Théorie de l'Information (3)

Philippe Duchon

ENSEIRB

2007-08

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$
- $\mathcal{C}_2 = \{00, 01, 001\}$
- $\mathcal{C}_3 = \{0, 01, 10\}$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$
- $\mathcal{C}_3 = \{0, 01, 10\}$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$ déchiffrable
- $\mathcal{C}_3 = \{0, 01, 10\}$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$ déchiffrable
- $\mathcal{C}_3 = \{0, 01, 10\}$ ambigu : $C(ac) = 010 = C(ba)$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$ déchiffrable
- $\mathcal{C}_3 = \{0, 01, 10\}$ ambigu : $C(ac) = 010 = C(ba)$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$ déchiffrable : préfixe
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$ déchiffrable
- $\mathcal{C}_3 = \{0, 01, 10\}$ ambigu : $C(ac) = 010 = C(ba)$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$ déchiffrable : préfixe
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$ déchiffrable : longueur fixe, donc préfixe
- $\mathcal{C}_6 = \{0, 01, 11\}$

Des exemples !

Pour chaque code, donné par l'ensemble des mots de code des symboles, dire s'il est ou non uniquement déchiffrable. S'il l'est, donner l'algorithme de décodage ; sinon, donner deux mots ayant le même codage.

- $\mathcal{C}_1 = \{0, 11, 101\}$ déchiffrable
- $\mathcal{C}_2 = \{00, 01, 001\}$ déchiffrable
- $\mathcal{C}_3 = \{0, 01, 10\}$ ambigu : $C(ac) = 010 = C(ba)$
- $\mathcal{C}_4 = \{000, 001, 01, 1\}$ déchiffrable : préfixe
- $\mathcal{C}_5 = \{000100, 100101, 010101, 111000\}$ déchiffrable : longueur fixe, donc préfixe
- $\mathcal{C}_6 = \{0, 01, 11\}$ déchiffrable, mais délai non borné

D'autres propriétés combinatoires

- Les propriétés utiles des codes qui sont de nature combinatoire ne s'arrêtent pas à la déchiffrabilité. Même si un code est non ambigu, il peut être difficile de prendre une séquence codée "en marche" : c'est le problème de la synchronisation, ou du formatage.

D'autres propriétés combinatoires

- Les propriétés utiles des codes qui sont de nature combinatoire ne s'arrêtent pas à la déchiffrabilité. Même si un code est non ambigu, il peut être difficile de prendre une séquence codée "en marche" : c'est le problème de la synchronisation, ou du formatage.
- Autre problème potentiel, le "délai" : nombre de symboles de code qu'on peut avoir besoin d'examiner au-delà de ceux qui codent une lettre, avant de pouvoir décoder cette lettre.

D'autres propriétés combinatoires

- Les propriétés utiles des codes qui sont de nature combinatoire ne s'arrêtent pas à la déchiffrabilité. Même si un code est non ambigu, il peut être difficile de prendre une séquence codée "en marche" : c'est le problème de la synchronisation, ou du formatage.
- Autre problème potentiel, le "délai" : nombre de symboles de code qu'on peut avoir besoin d'examiner au-delà de ceux qui codent une lettre, avant de pouvoir décoder cette lettre.
- Exemple de $\mathcal{C}_6 = \{0, 01, 11\}$: déchiffrable, mais pour décoder $011\cdots1$ il faut connaître la **parité** du nombre de 1 consécutifs avant d'être capable de décoder la première lettre (délai non borné).

Codes préfixes

Un mot m est un **préfixe** d'un mot m', s'il existe un mot w tel que $m' = m.w$.

Définition

*Un code $C = \{M_1, \ldots, M_k\}$ est dit **préfixe** s'il n'existe pas deux mots de code M_i, M_j (autres que $i = j$) tels que M_i soit un préfixe de M_j.*

Théorème

Tout code préfixe est uniquement déchiffrable.

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu
- Soit v le **plus long préfixe commun** de u et u' : $u = v.s_i.u_1$, $u' = v.s_j.u_2$ avec $i \neq j$

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu
- Soit v le **plus long préfixe commun** de u et u' : $u = v.s_i.u_1$, $u' = v.s_j.u_2$ avec $i \neq j$
- $w = C(v).C(s_i.u_1) = C(v).C(s_j.u_2)$, donc $C(s_i.u_1) = C(s_j.u_2)$

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu
- Soit v le **plus long préfixe commun** de u et u' : $u = v.s_i.u_1$, $u' = v.s_j.u_2$ avec $i \neq j$
- $w = C(v).C(s_i.u_1) = C(v).C(s_j.u_2)$, donc $C(s_i.u_1) = C(s_j.u_2)$
- 2 cas sont possibles :

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu
- Soit v le **plus long préfixe commun** de u et u' : $u = v.s_i.u_1$, $u' = v.s_j.u_2$ avec $i \neq j$
- $w = C(v).C(s_i.u_1) = C(v).C(s_j.u_2)$, donc $C(s_i.u_1) = C(s_j.u_2)$
- 2 cas sont possibles :
  - $\ell_i \leq \ell_j$: alors $C(s_i)$ est un préfixe de $C(s_j)$ (comme préfixe de longueur ℓ_i du mot $C(s_j.u_2)$), le code n'est pas préfixe ;

Preuve du théorème des codes préfixes

- Par l'absurde : on suppose que le code est ambigu.
- Soit $w = C(u) = C(u')$ un mot de code ambigu
- Soit v le **plus long préfixe commun** de u et u' : $u = v.s_i.u_1$, $u' = v.s_j.u_2$ avec $i \neq j$
- $w = C(v).C(s_i.u_1) = C(v).C(s_j.u_2)$, donc $C(s_i.u_1) = C(s_j.u_2)$
- 2 cas sont possibles :
  - $\ell_i \leq \ell_j$: alors $C(s_i)$ est un préfixe de $C(s_j)$ (comme préfixe de longueur ℓ_i du mot $C(s_j.u_2)$), le code n'est pas préfixe ;
  - $\ell_j < \ell_i$: inversement, $C(s_j)$ est un préfixe (strict) de $C(s_i)$, et le code n'est encore pas préfixe

Décodage d'un code préfixe

- **Remarque :** correspondance naturelle entre les mots de A^* et les nœuds d'un *arbre* k-aire infini :

Décodage d'un code préfixe

- **Remarque :** correspondance naturelle entre les mots de A^* et les nœuds d'un *arbre* k-aire infini :
  - la racine correspond au mot vide ϵ

Décodage d'un code préfixe

- **Remarque :** correspondance naturelle entre les mots de A^* et les nœuds d'un *arbre* k-aire infini :
  - la racine correspond au mot vide ϵ
  - les k fils d'un nœud w sont $w.s_1, w.s_2, \ldots, w.s_k$

Décodage d'un code préfixe

- **Remarque :** correspondance naturelle entre les mots de A^* et les nœuds d'un *arbre* k-aire infini :
  - la racine correspond au mot vide ϵ
  - les k fils d'un nœud w sont $w.s_1, w.s_2, \ldots, w.s_k$
  - les **préfixes** d'un mot correspondent exactement aux **ancêtres** de son nœud

Décodage d'un code préfixe

- **Remarque :** correspondance naturelle entre les mots de A^* et les nœuds d'un *arbre* k-aire infini :
  - la racine correspond au mot vide ϵ
  - les k fils d'un nœud w sont $w.s_1, w.s_2, \ldots, w.s_k$
  - les **préfixes** d'un mot correspondent exactement aux **ancêtres** de son nœud
- En règle générale, on peut étiqueter chaque nœud de l'arbre par le(s) mot(s) qu'il code ; le code est déchiffrable si et seulement si aucun nœud ne reçoit plus d'une étiquette

Décodage d'un code préfixe (2)

- On part de l'arbre infini T_k

Décodage d'un code préfixe (2)

- On part de l'arbre infini T_k
- On "marque" les nœuds correspondant aux mots du code (codage des symboles)

Décodage d'un code préfixe (2)

- On part de l'arbre infini T_k
- On "marque" les nœuds correspondant aux mots du code (codage des symboles)
- **Remarque :** dire que le code est préfixe revient à dire qu'aucun nœud marqué n'a d'ancêtre marqué

Décodage d'un code préfixe (2)

- On part de l'arbre infini T_k
- On "marque" les nœuds correspondant aux mots du code (codage des symboles)
- **Remarque :** dire que le code est préfixe revient à dire qu'aucun nœud marqué n'a d'ancêtre marqué
- On élimine de l'arbre tous les nœuds qui ne sont ni des nœuds marqués, ni des préfixes (ancêtres) de nœuds marqués (les nœuds marqués deviennent les feuilles)

Décodage d'un code préfixe (2)

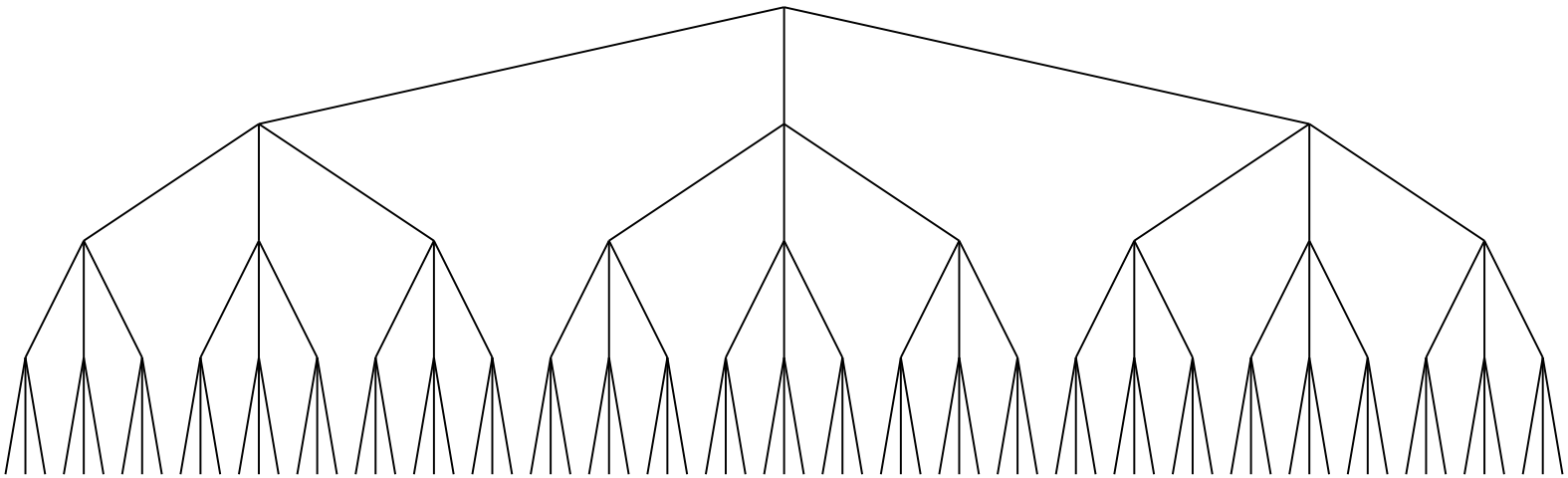
- On part de l'arbre infini T_k
- On "marque" les nœuds correspondant aux mots du code (codage des symboles)
- **Remarque :** dire que le code est préfixe revient à dire qu'aucun nœud marqué n'a d'ancêtre marqué
- On élimine de l'arbre tous les nœuds qui ne sont ni des nœuds marqués, ni des préfixes (ancêtres) de nœuds marqués (les nœuds marqués deviennent les feuilles)
- On a (presque !) un automate déterministe qui reconnaît les mots codants !

Décodage d'un code préfixe (2)

- On part de l'arbre infini T_k
- On "marque" les nœuds correspondant aux mots du code (codage des symboles)
- **Remarque :** dire que le code est préfixe revient à dire qu'aucun nœud marqué n'a d'ancêtre marqué
- On élimine de l'arbre tous les nœuds qui ne sont ni des nœuds marqués, ni des préfixes (ancêtres) de nœuds marqués (les nœuds marqués deviennent les feuilles)
- On a (presque !) un automate déterministe qui reconnaît les mots codants !
- L'arbre minimal contenant **tous** les mots codants serait obtnu en greffant, à chaque feuille, une copie de l'arbre (et en itérant à l'infini)

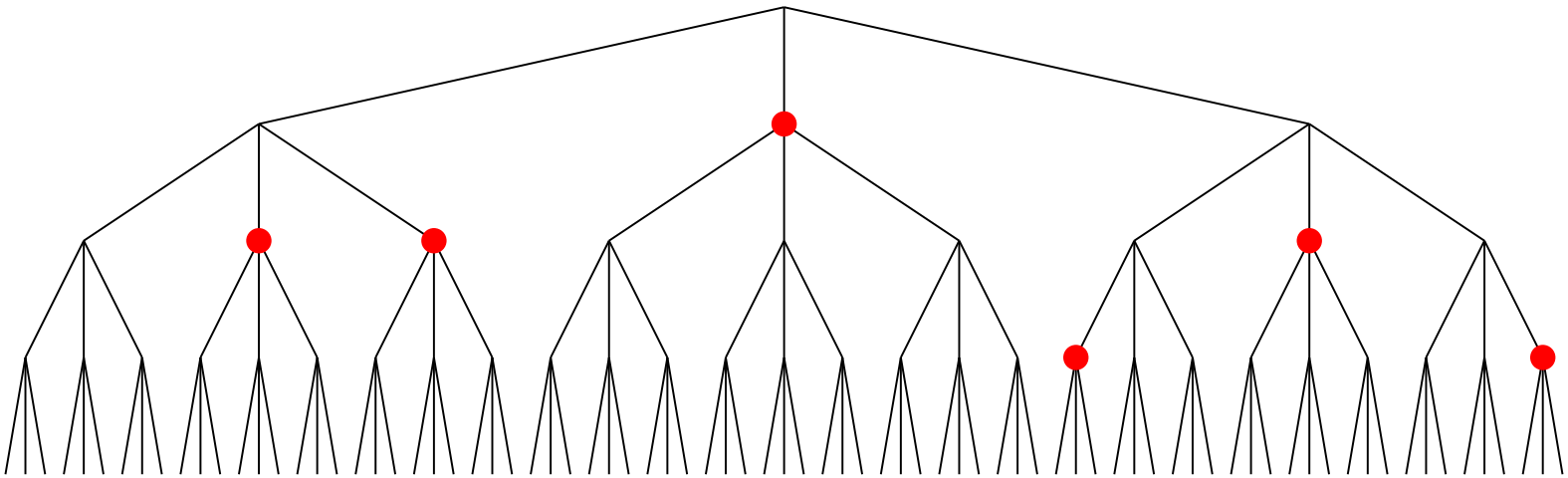
Un exemple de décodage

$$\mathcal{C} = \{01, 02, 1, 200, 21, 222\}$$



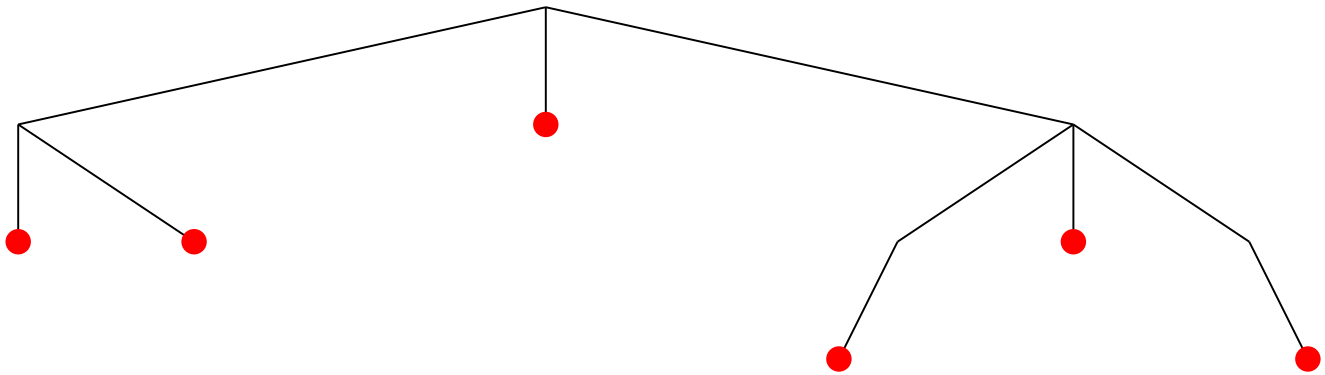
Un exemple de décodage

$$\mathcal{C} = \{01, 02, 1, 200, 21, 222\}$$



Un exemple de décodage

$$\mathcal{C} = \{01, 02, 1, 200, 21, 222\}$$



Condition de Kraft

- k (fixé) désigne la taille de l'alphabet de codage (souvent, $k = 2$), N la taille de l'alphabet de source
- Soit $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_N\}$ le **multi**-ensemble des longueurs des mots de code
- On appelle **fonction de Kraft** du code, la quantité

$$K(\mathcal{L}) = \sum_{i=1}^{N} k^{-\ell_i}$$

- Par extension, si C est un ensemble de mots : $K(C) = \sum_{w \in C} k^{-\ell(w)}$
- On dit qu'un code satisfait la **condition de Kraft** si $K(\mathcal{L}) \leq 1$

Théorème de Kraft

Il existe un code **préfixe** dont le multi-ensemble des longueurs est $\mathcal{L}$, **si et seulement si** $\mathcal{L}$ satisfait la condition de Kraft.

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$
  - si n est un nœud interne, $p_T(n)$ est la somme des poids des fils de n

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$
  - si n est un nœud interne, $p_T(n)$ est la somme des poids des fils de n
  - le poids $p(T)$ est défini comme le poids de la racine

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$
  - si n est un nœud interne, $p_T(n)$ est la somme des poids des fils de n
  - le poids $p(T)$ est défini comme le poids de la racine
- Si T est l'arbre des mots de code d'un code préfixe C, $p(T) = K(C)$

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$
  - si n est un nœud interne, $p_T(n)$ est la somme des poids des fils de n
  - le poids $p(T)$ est défini comme le poids de la racine
- Si T est l'arbre des mots de code d'un code préfixe C, $p(T) = K(C)$
- Si un arbre T est **inclus** dans un autre arbre T', alors pour tout nœud n de T, on a $p_T(n) \leq p_{T'}(n)$; en particulier, $p(T) \leq p(T')$

Preuve (1) : condition nécessaire

Lemme

Tout code préfixe satisfait la condition de Kraft.

- On définit, pour chaque nœud d'un **arbre k-aire fini** T, un poids :
  - si n est une feuille de profondeur i, $p_T(n) = k^{-i}$
  - si n est un nœud interne, $p_T(n)$ est la somme des poids des fils de n
  - le poids $p(T)$ est défini comme le poids de la racine
- Si T est l'arbre des mots de code d'un code préfixe C, $p(T) = K(C)$
- Si un arbre T est **inclus** dans un autre arbre T', alors pour tout nœud n de T, on a $p_T(n) \leq p_{T'}(n)$; en particulier, $p(T) \leq p(T')$
- Si T est l'arbre k-aire complet à ℓ niveaux, $p(T) = 1$ (il y a k^ℓ feuilles de poids $k^{-\ell}$ chacune)

Preuve (2) : condition suffisante

- Étant donné le multi-ensemble de longueurs $\mathcal{L}$, on construit un code préfixe
- On choisit les mots de code un par un, en s'assurant de le faire en conservant un ensemble préfixe
- Une condition permet d'être assuré de "ne pas être coincé" : *choisir les mots de code par longueurs croissantes*

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe d'assortiment de longueurs $\mathcal{L}$, il existe un mot de longueur ℓ' dont aucun préfixe n'est un mot de code.

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

- Il y a $k^{\ell'}$ mots de longueur ℓ'

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

- Il y a $k^{\ell'}$ mots de longueur ℓ'
- Un mot de code (de longueur $\ell \leq \ell'$) a $k^{\ell'-\ell}$ suffixes de longueur ℓ'

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

- Il y a $k^{\ell'}$ mots de longueur ℓ'
- Un mot de code (de longueur $\ell \leq \ell'$) a $k^{\ell'-\ell}$ suffixes de longueur ℓ'
- donc le nombre de mots de longueur ℓ' qui ont un mot de code pour préfixe, est au plus de

$$\sum_{i=1}^{N} k^{\ell'-\ell_i} = k^{\ell'} \sum_{i=1}^{N} k^{-\ell_i} = k^{\ell'} K(\mathcal{L}) \leq k^{\ell'} - 1$$

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

- Il y a $k^{\ell'}$ mots de longueur ℓ'
- Un mot de code (de longueur $\ell \leq \ell'$) a $k^{\ell'-\ell}$ suffixes de longueur ℓ'
- donc le nombre de mots de longueur ℓ' qui ont un mot de code pour préfixe, est au plus de $k^{\ell'} - 1$

Preuve du lemme

Lemme

Soit $\mathcal{L} = (\ell_1 \leq \ell_2 \leq \cdots \leq \ell_N)$ tel que $K(\mathcal{L}) < 1$, et ℓ' tel que $K(\mathcal{L}) \leq 1 - k^{-\ell'}$ (i.e., $\mathcal{L} \cup \{\ell'\}$ satisfait la condition de Kraft). Alors pour tout code préfixe C, d'assortiment de longueurs $\mathcal{L}$, il existe un mot w de longueur ℓ' dont aucun préfixe n'est un mot de code (i.e., $C \cup \{w\}$ est un code préfixe).

- Il y a $k^{\ell'}$ mots de longueur ℓ'
- Un mot de code (de longueur $\ell \leq \ell'$) a $k^{\ell'-\ell}$ suffixes de longueur ℓ'
- donc le nombre de mots de longueur ℓ' qui ont un mot de code pour préfixe, est au plus de $k^{\ell'} - 1$
- Par conséquent, il en existe au moins un qui n'a pas de tel préfixe, et on peut l'ajouter au code en conservant la condition préfixe.

Théorème de MacMillan

- Le théorème de Kraft nous *caractérise* les multi-ensembles de longueurs qui sont les longueurs de codes préfixes.

Théorème de MacMillan

- Le théorème de Kraft nous *caractérise* les multi-ensembles de longueurs qui sont les longueurs de codes préfixes.
- Le théorème de MacMillan nous dit que ce sont aussi *exactement* les multi-ensembles de longueurs des codes uniquement déchiffrables.

Théorème de MacMillan

- Le théorème de Kraft nous *caractérise* les multi-ensembles de longueurs qui sont les longueurs de codes préfixes.
- Le théorème de MacMillan nous dit que ce sont aussi *exactement* les multi-ensembles de longueurs des codes uniquement déchiffrables.

Théorème de MacMillan

Tout code uniquement déchiffrable satisfait la condition de Kraft.

Preuve du théorème de MacMillan

- **Remarque :** si C est déchiffrable, alors pour tout $r \geq 1$, C^r (ensemble de tous les mots que l'on peut former en concaténant exactement r mots de C) est déchiffrable, et contient exactement N^r mots.

Preuve du théorème de MacMillan

- **Remarque :** si C est déchiffrable, alors pour tout $r \geq 1$, C^r (ensemble de tous les mots que l'on peut former en concaténant exactement r mots de C) est déchiffrable, et contient exactement N^r mots.
- On vérifie que si A et B sont deux ensembles de mots tels que tous les mots de $A.B$ sont distincts, alors $K(A.B) = K(A).K(B)$; on en déduit que $K(C^r) = (K(C))^r$, par récurrence sur r.

Preuve du théorème de MacMillan

- **Remarque :** si C est déchiffrable, alors pour tout $r \geq 1$, C^r (ensemble de tous les mots que l'on peut former en concaténant exactement r mots de C) est déchiffrable, et contient exactement N^r mots.
- On vérifie que si A et B sont deux ensembles de mots tels que tous les mots de $A.B$ sont distincts, alors $K(A.B) = K(A).K(B)$; on en déduit que $K(C^r) = (K(C))^r$, par récurrence sur r.
- Supposons que l'on ait $K(C) > 1$: alors, si $L = \max_i(\ell_i)$, on prend r tel que $K(C)^r > rL$

Preuve du théorème de MacMillan

- **Remarque :** si C est déchiffrable, alors pour tout $r \geq 1$, C^r (ensemble de tous les mots que l'on peut former en concaténant exactement r mots de C) est déchiffrable, et contient exactement N^r mots.
- On vérifie que si A et B sont deux ensembles de mots tels que tous les mots de $A.B$ sont distincts, alors $K(A.B) = K(A).K(B)$; on en déduit que $K(C^r) = (K(C))^r$, par récurrence sur r.
- Supposons que l'on ait $K(C) > 1$: alors, si $L = \max_i(\ell_i)$, on prend r tel que $K(C)^r > rL$
- Tous les mots de C^r sont de longueur au plus rL : donc il existe forcément une longueur $\ell \leq rL$ telle que la somme des poids des mots de C^r ayant exactement longueur ℓ, soit > 1

Preuve du théorème de MacMillan

- **Remarque :** si C est déchiffrable, alors pour tout $r \geq 1$, C^r (ensemble de tous les mots que l'on peut former en concaténant exactement r mots de C) est déchiffrable, et contient exactement N^r mots.
- On vérifie que si A et B sont deux ensembles de mots tels que tous les mots de $A.B$ sont distincts, alors $K(A.B) = K(A).K(B)$; on en déduit que $K(C^r) = (K(C))^r$, par récurrence sur r.
- Supposons que l'on ait $K(C) > 1$: alors, si $L = \max_i(\ell_i)$, on prend r tel que $K(C)^r > rL$
- Tous les mots de C^r sont de longueur au plus rL : donc il existe forcément une longueur $\ell \leq rL$ telle que la somme des poids des mots de C^r ayant exactement longueur ℓ, soit > 1
- Cela implique qu'il y ait plus de k^ℓ tels mots : contradiction (ils ne peuvent pas être tous distincts).

Récapitulons

- On a défini une propriété combinatoire souhaitable des codes : la **déchiffrabilité**

Récapitulons

- On a défini une propriété combinatoire souhaitable des codes : la **déchiffrabilité**
- On a prouvé qu'une condition suffisante pour qu'un code soit déchiffrable est qu'il soit **préfixe**

Récapitulons

- On a défini une propriété combinatoire souhaitable des codes : la **déchiffrabilité**
- On a prouvé qu'une condition suffisante pour qu'un code soit déchiffrable est qu'il soit **préfixe**
- De plus, l'algorithme de décodage d'un code préfixe est **simple**

Récapitulons

- On a défini une propriété combinatoire souhaitable des codes : la **déchiffrabilité**
- On a prouvé qu'une condition suffisante pour qu'un code soit déchiffrable est qu'il soit **préfixe**
- De plus, l'algorithme de décodage d'un code préfixe est **simple**
- Par ailleurs, on a **caractérisé** les assortiments de longueurs des codes préfixes

Récapitulons

- On a défini une propriété combinatoire souhaitable des codes : la **déchiffrabilité**
- On a prouvé qu'une condition suffisante pour qu'un code soit déchiffrable est qu'il soit **préfixe**
- De plus, l'algorithme de décodage d'un code préfixe est **simple**
- Par ailleurs, on a **caractérisé** les assortiments de longueurs des codes préfixes
- Et on a montré qu'il n'**existe pas** d'assortiments de longueurs qui soient ceux de codes déchiffrables, mais pas de codes préfixes.