EPFL - Information, Calcul et Communication - Partie théorique

Série 11: Solutions

1 Comparaisons d'entropies

On indique ci-dessous en **gras** le mot avec la plus grande entropie:

1. EPFL et EEPPFFLL: Les entropies sont égales (et valent 2) car dans les deux cas, les 4 lettres E, P, F et L apparaissent toutes avec probabilité 1/4.

2. AAAH et **HAHA**: L'entropie du premier mot vaut $(3/4)\log_2(4/3) + (1/4)\log_2(4) \simeq 0.81$, tandis celle du deuxième vaut 1. On peut aussi voir qu'il y a moins de possibilités de former des mots différents avec les 4 lettres AAAH qu'avec les 4 lettres HAHA, donc moins d'entropie dans le premier mot.

3. **MEDITERRANNEE** et MEDETERRENNEE: L'entropie du premier mot est clairement plus grande, car le nombre de lettres dans chaque mot est le même, mais toutes les voyelles sont remplacées par des E dans le second mot.

4. ABB et **ABBA**: L'entropie du premier mot vaut $(2/3)\log_2(3/2) + (1/3)\log_2(3) \simeq 0.91$, tandis que celle du second vaut 1.

5. **ACD** et ACDC: L'entropie du premier mot vaut $\log_2(3) \simeq 1.58$, tandis que celle du second vaut 1.5.

6. ABR et **ABRI**: L'entropie du premier mot vaut $\log_2(3)$, tandis que celle du second vaut $\log_2(4)$.

7. CALC et **CALCUL**: L'entropie du premier mot vaut 1.5, tandis que celle du second vaut $(2/3)\log_2(3) + (1/3)\log_2(6) \simeq 1.92$.

2 Est-ce que l'entropie augmente ou diminue?

a) La réponse est non. Il se peut que l'entropie augmente (par exemple, avec ABB et ABBA) ou diminue (par exemple, avec ACD et ACDC).

b) Ici, la réponse est oui (par exemple, avec ABR et ABRI). La démonstration formelle de ce fait dépasse le cadre de ce cours, mais une intuition simple est la suivante: rajouter une lettre qui ne fait pas partie de la séquence amène quelque chose de *nouveau* à celle-ci et augmente donc le nombre de possibilités de former des mots différents, ce qui implique une augmentation de l'entropie.

3 Un peu de magie noire

a) Calcul de l'entropie de la séquence (que l'on va appeler $\mathcal{X}$ comme dans le cours): le A a une probabilité d'apparition de $\frac{5}{12}$, le V et le D de $\frac{2}{12}$ et les 3 lettres restantes de $\frac{1}{12}$. Donc

$$\begin{aligned} H(\mathcal{X}) &= \frac{5}{12}\log_2\left(\frac{12}{5}\right) + 2\,\frac{2}{12}\log_2\left(\frac{12}{2}\right) + 3\,\frac{1}{12}\log_2(12) \\ &= \log_2(12) - \frac{5}{12}\log_2(5) - \frac{1}{3}\log_2(2) = \frac{5}{3} + \log_2(3) - \frac{5}{12}\log_2(5) \end{aligned}$$

ce qui donne numériquement: $H(\mathcal{X}) \simeq 2.28$.

b) Avec l'algorithme de Shannon-Fano, on peut trouver les dictionnaires suivants:

lettre	nb app.	1) nb Q	mot de code	2) nb Q	mot de code	3) nb Q	mot de code
A	5	2	11	2	11	1	1
V	2	2	10	2	10	3	011
D	2	3	011	2	01	3	010
K	1	3	010	3	001	3	001
E	1	3	001	4	0001	4	0001
R	1	3	000	4	0000	4	0000

Avec les dictionnaires 1) et 2), la séquence codée contient 29 bits; avec le dictionnaire 3), la séquence codée contient 28 bits.

c) Avec l'algorithme de Huffman, on trouve le dictionnaire suivant:

lettre	nb apparitions	mot de code
A	5	1
V	2	011
D	2	010
K	1	001
E	1	0001
R	1	0000

et la séquence codée contient 28 bits. On peut aussi trouver des dictionnaires différents suivant l'arbre que l'on construit, mais la longueur de la séquence codée reste invariablement de 28 bits dans ce cas.

d) Avec l'algorithme de Shannon-Fano, la longueur moyenne du code $L(\mathcal{C}_{SF}) \simeq 2.42$ ou 2.33. Avec l'algorithme de Huffman, la longueur moyenne du code $L(\mathcal{C}_H) \simeq 2.33$, et l'entropie de la séquence vaut $H \simeq 2.28$. On vérifie donc bien les inégalités du cours:

$$H(\mathcal{X}) \leq L(\mathcal{C}_H) \leq L(\mathcal{C}_{SF}) \leq H(\mathcal{X}) + 1$$

4 De bien mauvais dicos

Explorons toutes les raisons pour lesquelles les minions ont failli à trouver de bons dictionnaires:

a) Le dictionnaire proposé ici n'utilise même pas un code binaire, mais un code "unaire". L'encodage du mot BANANA avec ce dictionnaire donne 1111111111, autrement dit quelque chose de tout à fait illisible! Il faudrait rajouter des caractères de séparation entre les mots de code si on voulait y comprendre quelque chose.

b) Ce dictionnaire est d'un côté celui qui utilise le moins de bits (7) pour represénter le mot BANANA, mais le code binaire ainsi généré n'est pas sans préfixe et est donc difficilement décodable: en particulier, quant on lit 11, on ne sait jamais si on doit interpréter ça comme B ou NN. De manière correspondante, la longueur moyenne du code binaire utilisé ici est $\frac{7}{6}$, qui est en dessous de la limite de Shannon (= l'entropie du mot $\simeq 1.46$; voir calcul ci-dessous).

c) Ici, le code est sans préfixe et donc décodable de manière unique, mais la longueur moyenne du code $L(\mathcal{C}) = \frac{10}{6} = \frac{5}{3}$ n'est pas optimale; le mot de code associé à la lettre B est en fait inutilement trop long.

d) Ce dictionnaire utilise un code sans préfixe (du fait que tous les mots de code ont la même longueur) et est donc décodable de manière unique, mais il n'atteint pas la performance optimale, car sa longueur moyenne $L(\mathcal{C}) = 2$ (forcément, puisque tous les mots de code sont de longueur 2), qui est plus éloignée de l'entropie du mot que nécessaire.

e) Le code utilisé ici est aussi sans préfixe, donc décodable de manière unique, mais le mot de code court (1) est attribué à la lettre la moins fréquente. La longueur moyenne du code résultante vaut $L(\mathcal{C}) = \frac{11}{6} \simeq 1.88$, qui est loin d'être optimale.

f) Ce code cumule deux défauts: sa longueur moyenne n'est pas optimale et il n'est pas sans préfixe (donc difficilement décodable).

Finalement, un bon dictionnaire est le suivant (qu'on obtient indifféremment par l'algorithme de Shannon-Fano ou de Huffman):

A	N	B
0	10	11

Il utilise 9 bits en tout pour la représentation du mot BANANA et sa longueur moyenne vaut donc $L(\mathcal{C}) = \frac{9}{6} = \frac{3}{2}$, qui est très proche de l'entropie du mot: $H(\mathcal{X}) = \frac{1}{2}\log_2(2) + \frac{1}{3}\log_2(3) + \frac{1}{6}\log_2(6) = \frac{2}{3} + \frac{1}{2}\log_2(3) \simeq 1.46$.

5 A la recherche d'un trésor

a) On doit chaque fois se mettre au point sur la grille qui est au milieu des possibilités qui nous restent, de sorte à diviser cet ensemble de possibilités en quatre parties égales; vu qu'il y a 64 possibilités au départ, le nombre de questions à poser est $\log_4(64) = 3$.

b) (5,5), (7,7), (6,8) (et la suite des réponses correspondantes de l'oracle est NE, NO, SE).

c) Deux possibilités:

- En vous basant sur la suite des questions ci-dessus, vous encodez chaque réponse obtenue avec 2 bits, p.ex.: NE=00, NO=01, SE=10, SO=11 (on peut aussi dire, N=0, S=1, O=0, E=1 d'ailleurs, mais se rappeler alors qu'on indique d'abord la direction nord-sud avant la direction est-ouest). Dans le cas présent, la séquence est donc 000110.

- Sans rapport avec le jeu des questions ci-dessus, on peut aussi simplement encoder la position de la case avec 6 bits également: 3 bits pour la position horizontale (de 1 à 8: plus précisément, on va encoder le nombre moins 1, comme dans l'exercice sur le codage par plages: ainsi, 000 encode 1, 001 encode 2, jusqu'à 111 qui encode 8), et 3 bits pour la position verticale. Avec cette représentation, la position du trésor ci-dessus est (6,7), et donc l'encodage est 101110 (on utilise la convention que l'abscisse vient avant l'ordonnée: pas besoin donc d'utiliser un bit pour ça: ça fait partie de la manière dont on définit notre dictionnaire).

6 Pour le plaisir: minimiser le nombre de pesées*

a) Observer tout d'abord la ressemblance entre ce problème et le jeu des questions vu au cours. Ici, à la place d'un "oracle" qui nous donne des réponses "oui" ou "non", on a maintenant une balance qui nous indique "penche à gauche", "penche à droite", ou "reste stable". Chaque pesée nous permet donc théoriquement de diviser par 3 (et non par 2) l'ensemble des possibilités. Il s'agit d'effectuer les bonnes pesées pour être le plus efficace possible.

"Numérotons" les 9 pièces par des lettres: ABCDEFGHI. Une d'entre elles est plus légère: il y a donc 9 possibilités en tout. Pour diviser par 3 l'ensemble des possibilités, nous effectuons la pesée suivante:

ABC-DEF

(signifiant qu'on place ABC à gauche et DEF à droite de la balance, les autres pièces GHI restant sur la table.)

- Si la balance penche à gauche (ce qu'on note ABC>DEF: le poids de ABC est plus grand que celui de DEF), alors on sait que la pièce plus légère est dans DEF.

- Si la balance penche à droite (ce qu'on note ABC<DEF), alors on sait que la pièce plus légère est dans ABC.

- Si la balance reste stable (ce qu'on note ABC=DEF), alors on sait que la pièce plus légère est dans GHI.

Ainsi, on a bien réduit par 3 l'ensemble des possibilités. Il suffit ensuite de répéter l'opération avce les trois pièces restantes XYZ. Plus précisément, on effectue la pesée:

X-Y

Si X>Y, la pièce plus légère est Y; si X<Y, la pièce plus légère est X; si X=Y, alors forcément, Z est la pièce plus légère, toutes les autres ayant le même poids.

En conclusion, 2 pesées suffisent à trouver la pièce défectueuse. Il est à noter que $2 = \log_3(9)$. C'est la définition de l'*entropie ternaire* (alors que dans le cours et les exercices précédents, nous avons affaire à l'entropie binaire).

b) Pour ce problème, "numérotons" les 4 pièces par des lettres: ABCD et appelons X la pièce qu'on a dans notre poche et qu'on sait avoir le bon poids. On a de nouveau 9 possibilités:

- ou bien une des 4 pièces est plus légère que les autres;

- ou bien une des 4 pièces est plus lourde que les autres;

- ou bien toutes les pièces ont le même poids.

Pour diviser cet ensemble de 9 possibilités en 3 parties égales, voici la première pesée à effectuer:

AB-CX

(avec donc D qui reste sur la table).

1) Si AB=CX, alors il nous reste bien 3 possibiliés: soit la pièce D est plus lourde, soit elle est plus légère, soit toutes les pièces ont le même poids. On effectue donc une deuxième pesée D-X qui nous donne la réponse.

2) Si AB<CX, alors il reste également 3 possibilités: soit l'une des pièces A ou B est plus légère, soit la pièce C est plus lourde. On effectue alors la pesée A-B: si A=B, c'est C qui est plus lourde; si A<B, c'est A qui est plus légère; si A>B, c'est B qui est plus légère.

3) Si AB>CX, alors il reste également 3 possibilités: soit l'une des pièces A ou B est plus lourde, soit la pièce C est plus légère. On effectue alors la pesée A-B: si A=B, c'est C qui est plus légère; si A<B, c'est B qui est plus lourde; si A>B, c'est A qui est plus lourde.

Ainsi, au total, 2 pesées suffisent à répondre aux questions de l'énoncé. Ce n'est à nouveau pas un hasard que $2 = \log_3(9)$.