Master 2 – Ingénierie Informatique (Parcours LP, SRI, LC)

Formats de Documents et Compression - Examen du 17-12-2009

Notes manuscrites et sujets de TD autorisés. Livres non autorisés.
La rigueur des raisonnements, la clarté des explications, mais aussi la qualité de la présentation influera sur la note. Deux ou trois lignes de texte ***au maximum*** *pour les réponses explicatives.*

Problème 1

Partie I

1. Simuler la méthode *Move-to-Front* (MTF) vue en cours sur la chaîne $s =$"`barbamaman`" et calculer la valeur moyenne des codes émis, en supposant que le premier élément de la liste est toujours codé par 0 et que la liste initiale est $[a, b, m, n, r]$.
2. A partir de la chaine s, écrire la chaîne s' obtenue en appliquant la procédure suivante :
   – Calculer tous les mots obtenus en conjuguant circulairement le mot s (les deux premiers : `nbarbamama`, `anbarbamam`, etc.). Il y a 10 de tels mots, y compris le mot s lui-même.
   – Trier ces mots en ordre lexicographique et les disposer en colonne, (cela donne une table 10x10).
   – Le mot s' est obtenu en lisant la dernière colonne de la table.
   Le mot s' est donc une permutation du mot s et la procédure appliquée s'appelle la *Transformée de Burrows-Wheeler* (BWT), c'est pourquoi on notera par la suite $s' = bwt(s)$.
3. Simuler la méthode *Move-to-Front* sur la chaîne $bwt(s)$ et calculer la valeur moyenne des codes émis pour cette chaine.
4. Expliquer cette différence entre les deux valeurs moyennes obtenues.
5. Expliquer en particulier quel est l'effet de la BWT sur un texte qui contient plusieurs occurrences d'un même facteur (comme par exemple `the` en Anglais ou `est` ou `que` en Français).
6. Citer une méthode autre que MTF qui est susceptible de compresser $bwt(s)$ mieux que s et expliquer pourquoi.
7. Si $longueur(s) = n$, combien de comparaisons de caractères sont nécessaires (dans le pire des cas) pour trier deux conjugués de s ?
8. Quel est donc la complexité (dans le pire des cas) attendue d'un algorithme optimal qui trie les n conjugués de s, exprimée en nombre de comparaisons de caractères ?
9. Est-il nécessaire que l'encodeur maintient en mémoire les n conjugués de s (grosso modo, cela fait n copies du fichier pour un espace de l'ordre de $O(n^2)$) ? Est il possible de calculer la BWT en utilisant un espace $O(n)$?
10. Quels sont les avantages si on divise le fichier en k blocs de taille n/k ? Quel est l'inconvénient ?
11. Expliquer comment le décodeur peut reconstruire la première colonne p de la table en connaissant uniquement $bwt(s)$. Est-il possible d'effectuer cette reconstruction en temps $O(n)$ et espace constant ?
12. Expliquer pourquoi le décodeur ne peut pas reconstruire s en connaissant uniquement $bwt(s)$ et indiquer quelle est l'information **minimale** additionnelle dont on doit disposer à votre avis pour que la reconstruction soit possible et quel est l'ordre de grandeur de cette information.

Partie II (facultative)

13. Remplir la table ci-dessous en écrivant le mot p et le mot $bwt(s)$ obtenus pour $s =$ `barbamaman`.

	0	1	2	3	4	5	6	7	8	9
p										
bwt(s)										
	0	1	2	3	4	5	6	7	8	9

14. Ecrire la permutation t des 10 places, obtenue en faisant correspondre lettre égales dans p et $btw(s)$ en respectant l'ordre. Par exemple, si la première a apparaît à la 3^e place dans p et à la 5^e place dans $btw(s)$, alors 5 correspond à 3 et écrire un 5 sous le 3.

	0	1	2	3	4	5	6	7	8	9
t =										
w =										

15. Ecrire dans la 3^e ligne de la table le mot w tel que $w[i] = p[t^i(0)]$. Quel rapport il y a entre s et w ? Quelle autre information l'encodeur doit vous transmettre pour reconstruire s ?

Exercice 2

Compresser le texte "`avada kedavra`" par le codage de Huffman adaptatif. En plus de fournir la chaîne compressée, vous devrez détailler toutes les étapes de construction de l'arbre de codage.
Supposez maintenant que chaque noeud de l'arbre est représenté par une structure (enregistrement) alors que l'arbre est représenté par un tableau de ces structures. Vous devrez choisir judicieusement les champs à inclure dans ces structures pour représenter convenablement l'arbre et pour effectuer sa construction. Vous devrez donc écrire le pseudo-code de la partie de l'algorithme qui met à jour le tableau à chaque étape de traitement d'un caractère de la chaîne.
Simuler votre algo sur la partie initiale "`avada`" de la chaîne.

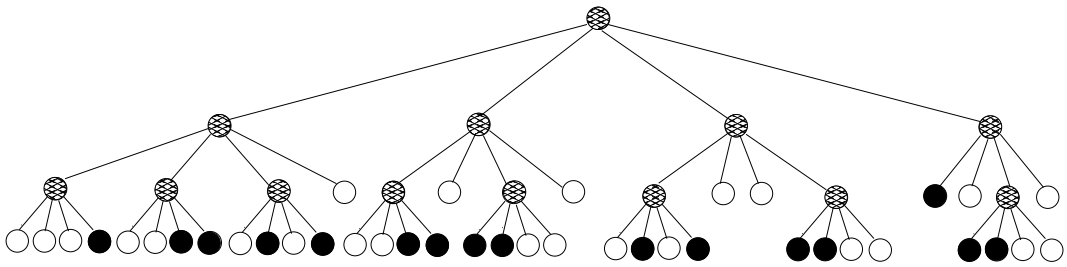
Exercice 3

Le message suivant a été compressé par la méthode LZ78. Retrouver le message original en détaillant les étapes de construction du dictionnaire :

$$(0,s);(0,i);(0,n);(0,g);(2,n);(4,\sqcup);(0,a);(0,\sqcup);(1,o);(3,g);(8,s);(5,g);$$
$$(8,t);(0,h);(0,e);(11,o);(10,\sqcup);(1,i);(10,i);(17,a);(16,n);(4,EOF)$$

Exercice 4

Le *quadtree* suivant représente une image binaire de 16 pixels par 16.



En sachant que les quadrants d'un carré sont pris en considérations dans l'ordre de la figure suivante, reconstruire l'image correspondante au quadtree.

1	2
3	4

Si on suppose que chaque noeud de l'arbre occupe autant de place qu'un pixel de la figure d'origine, quel est le rapport de compression de cette image quand elle est représentée par ce quadtree ?