Examen de théorie de l'information
Université Paris-Est Marne-la-Vallée. Master 1
Janvier 2013. Durée 2 heures. Documents autorisés.

Les exercices sont indépendants.

Exercice 1.— 4 points

a) On considère un mot w (qui se termine par \$) tel que la transformée de Burrows-Wheeler de w est $\mathrm{BWT}(w) = AGGGA\$CAC$. On rappelle que la lettre \$ est supposée inférieure à toute autre lettre du mot. Retrouvez le mot w et indiquez le calcul effectué pour le retrouver.
b) Trouver un mot qui n'est pas la transformée de Burrows-Wheeler d'un autre mot.

Exercice 2.— 6 points
On considère un codage de convolution de taux de transmission 1 : 2 défini de la façon suivante. Chaque bit i_n est codé par un bloc de deux bits $(t_n^{(0)}, t_n^{(1)})$. Ces bits sont calculés modulo 2 par les formules suivantes

$$\begin{aligned} t_n^{(0)} &= i_{n-3} + i_{n-2} + i_{n-1} + i_n, \\ t_n^{(1)} &= i_{n-3} + \qquad\qquad + i_n. \end{aligned}$$

a) Le codage est-il à fenêtre glissante ? Pourquoi ? Le décodage dans un canal sans bruit est-il à fenêtre glissante ? Pourquoi ?
b) Dessiner le transducteur de codage.
c) Le décodage dans un canal bruité est-il catastrophique ? Pourquoi ?

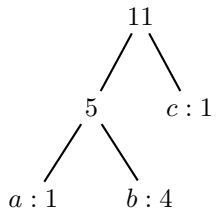
Exercice 3.— 4 points
a) Montrer que l'ensemble $X = \{b, abb, abbba, bbba, baabb\}$ n'est pas un code.
b) Montrer que l'ensemble $Y = \{abb, abbba, bbba\}$ est un code.

Exercice 4.— 8 points
On considère une alphabet source A pondéré (chaque lettre $a \in A$ a un poids $p(a)$ dans $\mathbb{N}$) et ordonné : on suppose que $a < b < c < \cdots$. On considère un alphabet $B = \{0, 1\}$ ordonné par $0 < 1$. On souhaite coder une séquence source sur l'alphabet A en une séquence sur l'alphabet B par un codage c qui transforme chaque lettre de A en un mot de B tel que $X = \{c(a) \mid a \in A\}$ est un ensemble préfixe, et tel le codage c soit *ordonné* : si $a < a'$ alors $c(a) < c(a')$ pour toute lettre $a \in A$.

Le codage doit permettre de minimise $\sum_{a\in A} |c(a)| \times p(a)$, où $|c(a)|$ est la longueur du mot $c(a)$. Il s'agit de la même optimisation que pour le codage de Huffman mais le codage doit être en plus ordonné.

Pour cela, on désire construire un arbre binaire complet où chaque nœud a un poids. Chaque feuille représente un lettre a de poids $p(a)$ et le chemin de la racine à la feuille a représente le mot codé $c(a)$. Le poids de la racine est égal à la quantité à optimiser $\sum_{a\in A} |c(a)|p(a)$. De plus, la suite des feuilles obtenues en ordre interne (ou infixe) est ordonnée. Un tel arbre est appelé *arbre de Huffman ordonné*.

Par exemple l'arbre ci-dessous pour la source $A = \{a, b, c\}$ avec $p(a) = p(c) = 1$ et $p(b) = 4$ est un arbre de Huffman ordonné. En effet, la projection des feuilles (l'ordre interne) pour cet arbre est a, b, c.



a) Montrer que l'algorithme classique de Huffman ne permet pas toujours d'obtenir un arbre de Huffman ordonné.

b) Trouver un algorithme de programmation dynamique permettant de construire un arbre de Huffman ordonné en $O(n^3)$ si n est la taille de l'alphabet A.

c) Donner un arbre de Huffman ordonné pour $A = \{a, b, c, d, e\}$ avec les poids suivants.

	a	b	c	d	e
poids	25	20	12	10	14