COMPRESIA ENTROPICĂ

2.1 CONCEPTE FUNDAMENTALE

2.1.1 Entropia unei surse staţionare

Se consideră un *alfabet* finit **X** constituit din M simboluri şi se defineşte un *mesaj* ca o secvenţă de simboluri. O sursă discretă de informaţie este numită stochastică dacă selecţia simbolurilor care formează mesajele este făcută din cadrul alfabetului **X** în conformitate cu o distribuţie de probabilitate $p_i \triangleq \mathbb{P}(x_i)$. Din punct de vedere probabilistic, mulţimea mesajelor poate fi privită ca un proces aleatoriu discret adică, precum o secvenţă $(\xi_n)_{n=0}^{\infty}$ de variabile aleatoare, fiecare dintre ele luând valori în mulţimea **X** cu probabilitatea de distribuţie $\{p_i\}$.

Se presupune în plus că sursa este *staţionară*, adică

$$P\{\xi_{i1} = x_1, \ldots, \xi_{ik} = x_k\} = P\{\xi_{i1+h} = x_1, \ldots, \xi_{ik+h} = x_k\} \qquad (2.1)$$

pentru toate numerele naturale $i_1, \ldots, i_k$, h şi toate $x_1, \ldots, x_k \in$ **X.**

O caracteristică globală a unei astfel de surse din punctul de vedere al informaţiei este *entropia sursei.*

$$H(x) \triangleq \sum_{i=1}^{M} p_i \log_2 \frac{1}{p_i} \qquad (2.2)$$

care este exprimată în biţi/simbol şi indică cantitatea medie de informaţie a alfabetului **X**.

Teorema 2.1. - Entropia H(x) a unei surse cu un alfabet conţinând **M** simboluri, satisface inegalitatea

$$H(x) \le \log_2 M \qquad (2.3)$$

egalitatea fiind satisfăcută pentru simboluri echiprobabile.

În scopul transmiterii la distanţă sau pentru stocarea informaţiei sursei este eficient să se treacă de la reprezentarea oferită de alfabetul sursei **X** la o reprezentare printr-un alt alfabet **Y**. Procesul este numit *codarea datelor*. Un cod este o corespondenţă între *mulţimea mesajelor sursei şi mulţimea cuvintelor de cod.*

Se consideră alfabetul sursei **X** constituit din simbolurile $x_1, \ldots, x_M$. Fiecărui simbol i se asociază o secvenţă finită de simboluri ale alfabetului codului **Y**. În practică, alfabetul **Y** este constituit din două simboluri **0** sau **1**. În acest fel fiecărui simbol i se asociază o secvenţă finită de biţi numită *cuvânt de cod*.

O codare eficientă (compresie) urmăreşte minimizarea *lungimii medii* a cuvintelor de cod.

$$\overline{n} \triangleq E\{n\} = \sum_{i=1}^{M} p_i n_i \qquad (2.4)$$

unde n_i este lungimea cuvântului de cod (număr de biţi) care reprezintă simbolul x_i, iar n este o variabilă aleatoare care reprezintă lungimea cuvântului de cod (luând valoarea n_i cu probabilitatea p_i, i = 1, 2, ..., M).

Codurile realizează o corespondenţă între mesajele sursei (secvenţe de simboluri - cuvinte ale alfabetului sursei) şi cuvintele de cod (secvenţe de biţi - cuvinte ale alfabetului **Y**). După modul de stabilire a acestei corespondenţe codurile pot fi clasificate în:

i) bloc-bloc;
ii) bloc-variabile;
iii) variabile-bloc;
iv) variabile-variabile;

Codurile bloc-bloc sunt coduri care stabilesc o corespondenţă între mesaje ale sursei de lungime fixată şi cuvintele de cod de lungime fixată.

Codurile variabile-variabile stabilesc o corespondenţă între mesaje ale sursei de lungime variabilă şi cuvinte de cod variabile.

EXEMPLUL 2.1

Se consideră
X = { a, b, c, d, e, f, g, h, spaţiu }
Y = { 0, 1}

Simboluri	Cuvinte de cod
a	000
b	001
c	010
d	011
e	100
f	101
g	110
Spaţiu	111

Tab. 2.1 Tabel de codare bloc-bloc

Codul din tabelul 2.1 este un cod bloc-bloc şi în aceeaşi categorie se încadrează codurile ASCII care pun în corespondenţă un alfabet de 64 (sau 256) simboluri cu cuvinte de cod de 6 (sau 8) biţi.

EXEMPLUL 2.2

Se consideră
X = { a, b, c, d, e, f, g, h, spaţiu }
Y = { 0, 1}

Simboluri	Cuvinte de cod
aa	0
bbb	1
cccc	10
ddddd	11
eeeeee	100
fffffff	101
gggggggg	110
spaţiu	111

Tab. 2.2. Tabel de codare variabile-variabile

Se observă că pentru un mesaj :

aa_bbb_cccc_ddddd_eeeeee_fffffffggggggg (40 simboluri in total)
se obține o lungime cod de 30 față de o lungime de cod de 120 dacă codarea s-ar fi făcut conform tabelului 2.1 .
Observație : S-a folosit simbolul "_" (underscore) pentru a nota caracterul spațiu

Un cod este *distinct* dacă fiecare cuvânt de cod este discernabil în raport cu celelalte (corespondența între mesajele sursei şi cuvintele de cod este biunivocă).
Un cod distinct este *unic decodabil* dacă fiecare cuvânt de cod este identificabil când se află imersat într-o secvență de cuvinte de cod.

Codurile din Exemplul 2.1 şi Exemplul 2.2 sunt ambele distincte, dar codul din Exemplul 2.2 nu este unic decodabil. Spre exemplu codul 11 poate fi decodat fie ca ddddd fie ca bbbbbb.
Un cod unic decodabil este un *cod prefix* dacă are proprietatea de prefix care pretinde ca nici un cuvânt de cod să nu fie prefixul altui cuvânt de cod. Codul care are drept cuvinte de cod {1, 100000, 00} este un exemplu de cod care este unic decodabil dar care nu are proprietatea de prefix.
Codurile prefix sunt *decodabile instantaneu* adică se bucură de proprietatea că mesajul codat poate fi despărțit în cuvinte de cod fără a fi necesară examinarea în avans (lookahead) a următorilor biți. Deci decodarea este posibilă imediat ce s-a primit ultimul bit al cuvântului de cod fără a mai fi necesară recepționarea altor biți. Spre exemplu dacă se foloseşte codul care are cuvintele de cod {1, 100000, 00}, decodarea mesajului 1000000001 şi identificarea lui 1 ca fiind primul cuvânt de cod necesită examinarea celui de-al zecelea simbol (era posibil ca primul cuvânt de cod să fie 100000).
Un *cod prefix minimal* este un cod prefix cu proprietatea ca dacă x este un prefix al unui cuvânt de cod atunci şi x sigma este fie un cuvânt de cod fie un prefix al unui cuvânt de cod oricare ar fi litera sigma aparținând alfabetului **Y**. Spre exemplu considerând codul {00, 01, 10} se observă că acesta este un cod prefix care nu este minimal. Astfel 1 este prefix al cuvântului de cod 10 şi ar impune ca şi 11 să fie un cuvânt de cod fie un prefix al unui cuvânt de cod valid. Dacă în exemplul de mai sus se înlocuieşte cuvântul de cod 10 cu cuvântul de cod 1 se obține un cod prefix minimal cu cuvinte de cod fie mai scurte.

2.1.2 Codarea alfabetului sursei

O reprezentare grafică utilă a codurilor prefix constă în asocierea fiecărui cuvânt de cod cu un nod terminal (frunză) dintr-un arbore binar. Pornind de la rădăcina arborelui primele două ramuri corespund selecției între 0 şi 1 ca primă cifră a cuvântului de cod. Cele două ramuri pornind din nodurile de ordinul întâi corespund celei de a doua cifre a cuvântului de cod şi procesul continuă mai departe (a se vedea fig.2.1).
Cuvintele de cod fiind asignate numai nodurilor terminale, nici un cuvânt de cod nu poate fi prefix al unui alt cuvânt de cod.
O condiție necesară şi suficientă ca un cod să fie cod prefix este dată de teorema următoare:

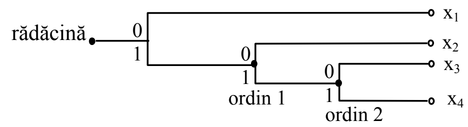

Fig.2.1 Arbore binar de codare

Simboluri	Cuvinte de cod
x_1	0
x_2	10
x_3	110
x_4	111

Tab.2.3 Tabelul de codare asociată arborelului

TEOREMA 2.2 Inegalitatea lui Kraft
Un cod binar care este un cod prefix având lungimile cuvintelor de cod $n_1, n_2, \ldots, n_M$ există dacă şi numai dacă

$$\sum_{i=1}^{M} 2^{-n_i} \le 1 \qquad (2.5)$$

Observație:
Dacă $p_i=2^{-n_i}$, i=1, 2, ..., M condiția (2.5) devine o egalitate.
În acest caz se obține pe baza relațiilor (2.2) şi (2.4) că $\overline{n} = H(x)$.
În general însă $\overline{n} \geq H(x)$ şi se urmăreşte ca acest n să fie cât mai aproape de H(x).
Un cod prefix poate fi determinat astfel încât n să satisfacă teorema următoare:
TEOREMA 2.3
Se poate determina un cod binar prefix pentru orice alfabet al sursei de entropie H(x); codul având o lungime medie care să satisfacă inegalitatea:

$$H(x) \le \overline{n} < H(x)+1 \qquad (2.6)$$

2.1.3 Clasificarea metodelor

Există mai multe criterii de clasificare a metodelor de compresie entropică.Toate pleacă însă de la ideea de a reduce redundanța naturală a mesajului inițial, şi acest lucru se poate face doar dacă anumite simboluri sau grupe (subşiruri) de simboluri, pe care le vom numi in general *forme* se repetă. In acest sens, putem clasifica metodele de compresie entropică în două categorii:

a) metode bazate pe frecvența de apariție a unei forme; aceste metode se realizează în principiu în două treceri asupra textului, prima trecere permițând tocmai stabilirea acestor frecvențe

b) metode bazate pe apariţia grupată (consecutivă sau diseminată) a aceleiaşi forme; de regulă aceste metode se realizează printr-o singură trecere pe text, eventual cu reveniri pe o zonă predeterminată

După un alt criteriu, cel al construcţiei arborilor de codare-decodare, metodele de compresie entropică pot fi clasificate în:

(i) metode statice
(ii) metode dinamice

O metodă *statică* este o metodă care stabileşte corespondenţa între mulţimea mesajelor şi mulţimea cuvintelor de cod, care nu se modifică în timp. Este fixată înainte de a începe codarea. În acest fel un mesaj dat este reprezentat de un acelaşi cuvânt de cod de fiecare dată când el apare în cadrul mesajului. Se urmăreşte construcţia unui cod unic decodabil care să minimizeze lungimea medie de cod. Un asemenea procedeu este optimal.

O metodă este dinamică, dacă corespondenţa între mulţimea mesajelor şi mulţimea cuvintelor de cod se modifică în timp. Asignarea cuvintelor de cod se bazează pe frecvenţa relativă de apariţie a mesajelor, evaluată până la momentul curent al codării.

Nu se poate face însă o delimitare clară între cele doua grupe de metode, chiar dacă în categoria (*a)* predomină metodele dinamice, iar in categoria (*b)* cele statice.

2.2 COMPRESIA STATICĂ

2.2.1 Algoritmul Huffman static

Pas 1. Se ordonează cele M simboluri în ordine descrescătoare a probabilităţii.

Pas 2. Se grupează ultimele două simboluri x_{M-1} şi x_M într-un *simbol echivalent* (contopire) cu probabilitatea $p_{M-1}+p_M$.

Pas 3. Se repetă paşii 1 şi 2 până ce se ajunge la un singur *simbol.*

Pas 4. În arborele obţinut prin parcurgerea paşilor anteriori se asignează simbolurile binare 0 şi 1 pentru fiecare pereche de ramuri care pleacă dintr-un nod intermediar. Cuvântul de cod pentru fiecare simbol poate fi citit ca secvenţa binară determinată prin parcurgerea nodurilor dinspre rădăcină până la nodurile terminale asociate simbolurilor.

Exemplul 2.3

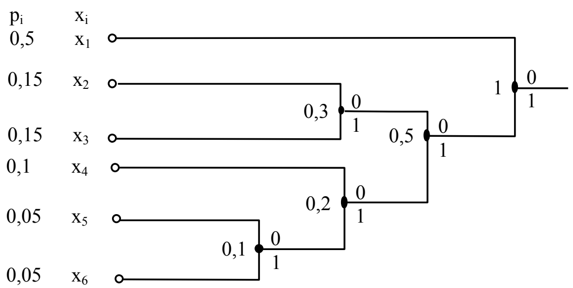

Fig. 2.2 Arbore binar de codare

Se obţine codarea Huffman:

Simboluri	Cuvinte de cod
x_1	0
x_2	100
x_3	101
x_4	110
x_5	1110
x_6	1111

Tab. 2.4 Tabelul de codare Huffman

Se constată că entropia H(x) = 2,09 biti/simbol iar lungimea medie $\overline{n}$ = 2,1 biţi/simbol şi cele două valori satisfac inegalitatea (2.6).

Exemplul 2.4

Pentru codul prezentat în exemplul 2.3 să se folosească arborele pentru a decoda secvenţa recepţionată (compactată) 1100100101110.

Soluţie:

Pornind de la rădăcină se urmăresc ramurile la fiecare nod conform biţilor secvenţei până se atinge un nod terminal (deci şi un simbol). Apoi se reporneşte procedura de la rădăcină. Se constată că secvenţa decodată este $x_4x_1x_2x_3x_4$.

Trebuie precizat că algoritmul Huffman nu duce la o codare unică, dar toate codările obţinute conduc la o aceeaşi lungime medie a codului.

Algoritmul descris mai sus se ocupă de *codarea simbol cu simbol*. Pentru a obţine performanţe mai bune în ceea ce priveşte $\overline{n}$, se consideră o codare a unui bloc de ν simboluri şi asignarea unor cuvinte de cod pentru noul alfabet $Y=X^{\nu}$, care conţine M^{ν} simboluri y_i. Probabilitatea simbolurilor y_i se obţine ca un produs al probabilităţilor celor ν simboluri din X (independenţa). În baza teoremei 2.3 se construieşte codul pentru alfabetul Y cu lungimea de cod $\overline{n}_{\nu}$ care satisface

$$H(Y) \leq \overline{n}_{\nu} < H(Y)+1 \qquad (2.7)$$

Având în vedere independenţa simbolurilor, se demonstrează că $H(Y)=\nu H(X)$ şi se obţine din (2.7) relaţia

$$H(X) \leq \overline{n}_{\nu}/\nu < H(X)+1/\nu \qquad (2.8)$$

Mărimea $\overline{n}_{\nu}/\nu$ este numărul mediu de biti/simbol al lui X. Prin alegerea lui ν suficient de mare se poate face ca aceasta să se apropie de H(X).

Se defineşte eficienţa unui cod ca

$$\varepsilon \stackrel{\Delta}{=} \frac{[\nu H(x)]}{\frac{\overline{n}_\nu}{\nu}} \qquad (2.9)$$

şi redundanţa $\eta \stackrel{\Delta}{=} 1-\varepsilon$

EXEMPLUL 2.5

Se consideră un alfabet al sursei $X=\{x_1, x_2, x_3\}$ cu $p_1=0,5$; $p_2=0,3$ şi $p_3=0,2$. Să construim un nou alfabet $Y=X^2=\{x_1x_1, x_1x_2, ..., x_3x_3\}$.
Să se construiască tabelul simbolurilor y_i, $i=1,...,9$ cu probabilităţile ordonate descrescător şi apoi pentru acest alfabet să se aplice algoritmul Huffman.

Soluţie:

x_I	p_I	c_I
x_1	0,5	0
x_2	0,3	10
x_3	0,2	11

Tab. 2.5 Tabelul de codare pentru sursa X

$\overline{n} = 0,5 \cdot 2 + 0,5 \cdot 1 = 1,5$ biţi /simbol

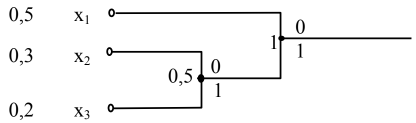

Fig. 2.3 Arbore de codare pentru alfabetul X

y_I	p_I	c_i
$Y_1=x_1\,x_1$	0,25	00
$Y_2=x_1\,x_2$	0,15	010
$Y_3=x_1\,x_3$	0,1	100
$Y_4=x_2\,x_1$	0,15	011
$Y_5=x_2\,x_2$	0,09	1100
$Y_6=x_2\,x_3$	0,06	1101
$Y_7=x_3\,x_1$	0,1	101
$Y_8=x_3\,x_2$	0,06	1110
$Y_9=x_3\,x_3$	0,04	1111

Tab. 2.6 Tabelul de codare pentru sursa Y

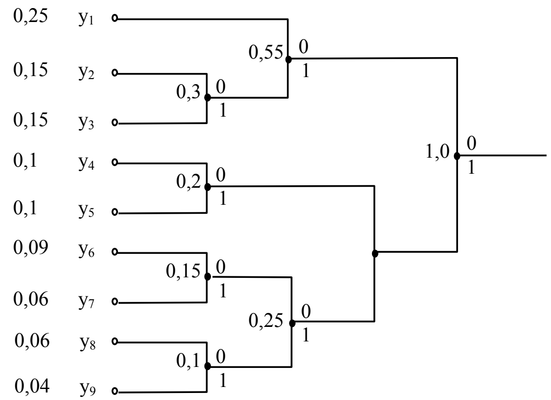

Fig. 2.4 Arbore de codare pentru alfabetul $Y = X^2$

Se obţin astfel lungimile medii ca fiind

$\overline{n}_2 = 0,25 \cdot 2 + 0,5 \cdot 3 + 0,16 \cdot 4 = 2,24$ biţi/simbol

$\overline{n}_2/2 = 1,12$ biţi /simbol

Exemplul 2.6

Metoda Huffman valorifică ideea că într-un cod optimal la $p_i > p_j$ corespunde $n_i < n_j$, şi adaugă cerinţa ca cele mai puţin probabile două mesaje să aibă aceeaşi lungime. Tehnica codării constă în rescrierea tabelei de probabilităţi intercalând în ordine descrescătoare suma ultimelor două mesaje (cele mai puţin probabile), iteraţia oprindu-se când rămân în tabel două mesaje. Combinaţia de cod se citeşte urmând traseul săgeţilor de la dreapta la stânga (figura 5.4):

Mesaj	p_i	Log(1/p_I)	n_i	s_i^* Huffman
s_1	0,4	1,32	2	1
s_2	0,18	2,47	3	001
s_3	0,10	3,32	4	011
s_4	0,10	3,32	4	0000
s_5	0,07	3,83	4	0100
s_6	0,06	4,06	5	0101
s_7	0,05	4,32	5	00010
s_8	0,04	4,64	5	00011
			$\overline{n}$ / η	2,61 97,8%

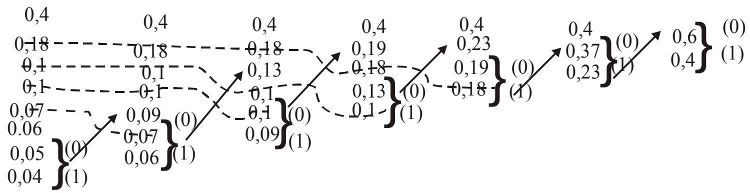

Fig 5.4

2.2.2 Algoritmul Shannon-Fano

Se presupune că cele M simboluri ale alfabetului sursei sunt ordonate în ordine descrescătoare a probabilității: $p_1 \geq p_2 \geq \ldots \geq p_M$

- Se notează cu $F_i = \sum_{k=1}^{i-1} p_k$, unde i =1, 2, ..., M și $F_1 = 0$
- Se determină un întreg n_i astfel încât

$$\log_2(1/p_i) \leq n_i < 1+\log_2(1/p_i) \qquad (2.10)$$

pentru fiecare i = 1, 2, ..., M

- Cuvântul de cod asociat simbolului x_i este reprezentarea binară a fracției F_i pe n_i biți

Observație:

Prin reprezentarea binară a unei fracții se înțelege

$b_1 b_2 \ldots b_k = b_1 \cdot 2^{-1} + b_2 \cdot 2^{-2} + \ldots + b_k \cdot 2^{-k}$, unde $b_i \in \{0, 1\}$, i = 1, ..., k

Se remarcă următoarele proprietăți ale algoritmului:

(i) Mesajele cu probabilitate mare de apariție sunt reprezentate prin cuvinte cod scurte

(ii) Cuvântul de cod pentru x_i va diferi de toate celelalte cuvinte de cod prin unul sau mai mulți biți. Prin aceasta se asigură decodarea unică. Pentru a demonstra această afirmație, se rescrie relația (2.10) sub forma

$$1/2^{n_i} \leq p_i < 1/2^{n_i+1}$$

Reprezentarea binară a lui F_i va fi diferită de toate celelalte prin unul sau mai mulți biți. Spre exemplu F_i va fi diferită față de F_{i+1} prin cel puțin bitul n_i deoarece $p_i \geq 1/2^{n_i}$. Prin aceasta cuvântul de cod pentru x_{i+1} va diferi de cel pentru x_i prin cel puțin un bit.

EXEMPLUL 2.6

Fiind dată o sursă care generează simboluri cu probabilitățile p_i specificate în coloana a doua a tabelului prezentat în continuare, să se aplice algoritmul de compresie Shannon-Fano.

Soluție:

Se remarcă ordonarea simbolurilor în sensul descreșterii probabilităților de apariție (coloanele 1 și 2). Se determină numărul de biți ai lungimii de reprezentare, completându-se astfel coloana 3 a tabelului.

Pentru simbolul x_1 numărul de biți se determină pe baza relației:

$$\log_2(128/27) \leq n_1 < 1+\log_2(128/27)$$

sau

$$2{,}245 \leq n_1 < 3{,}245$$

deducându-se că n_1 = 3 biți.

Cuvântul de cod c_1 se obține din $F_1 \triangleq 0$. În concluzie c_1 = 000. Pentru x_2 se determină n_2 = 3 biți și $F_2 = \sum_{i=1}^{1} p_i = p_1$ = 27/128. Reprezentarea binară a lui 27/128 este 0011011. Prin trunchierea acestei reprezentări la 3 biți se obține cuvântul de cod 001.

x_i	p_i	n_i	F_i	Reprezentare binară	cod c_i
x_1	27/128	3	0	.0000000	000
x_2	27/128	3	27/128	.0011011	001
x_3	9/128	4	54/128	.0110110	0110
x_4	9/128	4	63/128	.0111111	0111
x_5	9/128	4	72/128	.1001000	1001
x_6	9/128	4	81/128	.1001001	1010
x_7	9/128	4	90/128	.1011010	1011
x_8	9/128	4	99/128	.1100011	1100
x_9	3/128	6	108/128	.1101100	110110
x_{10}	3/128	6	111/128	.1101111	110111
x_{11}	3/128	6	114/128	.1110010	111001
x_{12}	3/128	6	117/128	.1110101	11010
x_{13}	3/128	6	120/128	.1111000	111100
x_{14}	3/128	6	123/128	.1111011	111101
x_{15}	2/128	6	126/128	.1111110	111111

Tab. 2.7 Tabelul de codare Shannon-Fano

Se constată că numărul de biți per simbol este $\overline{n}_v$ = 3,89 biți / simbol și se obține o eficiență ε = 62,4%, entropia fiind de 2,433 biți / simbol.

Codarea unei secvențe x_8, x_5, x_7, x_2 este făcută, conform tabelului, în șirul de biți 110010011011001.

Pentru decodare, decodorul va examina mai întâi primii 3 biți 110 (lungimea minimă a acestui cod), va concluziona că nu este cuvânt de cod și va încerca un grup de 4 biți 1100 și în acest moment se va decoda simbolul x_8 și procedura va fi reluată începând cu cel de al cincilea bit.

O formulare alternativă a algoritmului Shannon-Fano este:

Pas 1: O listă de simboluri cu probabilități date se sortează în ordine descrescătoare a probabilităților;

Pas 2: Se divide lista în două părți, astfel încât probabilitatea cumulată a porțiunii superioare să fie cât mai apropiată de probabilitatea cumulată a părții inferioare;

Pas 3: Părții superioare i se asignează o cifră binară 0, iar părții inferioare cifra binară 1;

Pas 4: Se aplică recursiv pașii 2 și 3 acestor două jumătăți divizându-se și analizând biții codului până când fiecare simbol a devenit o frunză a arborelui

In exemplul următor este ilustrată aplicarea algoritmului Shannon-Fano în această formulare alternativă.

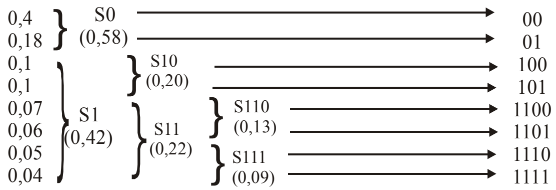
0,4
0,18
0,1
0,1
0,07
0,06
0,05
0,04
S0
(0,58)
S1
(0,42)
S10
(0,20)
S11
(0,22)
S110
(0,13)
S111
(0,09)
00
01
100
101
1100
1101
1110
1111

Fig 5.3