ALGORITMI DE COMPRESIE FOLOSIȚI ÎN SISTEMELE MODERNE DE ARHIVARE. CODAREA HUFFMAN

1. Obiectul lucrării

Lucrarea îşi propune familiarizarea cu unii dintre algoritmii de compresie cei mai utilizați (Huffman static, Huffman dinamic varianta FGK), precum şi prezentarea unui program de simulare pe calculator care ilustrează modul de compresie.

2. Noțiuni teoretice

2.1. Noțiuni generale despre compresie

Compresia este procesul de minimizare a spațiului ocupat sau a timpului necesar transmiterii unei anumite cantități de informație.

Metodele de compresie pot fi împărțite în:

-metode de compresie cu pierdere;

-metode de compresie fără pierdere.

Metodele de compresie cu pierdere de informație sunt folosite în special în transmisia semnalului audio şi video, unde pierderea de informație are ca rezultat o scădere a calității sunetului, respectiv imaginii, ele neconstituind obiectul prezentei lucrări.

Conceptul de compresie de date fără pierdere pare imposibil la prima vedere, dar o analiză mai atentă face ca această idee, de compresie fără pierdere, să aibă sens. Astfel, dacă ne gândim la prescurtările din viața de zi cu zi (abrevieri: prof., etc., CEC, ş.a.) acestea apar ca o formă primitivă a compresiei de date. Compresia de date fără pierdere, prezentă în programele de arhivare, în sistemele de transmisiune a datelor, a evoluat de-a lungul timpului pornind de la algoritmi simpli (suprimarea zerourilor, codarea pe şiruri) şi ajungând la algoritmii complecşi folosiți în prezent.

Avantajele compresiei sunt:

-reducerea spațiului necesar depozitării unei cantități de informație;

-scăderea timpului de transmitere a unor mesaje, ceea ce duce la scăderea costului per mesaj şi posibiltatea creşterii traficului într-o rețea de transmisiuni. Această scădere a timpului este efectul direct al micşorării cantității de informație, dar şi efectul indirect al micşorării pierderilor de timp datorate protocoalelor de comunicație.

- scăderea timpului de rulare a unui program datorită timpului de acces la disc.

Metodele de compresie fără pierderi au la bază ideea că, în general, cantitatea de informație prelucrată (transmisă, depozitată) conține o anumită redundanță care se datorează:

-distribuției caracterelor (unele caractere au o frecvență de apariție mult mai mare decât altele);

-repetării consecutive a unor caractere;

-distribuției grupurilor de caractere (unele grupuri sunt mult mai frecvente decât altele şi în plus există grupuri care nu apar deloc);

- distribuției poziției (unele caractere sau grupuri ocupă poziții preferențiale, predictibile în anumite blocuri de date).

Având în vedere toate aceste tipuri de redundanțe putem înțelege de ce o anumită tehnică de compresie poate da un rezultat bun pentru un anumit tip de surse, pentru altele însă rezultatul poate fi dezastruos. De aici este uşor de înțeles că noile direcții în studiul compresiei urmăresc obținerea unui algoritm care să ofere o compresie cât mai bună pentru tipuri de surse cât mai diferite.

Aprecierea cantitativă a compresiei realizate se face utilizând factorul de compresie definit ca: $F_c = \frac{n_u}{n_c}$, unde n_u este lungimea în biți a mesajului inițial şi n_c lungimea după

compresie, precum şi prin eficienţa codării definită prin $\eta = \frac{H(S)}{\bar{n} \cdot \log_2 m}$, unde $H(S)$ reprezintă entropia sursei informaționale ce este codată, m reprezintă numărul simbolurilor din alfabetul de codare iar $\bar{n}$ reprezintă lungimea medie a cuvintelor.

2.2. Clasificarea algoritmilor de compresie fără pierderi:

Algoritmii de compresie de date fără pierdere pot fi încadrați în una din următoarele categorii:

-Algoritmi statici, care se bazează pe o statistică bine cunoscută a sursei şi dau rezultate bune în cazul în care sursele au o statistică asemănătoare cu cea presupusă. În caz contrar, rezultatul poate fi o extensie în loc de o compresie.

-Algoritmi semiadaptivi sau în două treceri, algoritmi care folosesc statistica simbolurilor mesajului, statistică ce se obține printr-o parcurgere inițială a întregului mesaj. Aceşti algoritmi oferă rezultate pentru orice tip de sursă, dar nu pot fi folosiți într-o transmisiune.

-Algoritmi adaptivi, care sunt de obicei cei mai potriviti pentru orice tip de sursă. Tehnicile adaptive au ca idee construirea unui "dicționar de codare" al simbolurilor mesajului, paralel cu codarea pentru compresie a mesajului. Acest dicționar se va construi identic la recepție, pe baza informației recepționate.

Lucrarea va prezenta câteva din cele mai folosite metode de compresie fără pierderi.

2.3. Algoritm Huffman static

Codarea Huffman statică binară

Algoritmul Huffman (1952) constituie un algoritm optimal, în sensul că nici un alt algoritm nu asigură o mai mică lungime medie a cuvintelor. Sunt situații în care şi alți algoritmi pot da o lungime medie egală cu cea dată de algoritmul Huffman, dar niciodată mai mică.

Paşii algoritmului sunt:

1. Ordonarea mesajelor sursei în ordinea descrescătoare a probabilităților.
2. Formarea din ultimele două mesaje a unui mesaj restrâns $r_1 = s_{M-1} \cup s_M$ având $p(r_1) = p(s_{M-1}) + p(s_M)$
3. Includerea mesajului r_1 în mulțimea celorlalte mesaje, în ordinea descrescătoare a probabilităților, alcătuind şirul ordonat R_1.
4. Repetarea algoritmului de restrângere până când ultimul şir ordonat R_k conține doar două mesaje.
5. Cuvintele de cod corespunzătoare fiecărui mesaj se obțin în felul următor:
   - simbolului r_{k-1} i se aloca '0', iar lui r_k '1'
   - la fiecare diviziune în două se alocă, în aceeaşi ordine ca la prima diviziune, simbolurile '0' sau '1' şi operația se continuă până când se ajunge la mulțimi formate dintr-un singur mesaj.

Exemplu:

a) Să se codeze prin algoritmul Huffman binar sursa caracterizată de următoarea distribuție:

$$S: \begin{pmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ 0.3 & 0.25 & 0.15 & 0.15 & 0.10 & 0.05 \end{pmatrix}$$

b) Să se determine eficienţa codării şi factorul de compresie.

Soluţie:

S	p_i	c_i	Restrângeri			
			R_1	R_2	R_3	R_4
s_1	0.3	00	0.3	0.3	0.4	0.6 **0**
s_2	0.25	10	0.25	0.3	0.3 **00**	0.4 **1**
s_3	0.15	11	0.15	0.25 **10**	0.3 **01**	
s_4	0.15	010	0.15 **010**	0.15 **11**		
s_5	0.10	0110	0.15 **011**			
s_6	0.05	0111				

b) $\bar{n} = \sum_{i=1}^{6} p_i n_i = 2.45$ - lungimea medie a cuvintelor de cod

$H(S) = -\sum_{i=1}^{6} p_i \log_2 p_i = 2.39$ - entropia sursei informaționale S.

$\eta = \frac{2.39}{2.45} = 0.98 = 98\%$ - eficiența codării

$F_c = \frac{n_u}{\bar{n}} = \frac{3}{2.45} = 1.22$ - factorul de compresie iar n_u se obține din relația $n_u = \lceil \log_2 M \rceil$, M fiind numărul de mesaje din sursa S, $\lceil x \rceil$ reprezintă cel mai mic întreg mai mare sau egal cu x.

2.4. Algoritm Huffman dinamic.

Se porneşte de la premiza că varianta statică a algoritmului de compresie Huffman este bine înțeleasă.

Ideea de bază în codarea Huffman dinamică este folosirea pentru codarea simbolului t_{i+1} din mesaj, a unui arbore de codare (un dicționar de codare sub forma unui arbore binar) construit pe baza primelor i simboluri din mesaj. După transmiterea simbolului t_{i+1} se va revizui arborele de codare în vederea codării simbolului t_{i+2}. Există mai multe variante ale algoritmului Huffman dinamic (FGK, Δ) care diferă între ele prin modul de construcție al arborelui. În continuare vom prezenta algoritmul de compresie, ilustrându-l cu exemplul din figura 1.

Procedura generală de compresie pentru codarea Huffman dinamică are următorul algoritm:

1. inițializez arborele de codare cu un nod rădăcină;
2. transmit primul simbol în clar (de exemplu codul ASCII al simbolului);
3. construim încă două noduri (frunze) care pornesc din nodul rădăcină, unul la stânga, care nu conține nici un simbol, căruia îi ataşăm ponderea nulă (frunză goală) şi unul la dreapta care conține simbolul apărut, ponderea acestuia devenind egală cu 1;
4. citim următoarea literă din mesaj;

5. dacă litera este deja în arbore transmit codul ei din arbore. Codul literii din arbore se formează citind simbolurile de '0' respectiv '1' ataşate ramurilor, pornind de la nodul rădăcină până la nodul în care se află litera. Simbolurile de '0' respectiv '1' se ataşează ramurilor astfel: toate ramurile din dreapta vor avea ataşate simbolul '1', iar toate ramurile din stânga vor avea ataşate simbolul '0'. Apoi se trece direct la pasul 7;

6. dacă litera nu este în arbore atunci transmit codul nodului terminal care nu conţine nici un simbol (frunză goală) urmat de codul în clar(ASCII) al literei. Codul nodului fără nici o literă se formează ca şi în cazul nodului care conţine o literă;

7. reactualizez arborele; dacă mesajul s-a terminat, atunci codarea este terminată, iar în caz contrar reluăm procedeul începând cu pasul 4.

OBS. Diferitele variante de codare dinamică diferă doar prin modul de reactualizare al arborelui (tabelei de decodare).

Procedura generală de decompresie pentru codarea Huffman dinamică are următorul algoritm:

1. iniţializez arborele de codare cu un nod rădăcină;
2. citesc primul simbol transmis în clar (codul ASCII al simbolului);
3. transmit litera mai departe;
4. construim încă două noduri care pornesc din nodul rădăcină, unul la stânga, care nu conţine nici un simbol, căruia îi ataşăm ponderea nulă şi unul la dreapta care conţine simbolul apărut, ponderea acestuia devenind egală cu 1;
5. citesc codul transmis (bit cu bit) până când codul se află în arbore, adică până când am ajuns la un nod frunză; nod frunză = nod care se află în arbore pe ultimul nivel de ierarhie;
6. dacă nodul ataşat codului citit este un nod care nu conţine nici un simbol citesc litera transmisă în clar şi o transmit mai departe, apoi trecem direct la pasul 8;
7. dacă nodul ataşat codului citit conţine un simbol atunci decodez codul ataşat lui prin litera pe care o conţine şi transmit litera mai departe (la utilizator);
8. reactualizăm arborele; dacă mesajul s-a terminat, atunci decodarea este terminată, iar în caz contrar reluăm procedeul începând cu pasul 5.

Varianta FGK:

În continuare vom prezenta algoritmul de reactualizare al arborelui penru varianta FGK (Faller, Gallager, Knuth). Vaianta FGK urmăreşte minimizarea $\Sigma_j w_j l_j$ (sumă ce reprezintă lungimea în biţi a mesajului codat), unde w_j reprezintă ponderea frunzei j asociată simbolului j (numărul de apariţii ale simbolului j), iar l_j lungimea codului asociat frunzei respective. Dacă reactualizarea arborelui se face după procedeul de mai jos atunci această minimizare este satisfăcută.

Procedura de reactualizare arbore FGK:

Obs. În cele ce urmează ne vom referi prin noţiunea de nod frunză la un nod terminal din arbore.

1. Se presupune că suntem la pasul algoritmului Huffman dinamic în care s-a citit simbolul t_i. Notăm cu q nodul frunză corespunzătoare simbolului t_i dacă acesta este deja în arbore, sau frunză goală dacă simbolul nu este încă în arbore;
2. Dacă q este frunză goală atunci înlocuiesc nodul respectiv cu un nod părinte şi două noduri derivate (noduri fii) din acesta care nu conţin nici un caracter (sunt noduri nule, de pondere nulă), cele trei noduri nou create se notează astfel: nodul din stânga cu 1, nodul din dreapta cu 2 iar nodul părinte cu 3. Se incrementează numărul de ordine al celorlalte noduri. Apoi se notează cu q nodul derivat din dreapta tocmai creat. Tot acestui nod i se asociază şi simbolul citit t_i care nu este în arbore;

3. Ponderea nodului q căruia i s-a asociat simbolul citit t_i se incrementează cu o unitate. Se modifică ponderea nodurilor intermediare şi a nodului rădăcină astfel încât aceasta să fie egală cu suma ponderilor fiilor săi. Se schimbă nodul q cu nodul de pondere cea mai mică şi cu cel mai mare număr de ordine dacă există situații de acest tip până când se ajunge la nodul rădăcină.

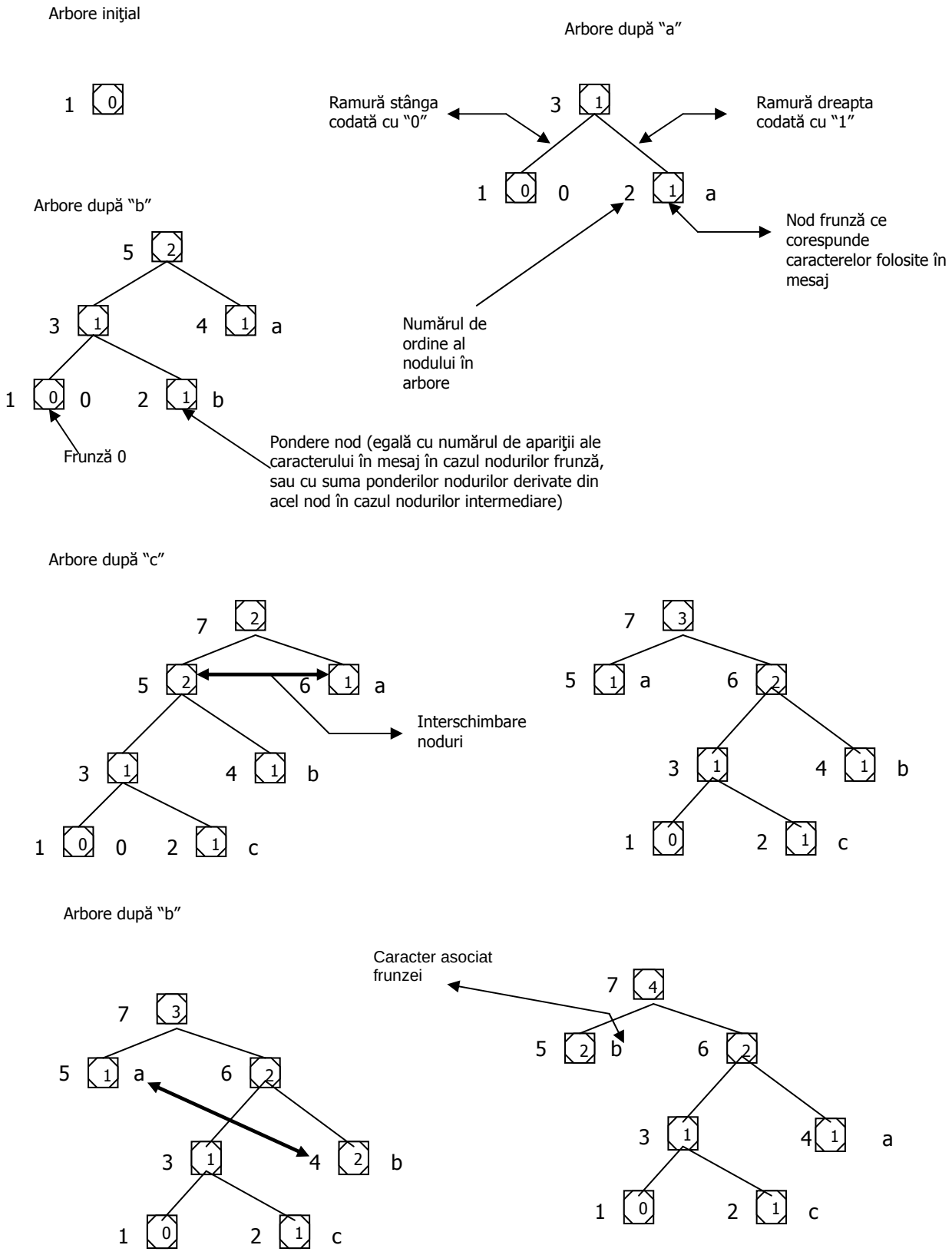

Figura 1. Evoluția arborelui Huffman FGK în cazul codării mesajului "abcb…"
Secvența binară obținută este "a0b00c11" (a,b,c reprezintă cei 8 biți ai codului ASCII)

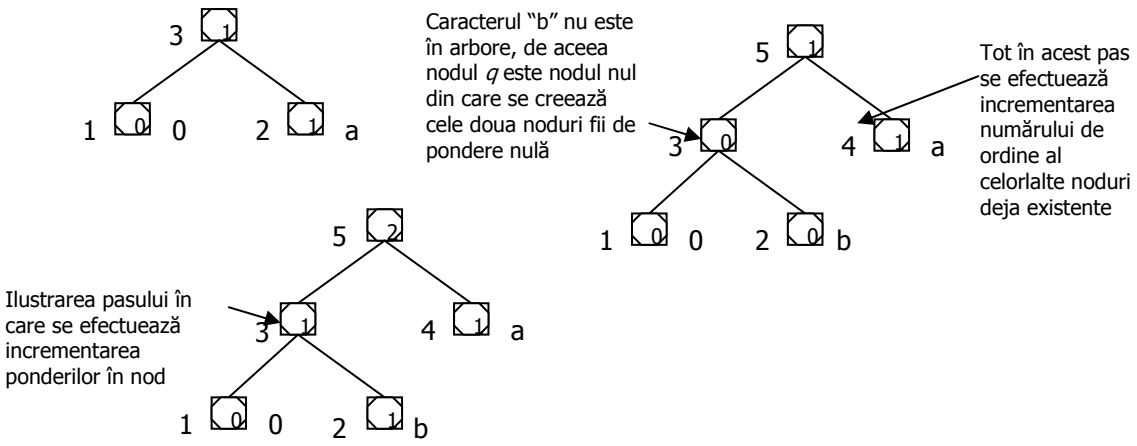

Figura 2. Reactualizarea arborelui la citirea primului simbol "b"

3. Desfăşurarea lucrării

a) Fie sursa informațională S caracterizată de distribuția $S:\begin{pmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ 0.1 & 0.2 & 0.3 & 0.15 & 0.05 & 0.2 \end{pmatrix}$. Să se realizeze o codare binară Huffman statică şi apoi să se calculeze eficiența codării şi factorul de compresie.

b) Pentru secvența "abcdeaa" desenați evoluția arborelui de codare dinamică Huffman FGK şi determinați secvența binară codată. Calculați raportul de compresie obținut.

c) Folosind programul de simulare Huffman FGK introduceți secvența de la punctul b) şi urmăriți evoluția arborelui şi a mesajului codat în concordanță cu rezolvarea de la punctul b).