iz ovog skupa

$$P(s_i),\ i=1,...N.$$

* **Primer:**
  - Izvor emituje šest simbola – **A, B, C, D, E, F**
  - Verovatnoće pojavljivanja

P(A)=0.5, P(B)=0.2, P(C)=0.1, P(D)=0.1, P(E)=0.07, P(F)=0.03

  - Primer sekvence

ADAABABCAEBAAACCBAFADABADABAEA

Efikasan prenos

* Neka sekvencu koju emituje diskretni izvor želimo da predstavimo u binarnom obliku
* ASCII kod – svaki simbol se predstavlja sa 7 bita
* Prethodni primer
  - Za prenos 30 slova iz prikazane sekvence potrebno je 30*7=210 bita.
  - Koliko god ima slova potrebno je sedam puta više bita za njihov prenos.
* Da li se poruka može predstaviti manjim brojem bita a da se ne naruši informacija koja se prenosi?
  - Želimo da pravilno rekonstruišemo svih 30 slova na strani prijema.

ASCII tabela – za štampani tekst

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Količina informacija

* **Informacija može imati više značenja**
  - sintaktički nivo – poruka nosi informacije ako na strani prijema postoji neizvesnost o tome koja će poruka biti primljena.
  - semantički nivo – zahteva se da korisnik razume značenje poruke (da je shvati)
  - pragmatički nivo – razmatra se vrednost informacija (korist koju izvlači korisnik)
* **Najjednostavniji način – pomoću logaritma**

$$Q(s_i)=\log[1/P(s_i)]$$

* **Funkcija mora da zadovolji sledeće**
  - Količina informacija ne može biti negativna.
  - Ako je verovatnoća pojave simbola ravna jedinici, događaj je siguran i simbol ne nosi nikakvu informaciju prijemniku log(1)=0;
  - Ako su simboli nezavisni, količine informacija koju oni nose se sabiraju

$$Q(s_i s_k)=\log[1/P(s_i, s_k)]= \log[1/P(s_i)P(s_k)]= Q(s_i)+Q(s_k)$$

* **Ako je baza logaritma 2 jedinica je Šenon (*Claude Shannon*).**
* **Prethodni primer:**

$$Q(\mathrm{A})=\log_2(1/0.5)=\mathrm{ld}(2)=1[\mathrm{Sh}],\ Q(\mathrm{F})=\mathrm{ld}(1/0.03)=5.06[\mathrm{Sh}],\ \ldots$$

Entropija

*** Entropija predstavlja prosečnu "meru neizvesnosti (neopredeljenosti)" posmatrača o tome šta će izvor da emituje.**

- Emitovanjem pojedinih simbola izvor emituje u proseku tačno potrebnu količinu informacija i upravo potpuno razrešava ovu neizvesnost.

$$H(S) = \overline{Q(s_i)} = \sum_{i=1}^{q} P(s_i)Q(s_i) = \sum_{i=1}^{q} P(s_i)\mathrm{ld}\left(\frac{1}{P(s_i)}\right) = -\sum_{i=1}^{q} P(s_i)\,\mathrm{ld}\,P(s_i) \quad \left[\frac{\mathrm{Sh}}{\mathrm{simb}}\right].$$

* Primer

H(S)=P(A)Q(A)+P(B)Q(B)+P(C)Q(C)+P(D)Q(D)+P(E)Q(E)+P(F)Q(F)
=0.5*1[Sh]+0.2*2.32[Sh]+...+0.07*5.06[Sh]=2.05 [Sh/simb]

Hafmenov postupak

- Hafmenov kod koji odgovara izvoru koji:

- Emituje šest simbola
- Verovatnoće zadate u drugoj koloni tabele

- Postupak

- Poređati po opadajućim verovatnoćama
- Sažimati po dva simbola i dati im isti prefiks

s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i
s_1	0.5	0	s_1	0.5	0	s_1	0.5	0	s_1	0.5	0	s_1	0.5	0
s_2	0.2	11	s_2	0.2	11	s_2	0.2	11	$s_3s_4s_5$ s_6	0.3*	10	s_2 $s_3s_4s_5$ s_6	0.5*	1
s_3	0.1	101	s_3	0.1	101	s_4s_5 s_6	**0.2***	100	s_2	0.2	11			
s_4	0.1	1000	s_4	**0.1**	1000	s_3	**0.1**	101						
s_5	**0.07**	10010	s_5 s_6	**0.1***	1001									
s_6	**0.03**	10011												

Srednja dužina kodne reči, efikasnost koda

- Srednja dužina kodne reči:

$$L_{sr} = 0.5*1 + 0.2*2 + 0.1*3 + 0.1*4 + 0.07*5 + 0.03*5 = 2.1 \ [b/s]$$

- Entropija izvora

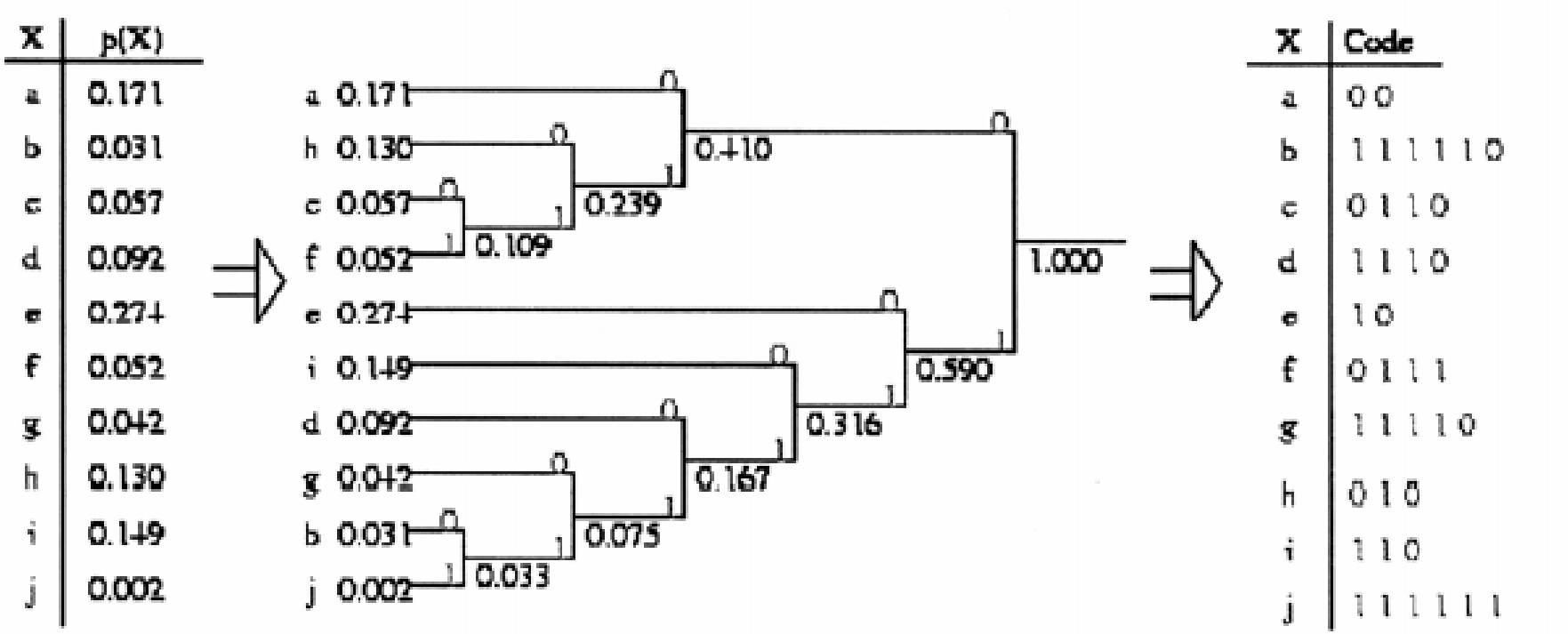
$$H(s) = \sum_{i=1}^{6} P(s_i) ld \frac{1}{P(s_i)} = 2.0502 \ [Sh/simb]$$

- Efikasnost

$$\eta = \frac{H(s)}{Lsr} \cdot 100\% = 97.63\%$$

Predstava pomoću stabla, drugi primer

- **Deset simbola izvorne liste, verovatnoće su im bitno različite!**
- **Dužina kodne reči bitno zavisi od verovatnoće pojavljivanja simbola kome je reč pridružena.**



Hafmenov kod primenjen na proširenje izvora

* **Posmatra se izvor koji emituje dva simbola sa sledećim verovatnoćama:**

s_i	s_1	s_2
$P(s_i)$	0.7	0.3

* **Potrebno je izvršiti binarno statističko kodovanje (po Hafmenovom postupku) izvora informacija, njegovog drugog i trećeg proširenja. Odrediti efikasnost svakog od postupaka i uporediti rezultate sa postavkom Prve Šenonove teoreme.**
  - Postupak statističkog kodovanja elemenata liste originalnog izvora je trivijalan:

s_i	$P(s_i)$	x_i
s_1	0.7	0
s_2	0.3	1

  - Entropija originalnog izvora, srednja dužina kodne reči i efikasnost

$$H(s) = 0.7ld\frac{1}{0.7} + 0.3ld\frac{1}{0.3} = 0.8813\ Sh/simb$$

$$L_{sr} = 0.7*1 + 0.3*1 = 1\ bit/simb$$

$$\eta = \frac{0.8813}{1}\cdot 100\% = 88.13\%$$

Hafmenov kod primenjen na proširenje izvora

* **Ako se umesto pojednih simbola posmatraju sekvence od po 2, 3 ili više (*n*) sukcesivnih simbola, tada se kaže da se posmatra drugo, treće ili *n-to proširenje izvora*.**
  - Ono se obično obeležava sa S^n a broj njegovih simbola je upravo q^n.
  - Drugim rečima, *n*-to proširenje izvora je izvor čiji su simboli sekvence od po *n* simbola prvobitnog izvora.
* **Postupak statističkog kodovanja elemenata liste II proširenja izvora:**

s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i	s_i	$P(s_i)$	x_i
$\sigma_1 = s_1 s_1$	0.49	1	σ_1	0.49	1	$\sigma_2\sigma_3\sigma_4$	0.51*	0
$\sigma_2 = s_1 s_2$	0.21	01	$\sigma_3\sigma_4$	**0.30***	00	σ_1	0.49	1
$\sigma_3 = s_2 s_1$	**0.21**	000	σ_2	**0.21**	01			
$\sigma_4 = s_2 s_2$	**0.09**	101						

  - Entropija II proširenja, srednja dužina kodne reči i efikasnost

$$H^2(s) = 2H(s) = 1.7626\ Sh/simb$$

$$L_{sr} = 0.49*1 + 0.21*2 + 0.21*3 + 0.09*3 = 1.81\ bit/simb$$

$$\eta = \frac{1.7626}{1.81} \cdot 100\% = 97.38\%$$

Prva Šenonova teorema – primer 2

* **Posmatrajmo binarno izvor bez memorije sa verovatnoćama pojavljivanja simbola P(0)=0.99 i P(1)=0.01.**
  - Originalni izvor ima entropiju H(S)=0.0808. On se može kodovati trivijalnim Hafmenovim kodom 0->0, 1->1, čime se jedan simbol izvorne liste predstavlja sa jednim bitom, pa je L_{sr}=1. Efikasnost je 8.08% a stepen kompresije 0%.
  - Drugo proširenje sastoji se od četiri simbola sa verovatnoćama P(00)=0.9801, P(01)=0.0099, P(10)=0.0099 i P(11)=10^{-4} i entropijom H(S)=0.1616. Srednja dužina kodne reči je L_{sr}=1.0299. Efikasnost je 15.7%.
  - Sa povećanjem reda proširenja efikasnost teži maksimalnoj vrednosti i postignuti stepen kompresije znatno raste.
  - U slučaju petog proširenja biće P(00000)=$(0.99)^5$=0.951 i ovom simbolu će biti dodeljena kodna reč '0', dok će preostale petobitne kombinacije iz izvorne liste imati znatno manje verovatnoće. To tnači da će pet nula iz izvorne liste biti zamenjene jednim binarnim simbolom iz kodne liste a da se ne izgubi informacija koja se prenosi!
  - Pri dovoljno velikom proširenju **100 simbola iz izvorne liste mogu se zameniti sa nešto više od osam simbola kodne liste** (i jedno i drugo su biti). Ova tvrdnja nije ništa drugo nego nešto drugačije interpretirana I Šenonova teorema!

Proširenje izvora, entropija proširenja

2. Primer

- Originalni izvor emituje poruke iz skupa S={0, 1} sa verovatnoćama

  P(0)=0.99, P(B)=0.01;

- Proširenje izvora "emituje" složene simbole iz sledećeg skupa:

  S^2={00, 01, 10, 11}.

- Računa se na osnovu verovatnoća složenih simbola $\sigma_i=s_i s_k$, za izvore bez memorije važi $P(s_i,s_k)=P(s_i)P(s_k)$.

  H(S)=0.99*ld(1/0.99)+0.01*ld(100)=1.5 [Sh/simb]

  P(00)=0.99*0.99=0.##,..., P(11)=0.01*0.01=0.0001

  $H(S^2)$=0.25*ld(4)+...+0.0625*ld(16)=3 [Sh/simb]= 2H(S).

Prva Šenonova teorema

* **Ako u tekstu postoji suvišnost, ona se može otkloniti kompresijom.**
  - Koliki stepen kompresije se maksimalno može postići?
  - Kolika je minimalna dužina kodne reči a da se sačuva kompletna informacija (da kod bude nedestruktivan)?

* **Prva Šenonova teorema – dovoljnim proširivanjem reda izvora i njegovim kodiranjem može se postići proizvoljno visoka efikasnost:**

$$\lim_{n\to\infty} \frac{L_{sr,n}}{nH(s)} = 1$$

- Ovaj izraz važi i za izvore sa memorijom!
- Kompresija može najviše ići do nivoa gde se svaki simbol u proseku predstavlja sa onoliko bita koliko iznosi entropija izvora.

Lempel Zivov (LZ) algoritam

* **Dve faze:**
  - Prvo se formira rečnik na osnovu dela sekvence koju emituje izvor;
  - Kada je rečnik jednom formiran, on se koristi za kompresiju ostalog dela sekvence koju emituje izvor.

* **Obično se koristi za kompresiju teksta**
  * Na prvih 256 pozicija slova, brojevi i specijalni znaci (prošireni ASCII).
  * Poznavanje ovog (manjeg, standardnog i nezavisnog od statistike prenošene sekvence!) dela rečnika je potreban i dovoljan uslov da se rekonstruiše rečnik i izvrši dekompresija samo na osnovu presretnute sekvence!
  * Ukupna veličina rečnika je obično 2048 ili 4096 adresa, pa se svaki složeni simbol upisan u rečnik predstavlja kombinacijom od 11 ili 12 bita.

LZ, kompresija

abbaabbaaba bb**abb**abb ->0110242 366 (tj. 000 001 001 000 010 100 010 011 110 110)

```
W=NIL;
loop
      read k;
      if wk u recniku
            w=wk;
      elde
            code of w->out
            wk-> tabela stringova
            w=k;
      end;
 end loop;
```

rečnik						
adresa	**sadržaj**	w	k	wk	?	out
0	a					
1	b	nil	a	a	+	
		a	b	ab	-	0
2	ab	b	b	bb	-	1
3	bb	b	a	ba	-	1
4	ba	a	a	aa	-	0
5	aa	a	b	ab	+	
		ab	b	abb	-	2
6	abb	b	a	ba	+	
		ba	a	baa	-	4
7	baa	a	b	ab	+	
		ab	a	aba	-	2

LZ vs. Hafmen

* **Neka je sekvenca koju treba komprimovati**

  abababababababababa…

* **ukupno:**
  - **dve podsekvence dužine 2 (ab,ba)**
  - **dve podsekvence dužine 3 (aba,bab)**
  - **dve podsekvence dužine 2 (abab,baba)**

rečnik	
adresa	**sadržaj**
0	a
1	b
2	ab
3	ba
4	aba
5	abab
6	bab
7	baba
8	ababa
...	...

* Ako rečnik ima 16 adresa (sa 4 bita)
  -> na adr 14. i 15. će biti sekvence dužine 8
* Ako rečnik ima 4096 adresa (sa 12 bita)
  -> na adr 4094. i 4095. će biti sekvence dužine 2048!

* U realnosti rečnik ima ukupno 4096 pozicija, prvih 256 pozicija osnovni simboli (0-255) a na pozicijama (256-4095) izvedeni simboli.
* Naravno, statistička zavisnost je znatno manja nego u navedenom primeru ali je sasvim dovoljna da za štampani tekst radi bolje nego Hafmen.

Primene algoritama za kompresiju

PRIMENE:
- Nedestruktivna kompresija pisanog teksta ili slike obično se zasniva na LZ kodovima (LZ77, LZW) ili kombinaciji LZ/Hafmen.

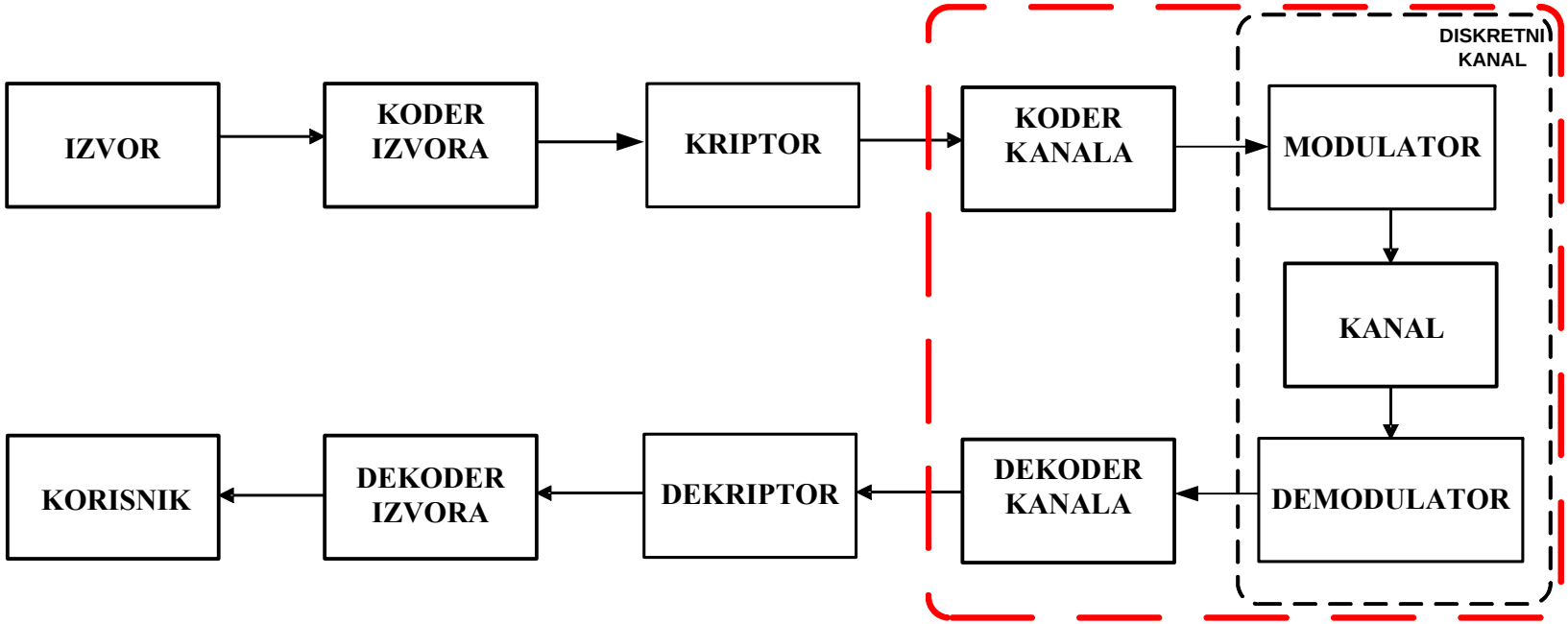
Utility	*Format*	*Compression*
pkarc (DOS) arc (Unix, Mac, etc.)	.arc, .ark	LZW
arj (DOS)	.arj	LZ77 + hashing, secondary static Huffman
Compuserve GIF	.gif	LZW
gzip	.gz	LZ77 + hashing, secondary static Huffman
lha, lharc	.lha, .lhz	LZ77 + tries, secondary static Huffman
squeeze (DOS)	.sqz	LZ77 + hashing
pkzip (DOS) zip (Unix) WinZip (Windows)	.zip	LZ77 + hashing, secondary static Huffman
zoo (DOS/Mac/Unix)	.zoo	LHA
freeze (Unix)	.F	LZ77 + hashing, secondary adaptive Huffman
yabba (Unix)	.Y	LZ78 variant
compress (Unix)	.Z	LZW

JPEG:
1. do irreversible compression on colour channels
2. compute the *Discrete Cosine Transform* for
3. "reduce" the DCT output: more reduction
4. Huffman encode the reduced output

MPEG:
1. do irreversible compression on colour channels (not on shade channel)
2. for each block of 16x16 in a frame, try to find a "similar" block in a previous (*or future*)
3. store the differences between blocks instead of storing entire blocks
4. Huffman encode the whole thing

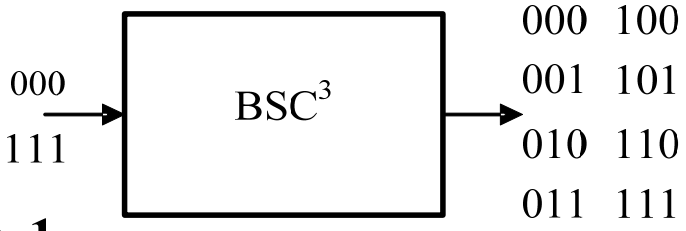
Osnovna blok šema

* Tri vrste kodova:
  - Statistički kodovi – zavise od osobina izvora;
  - Kodovi za očuvanje tajnosti (kriptografija);
  - Zaštitni kodovi – zavise od osobina kanala.



Ponavljanje poruke, pravilo odlučivanja

* **Kod zasnovan na ponavljanju poruke tri puta**
  - broj poruka je M=2, poruke bi mogle da se predstave sa ld(M)=1 bita;
  - dužina kodne reči je n=3, kodni količnik R=1/3 a protok Φ=v(X,Y)/3;



* **Pravilo odlučivanja br. 1**
  - Neka se 000 i 111 dekoduje kao 0, odnosno 1, a u svim ostalim slučajevima smatra da su se pojavile greške;
  - Verovatnoća neotkrivene greške za E_b/N_0=4.3dB tj. p=10^{-2}.

$$P_e^{(1)} = p^3 = 10^{-6}.$$

  - Ovaj nivo greške bez primene koda postiže se za E_b/N_0=10.5dB -> p=10^{-6}.

* **Pravilo odlučivanja br. 2**
  - Majoritetna logika: greške se detektuju ali i koriguju.
  - Ispravlja samo jednostruke, verovatnoća neotkrivene greške je sada nešto veća

$$P_e^{(2)} = \binom{3}{0} p^3 + \binom{3}{1} p^2 (1-p) = 0,000298.$$

Hemingov kod – konstrukcija pomoću šablona

* **Šablon**

1	0 0 **1**	z_1
2	0 **1** 0	z_2
3	0 1 1	i_1
4	**1** 0 0	z_3
5	1 0 1	i_2
6	1 1 0	i_3
7	1 1 1	i_4

* **Vektori**

I = [1101]
X = [1010101]
E = [0000010]
Y = [1010111]

- decimalni zapis sindroma pokazuje poziciju greške.

- ovaj kod može da detektuje i ispravlja greške.

* **Neka treba kodovati bite i_1=1, i_2=1, i_3=0, i_4=1.**
* **Pozicije prve jedinice u kolonama (počev od krajnje desne) određuju pozicije zaštitnih bita**

$$z_1, z_2, i_1, z_3, i_2, i_3, i_4$$

* **Preostale jedinice u pojedinim kolonama određuju kontrolne sume**

$$z_1 = i_1 \oplus i_2 \oplus i_4 = 1 \oplus 1 \oplus 1 = 1,$$
$$z_2 = i_1 \oplus i_3 \oplus i_4 = 1 \oplus 0 \oplus 1 = 0,$$
$$z_3 = i_2 \oplus i_3 \oplus i_4 = 1 \oplus 0 \oplus 1 = 0.$$

* **Greška na šestoj poziciji, e_6=1.**
* **Dekodovanje se obavlja pomoću sindroma**

$$s_1 = y_1 \oplus y_3 \oplus y_5 \oplus y_7 = 1 \oplus 1 \oplus 1 \oplus 1 = 0$$
$$s_2 = y_2 \oplus y_3 \oplus y_6 \oplus y_7 = 0 \oplus 1 \oplus 1 \oplus 1 = 1$$
$$s_3 = y_4 \oplus y_5 \oplus y_6 \oplus y_7 = 0 \oplus 1 \oplus 1 \oplus 1 = 1$$

S=[110]=6 (pozicija greške).

Pojava dvostruke greške

* **Poslato je [1101] a pri prenosu je pogrešno prenet 5. i 6. bit**

Koder :	*Kanal* :	*Dekoder* :
$z_1 = 1 \oplus 1 \oplus 1 = 1$	$x = [1\ 0\ 1\ 0\ 1\ 0\ 1]$	$s_1 = 1 \oplus 1 \oplus 0 \oplus 1 = 1$
$z_2 = 1 \oplus 0 \oplus 1 = 0$	$e = [0\ 0\ 0\ 0\ 1\ 1\ 0]$	$s_2 = 0 \oplus 1 \oplus 1 \oplus 1 = 1$
$z_3 = 1 \oplus 0 \oplus 1 = 0$	$y = [1\ 0\ 1\ 0\ 0\ 1\ 1]$	$s_3 = 0 \oplus 0 \oplus 1 \oplus 1 = 0$

* **Sindrom S=[011] ukazuje na treću poziciju koja se komplementira pa se rekonstruisana informaciona reč [0011] razlikuje od poslate za tri bita!**

* **U ovom primeru dekodovanje je čak pogoršalo performanse sistema. Zato se često dodaje još jedan zaštitni bit - ukupna provera na parnost.**

* **Osobine Hemingovog (7,4) koda**
  - **d=3, e_c=1, e_d=1;**
  - **ovaj kod može da detektuje jednu grešku, koju istovremeno i koriguje (ne može da detektuje dodatne greške).**

Tumačenje sindroma kod Heminga (8,4)

* **Modifikovano kodovanje i dekodovanje**

$$z_4 = \sum_{i=1}^{7} x_i, \qquad s_4 = \sum_{i=1}^{8} y_i$$

* **Sindromi**

  1) S=0, s_4=0 - nije bilo greške pri prenosu

  2) S>0, s_4=1 - neparan broj grešaka, sindrom pokazuje poziciju

  3) S>0, s_4=0 - paran broj grešaka

  4) S=0, s_4=1 - greška baš na bitu parnosti

* **Heming (8,4)**

  - ***d*=4, e_c=1, e_d=3; ovaj kod može da detektuje jednu grešku (koju i koriguje) a može da detektuje i još jednu dodatnu grešku.**
  - **Ako je s_4=1, postojao je neparan broj grešaka (na prvih sedam pozicija!), tj. jedna, tri, pet ili sedam grešaka. Za $p=10^{-3}$ verovatnoće da se ovo desi su, respektivno**

$$P_{e1} = \binom{7}{1} p(1-p)^6 \approx 7\cdot 10^{-3}, \quad P_{e3} = \binom{7}{3} p^3(1-p)^4 \approx 3.5\cdot 10^{-8}$$

$$P_{e5} = \binom{7}{5} p^5(1-p)^2 \approx 2.1\cdot 10^{-14}, \quad P_{e7} = \binom{7}{7} p^7 \approx 1\cdot 10^{-21}$$

Šenonov kapacitet kanala

* ***Druga Šenonova teorema:*** ***Sve dok je protok manji od kapaciteta kanala može se naći takav zaštitni kod da se verovatnoća greške proizvoljno smanji!***
  - moguć je pouzdan prenos (s proizvoljno malom verovatnoćom greške, označenom sa ε) kroz nepouzdan kanal!
  - kapacitet kanala je maksimalna moguća brzina kojom se informacije mogu (pouzdano) prenositi kroz dati kanal!
* **Neka se signal prenosi kroz kanal koji ima širinu propusnog opsega, označenu sa *B*, a u kome je prisutan šum snage P_n .**
* **Odnos signal-šum (*signal-to-noise ratio*) se definiše izrazom**

$$SNR = P_s / P_n$$

* ***Kapacitet kontinualnog kanala sa šumom*** **odredio je Šenon (*Claude Shannon*) i on se može izračunati pomoću izraza**

$$C / B = ld(1 + SNR)$$

Literatura

[1] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal,* vol. 27, pp. 379-423, July 1948; pp. 623-656, October 1948.

[2] S. Lin, D. J. Costello, *Error Control Coding*, Second Edition, Prentice Hall, New Jersey, 2004.

[3] D. Drajić, P. Ivaniš, “*Uvod u teoriju informacija sa kodovanjem*”, treće izdanje, Akademska misao, Beograd, 2009.

[4] D. J. Costello, Jr., J. Hagenauer, H. Imai, S. B. Wicker, “Applications of Error-Control Coding”, *IEEE Trans. Inform. Theory*. Vol 44 (1998), pp. 2531-2560

[5] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, John Wiley & Sons, Ltd, England, 2002.