

CAPITOLUL 8.

Compresia datelor

8.1 Introducere

Compresia datelor este un proces de codare prin care se urmărește micșorarea redundanței mesajului generat de o sursă, pentru a reduce resursele necesare memorării sau transmiterii acestui mesaj. Deoarece, de regulă, pentru memorarea sau transmiterea unui mesaj se folosește, eventual într-o formă intermediară, reprezentarea prin simboluri binare a acestuia, și pentru că cele mai multe metode de compresie folosesc metode de codare binar - binar, putem spune că obiectivul compresiei constă în reducerea numărului de simboluri binare necesar pentru reprezentarea mesajului.

După cum șirul simbolurilor emise de sursă este împărțit, pentru codare, în subșiruri de aceeași lungime sau de lungime variabilă, și după lungimea, constantă sau variabilă, a cuvintelor de cod, codurile de compresie se clasifică în bloc- bloc, bloc – variabil, variabil – bloc și variabil – variabil, bloc – bloc indicând aceeași lungime pentru subșirurile sursei și cuvinte de cod de lungime fixă, iar variabil – variabil corespunzând unor lungimi variabile ale subșirurilor și ale cuvintelor de cod.

Din punct de vedere al măsurii în care mesajul refăcut prin decompresie se aseamănă cu cel original, asupra căruia s-a acționat prin procesul de compresie, distingem două categorii de algoritmi de compresie: fără pierderi și cu pierderi.

Algoritmii de compresie fără pierderi sunt reversibili, prin decompresie obținându-se mesajul original întocmai. Algoritmii de compresie cu pierderi au ca rezultat diferențe relativ mici între mesajul rezultat prin decompresie și cel original și sunt utilizați în compresia imaginilor și a mesajelor video și audio.

În cele ce urmează ne referim la algoritmii de compresie fără pierderi, folosiți pentru texte sau date pur binare. Dacă se dispune de mesajul complet înainte de a începe procesul de compresie, se pot utiliza algoritmi statici de compresie, cum ar fi spre exemplu codarea Shannon – Fano și codarea Huffman statică, care țin seama de frecvența caracterelor în text, caracterelor cu probabilitatea mai mare de apariție în text alocându-li-se cuvinte de cod mai scurte. Cu algoritmii de compresie adaptivi, dinamici, procesul de compresie se realizează în timp real, probabilitatea de apariție a caracterelor fiind aproximată permanent, pe măsură ce sursa generează mesajul.

O măsură uzuală pentru aprecierea performanței unui algoritm de compresie aplicat asupra unui mesaj este raportul de compresie, care poate fi definit ca raportul dintre numerele de simboluri binare prin care mesajul este reprezentat înainte și după compresie.

În afara metodelor de uz general, folosite pentru compresia textelor, sunt folosite și metode care sunt specifice anumitor aplicații, exploatând semantica locală a datelor pentru a reduce redundanța. Aceste metode sunt numite metode dependente de semantică.

8.2 Metode dependente de semantică

- Împachetarea zecimală –

În codurile pentru reprezentarea caracterelor, de 7 biți (ASCII) și de 8 biți (EBCDIC), biții 5 – 7, respectiv 5 – 8, sunt identici pentru toate cifrele zecimale, așa încât s-ar putea renunța la ei atunci când textul conține un șir mai lung de cifre. În acest scop, atunci când în text apare un șir de cifre zecimale, pentru compresie se va introduce un caracter special, de control, care semnifică începutul împachetării zecimale, urmat de un alt caracter care va indica numărul de cifre împachetate, adică reprezentate numai prin 4 biți.

- Codarea relativă –

Când se transmit date numerice cu diferențe mici între valorile succesive, se pot transmite numai o valoare inițială și apoi diferențele între valori succesive, necesitând astfel mai puțini dișiți pentru transmiterea fiecărei valori.

- Codarea lungimii secvenței –

Dacă în mesaj se repetă același caracter de mai multe ori succesiv, secvența respectivă poate fi înlocuită cu un caracter de control, semnificând faptul că urmează un șir comprimat, urmat de caracterul din șir și de specificarea lungimii șirului.

Desigur, pot fi gândite o multitudine de metode de compresie dependente de semantică. Pentru compresia datelor care reprezintă documente bancare se pot exploata diferite tipuri de redundanță locală: secvențe lungi de zerouri în câmpuri numerice, secvențe lungi de blaturi (space) în câmpuri alfanumerice, număr redus de valori ale unor atribute (de exemplu tipul contului, tipul poliței de asigurare, sex, lună, etc.) și care pot fi reprezentate printr-un număr mic de biți.

Rutina de compresie utilizată pentru baze de date de către sistemul de management al informației al IBM, exploatează redundanța locală a diferitelor tipuri de câmpuri care pot fi întâlnite: *litere*, *cifre*, *blanc*, *alte*. Pentru fiecare tip de câmp se utilizează un cod specific. Fiecare cod definește cuvintele de cod pentru toate caracterele, favorizând însă, prin cuvinte

de cod mai scurte, caracterele tipului de câmp pentru care este definit. Stabilind, spre exemplu, că pentru reprezentarea unui text se va începe cu codul *litere*, textul 120107__abcd__opqr, în care _ semnifică blank, va fi reprezentat astfel: 1 în codul *litere*, 2, 0, 1, 0, 7 și _ în codul *cifre*, a în codul *blanc*, b, c, d și _ în codul *litere*, _ și o în codul *blanc*, p, q și r în codul *litere*.

8.3 Metode de compresie adaptive pentru texte sau date pur binare

8.3.1 Algoritmul Lempel – Ziv

Algoritmul de compresie Lempel – Ziv este de tipul variabil – bloc. Sunt mai multe versiuni ale acestui algoritm, iar în cele ce urmează vor fi prezentate două versiuni ale sale.

Într-una dintre cele mai cunoscute versiuni codorul împarte șirul simbolurilor generate de sursă în subșiruri de lungime variabilă, dar fără să depășească o lungime maximă impusă. Subșirurile sunt memorate în dicționarul codorului, în ordinea în care au fost create, fiecărui subșir corespunzându-i un pointer. Un nou subșir (intrare în dicționar) se creează căutând în dicționar cel mai lung subșir care este identic cu începutul datelor care așteaptă să fie codate (subșir rădăcină) și apoi adăugând la acest subșir următorul simbol de date, numit simbol de inovare. Decodorul alcătuiește un dicționar identic, pe baza pointerului primit, care corespunde unui subșir, deja existent în dicționar și a simbolului de inovare. Fiecare nou subșir este astfel transmis prin pointerul corespunzător subșirului rădăcină și simbolul de inovare.

Dicționarul se inițializează cu subșirurile formate dintr-un singur simbol. În principiu dicționarul tinde să devină din ce în ce mai mare și, în mod corespunzător, și pointerul tinde să devină mai lung. Este necesară o procedură, folosită atât de codor cât și de decodor, pentru a elimina din dicționar șirurile rar folosite.

Tabelele 8.1 și 8.2 prezintă exemple de aplicare a algoritmului pentru un șir de simboluri binare (00101100010111010101...) și pentru un șir de simboluri dintr-un alfabet cu patru simboluri distincte (notate a, b, c și d: aabddacbccabc...) și reprezentate în binar prin 00, 01, 10 și 11.

Tabel 8.1

Dicționar codor		Transmis în linie	Date repre zentate	Date rămase de transmis	Dicționar decodor	
Subșir	Pointer (cod)				Subșir	Pointer
0	0000				0	0000
1	0001			00101100010111010101...	1	0001
00	0010	0000 0	00	101100010111010101...	00	0010
10	0011	0001 0	10	1100010111010101...	10	0011
11	0100	0001 1	11	00010111010101...	11	0100
000	0101	0010 0	000	10111010101...	000	0101
101	0110	0011 1	101	11010101...	101	0110
110	0111	0100 0	110	10101...	110	0111
1010	1000	0110 0	1010	1...	1010	1000

Tabel 8.2

Dicționar codor		Transmis în linie	Date reprezentate	Date rămase de transmis	Dicționar decodor	
Subșir	Pointer				Subșir	Pointer
a	0000	(Inițializare dicționar)			a	0000
b	0001				b	0001
c	0010				c	0010
d	0011			abadabadacadacada...	d	0011
ab	0100	0000 01	ab	adabadacadacada...	ab	0100
ad	0101	0000 11	ad	abadacadacada...	ad	0101
aba	0110	0100 00	aba	dacadacada...	aba	0110
da	0111	0011 00	da	cadacada...	da	0111
ca	1000	0010 00	ca	dacada...	ca	1000
dac	1001	0111 10	dac	ada...	dac	1001
ada	1010	0101 00	ada	...	ada	1010

Desigur, un pointer de numai patru biți limitează dicționarul la numai 16 cuvinte (subșiruri). În practică, pentru ca algoritmul să devină performant, se va folosi un dicționar mult mai voluminos. Un dicționar cu 2^n subșiruri va necesita cuvinte de cod de lungime $n + 1$ pentru un alfabet al sursei binar, $n + 2$ pentru un alfabet cu patru simboluri distincte sau $n + 8$ pentru un alfabet cu 2^8 simboluri distincte (cazul codului de 8 elemente pentru reprezentarea caracterelor). În general este nevoie ca lungimea cuvintelor de cod să fie egală cu n plus numărul de simboluri binare necesare pentru reprezentarea simbolului de inovare, adică pentru reprezentarea simbolurilor distincte ale alfabetului sursei. La începutul procesului de

codare, când subșirurile care se introduc în dicționar sunt de lungime mică, algoritmul nu este eficient, dar pe măsură ce se avansează în procesul de codare subșirurile devin mai lungi și compresia devine mai eficientă. Atunci când dicționarul, finit, este plin, este necesar ca intrările mai vechi, care nu mai sunt folosite, să fie eliminate.

Un dezavantaj al algoritmului prezentat mai sus constă în faptul că simbolul de inovare este transmis necomprimat, ceea ce reduce sensibil randamentul compresiei. Spre exemplu, presupunând un alfabet format din 256 de simboluri, reprezentate fiecare prin opt biți și că entropia sursei este de 3 biți/simbol (octet), algoritmul va coda subșirurile lungi într-un cod apropiat de entropia sursei, dar va adăuga simbolul de inovare în forma necodată. Dacă subșirurile au o lungime medie de 5 simboluri (din alfabetul sursei), înseamnă că pointerii ar avea o lungime ceva mai mare de $5 \times 3 = 15$ simboluri binare, iar cuvintele de cod vor fi de lungime ceva mai mare de $15 + 8 = 23$ simboluri binare, rezultând un raport de compresie de aproximativ 48/23 (aprox. 2,08), în loc de 8/3 (aprox. 2,66).

Tabel 8.3

Dicționar codor		Transmis în linie	Date reprezentate	Date rămase de transmis	Dicționar decodor	
Subșir	Pointer				Subșir	Pointer
0	0000				0	0000
1	0001			00101100010111010101...	1	0001
		0000	0	0101100010111010101...		
00	0010	0000	0	101100010111010101...	00	0010
01	0011	0001	1	01100010111010101...	01	0011
10	0100	0011	01	100010111010101...	10	0100
011	0101	0100	10	0010111010101...	011	0101
100	0110	0010	00	10111010101...	100	0110
001	0111	0100	10	111010101...	001	0111
101	1000	0001	1	11010101...	101	1000
11	1001	0001	1	1010101...	11	1001
(11)	(1001)	1000	101	0101...	(11)	(1001)
1010	1011	0011	01	01...	1010	1011

Varianta Miller – Wegman a algoritmului Lempel – Ziv elimină acest dezavantaj al transiterii simbolului de inovare în forma necodată, amânând specificarea sa la iterația următoare, fiind primul simbol al subșirului următor transmis și fiind astfel transmis într-o

formă codată. Un nou subșir introdus în dicționar se formează din subșirul transmis anterior, la care se adaugă primul simbol (ca simbol de inovare) al actualului șir transmis (codat) și recepționat. Această variantă este exemplificată, în tabelele 8.3 și 8.4, folosind aceleași mesaje considerate mai sus.

Tabel 8.4

Dicționar codor		Transmis în linie	Date reprezentate	Date rămase de transmis	Dicționar decodor	
Subșir	Pointer				Subșir	Pointer
a	0000	(Inițializare dicționar)			a	0000
b	0001				b	0001
c	0010				c	0010
d	0011			abadabadacadacada...	d	0011
		0000	a	badabadacadacada...		
ab	0100	0001	b	adabadacadacada...	ab	0100
ba	0101	0000	a	dabadacadacada...	ba	0101
ad	0110	0011	d	abadacadacada...	ad	0110
da	0111	0100	ab	adacadacada...	da	0111
aba	1000	0110	ad	acadacada...	aba	1000
ada	1001	0000	a	cadacada...	ada	1001

Avantajul variantei Miller – Wegman a algoritmului Lempel – Ziv constă în faptul că un dicționar de mărime 2^n necesită un cuvânt de cod pentru transmisiune de lungime n și nu n plus numărul de simboluri binare folosite pentru reprezentarea simbolurilor sursei, ceea ce înseamnă o reducere semnificativă atunci când alfabetul sursei este mare.

8.3.2 Recomandarea ITU – T V.42 bis

Recomandarea V.42 bis prezintă, ca funcție opțională a modemurilor, o procedură de compresie a datelor, când acestea reprezintă texte, asemănătoare variantei Miller - Wegman a algoritmului Lempel – Ziv.

Dicționarele conțin un set de arbori, rădăcina fiecărui arbore corespunzând unui caracter din alfabetul în care este scris textul. Cu un alfabet ale cărei caractere sunt reprezentate prin opt biți vor fi 256 de arbori. Un arbore reprezintă setul șirurilor cunoscute începând cu un caracter specific, asociat rădăcinei arborelui, fiecare nod din arbore corespunzând unui șir din set. În fig. 9.1 este prezentat un exemplu pentru arborii corespunzători, la un moment oarecare, caracterelor A și B. Șirurile reprezentate de acești

arbori sunt: A, AR, ARD, B, BA, BAN, BAR, BAT, BI, BIT. Fiecărui nod i se alocă un cuvânt de cod, care identifică în mod unic nodul și, prin urmare, și șirul asociat.

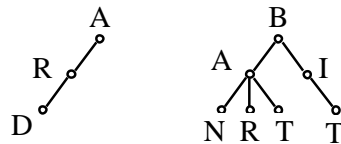


Fig. 8.1 Arbori în dicționar

Modemul care are implementată această procedură poate funcționa cu sau fără compresia datelor. Când modemul funcționează cu compresia activată el are două moduri de lucru, comutarea între ele fiind automată, în urma testării compresibilității datelor, prin care se estimează eficiența procesului de codare și se selectează modul care conduce la maximizarea eficienței: cu compresie sau transparent. În modul transparent nu se realizează procedura de compresie, ca fiind neeficientă, deși inițial s-a activat funcția de compresie. Pentru semnalarea la recepție a modului de lucru sunt utilizate trei cuvinte de cod, distincte de cele care rezultă prin procesul de compresie.

- Inițializarea dicționarelor –

În starea inițială fiecare arbore are doar nodul rădăcină. Cuvântul de cod asociat fiecărui nod rădăcină este 3 (numărul cuvintelor de cod cu rol de control) plus valoarea ordinală a caracterului reprezentat de nod (echivalentul numeric al codului binar al caracterului). Spre exemplu, dacă literei A îi corespunde, în alfabetul sursei, octetul 01000001, valoarea ordinală este 65_{10} și cuvântul de cod asociat, într-un cod de nouă elemente, va fi 001000100, după cum rezultă din operația $3 + 65 = 68$:

$$\begin{array}{r}
 00000011 + \\
 \underline{01000001} \\
 \mathbf{001000100} = 68_{10}
 \end{array}$$

Noile cuvinte de cod vor fi indicate de un numărător C_1 , setat inițial la valoarea 100000011 ($259_{10} = 2^8 + 3$) și va reprezenta cuvântul de cod asociat primului subșir care va fi creat și introdus în dicționar atunci când compresia va funcționa. Cuvintele începând de la 0 până la 258 inclusiv sunt deja alocate (256 rădăcini ale arborilor plus trei cuvinte de control).

- Funcțiunile dicționarelor –

Dicționarul din modemul transmițător trebuie să realizeze **selectarea** celui mai lung șir existent în dicționar, care coincide cu începutul textului rămas de transmis, primit de la terminalul de date. În linie se va transmite cuvântul de cod asociat nodului care reprezintă șirul selectat. Dicționarul se va **actualiza** prin introducerea unui nou șir, format din șirul

transmis anterior, la care se adaugă, drept caracter de inovare, primul caracter al șirului actual transmis, ceea ce înseamnă adăugarea unui nou nod la arborele corespunzător șirului transmis anterior. Cuvântul de cod asociat noului nod este indicat de numărătorul C_1 , numărător care se incrementează cu o unitate după fiecare șir (nod) nou introdus în dicționar. Desigur, dimensiunea dicționarului este finită. Dacă această dimensiune este mai mare de $512 = 2^9$, după ce numărătorul C_1 ajunge la starea 511 se va trece la reprezentarea cuvintelor de cod prin 10 biți. Când dicționarul este plin este necesar să se elimine anumite șiruri din dicționar pentru a **recupera** cuvintele de cod corespunzătoare. În acest scop numărătorul C_1 este readus la starea inițială (259) și dacă nodul căruia îi este asociat acest cuvânt de cod (259) este un nod frunză (nod de terminație a unei ramuri) șirul corespunzător va fi eliminat (este un șir vechi care n-a mai fost utilizat) și cuvântul de cod astfel disponibil va fi alocat unui nou șir, după care numărătorul este incrementat cu o unitate. Dacă nodul căruia îi este asociat cuvântul de cod reprezentat de starea numărătorului nu este un nod frunză, numărătorul se incrementează cu o unitate ș.a.m.d.

Dicționarul decodului va realiza funcțiile de actualizare și de recuperare. Pe baza cuvântului de cod primit, existent în dicționar, decodul va determina șirul recepționat și, totodată, va introduce un nou șir în dicționar și cuvântul de cod asociat, după aceleași reguli după care funcționează și codorul transmițătorului. De asemenea, numărătorul decodului va funcționa ca și numărătorul codului, iar recuperarea unui cuvânt de cod se va face ca și în codor.

8.4 Compresia datelor în transmisiunile facsimil

Terminalele facsimil permit transmiterea documentelor pe rețeaua telefonică cu comutație sau pe rețeaua digitală cu servicii integrate (ISDN – Integrated Services Digital Network). După capacitățile lor terminalele facsimil sunt clasificate de către ITU – T (Recomandarea T.0) în patru grupuri. Terminalele din grupurile 1 și 2, având performanțe reduse, nu mai sunt în fabricație și nici în exploatare. Terminalele din Grup 3 (G3) și cele din Grup 4 (G4) realizează transmisiuni de documente alb - negru și, opțional, color, pe rețeaua telefonică cu comutație (G3) și pe ISDN (G4). Pentru terminalele G3 se specifică (Recomandarea ITU - T T.4), ca opțiune, capacitatea de a funcționa la debitul de 64 Kb/s pe ISDN. Sunt două soluții tehnice pentru această opțiune: una bazată pe protocolul corespunzător aparatelor G4, numită opțiunea F (G3F), care poate interfuna direct cu terminale G4, cealaltă, numită opțiunea C (G3C) necompatibilă direct cu G4/G3F.

Considerând aria mesajului într-un plan vertical, aceasta trebuie analizată linie cu linie, de la stânga la dreapta și de sus în jos. La terminalele facsimil Grup 3, cele mai frecvente în prezent în exploatare, rezoluția standard este, pe verticală, de 3,85 linii/mm, echivalent cu 100 linii/inch și, pe orizontală, 7,7 puncte (pels – picture elements)/mm, echivalent cu 200 puncte/inch. Opțional pot fi realizate rezoluții mai fine, de 200, 300 sau 400 linii/inch și 300 sau 400 puncte/inch.

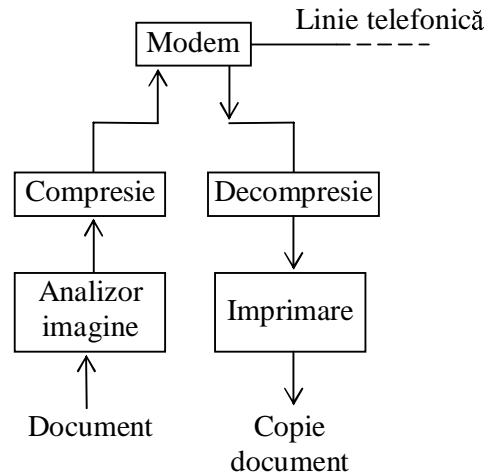


Fig. 8.2 Schema bloc a unui aparat telefacsimil

Schema bloc a unui aparat telefacsimil este prezentată în fig. 8.2. Cu 7,7 linii/mm, rezoluție verticală și 7,7 puncte/mm, rezoluție orizontală, la explorarea unei pagini A4 vor fi 2376 linii orizontale, fiecare cu 1728 pels, un total de peste 4,1 milioane de biți. În cursul explorării documentului unui punct alb îi corespunde bitul 0, iar unui punct negru îi corespunde bitul 1. Deoarece, datorită câmpurilor lungi de alb și negru din imaginea explorată, rezultă secvențe lungi de biți 0 și de biți 1, este foarte indicat să se folosească o metodă de compresie. Metodele de compresie recomandate pentru aparatele telefacsimil asigură, în funcție și de specificul documentelor explorate, rapoarte de compresie cuprinse între 10 și 20.

Pentru aparatele facsimil Grup 3 ITU – T recomandă (T.4) o metodă de compresie unidimensională, prin care secvențele de puncte albe și cele de puncte negre de diferite lungimi, din aceeași linie orizontală, sunt reprezentate prin cuvinte de cod de lungimi diferite. Alocarea cuvintelor de cod se bazează pe numeroase analize ale documentelor tipice explorate, urmărindu-se ca secvențelor de lungimi frecvente să li se aloce cuvinte de cod mai scurte, pentru a asigura o eficiență a compresiei cât mai ridicată. Desigur, în explorarea unei

linii, secvențele de alb și de negru alternează. Metoda de compresie este unidimensională pentru că se aplică linie cu linie, ținând seama doar de repetările din cadrul unei linii. Cuvintele de cod sunt grupate în două tabele, tabelul codurilor de terminație și tabelul codurilor de asamblare. Tabelul codurilor de terminație conține cuvintele de cod alocate secvențelor de puncte albe și celor de puncte negre cu lungimi de la zero la 63. Tabelul codurilor de asamblare conține cuvintele de cod alocate secvențelor ale căror lungimi sunt multipli lui 64 (64, 128, . . . , 1728 = 27 x 64). Fiecare secvență de puncte de aceeași culoare este reprezentată de un cod de terminație sau de un cod de asamblare și un cod de terminație. Secvențele cu lungimi de la zero la 63 sunt reprezentate prin codurile de terminație asociate. Secvențele cu lungimea de la 64 la 1728 sunt reprezentate prin codul de asamblare care reprezintă o lungime egală sau mai mică cu cea a secvenței. Acest cod este urmat de codul de terminație asociat diferenței dintre lungimea secvenței și lungimea reprezentată de codul de asamblare. Spre exemplu, o secvență de puncte negre, de lungime 140, va fi reprezentată prin codul de asamblare corespunzător unei secvențe de 128 puncte negre (000011001000), urmat de codul de terminație pentru o secvență de două puncte negre (11), adică prin 00001100100011.

Pentru a asigura sincronizarea de culoare a receptorului fiecare linie începe cu un cod corespunzător unei secvențe de puncte albe. Dacă linia explorată începe cu un câmp negru, se va transmite mai întâi codul corespunzător secvenței de puncte albe de lungime zero.

Nu se prevede utilizarea unui protocol de corectare a erorii pentru aparatele de Grup 3. Dacă intervine o grupare de erori este posibil ca receptorul să piardă sincronizarea de linii. Pentru a permite sincronizarea de linii fiecare linie de date se termină cu un cuvânt de cod unic, numit sfârșit de linie (EOL – End of line), reprezentat de 000000000001. De asemenea, înainte de a transmite prima linie se va transmite EOL. Dacă, în urma a unei desincronizări, receptorul eșuează în încercarea de a găsi EOL, după un anumit număr de linii va informa transmițătorul.

Alte metode de compresie, opționale pentru aparatele de Grup 3, realizează o codare bidimensională, prin care se ține seama nu numai de repetările dintr-o linie, ci și de pozițiile relative ale punctelor care reprezintă schimbarea de culoare din linii adiacente, linia anterioară fiind referință pentru linia în curs de codare. Unele dintre aceste metode, mai complexe și, în același timp, mai eficiente, nu pot fi utilizate decât dacă se realizează și controlul erorii.