Compression – méthode de Lempel–Ziv–Welsh

Thierry Lecroq

Université de Rouen
FRANCE

La méthode de Lempel–Ziv–Welsh

- Codage de facteurs
- Utilisation d'un dictionnaire
- Remplacement des facteurs répétés par leur indice dans le dictionnaire

Dictionnaire

Le dictionnaire est clos par préfixe : chaque préfixe d'un mot du dictionnaire est aussi présent dans le dictionnaire

Le dictionnaire est initialisé avec tous les caractères de l'alphabet

Dictionnaire

Le dictionnaire est clos par préfixe : chaque préfixe d'un mot du dictionnaire est aussi présent dans le dictionnaire

Le dictionnaire est initialisé avec tous les caractères de l'alphabet

Codage

La situation courante est la suivante :

- on a lu un facteur w du texte présent dans le dictionnaire
- on lit ensuite le caractère a qui suit w dans le texte

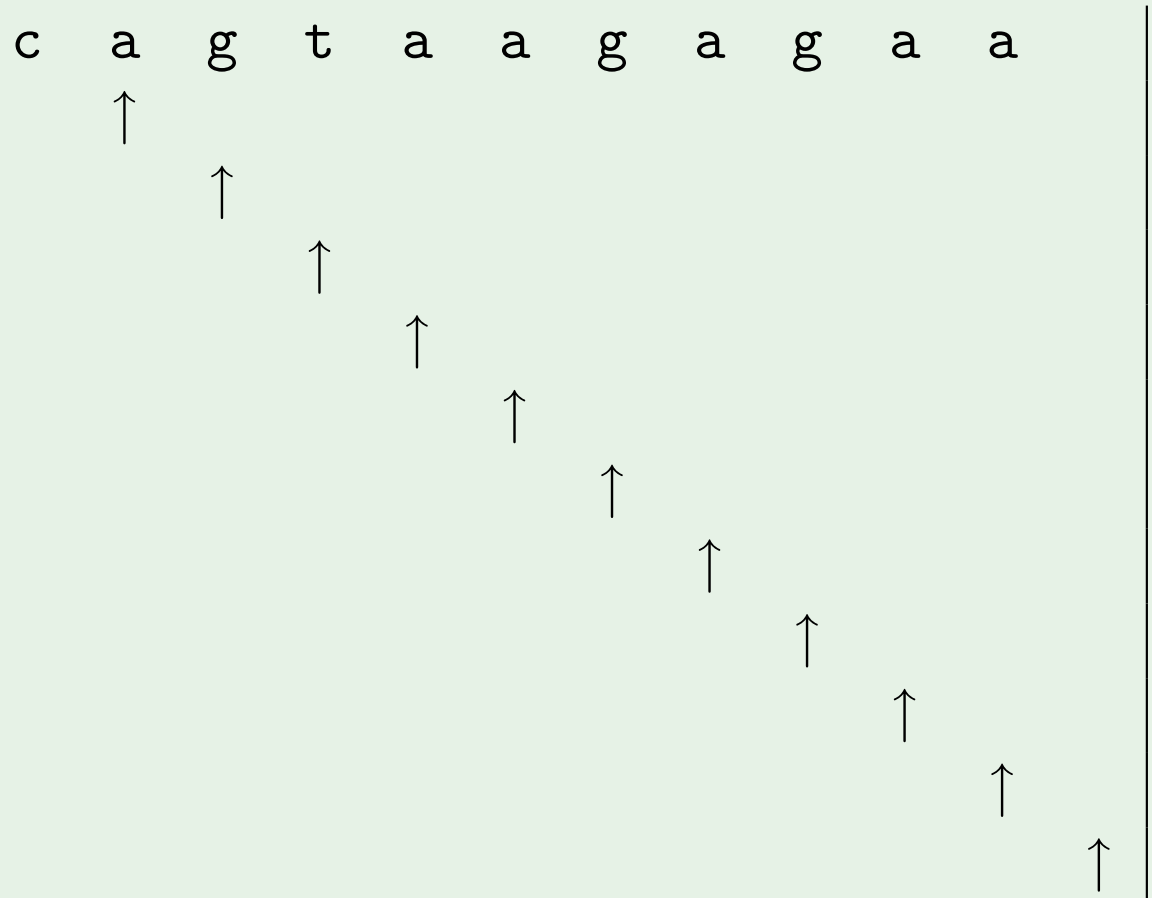
Codage

Alors

- si wa n'est présent pas dans le dictionnaire, on écrit l'indice de w dans le fichier et sortie, on ajoute wa dans le dictionnaire et on continue avec a pour nouvelle valeur de w
- si wa est présent dans le dictionnaire, on continue avec wa pour nouvelle valeur pour w

w est initialisé avec le premier caractère du texte

Exemple cagtaagagaa



w	écrit	ajouté
c	99	ca, 257
a	97	ag, 258
g	103	gt, 259
t	116	ta, 260
a	97	aa, 261
a		
ag	258	aga, 262
a		
ag		
aga	262	agaa, 263
a		
	97	
	256	

Décodage

Symétrique au codage : le dictionnaire est reconstruit pendant le décodage

Décodage

- Lire un indice i dans le fichier compressé
- Écrire dans le fichier de sortie le facteur w du dictionnaire d'indice i
- Ajouter dans le dictionnaire le facteur wa où a est la première lettre du facteur décodé suivant

Décodage

Problème

Si y contient un facteur $azazax$ tel que az est déjà dans le dictionnaire mais aza pas encore.
Pendant le codage, az est lu, son indice est écrit dans le fichier de sortie et aza est ajouté au dictionnaire.
Ensuite aza est lu et son indice est écrit dans le fichier de sortie.

Décodage

Problème

Pendant le décodage, l'indice de aza est lu alors que le traitement du mot az n'est pas encore terminé : le facteur aza n'est pas encore dans le dictionnaire.
Cependant comme c'est l'unique cas où cette situation peut se produire, le facteur aza est reconstruit, en prenant le dernier facteur ajouté lu az concaténé avec sa première lettre a : aza.

Exemple

lu	écrit	ajouté
99	c	
97	a	ca, 257
103	g	ag, 258
116	t	gt, 259
97	a	ta, 260
258	ag	aa, 261
262	aga	aga, 262
97	a	agaa, 263
256		

Implantation

Le dictionnaire est stocké dans une table D.
Il est organisé sous forme arborescente.
Chaque nœud p (correspondant à un facteur de y) de l'arbre possède trois composantes :

- *parent*(p) : un lien vers le nœud parent
- *étiq*(p) : la lettre contenue dans le nœud
- *code*(p) : l'indice associé à p

Codage(fentrée, fsortie)

```
1  compteur ← −1
2  pour a ∈ A faire
3     compteur ← compteur + 1
4     INSÉRER((D, (−1, a, compteur)))
5  compteur ← compteur + 1
6  INSÉRER((D, (−1, a, compteur)))
7  p ← −1
8  tantque  non fdf(fentrée) et a est le prochain caractère faire
9     q ← RECHERCHE((D, (p, a)))
10    si q = NIL alors
11       écrire codep sur 1 + log(compteur) bits dans fsortie
12       compteur ← compteur + 1
13       INSÉRER((D, (p, a, compteur)))
14       p ← RECHERCHE((D, (−1, a)))
15    sinon p ← q
16 écrire code(p) sur 1 + log(compteur) bits dans fsortie
17 écrire RECHERCHE((D, (−1, FIN))) sur 1 + log(compteur) bits dans fsortie
```

Codage

Une technique efficace pour coder D consiste à utiliser une table de hachage

Décodage

Le décodage ne nécessite pas de hachage.
Connaissant l'indice c d'un facteur, il suffit de remonter jusqu'à la racine pour reconstruire le renversé du facteur (utilisation d'une pile pour le remettre à l'endroit).
On suppose que la fonction $mot(c)$ effectue ce travail.

Décodage(fentrée, fsortie)

```
1  compteur ← −1
2  pour a ∈ A faire
3     compteur ← compteur + 1
4     Insérer((D, (−1, a, compteur)))
5  compteur ← compteur + 1
6  Insérer((D, (−1, a, compteur)))
7  p ← −1
8  c ← premier code sur 1 + log(compteur) bits dans fentrée
9  écrire mot(c) dans fsortie
10 a ← mot(c)[0]
11 tantque VRAI faire
12    d ← premier code sur 1 + log(compteur) bits dans fentrée
13    si d > compteur alors
14       compteur ← compteur + 1
15       parent(compteur) ← c
16       étiq(compteur) ← a
17       écrire mot(c)a dans fsortie
18       c ← d
19    sinon a ← mot(d)[0]
20       si a ≠ FIN alors
21          compteur ← compteur + 1
22          parent(compteur) ← c
23          étiq(compteur) ← a
24          écrire mot(c)a dans fsortie
25          c ← d
26       sinon STOP
```