Rappel : on a un alphabet fini $A = \{a_1 \dots a_K\}$
et une proba $p_1 \dots p_K$ proba sur A
≡ fréquence des lettres.

- on veut coder $X_1 X_2 \dots$ où X_i indep de (p).
$X_i \in A$

de manière à ce que $\ell(X_1 \dots X_n)$ soit de taille la + petite possible.
$\ell : \bigsqcup_{n \geq 0} \{0;1\}^n$

- on veut minimiser

$$R_n := E\left[\frac{|\ell(X_1 \dots X_n)|}{n}\right]$$
↖ longueur

et $R = \lim_n R_n$.
codage bon si R petit.

- on a introduit $H = \sum_{k=1}^{K} -p_k \log_2 p_k$

$$0 \leq H \leq \log_2 K$$
↑ ↑
nulle de Dirac / uniforme

- Codage naïf (ne dépend pas de p) on injecte :
$A = \{a_1 \dots a_K\} \hookrightarrow \{0;1\}^{\lceil \log_2 K \rceil}$
alors $R_n = n R_1 = n \lceil \log_2 K \rceil$. $R \approx \lceil \log_2 K \rceil$

- Codage "entropique" : on a identifié les suites $x_1 \dots x_n$ typique au nombre de $\approx 2^{nH}$
celle ci on les codes par des suites dans $\{0;1\}^{nH}$
$\rightsquigarrow R_n \approx n(H + C\varepsilon)$
alors $R \approx H + C\varepsilon$. ▷ meilleur

- Peut on faire mieux ?

on a toujours $R_n = \sum_{x_1 \cdots x_n} |\varphi(x_1 \cdots x_n)| p_{x_1 \cdots x_n}$

$$\geq \sum_{x_1 \cdots x_n \text{ typiques } \in T_\varepsilon^n} |\varphi(x_1 \cdots x_n)|$$

$p_{x_1 \cdots x_n} \geq 2^{-n(H+\varepsilon)}$ par def de T_ε^n.

$$R_n \geq 2^{-n(H+\varepsilon)} \sum_{x_1 \cdots x_n \text{ typique}} |\varphi(x_1 \cdots x_n)|$$

(il y a $|T_\varepsilon^n| \geq 2^{n(H-\varepsilon)}$.)

éléments 2 à 2 $\neq$ dans $\coprod_{m \geq 0} \{0;1\}^m$ (sinon on ne pourrait pas décoder)

on $\geq 2^{n(H-\varepsilon)}$ éléments $\neq$ dans $\coprod_{m \geq 0} \{0;1\}^m$

de quelle manière minimiser la somme des longueurs ?

mettre $\emptyset, \{0\}, \{1\}, \underbrace{\text{mots de taille 2}}, \underbrace{\text{mots de taille 3}}, \ldots$

Si on prend tous les mots de taille $\leq p$

cela donnera $\underbrace{1 + 2^1 + \cdots + 2^p}_{2^{p+1}-1}$ mots

on veut que cela fasse $\geq 2^{n(H-\varepsilon)}$

$$\boxed{p \approx n(H-\varepsilon) - 1.}$$

contraintes pour coder tous les mots typiques par des mots $\neq$.

$$\{\varphi(x_1 \cdots x_n) \mid x_1 \cdots x_n \in T_\varepsilon^n\} = \coprod_{m=0} \{0;1\}^m$$

La somme des longueurs est alors

$$\sum_{m=0}^{p} \sum_{q \in \{0;1\}^m} \text{longueurs } q = \sum_{m=0}^{p} 2^m m$$

$$= x\left(\sum_{m=0}^{p} x^m\right)' \Big|_{x=2}$$

$$= x\left(\frac{x^{p+1}-1}{x-1}\right)' \Big|_{x=2}$$

$$= x \left. \frac{(p+1)x^p(x-1) - (x^{p+1}-1)}{x-1} \right|_{x=2}$$

$$= 2\left[(p+1)2^p - 2^{p+1} + 1\right].$$

$$R_n \geq 2^{-n(H+\varepsilon)}\, 2\left((p+1)2^p - 2^{p+1} + 1\right) \qquad p \simeq n(H-\varepsilon)-1$$

$$R_n \geq 2^{-n(H+\varepsilon)}\, 2\left((n(H-\varepsilon)-1)\,2^{n(H-\varepsilon)-1}\right)(1+o(1)) \qquad n \to \infty$$

$$\boxed{R_n \geq \underbrace{2^{-n\varepsilon}}_{\text{mauvaise}}\, 2\,\boxed{n(H-\varepsilon)}}$$

$\to 0$ lorsque $n \to \infty$

On peut affiner pour voir que $\forall$ codage $\forall \varepsilon > 0$

lorsque n grand $R_n \geq n(H-\varepsilon) \Rightarrow \boxed{R \geq H-\varepsilon}$

Cas particulier : codage lettre à lettre :

cas où $\varphi(x_1 \dots x_n) = \varphi(x_1) \dots \varphi(x_n)$.

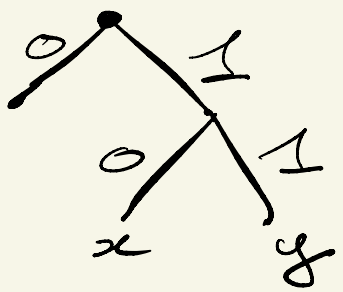
- pour éviter les ambiguïté on doit avoir

si $i \neq j$: $\varphi(x_i)$ n'est pas un préfixe de $\varphi(x_j)$

$\Leftrightarrow$ on peut placer les x_i sur les feuilles d'un arbre binaire et le chemin de 0 et de 1 qui mène à x_i est le code de x_i

$\varphi(x) = 10$
$\varphi(y) = 11$
$\varphi(z) = 0$

$\Leftrightarrow$



Codage de Huffmann : $A = \{a_1, \dots, a_K\}$

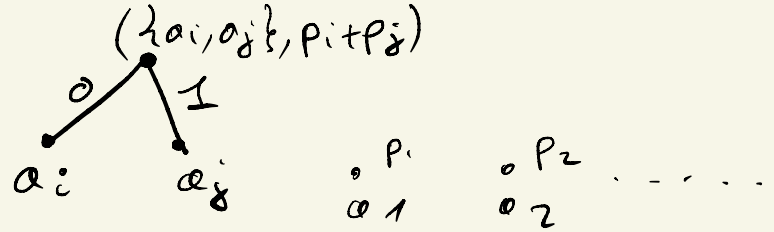
$\{p_1, \dots, p_K\}$

Codage lettre à lettre

construit récursivement (sur K).

Objectif : arbre binaire dont les feuilles sont les a_i

1) je sélectionne les 2 éléments a_i et a_j tel que p_i et p_j sont minimaux $\left(p_i = \min_{1 \le k \le K} p_k \right.$, $\left. p_j = \min_{\substack{1 \le k \le K \\ k \neq i}} p_k\right)$



3) Je construis les lettres $(\{a_i, a_j\}, a_1, a_2 \dots \cancel{a_i} \dots \cancel{a_j} \dots)$

alphabet à $K-1$ lettres $(p_i + p_j, p_1, p_2 \dots \cancel{p_i} \dots \cancel{p_j} \dots)$

Exo : $\{\Pi, P, R, U, Y, Z\}$ $\quad K = 6$.

p : $1/16$ $1/16$ $1/16$ $1/4$ $1/2$ $1/16$

1) H ?

$$H = -(p_\Pi \log_2 p_\Pi + p_P \log_2 p_P + \dots)$$

$$= -\left(4 \times \frac{1}{16} \log_2 \frac{1}{16} + \frac{1}{4} \log_2 \frac{1}{4} + \frac{1}{2} \log_2 \frac{1}{2}\right)$$

$\left(\log_2 \frac{1}{2^k} = -k\right)$

$$= 4 \times \frac{1}{16} \times 4 + \frac{1}{4} \times 2 + \frac{1}{2} \times 1$$

$$= 2$$

2) Huffmann :

① Π P R U Y Z.

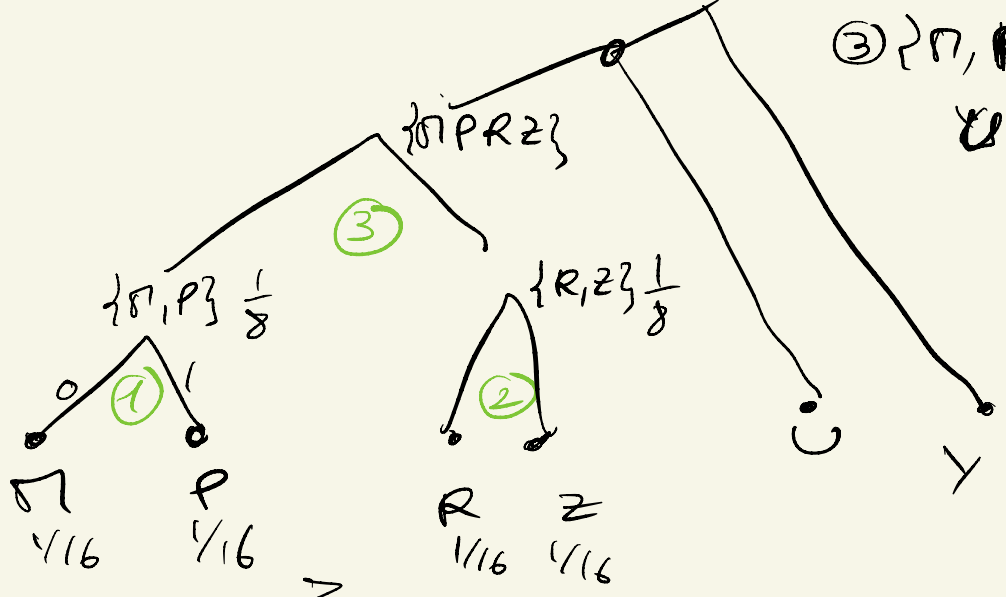
$1/4$ $1/2$.

p_Π et p_P sont minimaux.

② $\{\Pi, P\}$ $1/8$ R U Y Z

$1/16$ $1/4$ $1/2$ $1/16$

p_R p_Z minimaux

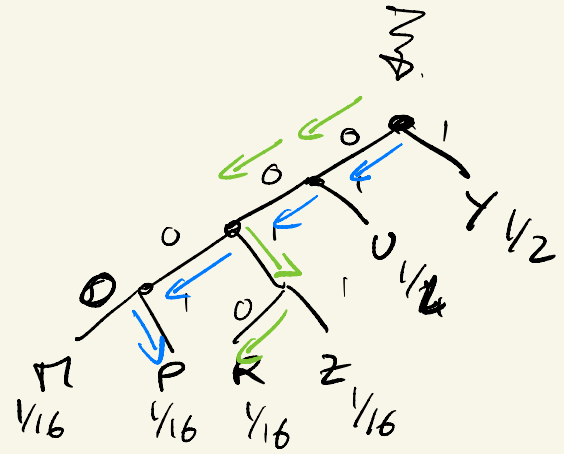


③ $\{\Pi, P\}$ $1/8$ $\{R, Z\}$ $1/8$ U $1/4$ Y $1/2$.

$\{\Pi, P\}$ et $\{R, Z\}$ minimaux

④ $\{\Pi\ P\ R\ Z\}$ $1/4$ U $1/4$ Y $1/2$

$\{\Pi\ P\ R\ Z\}$ et U sont minimaux



Le codage

$\varphi(\Pi) = 0000$ $\quad \varphi(U) = 01$

$\varphi(P) = 0001$ $\quad \varphi(Y) = 1$

$\varphi(R) = 0010$

$\varphi(Z) = 0011$

- $\varphi(Y U \Pi) = \underbrace{1}_{Y}\ \underbrace{0\ 1}_{U}\ \underbrace{0000}_{\Pi}$

- $\varphi^{-1}(0\,0\,1\,0\,0\,0\,0\,1)$? on part de la racine et on suit le chemin arriver sur une feuille on la note puis on recommence.

$\underbrace{0\ 0\ 1\ 0}_{R}\ \underbrace{0\ 0\ 0\ 1}_{P}$

$= RP$

- nbre moyen de bits par message ?

$R_n = n R_1$

et $R_1 = p_\Pi |\varphi(\Pi)| + p_P |\varphi(P)| \dots$

$= 4 \underbrace{\frac{1}{16} \times 4}_{\Pi, P, R, Z} + \underbrace{\frac{1}{4} \times 2}_{U} + \underbrace{\frac{1}{2} \times 1}_{Y} = 2 = H$

En général on ne peut pas espérer $R_1 = H$ pour des problèmes d'arrondis

Exo : Alphabet : {A, C, F, R}

P : 1/8, 1/4, 1/8, 1/2

1) Calculer H

2) Le codage de Huffman (et son arbre)

3) R = nbre moyen de bits par lettre

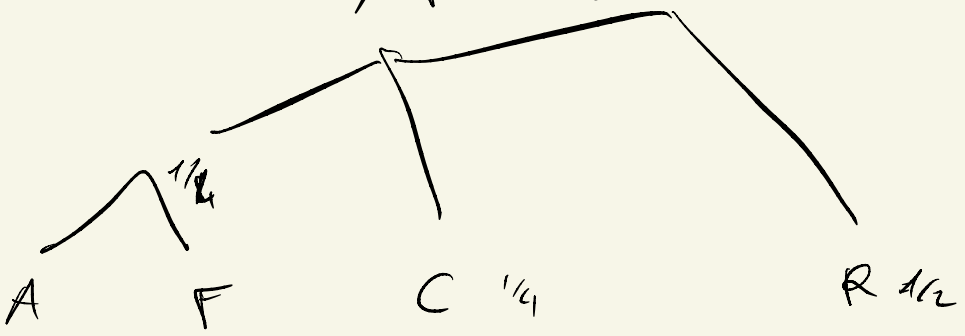
4) φ(ARC) ?

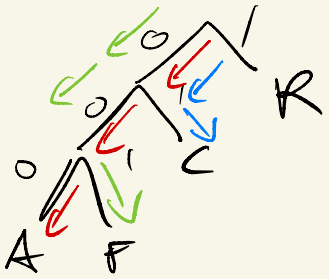
5) φ^{-1}(00100001) ?

1) $H = -\left(\frac{1}{8}\log_2\frac{1}{8} + \frac{1}{4}\log_2\frac{1}{4} + \frac{1}{8}\log_2\frac{1}{8} + \frac{1}{2}\log_2\frac{1}{2}\right)$

$= + \frac{3}{8} + \frac{2}{4} + \frac{3}{8} + \frac{1}{2} = \frac{7}{4}$

2) A et F min, puis {A, F} et C





$\varphi(R) = 1$

$\varphi(C) = 01$

$\varphi(F) = 001$

$\varphi(A) = 000$

3) $R = \frac{1}{8}(|\varphi(A)| + |\varphi(F)|) + \frac{1}{4}|\varphi(C)| + \frac{1}{2}|\varphi(R)|$

$= \frac{1}{8} 2 \times 3 + \frac{2}{4} + \frac{1}{2} = 7/4$

4) $\varphi(ARC) = \underbrace{000}_{A}\ \underbrace{1}_{R}\ \underbrace{01}_{C}$

5) $\varphi^{-1}(00\,1\,00000\,1) = FAC$

F A C

Exo: Alphabet : $\{A, B\}$ $\quad \varepsilon < \frac{1}{2}$ $\quad \varepsilon$ petit.

p $\{\varepsilon, 1-\varepsilon\}$.

1) Calculer H
2) Calculer le code de Huffmann
3) Calculer R et commenter son l'efficacité.

Exo: Alphabet infini : $\{1, 2, 3, 4, \ldots\} = \mathbb{N}^*$

$\mathbb{P}(X = i) = \frac{1}{2^i}$ $\quad \left(\sum_{i \in \mathbb{N}^*} \frac{1}{2^i} = 1\right)$.

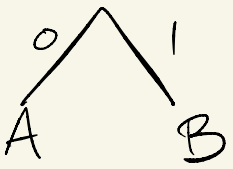
1) Calculer H
2) S'inspirer du code de Huffmann pour trouver un codage
3) Trouver pour ce codage le R_1 correspondant

Exo: $\{A, B\}$ $\quad K = 2$ $\quad \lfloor \ln_2 K \rfloor = 1$.

ε $\quad 1-\varepsilon$

1) $H = -(\varepsilon \ln \varepsilon + (1-\varepsilon)\ln(1-\varepsilon))$.

2) Arbre :



3) $R_1 = 1$

Efficacité : a t'on R proche de H ?

Non : $R_1 = 1$ $\quad H \xrightarrow[\varepsilon \to 0]{} 0$

Donc on ne peut pas espérer approcher H à une constante multiplicative près.

Indication pour l'alphabet ∞ :

S'inspirer du cas $A = \{1, 2, \dots, n\}$

$\mathbb{P}(X=i) = \frac{1}{2^i}$ si $i = 1, 2, \dots n-1$

$\mathbb{P}(X=n) = \frac{1}{2^{n-1}}$

1) $H = -\sum_{k\geq 1} p_k \log_2 p_k$

$= -\sum_{k\geq 1} \frac{1}{2^{-k}} \log_2 \frac{1}{2^{-k}} = \sum_{k\geq 1} \frac{k}{2^k}$

$= z \underbrace{\left(\sum_{k\geq 0} z^k\right)'}_{z \sum k z^{k-1}} \Big|_{z=\frac{1}{2}} = z\left(\frac{1}{1-z}\right)' \Big|_{z=\frac{1}{2}}$

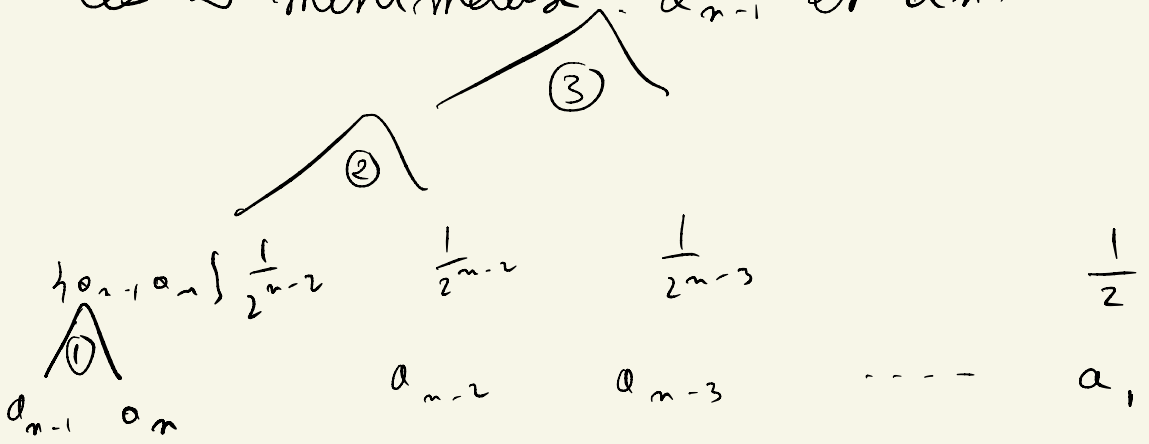
$\underbrace{\qquad}_{\sum k z^k}$

$= \frac{z}{(1-z)^2}\Big|_{z=1/2} = \frac{1/2}{(1-1/2)^2} = 2$

2) Si $A_n = \{ a_1, \dots, a_{n-1}, a_n \}$

$\frac{1}{2}, \frac{1}{4}, \frac{1}{2^{n-1}}, \frac{1}{2^{n-1}}$

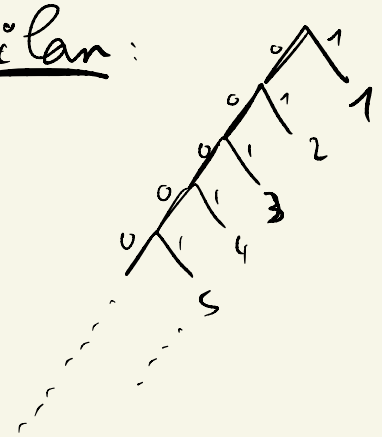
les 2 minimaux : a_{n-1} et a_n.



les 2 minimaux : $\{a_{n-1}, a_n\}$ et a_{n-2}

(Après un regroupement on est équivalent à A_{n-1})

Bilan :



$\varphi(i) = \underbrace{000\ldots0}_{i-1}1$ $\quad |\varphi(i)| = i$

$$R_1 = \sum_i p_i |\varphi(i)|$$

$$= \sum_i \frac{1}{2^i} i = 2 = H.$$

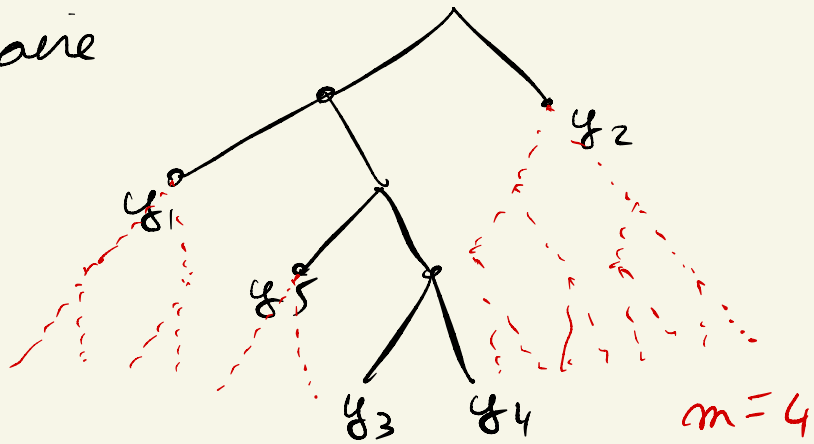
↑ comme pour H

• Peut-on atteindre $R_1 \approx H$ pour les codages lettre à lettre ?

Lemme (Kraft) si $y_1 \ldots y_k$ sont des mots $\in \{0;1\}^*$ tq y_i n'est pas le préfixe de y_j $\forall i \neq j$

si $\ell_i = |y_i|$ alors $\sum_{i=1}^{k} 2^{-\ell_i} \leq 1$

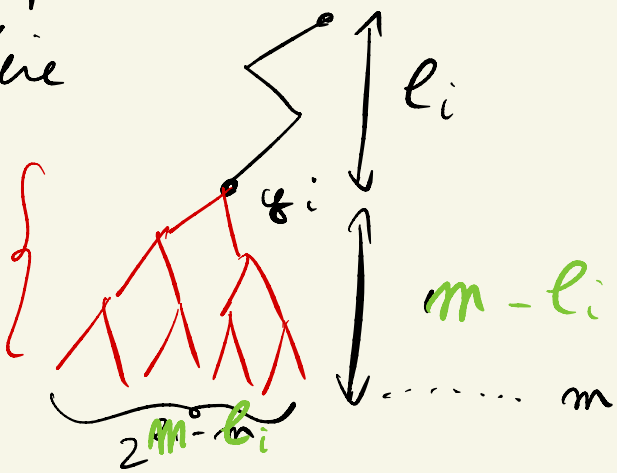
Dém : ∃ un arbre binaire dont les y_i sont les feuilles



$m = \max_{1 \leq i \leq k} \{\ell_i\}$

on complète l'arbre jusqu'à la hauteur m

c'est à dire



arbre binaire complet de hauteur $\ell - m$

y_i a $2^{\ell_i - m}$ descendants à hauteur m.

les descendants de 2 y_i ≠ sont ≠

donc $\sum_i 2^{m-\ell_i} \leq$ #total d'individus à la hauteur $m = 2^m$

donc $\boxed{\sum_i 2^{-\ell_i} \leq 1}$ •

Réciproque : Si ℓ_i entier tels que $\sum 2^{-\ell_i} \leq 1$

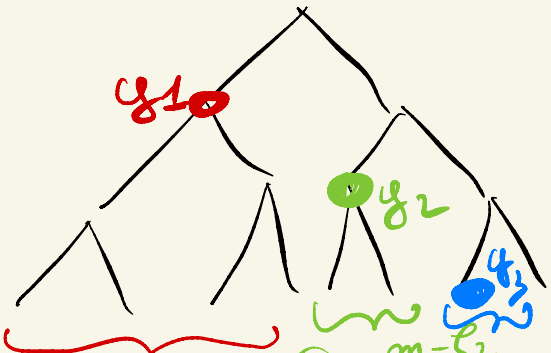
alors $\exists$ des mots $y_1 \dots y_k$ tel que $|y_i| = \ell_i$

et y_i n'est pas préfixe de y_j si $i \neq j$ $\quad y_i \in \bigcup_{m} \{0,1\}^m$

Preuve : on veut $y_1 \dots y_k$ des feuilles d'un arbre binaire y_i à hauteur ℓ_i

Supposons $\ell_1 \leq \ell_2 \dots \leq \ell_k$ ← Important. on pose $m = \max \ell_i = \ell_k$.

on regarde l'arbre binaire complet de hauteur m



← hauteur m il y a 2^m feuilles.

① $2^{m-\ell_1}$ premiers individus ce doit être les descendants de y_1 : ($2^{3-1} = 4$)

② $2^{m-\ell_2}$ suivants ce doit être les descendants de y_2

③ $2^{m-\ell_3}$ suivants doivent descendre de y_3

Ex : $\ell_1 = 1$ $\quad \ell_2 = 2$ $\quad \ell_3 = 3$ $\quad (m=3)$

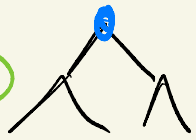
$(2^{-1} + 2^{-2} + 2^{-3} \leq 1$ ✓)

Si y_i est de longueur ℓ_i il doit avoir

$2^{m-\ell_i}$ descendant à la hauteur m.

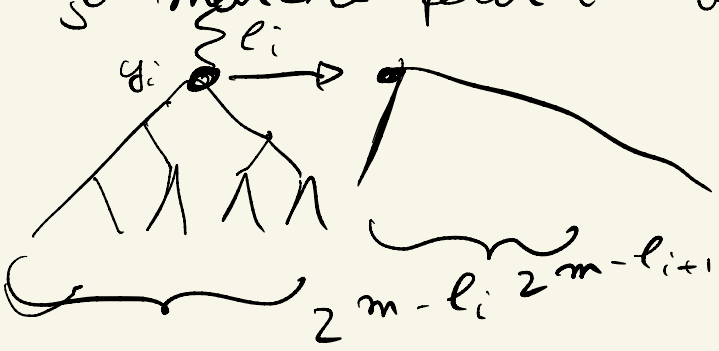
Pourquoi ça marche : pourquoi le premier ancêtre des $2^{m-\ell_i}$ individus est à hauteur ℓ_i ?

Problème :



un ancêtre de 2^{2-1} individu n'est pas de longueur 1 mais 0.

Si ça marche pour i alors comme $l_{i+1} \geq l_i$
$2^{m-l_{i+1}} \leq 2^{m-l_i}$



les $2^{m-l_{i+1}}$ suivants sont les premiers descendants des voisins de droite de y_i $\Rightarrow$ ils ont un ancêtre commun sans autres descendants

Le Problème n'arrive pas □

On cherche un codage Lethe à Lethe optimal :
on cherche $y_1 \dots y_K$ qui code $a_1 \dots a_K$
$\varphi(a_i) = y_i$ et qui minimise.

$$R_1 = \sum_{i=1}^{K} p_i |y_i| = \sum_{i=1}^{K} p_i l_i$$

Le Lemme de Kraft nous donne une CNS sur $l_1 \dots l_K$ pour que il existe des mots $y_1 \dots y_K$ ayant ces longueurs.

Donc on cherche $l_1 \dots l_K$ tel que $\sum_{i=1}^{K} 2^{-l_i} \leq 1$
et qui minimise $R_1 = \sum_{i=1}^{K} p_i l_i$.

Cherchons $x_1 \dots x_K \in \mathbb{R}^+$ tels que $\Psi(x) := \sum_{1}^{K} 2^{-x_i} \leq 1$
et $R_1 = \sum_{i=1}^{K} p_i x_i$ soit le + petit possible

Pour minimiser R_1 mieux prendre x_i petit.
Si $\Psi(x) = \sum 2^{-x_i} < 1$ on peut sans risque diminuer les x_i

$\longrightarrow$ optimal atteint en un point $\psi(x) = \sum_1^K 2^{-x_i} = 1$

But : minimiser $\sum_1^K p_i x_i$ sous la contrainte $\sum_1^K 2^{-x_i} = 1$

Méthode multiplicateur de Lagrange :

on considère $\varphi(x, \lambda) = \sum_1^K p_i x_i + \lambda(\psi(x) - 1)$

Alors l'optimum est atteint en un (x, λ) tel que

$$\partial_{x_i} \varphi = 0 \quad \text{et} \quad \partial_\lambda \varphi = 0$$

En effet, $\partial_\lambda \varphi = \psi(x) - 1 = 0$

et si h est une direction selon laquelle $\nabla \psi \cdot h = 0$ c'est à dire ψ ne bouge pas dans la direction h (donc la contrainte continue à être respecté) alors dans cette direction $\sum_1^K p_i x_i$ doit être optimal donc $\left(\nabla \sum_1^K p_i x_i\right) h = 0$

$h \in \nabla(\psi - 1)^\perp \Rightarrow h \in \left(\nabla \sum_1^K p_i x_i\right)^\perp$

$\Rightarrow -\lambda \nabla(\psi - 1) = \nabla \sum_1^K p_i x_i$

$\Rightarrow \nabla \varphi = 0 \Rightarrow \forall i \ \partial_{x_i} \varphi = 0$ et $\partial_\lambda \varphi = 0$

Analyse on cherche x, λ tq $\nabla \varphi(x, \lambda) = 0$

où $\varphi(x, \lambda) = \sum_{i=1}^K x_i p_i + \lambda\left(\sum_{i=1}^K 2^{-x_i} - 1\right)$

$\partial_{x_i} \varphi = 0 \Rightarrow p_i + \lambda(-\ln 2) 2^{-x_i} = 0 \quad (*)$

$$\sum_i : \underbrace{\sum_{i=1}^K p_i}_{1} - \ln 2 \, \lambda \underbrace{\sum_{i=1}^K 2^{-x_i}}_{1} = 0$$

Donc $\lambda = \frac{1}{\ln 2}$

$(*)$: $2^{-x_i} = p_i \Rightarrow x_i = -\log_2 p_i$

Donc :

$$\min\left\{\sum_{i=1}^{R} x_i p_i : x_i \geq 0 \text{ et } \sum_{1}^{R} 2^{-x_i} = 1\right\}$$

$$= \sum_{i=1}^{R} \underbrace{-\log_2 p_i}_{=\bar{x}_i} p_i \qquad \left(\text{on a } \sum 2^{-(-\log_2 p_i)} = 1\right)$$

Je pose $l_i = \lceil -\log_2 p_i \rceil = \lceil \bar{x}_i \rceil$.

et alors $\sum_i 2^{-l_i} = \sum 2^{-\lceil \bar{x}_i \rceil} \leq \sum 2^{-\bar{x}_i} = 1$

Donc par le Lemme de Kraft,

$\exists$ $y_1 \dots y_R$ de longueurs $l_1 \dots l_R$ avec condition de préfixe

et pour ce code ($\varphi(a_i) = y_i$)

on a $\boxed{R_1 = \sum p_i l_i \leq \sum_i p_i (\bar{x}_i + 1)}$

$$= \sum_i p_i (-\log_2 p_i) + 1 = \boxed{H + 1}$$

Et pour tout autre code (on aura $\sum 2^{-l_i} \leq 1$)

$$\boxed{R_1 = \sum p_i l_i \geq} \min\left\{\sum x_i p_i : \sum 2^{-x_i} \leq 1\right\}$$

$$= \sum \bar{x}_i p_i = H$$

Théorème : pour tout codage (lettre à lettre) $(y_1 \dots y_R)$ avec condition de préfixe :

- $\boxed{R_1 \geq H}$
- Il en existe un tel que $\boxed{R \leq H + 1}$

Exo : $A = \{E, N, S, T, Z\}$ avec $\frac{5}{18}\ \frac{2}{9}\ \frac{1}{3}\ \frac{1}{9}\ \frac{1}{18}$

1) Calculer H
2) Calculer le codage
3) Quel est le code de EST
4) on change le premier bit de EST et est ce que on obtient le code d'un autre mot ?

1) $H = -\left(\frac{5}{18}\log_2\frac{5}{18} + \frac{2}{9}\log_2\frac{2}{9} + \frac{1}{3}\log_2\frac{1}{3} + \frac{1}{9}\log_2\frac{1}{9} + \frac{1}{18}\log_2\frac{1}{18}\right)$

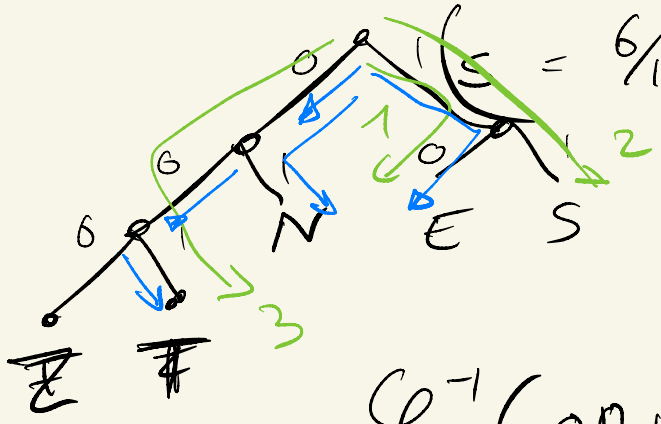
$= \dots$

2) ① T, Z $\Rightarrow$ $\frac{3}{18} = \frac{1}{6}$ ($T: 4/9$, $Z: 1/18$) $\rightarrow 7/18$

$N = 4/18$ $\rightarrow$

$E = 5/18$ $\rightarrow 11/18$

$S = 6/18$



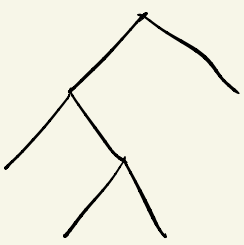
$\varphi(BST) = \underbrace{10}_{E}\ \underbrace{11}_{S}\ \underbrace{001}_{T}$

$\varphi^{-1}(00111001) \overset{\text{oui}}{=} TEN$ □

Théorème : Il n'y a pas de codage tel que $R_1 \leq R_1^{\text{Huffmann}}$.

Conséquence : $R_1^{\text{Huffmann}} \leq H+1$

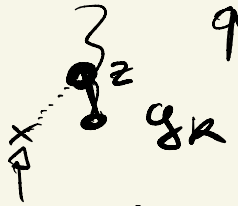
Dém : Partons d'un autre codage lettre à lettre sans préfixe. Les $g_i = \ell(a_i)$ sont les feuilles d'un arbre binaire



Regardons $\ell_1 \dots \ell_k$ les longueurs de $g_1 \dots g_k$

et supposons $\ell_1 \leq \dots \leq \ell_K$.

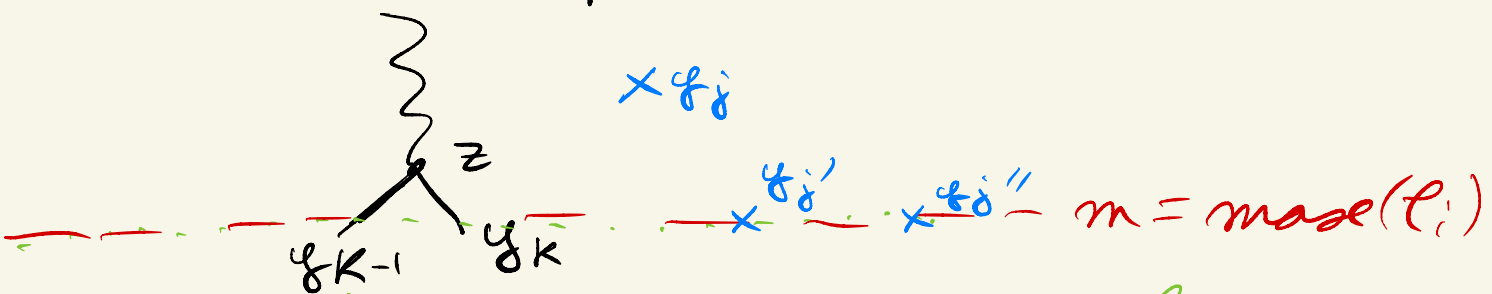
Supposons : que z le père de y_K n'ait qu'un fils dans l'arbre (son autre fils possible n'est pas un y_i)



on peut améliorer strictement le code en remplaçant y_K par z $\Rightarrow$ les codes produits seront + courts.

$\Rightarrow$ on améliore le code tant que cette situation se présente

$\Rightarrow$ ça nous ramène où y_K (code de longueur max) a pour voisin qui est l'un des y_i



Un des y_i de même longueur (on peut supposer que c'est y_{K-1})

Pour ce code $R = \sum p_i \ell_i$

y_{K-1} et y_K de longueur max $\Rightarrow$ on ne perd rien à leur associer les lettres de proba minimale. Si p_i est de proba $< p_K, p_{K-1}$

Je fais un nouveau code $\tilde{R}$ où la lettre a_i est codé par y_K.

et alors $\tilde{R} - R = (p_i m + p_K \ell_i) - (p_i \ell_i + p_K m)$

$= \underbrace{(p_i - p_K)}_{<0} \underbrace{(m - \ell_i)}_{\geq 0} \leq 0$

⟹ on y gagne à donner à y_k y_{k-1} les codes des lettres les moins probables.

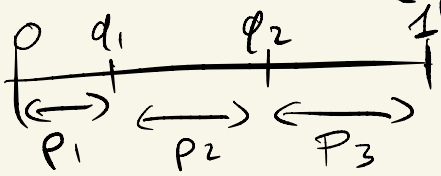
⟹ on ne perd rien à faire la première étape de Huffmann

⟹ on recommence sur le reste ▨

Exo 15 : (codage entropique) pas un codage lettre à lettre.

on considère $A = \{1; \dots; K\}$

et une proba $\{p_1, \dots, p_K\}$ $\quad \sum p_i = 1$



$q_k = p_1 + \dots + p_k \qquad q_0 = 0 \qquad q_K = 1$

$q_k - q_{k-1} = p_k$

Pour coder $x_1 \dots x_n \qquad x_i \in \{1; \dots; K\}$

on définit une suite d'intervalles

$I_0 \supseteq I_1 \supseteq I_2 \dots \supseteq I_n$

$I_k = [a_k ; b_k] \qquad a_k \uparrow \qquad b_k \downarrow$

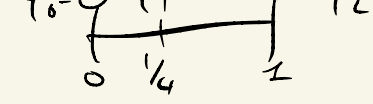
$$I_k = \underbrace{a_{k-1}}_{\text{borne gauche de } I_{k-1}} + \underbrace{(b_{k-1} - a_{k-1})}_{\text{largeur de } I_{k-1}} \times [q_{x_{k-1}} ; q_{x_k}[$$

Le code de $x_1 \dots x_n$ est $z_1 \dots z_{L_n}$ où

$$\sum_{i=1}^{L_n} \frac{z_i}{2^i} \in I_n$$ (de longueur L_n la + petite possible)

1) $A = \{0, 1\}$ $\quad p_0 = 1/4 \quad p_1 = 3/4$

coder les 3 messages 0 ; 1 ; 10.



$I_0 = [0; 1]$

$I_1 = 0 + 1\,[q_0 ; q_1[= [0 ; 1/4[$ (0 1ère étape)

$a_1 = 0$ $\quad b_1 = 1/4$ $\quad b_1 - a_1 = \frac{1}{4}$

$I_2 = 0 + \frac{1}{4}\,[q_1 ; q_2[$ (1 2ème étape)

$= \frac{1}{4}\,[\frac{1}{4} ; 1]$

$= [\frac{1}{16} ; \frac{1}{4}[$

$a_2 = \frac{1}{16}$ $\quad b_2 = 1/4$

$b_2 - a_2 = .3$