Laborator PS

ALGORITMI DE COMPRESIE

Algoritmul Shannon-Fano - exemplu

Prof. dr. ing. Dan STEFANOIU

As. Ing. Alexandru DUMITRASCU

EXEMPLUL ALG. SHANNON-FANO DE COMPRESIE

Setul de date *D*: **IT IS BETTER LATER THAN NEVER.**

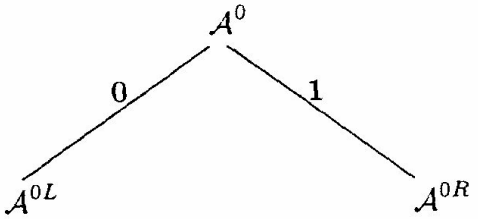
Constructia arborelui binar

Alfabetul asociat setului de date *D*:

A^0		E	T	R	A	I	N	.	B	H	L	S	V
N(s)	5	5	5	3	2	2	2	1	1	1	1	1	1

$N(A^0)$=13 si arborele binar se construieste in cinci iteratii de divizare in subalfabete

Prima iteratie:



(simbolii cu ponderi egale sunt aranjati in ordine lexicografica)

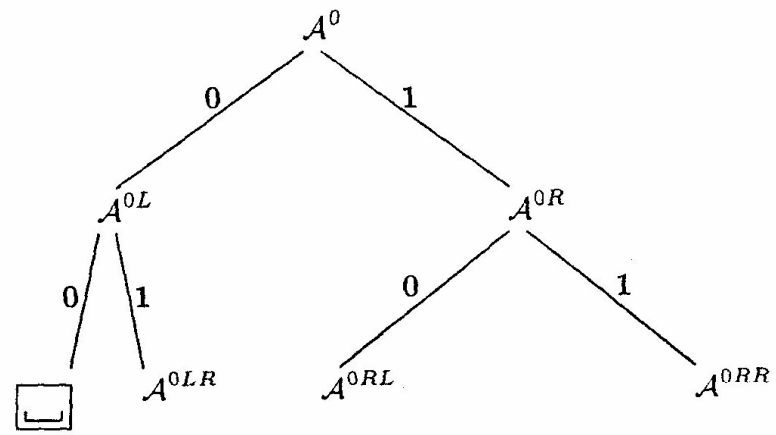
$\mathcal{A}^{0L}$	␣	E	T
$\mathcal{N}(s)$	5	5	5

$\mathcal{N}(\mathcal{A}^{0L}) = 15$

$\mathcal{A}^{0R}$	R	A	I	N	.	B	H	L	S	V
$\mathcal{N}(s)$	3	2	2	2	1	1	1	1	1	1

$\mathcal{N}(\mathcal{A}^{0R}) = 15$

A doua iteratie:



$\mathcal{A}^{0LR}$	E	T
$\mathcal{N}(s)$	5	5

$\mathcal{N}(\mathcal{A}^{0LR}) = 10$

$\mathcal{A}^{0RL}$	R	A	I
$\mathcal{N}(s)$	3	2	2

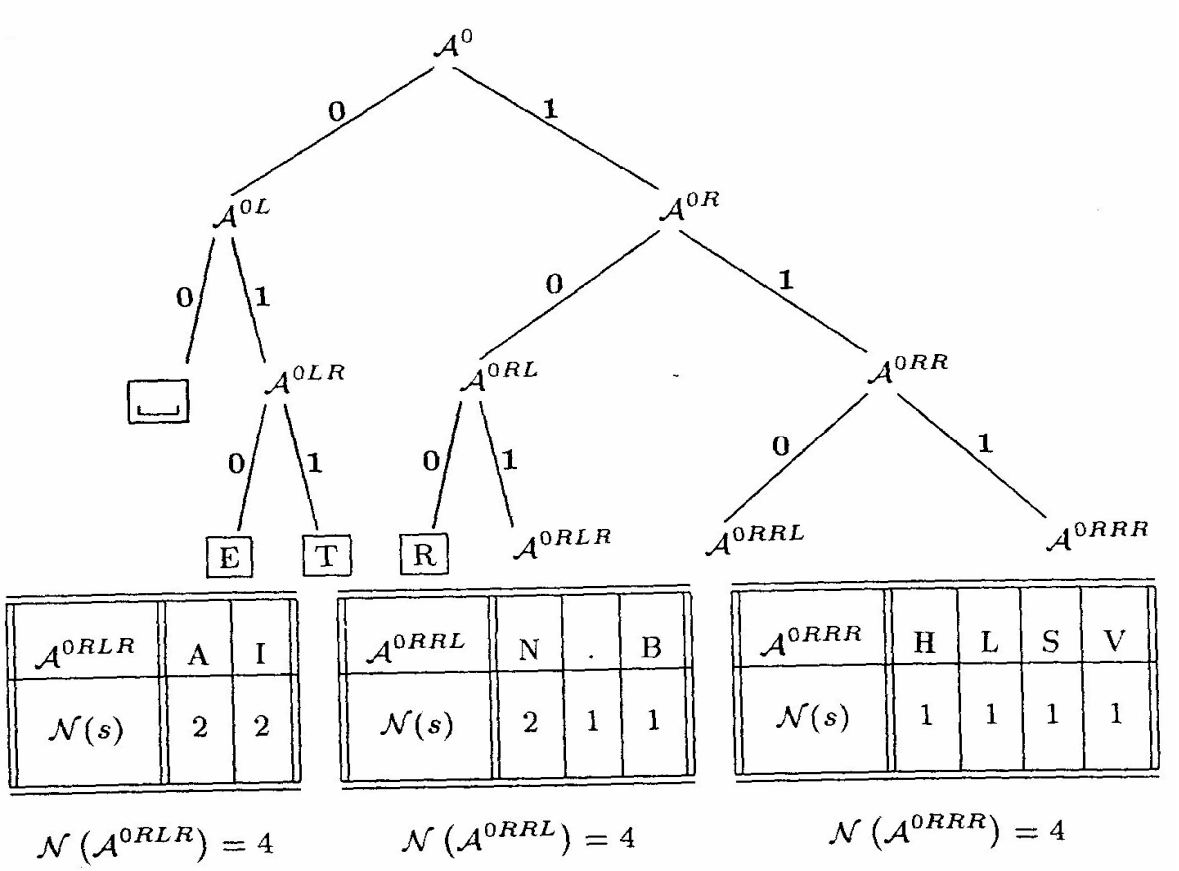
$\mathcal{N}(\mathcal{A}^{0RL}) = 7$

$\mathcal{A}^{0RR}$	N	.	B	H	L	S	V
$\mathcal{N}(s)$	2	1	1	1	1	1	1

$\mathcal{N}(\mathcal{A}^{0RR}) = 8$

Obs.: Simbolul „spatiu” este cel mai frecvent simbol din alfabet si a atins deja o frunza, fiind recodificat pe 2 biti: 00 (in loc de codul original pe 8 biti 00100000, adica 32).

A treia iteratie:



$\mathcal{A}^{0RLR}$	A	I
$\mathcal{N}(s)$	2	2

$\mathcal{N}(\mathcal{A}^{0RLR}) = 4$

$\mathcal{A}^{0RRL}$	N	.	B
$\mathcal{N}(s)$	2	1	1

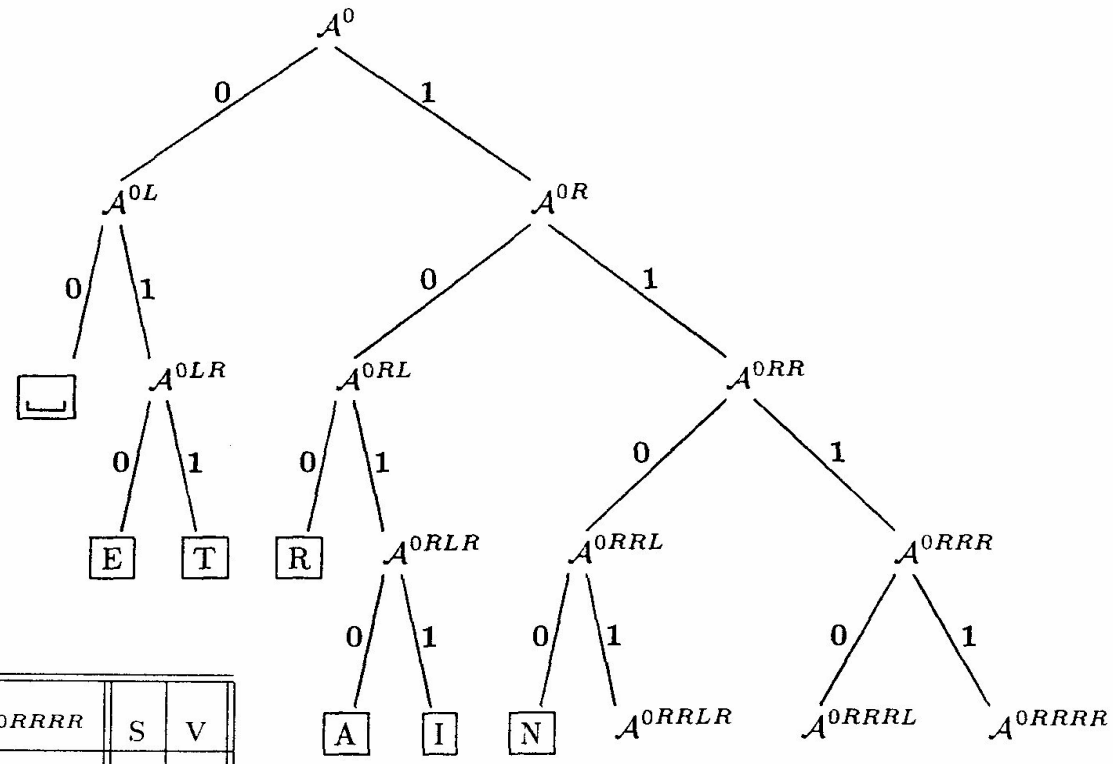
$\mathcal{N}(\mathcal{A}^{0RRL}) = 4$

$\mathcal{A}^{0RRR}$	H	L	S	V
$\mathcal{N}(s)$	1	1	1	1

$\mathcal{N}(\mathcal{A}^{0RRR}) = 4$

Obs.: Simbolii cei mai frecventi sunt recodificati in aceasta iteratie, iar noile lor coduri nu depasesc lungimea de 3 biti (in loc de 8).

A patra iteratie:



$\mathcal{A}^{0RRLR}$	.	B
$\mathcal{N}(s)$	1	1

$\mathcal{N}(\mathcal{A}^{0RRLR}) = 2$

$\mathcal{A}^{0RRRL}$	H	L
$\mathcal{N}(s)$	1	1

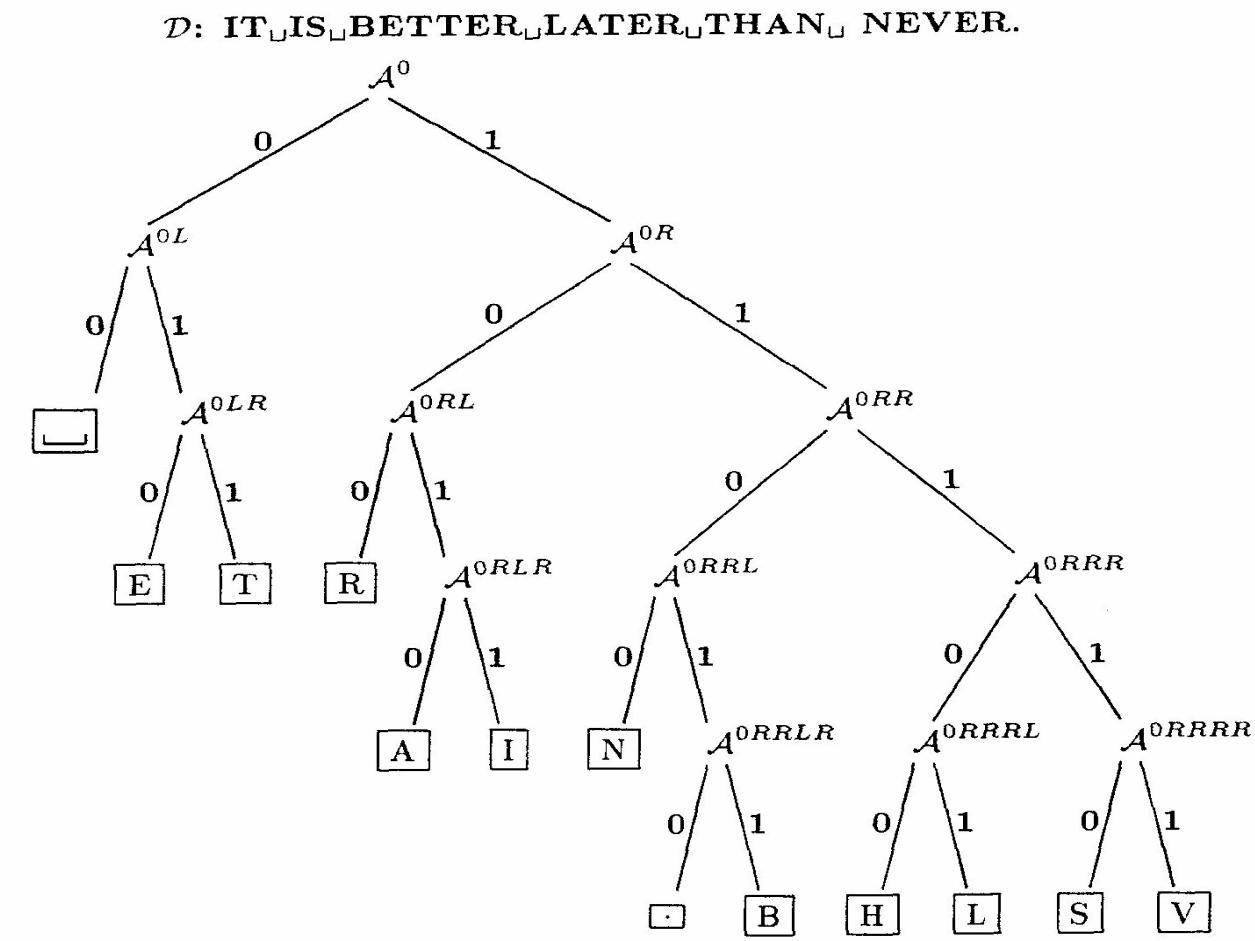
$\mathcal{N}(\mathcal{A}^{0RRRL}) = 2$

$\mathcal{A}^{0RRRR}$	S	V
$\mathcal{N}(s)$	1	1

$\mathcal{N}(\mathcal{A}^{0RRRR}) = 2$

Obs.: Numai simbolii foarte rari din setul de date au mai ramas de recodificat, dar codurile lor vor avea o lungime de 5 biti in loc de 8 biti.

A cincea iteratie:



Obs.: Noile coduri nu depasesc 5 biti => minimizarea redundantei si deci maximizarea ratei de compresie.

Structura setului de date comprimate

Dupa etapa anterioara, informatia de pe fluxul de iesire este:

- Informatia auxiliara:

Cod	13	32	5	69	5	84	5	82	3	65	2	73	2
Info	N	␣	$\mathcal{N}$	E	$\mathcal{N}$	T	$\mathcal{N}$	R	$\mathcal{N}$	A	$\mathcal{N}$	I	$\mathcal{N}$
Nr. biţi	8	8	32	8	32	8	32	8	32	8	32	8	32

Cod	78	2	46	1	66	1	72	1	76	1	83	1	86	1
Info	N	$\mathcal{N}$	.	$\mathcal{N}$	B	$\mathcal{N}$	H	$\mathcal{N}$	L	$\mathcal{N}$	S	$\mathcal{N}$	V	$\mathcal{N}$
Nr. biţi	8	32	8	32	8	32	8	32	8	32	8	32	8	32

Codurile numerice sunt exprimate in zecimal, dar in realitate aceste coduri sunt binare si au cate 8 biti lungime pentru simbolii alfabetului. Aranjarea informatiei auxiliare poate fi realizata si in alta maniera, dar modul de aranjare trebuie cunoscut si la decompresia datelor.

- Informatia utila:

Cod	1011	011	00	1011	11110	00	11011	010	011	011
Info	I	T	␣	I	S	␣	B	E	T	T
Nr. biţi	4	3	2	4	5	2	5	3	3	3

Cod	010	100	00	11101	1010	011	010	100	00	011
Info	E	R	␣	L	A	T	E	R	␣	T
Nr. biţi	3	3	2	5	4	3	3	3	2	3

Cod	11100	1010	1100	00	1100	010	11111	010	100	11010
Info	H	A	N	␣	N	E	V	E	R	.
Nr. biţi	5	4	4	2	4	3	5	3	3	5

Codurile informatiei utile si cele ale informatiei auxiliare se succed fara separatori intre ele. Intre informatia auxiliara si cea utila ar putea sa apara un separator sub forma unui simbol virtual, in absenta numarului de simboli ai alfabetului, *N*. In acest caz, separatorul de informatie trebuie sa aiba un cod mai mare de 255, de exemplu 256, dar reprezentarea lui ar fi pe 16 biti, in loc de 8 biti, cat ocupa *N*.

Analiza performantelor de compresie

Vom face o analiza a entropiei (numarul de biti efectiv alocati in urma compresiei). Exista doua tipuri de entropii: cea a informatiei auxiliare $H_a(D)$ si cea a informatiei utile $H_u(D)$.

$H_a(D)=8\#A^0+32\#A^0=520$ biti

$$H_u(D)=\sum_{s\in D} H_u(s)=\sum_{s\in A^0} H_u(s)N(s)=103 \text{ biti}$$

Obs.: Valoarea entropiei $H_u(D)$ e foarte apropiata de cea ideala (101,625 biti) => metoda de compresie eficienta ?

Entropia totala este: $H(D) = H_a(D) + H_u(D) = 623$ biti

Concluzii:

1. In urma compresiei setul original de date va fi expandat, datorita dimensiunii lui reduse.
2. Tipul static al modelului determina expandarea datelor.
3. Un model dinamic ar produce performante mai bune, dar ar creste complexitatea metodei de compresie/decompresie.

Decompresia datelor

Se citeste informatia auxiliara (5 octeti consecutivi indica o pereche simbol-contor).
Se construieste arborele binar asociat.
Se citeste informatia utila secvential, bit cu bit.
Fiecare simbol al setului de date este decriptat folosind arborele binar.

Ex: primul bit este 1 => deplasarea $A^0 \rightarrow A^{0R}$
al doilea bit 0 => deplasarea spre A^{0RL}
al treilea bit 1 => deplasarea spre A^{0RLR}
al patrulea bit 1 => atingerea simbolului *I*, care va fi inscris pe fluxul de iesire
Atingerea unei frunze reinitiaza cautarea din radacina arborelui.

Obs: Avantajul acestei metode consta in faptul ca nu este necesara cunoasterea dimensiunilor noilor coduri de compresie, care, in general, variaza de la un simbol la altul.