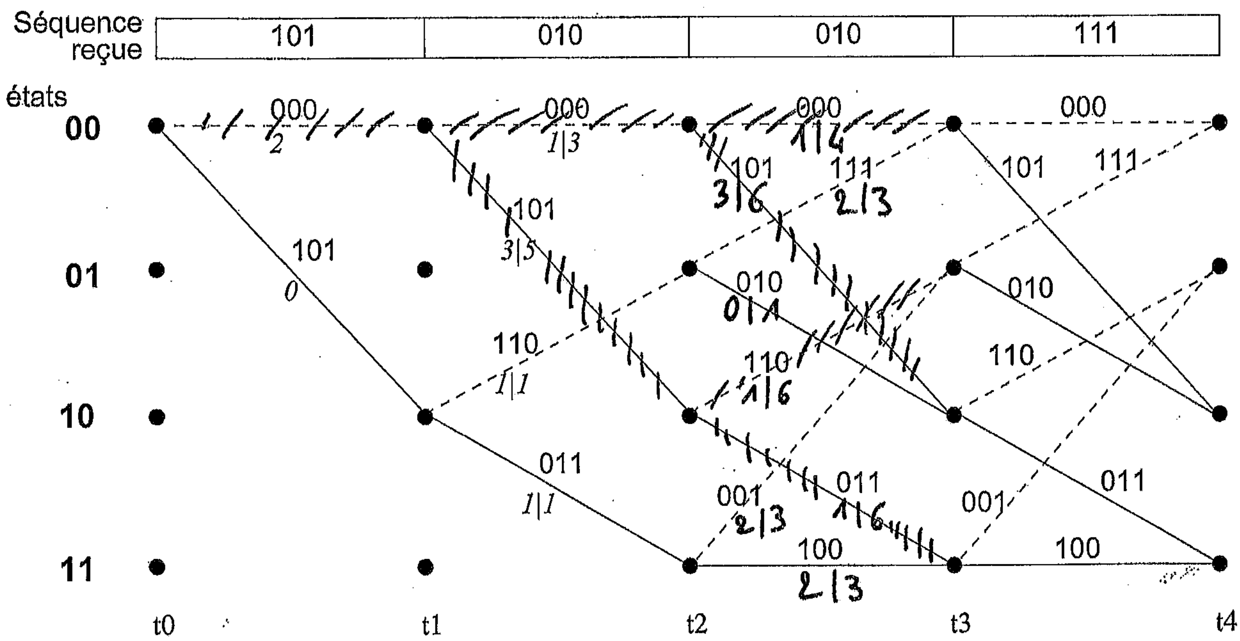
2.1

2 branches supprimées entre t_2 et t_3 sont issues de la même branche entre t_1 et t_2, celle-ci est donc aussi supprimée

De même pour les 2 autres branches supprimées entre t_2 et t_3.

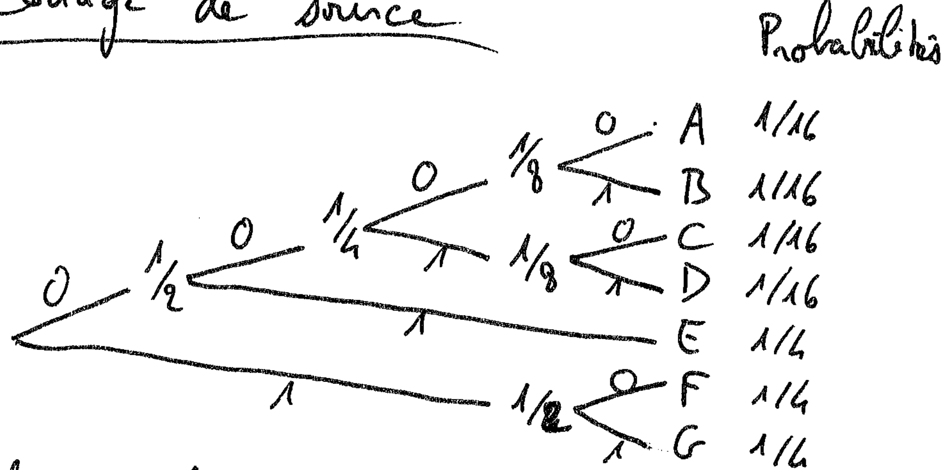
Les 2 branches $t_1 - t_2$ ainsi supprimées sont elles aussi issues de la même branche 00→00 entre t_0 et t_1, que l'on supprime donc.

On peut donc dès à présent décoder le 1er triplet : 101, qui correspond à 1.



2.2) Codage de source

a)



Codage : A : 0000 E : 01
B : 0001 F : 10
C : 0010 G : 11
D : 0011

b) Pour A, B, C, D : $I = -\log_2\left(\frac{1}{16}\right) = 4$ = longueur du code

Pour E, F, G : $I = -\log_2\left(\frac{1}{4}\right) = 2$ = ―――

On note n_i la longueur du code du ieme symbole x_i

→ longueur moyenne : $L = \sum_{i=1}^{7} n_i P(x_i)$

→ entropie : $H = \sum_{i=1}^{7} I(x_i) P(x_i)$

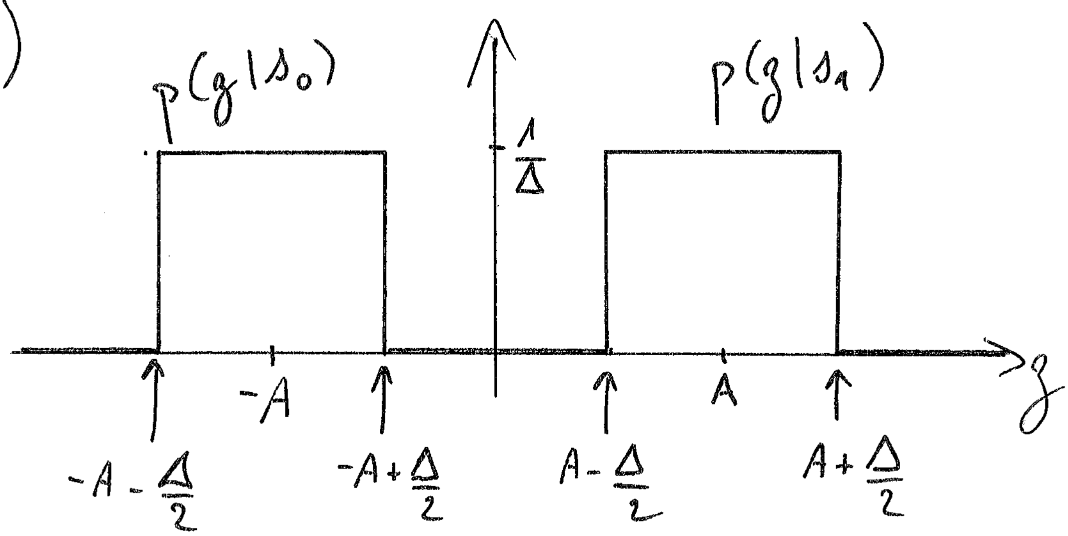
Or on a montré que $\forall i$, $I(x_i) = n_i$. Donc $L = H$

c) Cette inégalité indique que pour $N \to \infty$, $H(X) \to \frac{L}{N}$ ce qui maximise l'efficacité du codage, puisqu'on a alors 1 bit d'information par élément binaire.

Ici, on a déjà $L = H$ donc une efficacité de 100 %. Il est donc inutile de grouper les symboles avant de les coder

2.3

a)

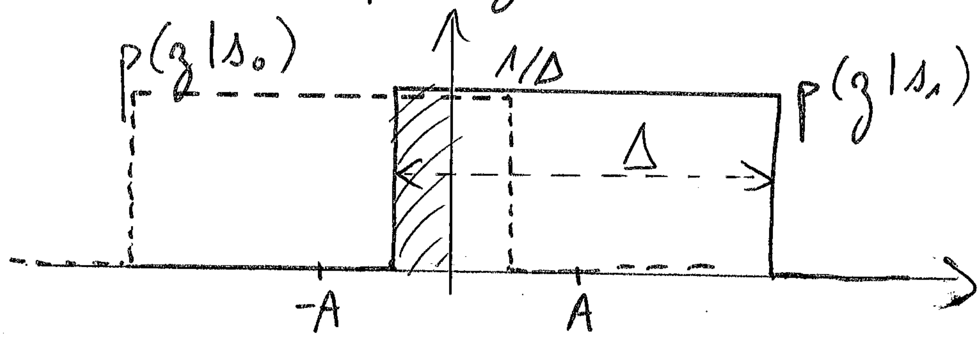


b) Si $\Delta < 2A$,

$P(z < 0 \mid s_1) = P(z > 0 \mid s_0) = 0$,

donc il ne peut y avoir d'erreur.

c)



$$P(r_0 \mid s_1) = P(z < 0 \mid s_1)$$

$$= \int_{-\infty}^{0} p(z \mid s_1)\, dz$$

$$= \left(\frac{\Delta}{2} - A\right) \times \frac{1}{\Delta} \quad \text{(aire hachurée)}$$

$$= \frac{1}{2} - \frac{A}{\Delta}$$

De même pour $P(r_1 \mid s_0)$

d) $e = \{(s_0, r_1) ; (s_1, r_0)\}$

$$P_e = P(s_0, r_1) + P(s_1, r_0)$$

$$= P(r_1 \mid s_0) P(s_0) + P(r_0 \mid s_1) P(s_1)$$

$$= 1/2 - A/\Delta$$