TD 2 : Information selon Shannon et codage optimal

1 Inégalité de Kraft (suite)

Exercice 1

1. Utilisez l'inégalité de Kraft pour montrer qu'il est possible de construire un code préfixe binaire pour les caractères $a \dots f$ avec les longueurs de code suivants :

caractère	a	b	c	d	e	f
longueur	2	3	4	2	4	3

2. Construisez effectivement un arbre de codage pour cet exemple.
3. Vous constatez qu'un code n'est pas utilisé. Lequel ?
4. Évidemment, une telle situation est embêtante surtout pour le décodage de messages (certains messages ne peuvent pas être décodés). Montrez que tout alphabet fini avec au moins deux caractères A_1 peut être codé par un code préfixe binaire, donc par une fonction $c : A_1 \to \{0,1\}^+$, de telle manière qu'il n'y ait pas de codes non utilisés. (Bien sûr, ceci n'est pas possible pour n'importe quelles contraintes sur les longueurs des codes.) Donnez un tel code pour les caractères $a \dots f$.
5. Montrez que ceci n'est pas toujours possible si on utilise un code ternaire (donc un arbre de décodage où chaque noeud est une feuille ou a exactement trois successeurs).

2 Entropie et longueur de codes

Exercice 2 Dans cet exercice, nous nous intéressons à un codage des caractères a, b, c, d par des séquences de nombres binaires, et du temps de transmission d'un texte T de 100 caractères sur le canal $\mathcal{C}$ qui peut transmettre 20 bit/sec, donc 20 chiffres binaires par seconde.

1. Supposons que l'occurrence des caractères est équiprobable, et que les caractères $a, \dots d$ sont codés par $00, \dots 11$ respectivement. Combien de temps faut-il pour communiquer le texte sur le canal $\mathcal{C}$?
2. Supposons maintenant une source d'information S dont la probabilité des caractères est indiquée dans ce tableau :

caractère	a	b	c	d
probabilité	0.5	0.2	0.2	0.1

et que nous continuons à coder les caractères comme dans (1). Quelle est l'espérance de la taille du code du texte T, et, en moyenne, son temps de transmission sur $\mathcal{C}$?

3. Nous essayons d'optimiser et nous proposons le code suivant :

caractère	a	b	c	d
code	0	10	110	1110

Vérifiez bien qu'il s'agit d'un code préfixe ! Pour la distribution de probabilité de (2), nous posons de nouveau la question de la taille moyenne de T et du temps de transmission sur $\mathcal{C}$.

4. Calculez l'entropie de la source S de (2) pour obtenir une borne inférieure du temps de transmission de T sur $\mathcal{C}$.
5. Il s'avère que le code de (3) n'est pas encore optimal. Utilisez l'algorithme de Huffman pour calculer un code optimal. Comparez avec l'entropie.

Exercice 3 Une source d'information émet les caractères `a,b,c,d` selon la distribution de probabilité suivante :

a	b	c	d
0.4	0.2	0.3	0.1

1. Quelle est l'entropie de cette source d'information ?
2. Construisez l'arbre de Huffman et attribuez un code binaire optimal à chacun des caractères `a,b,c,d`. Calculez la taille moyenne du code, et comparez-la à l'entropie.
3. Un collègue vous dit qu'il arrive à coder un texte de 24 caractères avec 18 bits. Vous lui dites qu'il se trompe. Quelle est votre justification ?
4. Votre collègue vous présente un exemple : `aaaaaaaaccccccccbbbbdddd` codé par `00111.10111.0111.1111`. Son idée : Coder les caractères `a` ... `d` par `00` ... `11`, suivi du nombre $n-1$ codé en binaire, pour n occurrences consécutives du caractère. Par exemple, ceci donne `0111` pour la séquence de 4 `b`. Où est le malentendu ?

Exercice 4 Les deux parties du théorème de Shannon donnent des bornes (inférieure et supérieure) pour un code optimal, en fonction de l'entropie de la source d'information que le code est censé coder. Cet exercice a pour but d'explorer ce rapport plus en détail.

1. Une source émet les caractères `a,b,c,d,e` selon la distribution de probabilité suivante :

a	b	c	d	e
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$

   Construisez l'arbre de Huffman, calculez la taille moyenne du code et comparez avec l'entropie. Constat ?
2. Vous observez que toutes les probabilités de l'exemple précédant ont la forme 2^{-k}. A quelle profondeur de l'arbre de codage se trouve un caractère dont la probabilité est 2^{-k} ?
3. Faites de même pour la distribution suivante :

a	b	c	d	e
$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{2}{15}$	$\frac{1}{12}$

4. Nous généralisons l'observation du point (1) : Supposons qu'une source d'information émet les caractères $c_1 \dots c_n$ avec des probabilités associées $p_1 \dots p_n$ qui sont toutes des puissances négatives de 2, donc de la forme $p_i = 2^{-k_i}$ et telles que $\sum_{i=i}^{n} p_i = 1$. Alors, la taille moyenne du code construit par l'algorithme de Huffman est égale à l'entropie de la source.

   Pour cela, nous montrons les invariants suivants de l'algorithme de Huffman pour l'ensemble $\mathcal{A} = \{a_1 \dots a_t\}$ des arbres qu'il manipule :

   (a) A tout instant, tout arbre de l'ensemble $\mathcal{A}$ a une racine avec une valeur qui est de la forme 2^{-k}.

   (b) Tant qu'il existent encore deux arbres dans l'ensemble $\mathcal{A}$, il existent au moins deux arbres dans $\mathcal{A}$ dont la probabilité est minimale.

   (c) A tout instant, l'entropie de la source est égale à $lc(a_1) + \dots lc(a_t)$, où, pour un arbre a avec racine 2^{-k}, nous définissons $lc(a) = lnm^{+k}(cpf(a))$, et cpf est défini comme sur les transparents du cours et $lnm^{+k}(E) = \sum_{(m,p)\in E} p * (|m| + k)$.

   Assurez-vous que ces propriétés tiennent pour l'exemple du point (1). Démontrez ensuite qu'il s'agit d'un invariant, à savoir, que les propriétés sont satisfaites au début de l'algorithme ; qu'elles sont préservées par chaque itération ; et qu'à la fin, l'invariant implique la proposition (égalité de la taille moyenne et de l'entropie).

5. En applicant un raisonnement similaire au point (4), vous pouvez généraliser (3) et montrer que si deux caractères ont des probabilités qui ne sont pas des puissances négatives de 2, alors l'algorithme de Huffman construit un arbre dont la longueur moyenne est strictement supérieure à l'entropie.

Exercice 5 [Preuve du théorème de Shannon] *Exercice facultatif, à faire en privé*

(a) Arbre 1 (b) Arbre 2

FIGURE 1: Arbres avec probabilité

Le but de cet exercice est de prouver le premier théorème de Shannon. La présentation est inspirée de J. L. Massey : *Applied Digital Information Theory I*[1], pp. 35 ff.

Nous donnons d'abord la fonction *cpf* (code / mots aboutissant à des feuilles, et probabilité) :

```
cpf(L(p)) = { ([], p) }
cpf(N(p, g, d)) = 0 · cpf(g) ∪ 1 · cpf(d)
```

La fonction prend en argument un arbre avec probabilités et renvoie un ensemble des couples (mot, probabilité) des chemins vers des feuilles, et les probabilités associées. Ici, $b \cdot E$ préfixe chaque code de l'ensemble E avec le caractère binaire b, par exemple $1 \cdot \{(0, 0.2), (1, 0.5)\} = \{(10, 0.2), (11, 0.5)\}$.

1. D'abord quelques lemmes préliminaires. Nous utiliserons souvent une propriété de distributivité : Comment est-ce qu'on peut réécrire $\sum_{(m,p)\in(E_1 \uplus E_2)} p$ en fonction de $\sum_{(m,p)\in E_1} p$ et $\sum_{(m,p)\in E_2} p$? Nous considérons ici des multi-ensembles E_1, E_2, c'est à dire, des structures qui se comportent comme des ensembles finis (surtout : l'ordre des éléments n'importe pas) mais qui peuvent contenir un élément plusieurs fois. Ici, $\uplus$ est l'union de multi-ensemble qui préserve le nombre d'occurrences des éléments, par exemple $\{1, 2, 2, 4\} \uplus \{2, 3, 4\} = \{1, 2, 2, 2, 3, 4, 4\}$.
2. Définissez la fonction $prob(a)$ qui renvoie la probabilité attachée à la racine d'un arbre avec probabilités a. *Note :* La fonction (très simple) se définit par distinction de cas (comme *cpf*), mais elle n'est pas récursive.
3. Nous montrons une première propriété : $prob(a) = \sum_{(m,p)\in cpf(a)} p$.
   Pour la preuve, on a besoin d'un principe d'**induction sur la structure d'un arbre** : Soit $\mathcal{P}$ un prédicat sur des arbres, alors la preuve de $\forall a.\mathcal{P}(a)$ se réduit à prouver :
   — *cas de base :* la propriété est satisfaite pour toute feuille : $\mathcal{P}(L(p))$
   — *cas d'hérédité :* si la propriété est satisfaite pour un sous-arbre gauche g et un sous-arbre droit d, alors elle est satisfaite pour un arbre composé : $\mathcal{P}(g) \wedge \mathcal{P}(d) \longrightarrow \mathcal{P}(N(p, g, d))$
   Pour la propriété à prouver ici, il faut donc montrer :
   — $prob(L(p)) = \sum_{(m,p)\in cpf(L(p))} p$
   — Si $prob(g) = \sum_{(m,p)\in cpf(g)} p$ et $prob(d) = \sum_{(m,p)\in cpf(d)} p$,
   alors $prob(N(p, g, d)) = \sum_{(m,p)\in cpf(N(p,g,d))} p$.
   Effectuez ces deux sous-preuves à l'aide des définitions des fonctions *prob* et *cpf* et de la propriété de distributivité du point (1).

1. http://www.isiweb.ee.ethz.ch/archive/massey_scr/adit1.pdf

4. Définissez la fonction *sai* qui calcule le multi-ensemble des sous-arbres intérieurs d'un arbre, c'est-à-dire, le multi-ensemble des sous-arbres qui ne sont pas des feuilles. Par exemple, dans le cas de la figure 1a, nous obtenons l'ensemble avec les deux arbres avec racine 0.7 et 1. En général, un arbre peut apparaître plusieurs fois comme sous-arbre, d'où l'importance d'utiliser des multi-ensembles.
5. Démontrez que $lnm(cpf(a)) = \sum_{s \in sai(a)} prob(s)$. Quelques remarques :
   — La preuve se fait par induction sur la structure de l'arbre a.
   — Nous rappelons la définition de longueur moyenne d'un ensemble (mot $\times$ probabilté) : $lnm(E) = \sum_{(m,p) \in E} |m| * p$
   — Pour lnm, vous aurez besoin d'un lemme de distributivité pour calculer $lnm(E_1 \cup E_2)$. Inspirez-vous du point (1).
   — En plus, on a besoin d'un lemme de distributivité pour $lnm(b \cdot E)$ pour l'opération de préfixage $(\cdot)$ défini au début de l'exercice (facile si on voit comment $(b \cdot E)$ modifie la longueur des mots qui apparaissent dans E).
6. On définit l'entropie de branchement H_b d'un arbre composé par :
   $H_b(N(p,g,d)) = -(\frac{prob(g)}{p} * \log_2(\frac{prob(g)}{p}) + \frac{prob(d)}{p} * \log_2(\frac{prob(d)}{p}))$
   Cette définition n'est pas récursive et calcule l'entropie localement, en ne considérant que les sous-arbres immédiats.
   Il est facile de vérifier $p * H_b(N(p,g,d)) = -(prob(g) * \log_2(prob(g)) + prob(d) * \log_2(prob(d))) + p * \log_2(p)$, en prenant en compte la propriété charactéristique d'un arbre avec probabilités. (*Astuce :* utilisez aussi la distributivité du logarithme sur la division.)
7. Démontrez $H(a) = -prob(a) * \log_2(prob(a)) + \sum_{s \in sai(a)} prob(s) * H_b(s)$ par induction sur la structure d'un arbre. Vous aurez besoin du lemme de (6) pour les noeuds composés.
8. Il est maintenant facile à voir que $H(a) = \sum_{s \in sai(a)} prob(s) * H_b(s)$ pour des arbres *complets* avec probabilités. Nous montrerons dans la suite (voir (10)) que toujours $H_b(s) \leq 1$. Nous obtenons donc l'estimation $H(a) \leq \sum_{s \in sai(a)} prob(s)$
9. Combinez les résultats précédents pour obtenir le Théorème de Shannon : $H(a) \leq lnm(cpf(a))$.
10. Montrons enfin $H_b(s) \leq 1$ pour tout arbre intérieur s. Montrez que, dans le cas d'un arbre s avec probabilités, la question se réduit à montrer que la fonction $h(x) =_{def} -x \log_2(x) - (1-x) \log_2(1-x)$ est ≤ 1 pour tout $x \in [0 \ldots 1]$. Si vous ne vous contentez pas avec la courbe de cette fonction (voir transparents), vous pouvez aussi calculer le maximum m de $h(x)$ de manière analytique (par la dérivée de h) et vérifier que $h(m) = 1$. Cette valeur maximale correspond à quelle situation dans un arbre avec probabilités ? Comparez les arbres de la figure 1.