Examen de théorie de l'information
Université de Marne-la-Vallée. M1 informatique
Janvier 2008. Durée 2 heures.
Tous les documents sont autorisés.

Les exercices sont indépendants.

Exercice 1.— 7 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

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

a) Le codage est-il à fenêtre glissante ? Pourquoi ?
b) Le décodage dans un canal sans bruit est-il à fenêtre glissante ? Pourquoi ?
c) On considère que la suite codée passe dans un canal binaire symétrique. Dessiner le tranducteur de codage.
d) Décrire une suite infinie de bits codée c_1 et une suite infinie de bits codée c_2 telles que c_1 et c_2 ne diffèrent que par un nombre fini (petit) de bits et les suites décodées de c_1 et c_2 à l'aide du transducteur diffèrent sur un nombre infini de bits.

Exercice 2.— 5 points
On considère un mot w (avec terminaison \$) tel que la transformée de Burrows-Wheeler de w est $\mathrm{BWT}(w) = aml\$baaa$. On rappelle que la lettre \$ est supposée inférieure à toute autre lettre du mot. Retrouvez le mot w.

Tout mot peut-il être la transformée de Burrows-Wheeler d'un autre mot ?

Exercice 3.— 8 points

On considère une source A uniforme de cardinal n ($p(a) = 1/n$ pour toute lettres $a \in A$). On désire coder chaque lettre $a \in A$ par un mot $c(a)$ d'un code préfixe sur un alphabet B à k lettres. Les mots de code $c(a)$ correspondent ainsi aux feuilles d'un arbre k-aire.

Les lettres de B ont des coûts différents et on ne peut donc pas construire le code en utilisant l'algorithme de Huffman. On note $\mathrm{cost}(b)$ le coût de la lettre b. Le coût d'un mot de code est la somme des coûts de ses lettres.

Pour avoir une efficacité maximale, on doit minimiser

$$\sum_{a\in A} \mathrm{cost}(c(a))p(a) = \sum_{a\in A} \frac{\mathrm{cost}(c(a))}{n}\,.$$

On doit donc minimiser le *coût* du code :

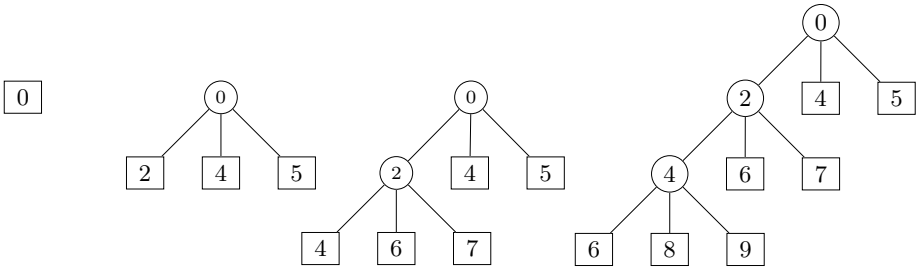
$$\sum_{a\in A} \mathrm{cost}(c(a))\,.$$

On suppose que $n-1$ est un multiple de $k-1$. Donc $n = q(k-1)+1$ pour un certain entier positif ou nul q.

On construit un arbre k-aire de la façon suivante. Cet arbre a q nœuds internes et n feuilles. Ansi le code préfixe sera complet et correspondra aux

feuilles de cet arbre. L'algorithme part d'un arbre constitué d'un seul nœud (une feuille) racine de coût nul. De façon itérative, on remplace chaque feuille de l'arbre de coût minimal par un nœud interne à k fils, un pour chaque lettre de B. Le coup de chaque nouvelle feuille est le coût de la lettre plus le coût du père. Comme le nombre de feuilles augmente de $k-1$, on obtient un arbre à n feuilles au bout de q étapes.

Example 0.1 Supposons que l'on recherche un code optimal pour la source $A = \{a, b, c, d, e, f, g\}$ à 7 lettres, le code étant dans un alphabet ternaire $B = \{0, 1, 2\}$ tel que $\text{cost}(0) = 2, \text{cost}(1) = 4$ et $\text{cost}(2) = 5$. L'algorithme construit par exemple les arbres suivants et on obtient le code $\{000, 001, 002, 01, 02, 1, 2\}$ de coût 45.



(a) (b) (c) (d) Une solution

a) Construire un arbre optimal pour une source uniforme à 13 lettres toujours avec un alphabet ternaire $B = \{0, 1, 2\}$ tel que $\text{cost}(0) = 2, \text{cost}(1) = 4$ et $\text{cost}(2) = 5$. Indiquez le code trouvé et son coût.

b) Quelle est la complexité de l'algorithme de construction de l'arbre en fonctions de n (on supposera k fixé et $n-1$ multiple de $k-1$) ? On précisera avec quelles structures de données.

c) Montrer que l'algorithme permet bien de construire un code de coût minimal.