UNIVERSITÉ TOULOUSE III PAUL SABATIER

Théorie de l'Information

Année 2016 - 2017
L1 Semestre 2

Contrôle Continu

— Durée du contrôle : 90 minutes
— Aucun document n'est autorisé. Les définitions essentielles sont résumées en annexe.
— Une calculette classique (sans moyen de communication) est autorisée. Tout autre appareil électronique est interdit.

1 Codage de caractères

Exercice 1 (5 pts) Voici des extraits de la table de codage Unicode pour l'alphabet arabe et palmyrénien.

أ 0623	س 0633	ك 0643
ؤ 0624	ش 0634	ل 0644

(a) Arabic

A	𐡚 1086A	𐡺 1087A
B	𐡛 1086B	𐡻 1087B
C	𐡜 1086C	𐡼 1087C

(b) Palmyrene

Codez les caractères Lam (U+0644) de l'alphabet arabe et Lamedh (U+1086B) de l'alphabet palmyrénien en UTF-8, UTF-16, UTF-32.

Exercice 2 (4 pts)

1. Quel code point U+x (pour x un nombre hexadécimal) est codé par le code UTF-8 `1110 1011. 1001 1100. 1010 1010` ?
2. Que pensez-vous du code UTF-8 : `11010110. 10101100. 10011010` ? Quel(s) code point(s) est-ce qu'il code ? Justifiez votre réponse.
3. Quel code point U+x est codé par le code UTF-16 `1101 1000. 0111 0101. 1101 1100. 1100 1101` ?

2 Codes

Exercice 3 (5 pts) *Les questions (5) et (6) sont des questions facultatives (bonus).*

Nous disons que le *miroir* d'un mot $x_1x_2\dots x_n$ est le mot $x_n\dots x_2x_1$. Par exemple, `caba` est le miroir du mot `abac`. Nous écrivons $\overline{m}$ pour le miroir d'un mot m.

Par extension, on peut définir le miroir $\overline{cod}$ d'une fonction de codage cod par $\overline{cod}(m) = \overline{cod(m)}$.

1. Pour la fonction de codage cod_1 définie par le tableau suivant :

x	a	b	c	d
$cod_1(x)$	0	10	110	111

   définissez la fonction de codage miroir $\overline{cod_1}$.
2. Soit $y_1 = 011010111$ le code d'un mot x_1 codé avec la fonction cod_1, donc $y_1 = cod_1(x_1)$. Donnez le mot source x_1.
3. Soit $y_2 = 111010110$ le code d'un mot x_2 codé avec la fonction $\overline{cod_1}$, donc $y_2 = \overline{cod_1}(x_2)$. Donnez le mot source x_2.
4. Montrez que si cod est une fonction de codage unique, alors $\overline{cod}$ l'est aussi. Pour cela, indiquez explicitement la fonction de décodage $\overline{dec}$ associée à $\overline{cod}$, et prouvez que $\overline{dec}$ est la fonction de décodage de $\overline{cod}$.
5. Informellement, un mot m est un *postfixe* de m' si m' finit avec le mot m. Par exemple, `ba` est un postfixe de `caba`. Nous écrivons alors $m \dashv m'$.

   Une définition plus formelle est : $m \dashv m'$ s'il existe r avec $m' = r \cdot m$, ou $\cdot$ est l'opération de concaténation.

   Donnez une autre définition qui utilise uniquement la relation "préfixe" $\preceq$ et l'opération "miroir".
6. Un *code postfixe* est un code *post* tel que pour tout a, a' de l'alphabet A, si $a \neq a'$, alors $post(a) \not\dashv post(a')$. Montrez que tout code postfixe est un code unique.

   Pour faire la demonstration, vous pouvez faire une preuve directe (complexe) ou utiliser les faits évoqués plus haut, et les propriétés connues des codes préfixes.
7. Pourquoi un code préfixe est-il préférable à un code postfixe ?

Rappel de notions définies en cours :
— une fonction de codage cod est *unique* s'il existe un décodage dec tel que pour tout message m, $dec(cod(m)) = m$.
— m est un préfixe de m' s'il existe r avec $m' = m \cdot r$. Notation : $m \preceq m'$.

3 Inégalité de Kraft

Exercice 4 (3 pts) Un collègue vous demande de construire un code préfixe binaire pour les caractères a, b, c, d, e, f respectivement avec des longueurs 2, 2, 2, 3, 4, 5.

1. Vérifiez tout d'abord, à l'aide de l'inégalité de Kraft, qu'un tel code peut exister.
2. Construisez effectivement un tel code.
3. Vous constatez que le code n'est pas optimal, parce que le code d'un caractère pourrait être raccourci. Quelle amélioration pouvez-vous proposer à votre collègue ?

4 Théorie de l'information et algorithme de Huffman

Exercice 5 (3 pts) Une source d'information émet les caractères `a` ... `f` selon la distribution de probabilité suivante :

a	b	c	d	e	f
0.1	0.15	0.3	0.15	0.1	0.2

1. Calculez l'entropie de cette source d'information.
2. Construisez l'arbre de Huffman pour trouver un codage optimal des caractères.
3. Calculez la longueur moyenne du code et comparez avec l'entropie.

A Définitions du cours

A.1 Unicode

UTF-32 Chaque caractère représenté par un mot de 32 bits

UTF-16 Un ou deux mots de 16 bits, construits comme suit :

1. U+0000 ... U+D7FF et U+E000 ... U+FFFF :
   représentés par *un* mot de 16 bits avec la même valeur
2. U+D800 ... U+DFFF : ne sont pas des code points valides
3. U+10000 ... U+10FFFF : deux mots.
   Algorithme pour conversion de U+x_{16} :
   (a) Calculer $(x')_{16} = (x)_{16} - (10000)_{16}$
   (b) Représenter $(x')_{16}$ en binaire $(b')_2$ avec 20 chiffres : $(x')_{16} = (b')_2$
   (c) Scinder $(b')_2$ en deux mots v et w de 10 bits
   (d) Résultat du codage :
   — Premier mot : $(110110v)_2$
   — Deuxième mot : $(110111w)_2$

UTF-8 Codage entre 1 et 4 octets, selon la table :

Intervalle	*Octet 1*	*Octet 2*	*Octet 3*	*Octet 4*
U+0 ... U+7F	0xxxxxxx			
U+80 ... U+7FF	110xxxxx	10xxxxxx		
U+800 ... U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
U+10000 ... U+10FFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

A.2 Inégalité de Kraft

Il existe un code préfixe binaire avec k codes $u_1 \dots u_k$ aux longueurs $|u_1| \dots |u_k|$ si et seulement si

$$\sum_{i=1}^{k} 2^{-|u_i|} \leq 1$$

A.3 Théorie de l'information

Entropie (selon Shannon) d'une source d'information avec des événements $\{x_i | i \in I\}$:

$$H =_{def} \sum_{i \in I} (-\log_2(P(x_i))) * P(x_i)$$

Longueur moyenne d'un ensemble (mot × probabilté) : $lnm(E) = \sum_{(m,p) \in E} |m| * p$

Notions de logarithme

— Définition du logarithme en base b : $\log_b(x) = y$ si et seulement si $x = b^y$.
— Calcul du logarithme binaire à l'aide du logarithme décimal : $\log_2(x) = \frac{\log_{10}(x)}{\log_{10}(2)}$