

Projet 2021-2022

Le code de Huffman

Clémentine GRETHEN et Victor MARTI



Types utilisés pour la compression

```
package arbre_new is new arbre(T_donnee=>T_octet);  
use arbre_new;
```

```
package lca_arbre is new LCA(T_cle=>integer,T_donnee=>T_arbre);  
use lca_arbre;
```

```
package lca_integer is new lca(T_cle=> integer,T_donnee=>T_octet);  
use lca_integer;  
type T_tab2 is array (1..257) of Unbounded_String;
```

Architecture de compression:

Module arbre
Générique : T_donnee
Type : T_arbre
T_arbre is access T_noeud
Type T_noeud is record
cle:integer;
donnee:T_Donnee;
gauche:t_arbre;
droit:T_arbre;

Opérations de manipulation d'un arbre.

package arbre_new EST
NOUVEAUarbre(T_donnee=>T_octet);

Module liste
Générique : T_cle,T_donnee,
procédure traiter(cle;donnee)
Types : T_lca
Type T_lca is access T_cellule
Type T_cellule is record
cle:T_cle;
donnee:T_donnee;
suivante: T_LCA;
Opérations de manipulation d'une liste chaînée.

package lca_arbre EST nouveau
LCA(T_cle=>integer,T_donnee=>T_arbre);

package lca_integer EST nouveau
lca(T_cle=>integer,T_donnee=>T_octet);

Compresser(module principal)

- gestion de la ligne de commande

-Lancement des opérations de calcul des fréquences, de construction de l'arbre de Huffman, de calcul des codes, d'encodage du fichier source et d'affichage si mode « -b ».

Test 1 compression: test1.txt

```
1 exemple de texte:  
2 exempte tempeste lexeme
```

Table de fréquence :

/\$:	/n	T	D	«	»	L	P	M	x	e
0	1	2	5	1	5		2	3	4	3	15

Arbre de Huffman théorique

Parcours infixe de l'arbre:
00000111101011100111

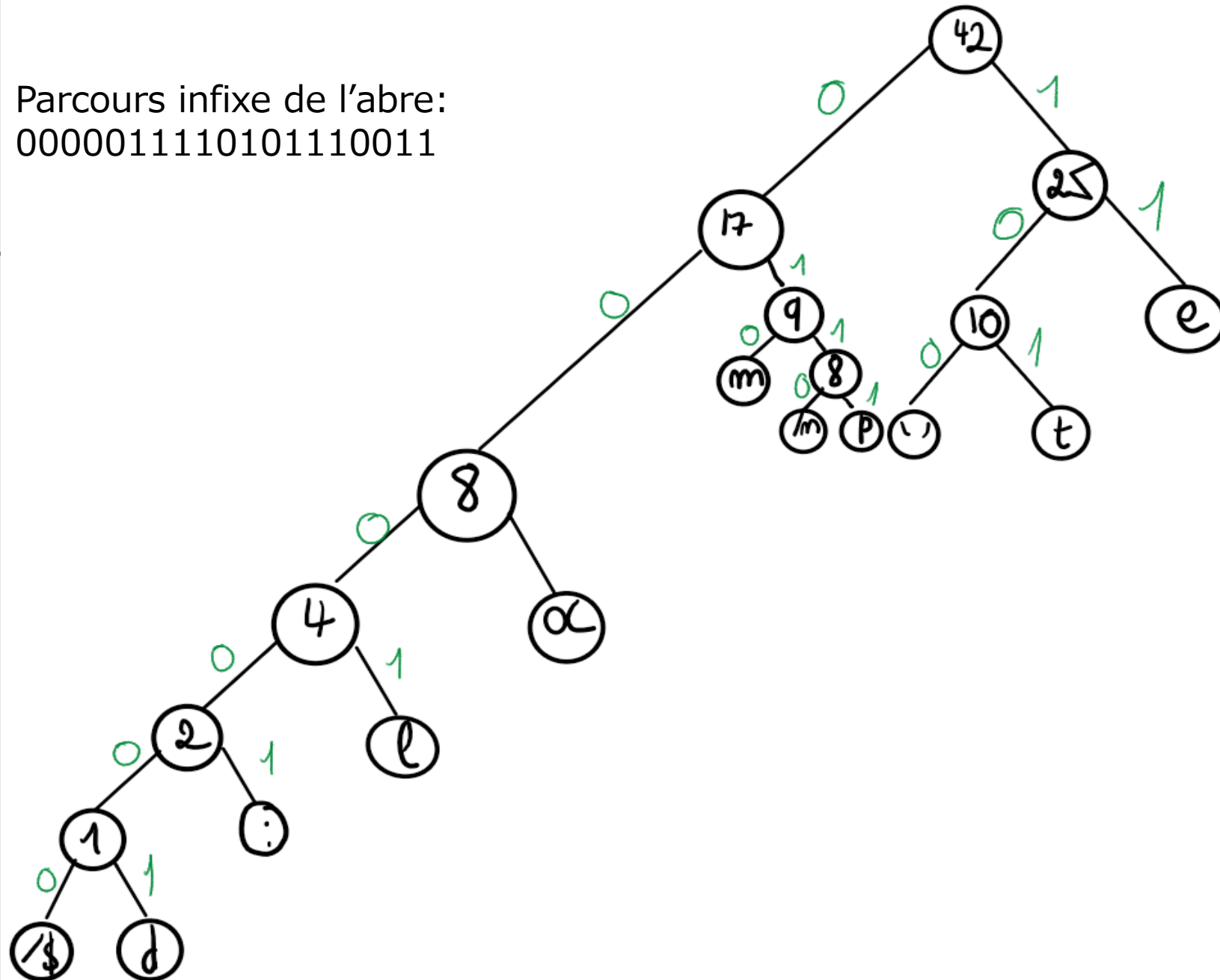


Table de huffman théorique:

/\$	000000
E	11
X	001
M	010
p	0111
L	0001
‘ ‘	100
D	000001
T	101
:	00001
/n	0110

Vérification de l'encodage du texte:

```
000000010110010000111010011011000111100001101101000010100111000000100000011101000110
01010110010100000011111010110011110011101001110001111000000011110010111001101111000
000101101100111010011110111100101110100111110111100000111001110101101100000000
```

En bleu et vert clair: le dernier caractère en parcours infixe, doublé.

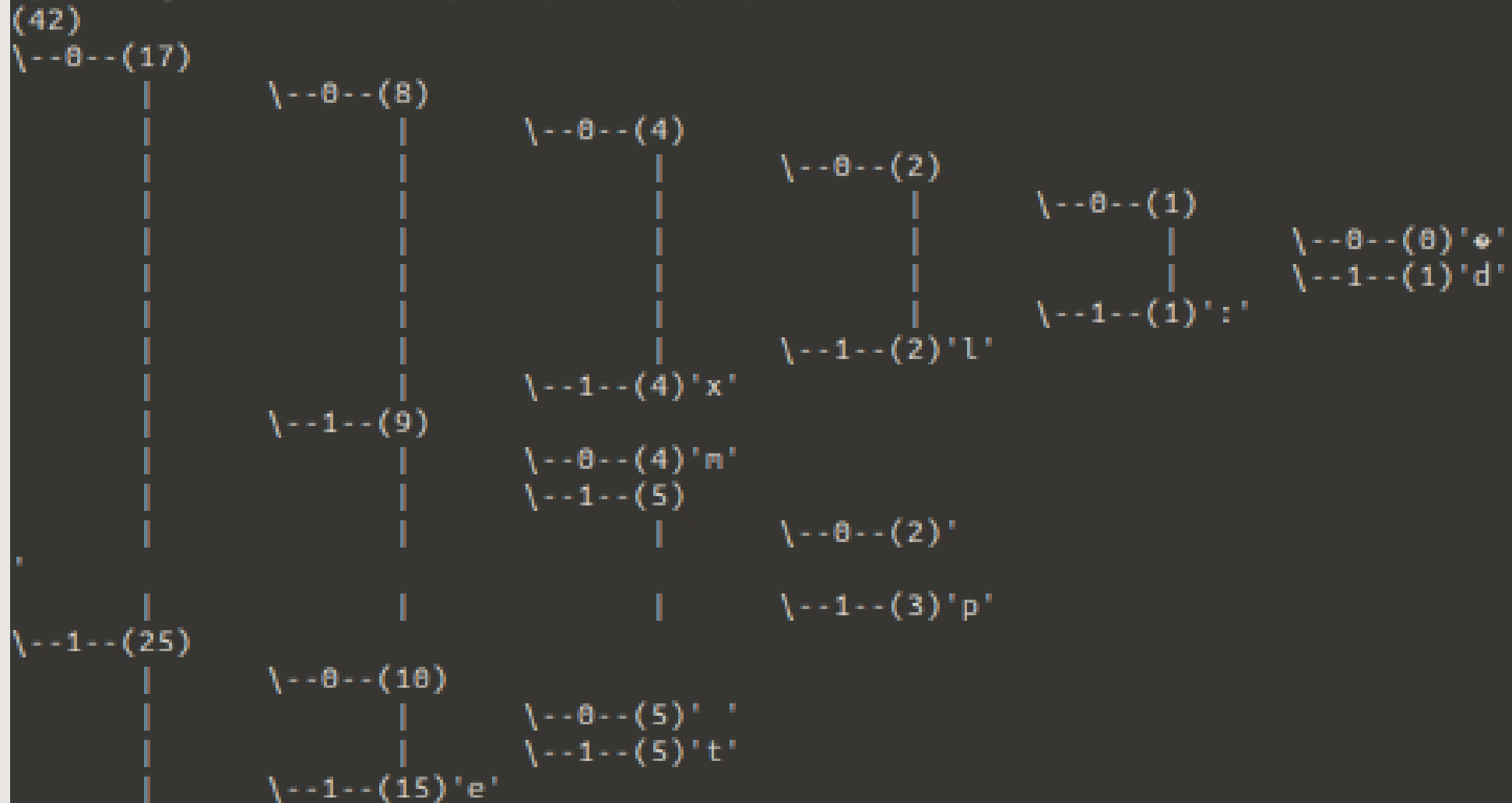
En violet: le parcours infixe:~correspond bien à la slide 6

En orange: le code du caractère de fin.

En vert foncé: les 0 rajoutés pour une écriture en octet

Arbre de Huffman avec l'algorithme:

egre cheng / ENS CMAB311 / 1400 / 3103 / 1 / compresser / 0 / test1.txt



Test dans aucun fichier:

```
'/$' -->0000000  
'e' -->11  
'x' -->001  
'm' -->010  
'p' -->0111  
'l' -->0001  
' ' -->100  
'd' -->000001  
't' -->101  
':' -->00001  
'  
' -->0110
```

Test2: fichier inexistant:

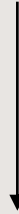
```
cgrethen@n7-ens-lnx031:~/MN06/src$ ./compresser -b inexistant.txt  
veuillez rentrer un ou des fichiers existants!!!  
cgrethen@n7-ens-lnx031:~/MN06/src$
```



```
-- on traite les exceptions relatives aux fichiers:  
exception  
when Ada.IO_Exceptions.STATUS_ERROR =>  
    Put_Line("attention un des fichiers est déjà ouvert");  
when Ada.IO_Exceptions.NAME_ERROR =>  
    Put_Line("veuillez rentrer un ou des fichiers existants!!!");
```

Test 3: test sans fichier

```
cgrethen@n7-ens-lnx031:~/MN06/src$ ./compresser -b  
Votre entrée n'est pas bonne: il faut au moins un texte à décompresser  
cgrethen@n7-ens-lnx031:~/MN06/src$
```



```
File(0, 0, 0);  
-- on traite les exceptions relatives aux fichiers:  
exception  
when Ada.IO_Exceptions.STATUS_ERROR =>  
    Put_Line("attention un des fichiers est déjà ouvert");  
when Ada.IO_Exceptions.NAME_ERROR =>  
    Put_Line("veuillez rentrer un ou des fichiers existants!!!");
```

Architecture de décompression

Types utilisés pour la décompression:

```
12     type T_octet is mod 2**8;
13
14     package LCA_decompress is
15     new LCA (T_donnee => unbounded_string, T_cle => T_oCtEt);
16     use LCA_decompress;
17
18
19     type T_tab is array (1..257) of T_octet;
20     type T_tab2 is array (1..257) of unbounded_string;
21
22
23
```

Test 1: décompression test1.txt.hff

- Texte initiale: exemple du sujet
- Permet de vérifier que la compression est juste.
- On obtient test1.txt:

```
1 exemple de texte :
2 exempte tempete lexeme
3 00
  19
```

```
cgrethen@n7-ens-lnx018:~/MN06/src$
cgrethen@n7-ens-lnx018:~/MN06/src$ xxd test1.txt
00000000: 6578 656d 706c 6520 6465 2074 6578 7465  exemple de texte
00000010: 203a 0a65 7865 6d70 7465 2074 656d 7065  :.exempte tempe
00000020: 7465 206c 6578 656d 650a 19          te lexeme..
cgrethen@n7-ens-lnx018:~/MN06/src$
```

Table de huffman affichée pour le test 1 si le mode -b :

```
0110 --> /$  
100 -->  
00001 --> :  
000001 --> d  
11 --> e  
0001 --> l  
010 --> m  
0111 --> p  
101 --> t  
001 --> x  
000000 --> /$
```

Avec la décompression

```
/ $ ' --> 000000  
' e ' --> 11  
' x ' --> 001  
' m ' --> 010  
' p ' --> 0111  
' l ' --> 0001  
' ' --> 100  
' d ' --> 000001  
' t ' --> 101  
' : ' --> 00001  
'  
' --> 0110
```

Avec la compression

Test 2: Fichier inexistant:

```
0
1 cgrethen@n7-ens-lnx018:~/MN06/src$ ./decompresser inexistant.txt.hff
2 veuillez rentrer un ou des fichiers existants!!!
3 cgrethen@n7-ens-lnx018:~/MN06/src$
```

Test 3: test sans aucun fichier:

```
cgrethen@n7-ens-lnx018:~/MN06/src$ ./decompresser  
Votre entrée n'est pas bonne: il faut au moins un texte à décompresser  
cgrethen@n7-ens-lnx018:~/MN06/src$
```