Science de l'information : Série 3

Exercice 3.1 :

1. Pour calculer le code optimal, nous calculons la longueur moyenne de chaque lettre L1 et L2. La longueur moyenne de L1 se trouve en plaçant les lettres sur un arbre de Huffman (c.f. Annexe). Nous trouvons donc que la lettre E (code {00}, probabilité 2/8) est de longueur 2, et que les autres sont de longueur 3, probabilité 1/8. Donc, nous trouvons la longueur moyenne de L1 est de 2/8*2 + 6*1/8*3 = 11/4 = 2.75 bits. Celle de L2 étant la même que celle de L1, et les deux sources étant indépendantes, L = L1 + L2 = **5.5 bits**. L'entropie et le code de Shannon-Fano nous donnent le même résultat (c.f. séries précédentes).

2. Ici, même démarche qu'avant : placer les issues possibles (cette fois les 13 mots de Mr. Burns) dans un arbre de Huffman (c.f. annexe). Tous les mots ont une probabilité d'apparition de 1/13. 3 mots, RI, PI et IN ont une longueur 3, les 10 autres ont une longueur de 4.Le calcul de la longueur moyenne (3*1/13*3 + 10*1/13*4) donne une longueur moyenne de **3.769 bits. Le Shannon-Fano a une longueur de 4, et l'entropie de B est de 3.70043. Cela respecte l'inégalité H(B) ≤ Huffman (B) ≤ Shannon-Fano(B).**

3. L'entropie de M, comme calculée dans la série 1, est de 5.34. Comme aucun code ne peut faire mieux que l'entropie, cela nous marque la borne inférieure. Pour la borne supérieur en revanche, il faut Se rappeler de Lisa, dont la longueur moyenne (ainsi que le Shannon-Fano et l'entropie) vaut 5.5. Or, la densité de probabilité de de M et de L ne sont pas égales : Lisa pourra avoir 64 (8*8 lettres) possibilités, et Homer seulement 56 (on compte pour ça qu'il tire ses lettres en même temps, donc 8 lettres pour M1 OU M2, et 7 lettres pour l'autres, du fait qu'il n'y a pas de remise). Depuis là, comme la densité de probabilité de M est moindre que L, la longueur moyenne de M sera forcément inférieur (ou égal selon les cas) à celle de L, et donc plus petite que 5.5.

   Ces deux points montrent que la longueur moyenne sera plus grande ou égale à 5.34 (entropie) et plus petite ou égale à 5.5 (Lisa), donc dans l'intervalle [5.34, 5.5].

4. Pour A, il est facile de réduire la taille, en enlevant un 0 à chaque code (a devient {10}, b {1000},…). Pour B en revanche, comme l'inégalité de Kraft donne 1, il est impossible de réduire la taille sans lui enlever la propriété du décodage unique.

Exercice 3.2

1. Les probabilités réparties avec i dans les lignes et j dans les colonnes

i\j	E	N	T	R	O	P	I
E	0	1/2	1/2	0	0	0	0
N	1/3	0	0	0	1/3	0	1/3
T	1/2	0	0	0	1/2	0	0
R	1/2	0	0	0	0	0	1/3
O	0	1/2	0	1/2	0	0	0
P	0	0	0	0	0	0	1
I	0	1	0	0	0	0	0

2.

   a. En sachant que $H(B_2|B_1) = H(B_1, B_2) – H(B_1)$ (théorie), et sachant que $H(B_1)$ vaut 2.72 et que $H(B_1, B_2)$ vaut 3.7, on trouve l'entropie conditionnelle = 3.7-2.72 ≈ **0.98 bits**

   b. La seconde manière se calcule de la manière suivante : il faut calculer les entropies de B2 sachant que B1 vaut une certaine lettre (pour chaque lettre) puis sommer les entropies selon leur probabilités d'apparition. Ainsi :

      H(B2|B1 = E) = H(B2|B1 = O) = H(B2|B1 = T) = H(B2|B1 = R) = 2*(1/2*log2(2)) = 1 bit

      H(B2|B1 = N) = 3*(1/3*log2(3) = 1.5849 bits

      H(B2|B1 = P) = H(B2|B1 = I) = 0 (car si B1 est P, B2 sera forcément I, donc l'entropie est de 0. Pareil pour B1 = I et B2 = N)

      En prenant le nombre de lettres commençant par chaque lettre (2 pour E par exemple), divisé par le nombre de mots (13) possibles, et multipliés par l'entropie de chaque lettre, nous trouvons :
      2/13*4 +3/13*1.5849 = **0.9811 bits** ce qui est égal à l'entropie conditionnelle trouvée en a.

3. Nous utilisons ici la méthode vue au 2.a, à savoir $H(L_2|L_1) = H(L_1, L_2) – H(L_1)$. $H(L_1)$ = 2.75 bits, et $H(L_1, L_2)$ = 5.5 bits, donc **$H(L_2|L_1) = 2.75$ bits**

4. Pareil, $H(M_2|M_1) = H(M_1, M_2) – H(M_1)$. $H(M_1) = 2.75$ bits, et $H(M_1, M_2) = 5.34$ bits, donc **$H(M_2|M_1) = 2.59$ bits**

5. **Bart a raison** (pour une fois). En effet, l'entropie conditionnelle se définit comme étant la quantité d'information supplémentaire. Donc l'information contenue dans M_1 (ici). Or, comme $H(M_2|M_1) < H(L_2|L_1)$, nous pouvons en déduire que ce garnement de Bart a raison, et qu'il y a plus d'information dans la seconde lettre de Lisa que dans celle de Bart.

Exercice 3.3

1.

   a. – Demander si lettre est 'E' (réponse non)

      - Demander si lettre est Voyelle (réponse oui)

      - Demander si lettre es 'O'(réponse non)

      → la lettre est 'I'

   b. Il faut 1 question pour arriver a E {1}, 3 pour O {011}, 3 pour I {010}, 4 pour N {0011}, 4 pour P {0010}, 4 pour R {0011} et 4 pour T {0000}. E a une densité de ¼, les autres de 1/8, donc la longueur moyenne est de

      ¼*1 + 2*(1/8*3) + 4*(1/8*4) = **3 questions**

      La longueur max quant à elle est de 4.

2. Le code de Lisa pose K questions, mais on se sait pas lesquelles. Mais il est à noter que dans cette optique il est impossible d'avoir un préfixe ! On ne peut pas répondre oui à la question Est-ce que la lettre est E, avoir un oui et continuer ! Du moment qu'on arrive à une lettre, l'algorithme s'arrête net, immédiatement, sans attente. Ainsi, cette manière de faire affirme que le code est sans préfixe, donc instantané, donc a décodage unique. Vlan !

3. Le nombre moyen de questions L revient à donner la longueur moyenne de l'encodage (les réponses étant limitées à oui/non, l'encodage est binaire). La longueur moyenne (optimale) L est plus petite ou égale à $H(S)/\log2(D)$ (ici $D = 2$ car le code est binaire), donc $L(S) \leq H(S)$.

   Ici K est la longueur maximum (aka le nombre max de questions). Donc, le nombre max de questions est forcément plus grand ou égal au nombre moyen de questions. De ce fait, comme 'L' est plus grand ou égal à H(S) et que K est plus grand ou égal à L, alors K est plus grand ou égal à H(S) !

4. Comme on encode le message binaire avec un code binaire, on n'a qu'à utiliser le même encodage que pour L_1 (aux questions précédentes). On exécute donc, sur la place publique, un code de Huffman, donc nous trouvons une longueur moyenne de 2.75 questions, comme pour L_1.

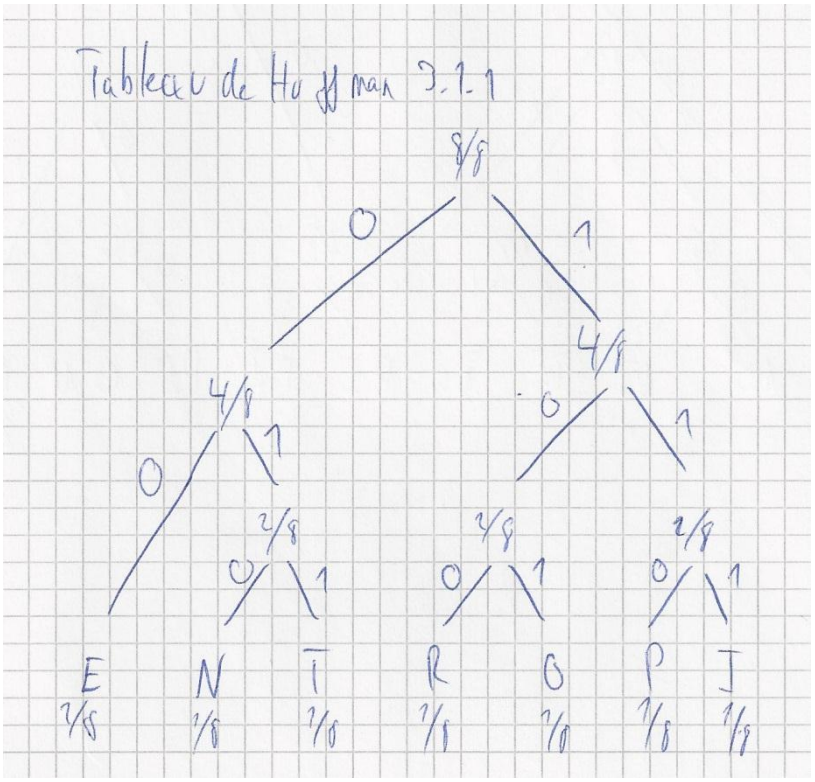
Excercice 3.4

1. La longueur moyenne se trouve en multipliant la longueur (binaire) de chaque lettre par sa probabilité d'apparition, et en les sommant… de se sommer. Nous trouvons donc une la longueur moyenne ainsi : 2*8.11+4*0.81+4*3.38+3*4.28+1*17.69+4*1.13+3*1.19+4*0.74+2*7.24+4*0.18 +3*0.02+4*5.99+2*2.29+2*7.69+3*5.20+4*2.92+4*0.83+3*6.43+3*8.87+1*7.44 +3*5.23+4*1.23+3*0.06+4*0.53+4*0.26+4*0.12 (vérifiez, je suis pas sûr du calcul). Ce qui donnerait une longueur moyenne de **2.4211 bits** .

2. L'entropie du français (calculée dans le livre de cours) est de 3.95 bits. L'entropie calculée grâce au morse étant de 2.4211 bits, nous pouvons affirmer que le code morse est un bon moyen de compression, car largement inférieur à l'entropie ''normale''.

Annexes

Annexe 1



Annexe 1