Laborator PS

ALGORITMI DE COMPRESIE

Algoritmul Huffman adaptiv - exemplu

Prof. dr. ing. Dan STEFANOIU

As. Ing. Alexandru DUMITRASCU

Exemplu – criptarea/decriptarea adaptiva Huffman

Criptarea adaptiva Huffman

Acest exemplu pune in evidenta cateva aspecte importante pentru algoritmul de generare si manevrare a codurilor in faza de compresie.

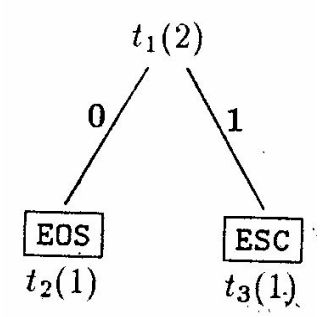
Obs.:

- Ori de cate ori simbolul citit nu se regaseste printre frunzle arborelui binar, sunt emise doua coduri succesive: primul este cel al simbolului de evitare ESC (cod preluat din arborele Huffman inainte de reactualizare), iar al doilea este codul original pe 8 biti (nemodificat) al simbolului.

- Odata cu mutarea unui nod in alta pozitie (in cursul reactualizarii arborelui), este posibil ca unul sau mai multi arbori descendenti sa fie, la randul lor, mutati in alte pozitii.

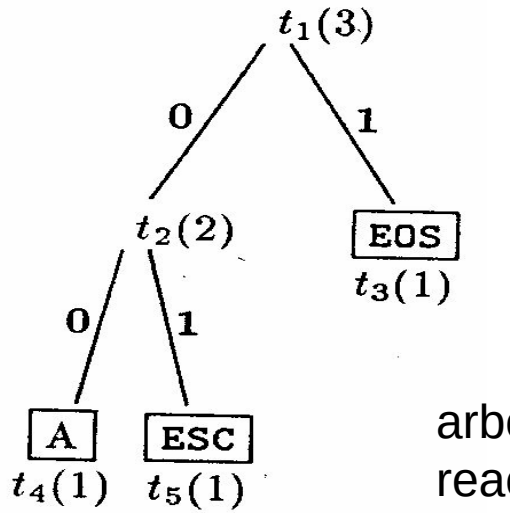
Reactualizarea codurilor frunzelor consta in reactualizarea partii mai semnificative a vechiului cod, caci partea mai putin semnificativa nu este afectata de mutare.

Se considera setul de date format din 4 simboli, primii 3 fiind diferiti: **ADIA**.
Evolutia arborelui Huffman este redata in figurile de mai jos.



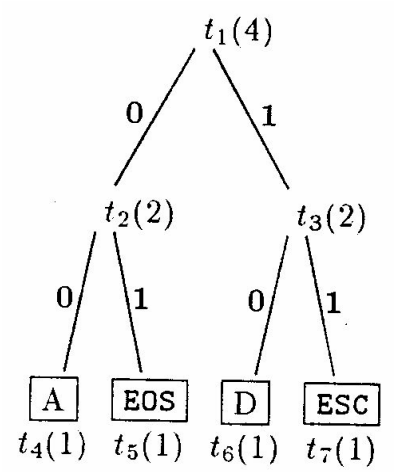
arborele initial

Se citeste simbolul „A” si se emit doua coduri (9 biti): ESC si A = 1 01000001.



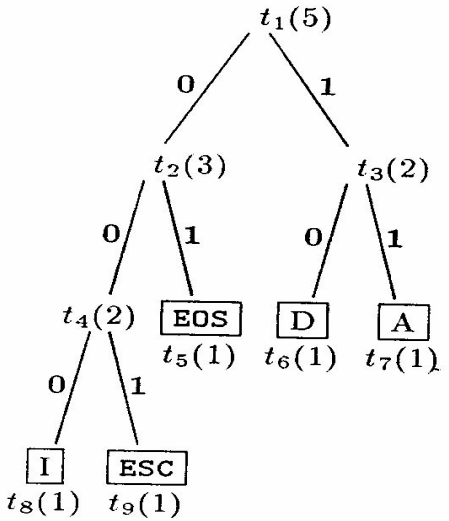
arborele reactualizat

Se citeste simbolul „D” si se emit doua coduri (10 biti): ESC si D = 01 01000100.



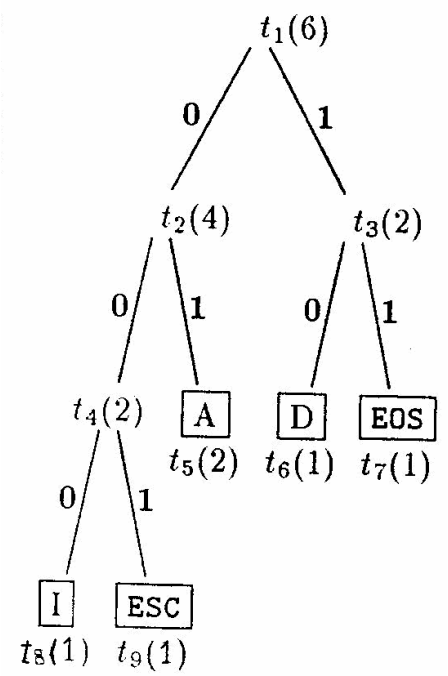
arborele reactualizat

Se citeste simbolul „I” si se emit doua coduri (10 biti): ESC si I = 11 01001001.



arborele reactualizat

Se citeste simbolul „A” si se emite codul (2 biti): A = 11.



arborele reactualizat

Decriptarea adaptiva a simbolilor in faza de decompresie

Spre deosebire de constructia adaptiva a unui cod, decriparea sa adaptiva este o operatie mult mai simpla, ea presupunand doar o parcurgere descendenta a arborelui.

Algoritmul de decompresie adaptiva:

Pas 1: Se citeste fluxul de intrare la nivel de bit (in mod normal se citeste cate un octet, in cadrul caruia se analizeaza fiecare bit).

In paralel, se construieste si se reactualizeaza arborele Huffman intocmai ca in faza de compresie. Initial, arborele este alcatuit doar din simbolii virtuali EOS si ESC. Primul simbol decriptat va fi EOS (cod 0), daca fluxul de intrare este vid, sau ESC (cod 1), in caz contrar. Simbolul ESC va fi urmat de codul original (pe 8 biti) al primului simbol valid din setul de date.

Pas 2: Se decripteaza simbolul curent de pe fluxul de intrare, prin parcurgerea arborelui Huffman existent, incepand de la radacina. Parcurgerea arborelui este ghidata de succesiunea bitilor fluxului de intrare.

- Daca simbolul decriptat este ESC, atunci se citeste urmatorul octet de pe fluxul de intrare. Acesta va fi codul original al simbolului curent.

- Daca simbolul decriptat este diferit de ESC, atunci simbolul a mai aparut pe fluxul de intrare, frunza ocupata de el in arborele Huffman oferind codul original (pe 8 biti) al acestuia.

In ambele situatii descrise mai sus, codul original al simbolului trebuie inscris pe fluxul de iesire. Daca ultimul simbol decriptat este EOS, algoritmul se incheie. In caz contrar, se trece la pasul urmator.

3. Se reactualizeaza arborele Huffman (conform algoritmului de reactualizare). Algoritmul se reia de la pasul 2.

Exemplu: **ADIA**

Fluxul de intrare in procedura de decompresie este constituit din urmatoarele coduri (33 de biti):

1 01000001 01 01000100 11 01001001 11 11

- se citeste bitul 1 si se decripteaza simbolul ESC, folosind arborele Huffman initial. Codul sau original nu este emis, insa, pe fluxul de iesire. Operatia care se executa in continuare este citirea urmatorului octet de pe fluxul de intrare (litera „A”), care este inscris nemodificat pe fluxul de iesire.

- se reactualizeaza arborele Huffman si se reseteaza parcurgerea sa din radacina.

- se citesc bitii 01 si se constata ca simbolul decriptat este tot ESC, folosind arborele Huffman reactualizat. In consecinta, urmatorul octet al fluxului de intrare (litera „D”) este inscris nemodificat pe fluxul de iesire.

- urmeaza o noua reactualizare a arborelui.

- urmatorii 2 biti cititi - 11 - conduc tot la simbolul de evitare ESC, motiv pentru care urmatorul octet citit (litera „I”) este si el descarcat direct pe fluxul de iesire.

- se reactualizeaza arborele, care va avea acum 5 frunze.

- se reia citirea fluxului de intrare. Dupa codul 11, se constata ca, folosind arborele Huffman curent, se ajunge la nodul frunza ocupat de litera „A”. Codul original al acestuia (01000001) este preluat din informatia de nod a arborelui si descarcat pe fluxul de iesire.

- o noua reactualizare a arborelui binar nu aduce noi frunze acestuia, ci o organizare diferita.

- utimii 2 biti cititi – 11 – conduc la decriptarea simbolului EOS, care arata ca decompresia a luat sfarsit si provoaca inchiderea fluxului de iesire.

Din acest exemplu se observa ca un acelasi cod (11) poate avea semnificatii total diferite la momente de timp diferite (ESC, „A” si EOS).

Cu toate acestea, decriptarea este exacta, deoarece structura arborelui Huffman este precisa, unic determinata de fiecare simbol decriptat si identica cu cea din faza de compresie.

Concluzie:

Metoda adaptiva realizeaza o rata de compresie strict pozitiva, pentru ca se emite un numar de biti mai mic decat numarul de biti ai setului de date initial, *D* (33 de biti ai setului comprimat fata de 41 de biti ai setului *D*).