Théorie de l'information. Examen de Rattrapage 2013-2014.

Ordinateur interdit. Aucun document papier autorisé à l'exception d'une feuille A4 Recto-Verso aide-mémoire.Calculatrice autorisée. Durée : 2h00.

Exercice 1. Entropie du code à répétition. (2 points)

On considère une source binaire d'ordre 1, S_1, émettant les symboles 0 et 1 avec les probabilités respectives p et q. On note S_n la source d'ordre n, émettant des n-uplets successifs de symboles de S_1 (S_n peut être considérée comme n sources S_1 *indépendantes*). Ainsi S_2 émet les 4 symboles 00, 01, 10 et 11, S_3 émet les 8 symboles 000, 001, 010, 011, 100, 101, 110 et 111, …

1. Calculer en fonction de p et q, *l'entropie* notée H_1 de la source S_1 d'ordre 1.
2. Calculer en fonction de H_1, *l'entropie* notée H_2 de la source S_2 d'ordre 2.
3. Calculer en fonction de H_1, *l'entropie* notée H_3 de la source S_3 d'ordre 3.
4. En déduire *l'entropie* notée H_n de la source S_n d'ordre n, toujours exprimée en fonction de H_1.

Exercice 2. Codage de Huffman. (6 points)

On considère une image numérique couleur dont la palette de couleurs est ramenée à $N = 8$ couleurs, dont la fréquence est donnée dans la table suivante :

Table des fréquences des couleurs présentes dans l'image :

Couleur s_i	noir	blanc	cyan	jaune	rouge	vert	marron	gris
Fréquence p_i	**0.25**	**0.15**	**0.05**	**0.10**	**0.20**	**0.05**	**0.10**	**0.10**

1. Quelle est la *quantité moyenne d'information* présente dans l'image (valeurs **littérale** et **numérique**) ?
2. Construire un *arbre de Huffman* en vue de la compression sans pertes de l'image.
3. En déduire un *code optimal de Huffman binaire* associé à chaque couleur.
4. Donner la *longueur moyenne* du code de Huffman construit (valeur **numérique**).
5. Indiquer (en %) le *taux de compression t* réalisé par rapport à une transmission standard (codage fixe **8 bits par couleur**).
6. La *longueur moyenne* du code binaire de Huffman obtenu *atteint-elle* la valeur de l'*entropie* (justifier) ?

Exercice 3. Canal de transmission bruité. (4 points)

Soit une *source binaire* X d'alphabet $\Omega_X = \{\alpha, \beta, \gamma\}$ de distribution de probabilités (*marginale*) $P_X = \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix}$ et un *récepteur binaire* Y d'alphabet identique. On considère un canal de transmission *binaire* bruité de *matrice de transition* :

$$P(Y|X) = \begin{pmatrix} 1/2 & 1/4 & 1/4 \\ 1/4 & 1/2 & 1/4 \\ 1/4 & 1/4 & 1/2 \end{pmatrix}$$

1. S'agit-il d'un canal de transmission *symétrique* (justifier).
2. Calculer les *matrices de probabilités conjointe* $P(X,Y)$, *marginale* P_Y et *conditionnelle* $P(X|Y)$ (valeurs **numériques**).
3. Calculer les *entropies* $H(X)$ et $H(X|Y)$ (valeurs **numériques**).
4. Déterminer l'*information mutuelle moyenne* $I(X,Y)$ ainsi que la *capacité* C *du canal* (valeurs **numériques**).

Rappel : Capacité C d'un canal : $C = \max_{P(X)} I(X,Y)$

Exercice 4. Code correcteur. (5 points)

On considère un code en bloc linéaire, de paramètres $(n,k)=(8,2)$, de matrice génératrice : $\mathbf{G}=\begin{pmatrix}1&1\\0&1\\1&1\\0&1\\1&0\\1&1\\1&0\\1&0\end{pmatrix}$

1. Ecrire la *matrice génératrice* sous forme systématique $\tilde{\mathbf{G}}$ permettant d'obtenir la *forme systématique* du code.
2. Donner la *table de code* obtenue par construction systématique du code.
3. En déduire la *distance minimale* d de ce code.
4. Combien d'erreurs peut-il toujours *détecter* ? Combien d'erreurs peut-il toujours *corriger* ?
5. Soit le message reçu $\boldsymbol{\omega}^{\mathrm{T}}=(0\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1)$ entaché de 2 erreurs. Donner le(s) *mot(s)-source(s)* issu(s) du décodage de $\boldsymbol{\omega}$ par minimum de distance de Hamming.
6. Soit le message reçu $\boldsymbol{\Omega}^{\mathrm{T}}=(0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 1\ \ 1\ \ 1)$ entaché de 3 erreurs. Donner le(s) *mot(s)-source(s)* issu(s) du décodage de $\boldsymbol{\Omega}$ par minimum de distance de Hamming.

Exercice 5. Chiffre de Vernam. (3 points)

On considère un *chiffrement de Vernam* (1917) appliqué sur un fichier segmenté en blocs de n bits : chaque bloc de *message clair* $\mathbf{m}$ de n bits est crypté pour donner le *chiffré* $\mathbf{c}$ de n bits à l'aide d'une clé secrète *unique* $\mathbf{k}$ de n bits également, par l'algorithme (utilisant l'opérateur OU exclusif $\oplus$ bit à bit) : $\mathbf{c}=\mathbf{m}\oplus\mathbf{k}$.
Le chiffrement est *symétrique* : la même clé est utilisée pour le chiffrement et le déchiffrement ; en outre, elle est *identique* pour le chiffrement de tous les blocs.

1. Exprimer *la clé* $\mathbf{k}$ à partir de l'interception d'un bloc de message clair $\mathbf{m}$ et du chiffré $\mathbf{c}$ correspondant.
2. Exprimer le *message clair* $\mathbf{m}$ issu du déchiffrement du chiffré $\mathbf{c}$ avec la clé $\mathbf{k}$.
3. *Application :* $n=3$ bits : on intercepte le bloc de message clair $\mathbf{m}=010$ et le chiffré $\mathbf{c}=101$ correspondant.
3.1. Déterminer *la clé* $\mathbf{k}$ (valeur *numérique*).
3.2. En déduire le *message clair* $\mathbf{m}'$ (valeur *numérique*) issu du déchiffrement du chiffré $\mathbf{c}'=011$.
4. On intercepte maintenant 2 *blocs de messages chiffrés* de n bits $\boldsymbol{\omega}$ et $\boldsymbol{\omega}'$ mais *sans connaître les messages clairs* $\boldsymbol{\mu}$ et $\boldsymbol{\mu}'$ correspondants. Montrer que le calcul de $\boldsymbol{\omega}\oplus\boldsymbol{\omega}'$ permet néanmoins d'apporter de l'information sur les messages clairs $\boldsymbol{\mu}$ et $\boldsymbol{\mu}'$, constituant ainsi une faille du chiffrement.
5. Proposer une amélioration du chiffrement (prévue par Vernam) pour le rendre plus sûr.

ANNEXE. Table de logarithme de base 2

p	$\log_2(p)$	$plog_2(1/p)$
0.01	-6.6438	0.0664
0.02	-5.6438	0.1128
0.03	-5.0588	0.1517
0.04	-4.6438	0.1857
0.05	-4.3219	0.2160
0.06	-4.0588	0.2435
0.07	-3.8365	0.2685
0.08	-3.6438	0.2915
0.09	-3.4739	0.3126
0.10	-3.3219	0.3321
0.11	-3.1844	0.3502
0.12	-3.0588	0.3670
0.13	-2.9434	0.3826
0.14	-2.8365	0.3971
0.15	-2.7369	0.4105
0.16	-2.6438	0.4230
0.17	-2.5563	0.4345
0.18	-2.4739	0.4453
0.19	-2.3959	0.4552
0.20	-2.3219	0.4643
0.21	-2.2515	0.4728
0.22	-2.1844	0.4805
0.23	-2.1202	0.4876
0.24	-2.0588	0.494134
0.25	-2.00	.50
0.26	-1.9434	0.5052
0.27	-1.8889	0.5100
0.28	-1.8365	0.5142
0.29	-1.7858	0.5179
0.30	-1.7369	0.5210
0.31	-1.6896	0.5237
0.32	-1.6438	0.5260
0.33	-1.5994	0.5278

p	$\log_2(p)$	$plog_2(1/p)$
0.34	-1.5563	0.5291
0.35	-1.5145	0.5301
0.36	-1.4739	0.5306
0.37	-1.4344	0.5307
0.38	-1.3959	0.5304
0.39	-1.3584	0.5297
0.40	-1.3219	0.5287
0.41	-1.2863	0.5273
0.42	-1.2515	0.5256
0.43	-1.2175	0.5235
0.44	-1.1844	0.5211
0.45	-1.1520	0.5184
0.46	-1.1202	0.5153
0.47	-1.0892	0.5119
0.48	-1.0588	0.5082
0.49	-1.0291	0.5042
0.50	-1.00	0.50
0.51	-0.9714	0.4954
0.52	-0.9434	0.4905
0.53	-0.9159	0.4854
0.54	-0.8889	0.4800
0.55	-0.8624	0.4743
0.56	-0.8365	0.4684
0.57	-0.8109	0.4622
0.58	-0.7858	0.4558
0.59	-0.7612	0.4491
0.60	-0.7369	0.4421
0.61	-0.7131	0.4350
0.62	-0.6896	0.4275
0.63	-0.6665	0.4199
0.64	-0.6438	0.4120
0.65	-0.6214	0.4039
0.66	-0.5994	0.3956

p	$\log_2(p)$	$plog_2(1/p)$
0.67	-0.5777	0.3871
0.68	-0.5563	0.3783
0.69	-0.5353	0.3693
0.70	-0.5145	0.3602
0.71	-0.4941	0.3508
0.72	-0.4739	0.3412
0.73	-0.4540	0.3314
0.74	-0.4344	0.3214
0.75	-0.4150	0.3112
0.76	-0.3959	0.3009
0.77	-0.3770	0.2903
0.78	-0.3584	0.2795
0.79	-0.3400	0.2686
0.80	-0.3219	0.2575
0.81	-0.3040	0.2462
0.82	-0.2863	0.2347
0.83	-0.2688	0.2231
0.84	-0.2515	0.2112
0.85	-0.2344	0.1992
0.86	-0.2175	0.1871
0.87	-0.2009	0.1747
0.88	-0.1844	0.1622
0.89	-0.1681	0.1496
0.90	-0.1520	0.1368
0.91	-0.1360	0.1238
0.92	-0.1202	0.1106
0.93	-0.1046	0.0973
0.94	-0.0892	0.0839
0.95	-0.0740	0.0703
0.96	-0.0588	0.0565
0.97	-0.0439	0.0426
0.98	-0.0291	0.0285
0.99	-0.0145	0.0143