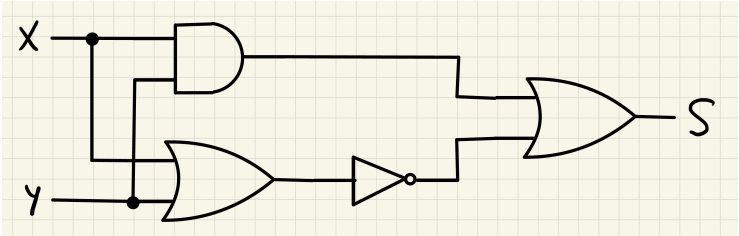
# 3 Circuits

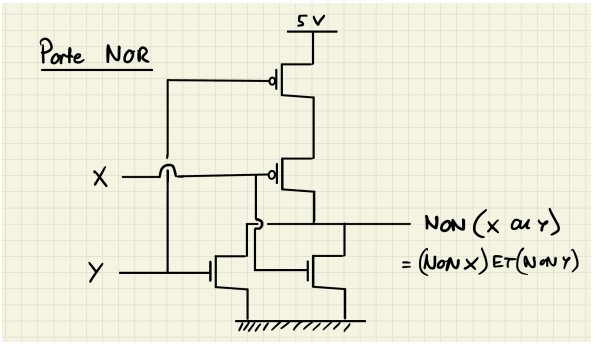
**Exercice 12 (automne 2022-2023).**

Voici un circuit possible:




**Exercice 13 (printemps 2023-2024).**

Voici un circuit possible:



# 4 Signaux

**Exercice 14 (automne 2020-2021).**

**a)** Le passage à travers un filtre à moyenne mobile ne change pas la fréquence d'une sinusoïde pure, donc on n'entendra aucune dissonance; les notes avec une haute fréquence seront seulement un peu plus atténuées que les autres.

**b)** Pour que la condition du théorème d'échantillonnage soit respectée, il faut (et il suffit) que la fréquence de la note jouée soit plus petite que 500 Hz. Les notes DO ($f_4 = 523.3$ Hz) et RÉ ($f_5 = 587.3$ Hz) subiront donc l'effet stroboscopique lors de leur reconstruction, et créeront des dissonances. Seules 9 notes de la mélodie sonneront correctement.

**c)** Pour éviter les dissonances, il faut filtrer la mélodie avec un filtre passe-bas idéal avant de l'échantillonner. En choisissant une fréquence de coupure $f_c$ comprise entre 495 et 523 Hz, on conserve un maximum de notes (9) de la mélodie (les autres se transforment en silences).

**d)** En appliquant l'algorithme de Huffman, qu'on sait être optimal dans ce contexte (sans que ceci n'ait été démontré en cours), on obtient (par exemple) le code binaire sans préfixe suivant:

$$\text{SOL} \longleftrightarrow 1 \qquad \text{LA} \longleftrightarrow 011 \qquad \text{SI} \longleftrightarrow 001 \qquad \text{DO} \longleftrightarrow 010 \qquad \text{RÉ} \longleftrightarrow 000$$

qui utilise $6 \times 1 + 6 \times 3 = 24$ bits en tout (= 2 bits par note), donc 12 bits de moins que la solution proposée initialement.

**e)** Le calcul de l'entropie binaire de la séquence de notes donne:

$$\begin{aligned} H &= \frac{6}{12} \times \log_2\left(\frac{12}{6}\right) + 2 \times \frac{2}{12} \times \log_2\left(\frac{12}{2}\right) + 2 \times \frac{1}{12} \times \log_2(12) \\ &= \frac{1}{2} + \frac{1}{3}\left(1 + \log_2(3)\right) + \frac{1}{6}\left(2 + \log_2(3)\right) = \frac{7}{6} + \frac{1}{2}\,\log_2(3) \end{aligned}$$

Donc en approximant ceci, on obtient $H \simeq 1.16 + \frac{1.58}{2} = 1.16 + 0.79 = 1.95$.

Le théorème de Shannon qu'on a démontré en cours affirme qu'aucun code binaire sans préfixe ne permet de représenter la séquence de notes ci-dessus avec une longueur de code plus petite ou égale à $H$, donc c'est une preuve que la performance obtenue avec le code binaire ci-dessus est très bonne.

**Exercice 15 (automne 2021-2022).**

**a)** On a l'égalité $120 \times 10^6 \times 8 = 50 \times 60 \times f_e \times 32$. En simplifiant, on trouve $2 \times 10^6 = 50 \times f_e \times 4$, donc $f_e = 10$ kHz. Il faut donc choisir $f_c < 5$ kHz pour respecter la condition de Nyquist.

**b)** Les fréquences de ce signal sont $f_0 = 6$ kHz, $f_0/2 = 3$ kHz, $f_0/3 = 2$ kHz, etc. Avec une fréquence de coupure juste un peu plus petite que 5 kHz, le signal sortant (qui ne subit pas l'effet stroboscopique) est donc:

$$X(t) = \sum_{n \geq 2} \frac{1}{n}\,\sin\left(\frac{2\pi f_0 t}{n}\right), \quad t \in \mathbb{R}$$

**Exercice 16 (automne 2022-2023).**

**a)** La bande passante vaut $f$, car chaque composante est de fréquence $f$ (et les deux composantes ne s'annulent pas).

**b)** En utilisant la formule donnée en indication, on trouve

$$\widehat{X}(t) = 2 \cdot \sin(\pi f T_0) \cdot \cos(2\pi f t - \pi f t_0)$$

et donc, l'amplitude de ce signal vaut $a = 2 \cdot |\sin(\pi f t_0)|$ (réponse $a = 2 \cdot \sin(\pi f t_0)$ aussi acceptée).

**c)** L'amplitude est maximale quand par exemple $\pi f t_0 = \pi/2$, i.e. $t_0 = 1/2f$.

**d)** Non: l'amplitude de sortie oscille avec la valeur de $f$ en entrée, mais ne décroît pas lorsque $f$ grandit (à noter qu'il est faux de dire que ce n'est pas un filtre passe-bas car la fréquence de sortie reste identique; en effet, le filtre à moyenne mobile, qui lui est un filtre passe-bas, a cette même propriété).

**Exercice 17 (printemps 2022-2023).**

**a)** Un tel filtre est un filtre passe-bas, car il atténue plus l'amplitude des hautes fréquences que celle des basses fréquences.

**b)** Selon l'indication, $X(t) = \sin(6\pi t + \frac{\pi}{2}) + \frac{1}{10}\sin(30\pi t)$, et donc

$$\widetilde{X}(t) = \frac{1}{3}\sin(6\pi t + \frac{\pi}{6}) + \frac{1}{150}\sin(30\pi t)$$

(car les deux fréquences $f_1$ et $f_2$ valent respectivement 3 Hz et 15 Hz).

**c)** D'après la formule, l'amplitude de la composante $\frac{1}{10}\sin(30\pi t)$ sortant du filtre à moyenne mobile vaut:

$$\left|\frac{\sin(\pi \cdot 15 \cdot 0.2)}{\pi \cdot 15 \cdot 0.2}\right| = \left|\frac{\sin(3\pi)}{3\pi}\right| = 0$$

donc c'est le filtre à moyenne mobile qui permet de mieux atténuer le bruit; il atténue même parfaitement celui-ci.

**d)** Vu que la bande passante du signal filtré vaut 3 Hz, on n'assistera pas à l'effet stroboscopique.

**Exercice 18 (automne 2023-2024).**

**a)** $\widehat{X}(t) = \frac{1}{8} \cdot \sin(6\pi t)$, et donc $B_{\widehat{X}} = 3$ Hz. D'autre part, selon le rappel ci-dessus:
$Y(t) = \frac{1}{2} \cdot (\cos(2\pi t) - \cos(8\pi t))$, et donc $\widehat{Y}(t) = \frac{1}{2} \cdot \cos(2\pi t)$ et $B_{\widehat{Y}} = 1$ Hz.

**b)** Non, car la bande passante du signal $\widehat{Z}$ vaut $B_{\widehat{Z}} = \max\{B_{\widehat{X}}, B_{\widehat{Y}}\} = 3$ Hz, et la fréquence d'échantillonnage $f_e$ vaut donc plus du double de cette valeur.

**c)** Encore non, car $Z(t) = \frac{1}{8} \cdot \sin(6\pi t) + \frac{1}{2} \cdot \cos(2\pi t) = \widehat{Z}(t)$ (car les termes $+\frac{1}{2} \cdot \cos(8\pi t)$ et $-\frac{1}{2} \cdot \cos(8\pi t)$ s'annulent).

**Exercice 19 (printemps 2023-2024).**

**a)** $\widehat{X}(t) = \frac{\sin(3\pi/4)}{3\pi/4} \cdot \sin\left(6\pi t - \frac{3\pi}{4}\right) + \frac{\sin(\pi)}{\pi} \cdot \sin(8\pi t - \pi) = \frac{\sin(3\pi/4)}{3\pi/4} \cdot \sin\left(6\pi t - \frac{3\pi}{4}\right)$

**b)** $B_{\widehat{X}} = 3$ Hz

**c)** $f_e > 6$ Hz

**d)** Dans ce cas, $\widehat{X}(t) = \frac{\sin(3\pi/2)}{3\pi/2} \cdot \sin\left(6\pi t - \frac{3\pi}{2}\right)$, donc la réponse à la question c) ne change pas.

**Exercice 20 (automne 2024-2025).**

**a)** En utilisant la formule donnée dans le rappel, on trouve:

$$Z(t) = \frac{1}{2} \cdot (\cos(\pi t) - \cos(7\pi t)) - \frac{1}{2} \cdot (\cos(3\pi t) - \cos(7\pi t)) = \frac{1}{2} \cdot (\cos(\pi t) - \cos(3\pi t))$$

donc la bande passante de $Z$ vaut 1,5 Hz.

**b1)** Oui, car il est toujours vrai que pour un nombre entier $m$, $X_I(mT_e) = X(mT_e)$, et ici $T_e = \frac{1}{2}$, donc

$$X_I(n) = X_I(2nT_e) = X(2nT_e) = X(n) \quad \text{pour tout } n \in \mathbb{Z}$$

**b2)** Vu que $T_e = \frac{1}{2}$ sec, $f_e = 2$ Hz, et donc la bande passante $f_{\max}$ du signal $X$ doit satisfaire la condition $f_{\max} < \frac{f_e}{2} = 1$ Hz.

**c)** Pour éviter l'effet stroboscopique, il faut, avant d'échantillonner le signal, supprimer la fréquence $f = 1{,}5$ Hz présente dans le signal $X$ avec un filtre passe-bas idéal de fréquence de coupure $f_c$ comprise entre 0,5 Hz et 1,5 Hz (strictement), de manière à conserver l'autre fréquence $f = 0{,}5$ Hz présente dans le signal $Z$. Le signal reconstruit est alors le même que le signal filtré et vaut

$$\widehat{Z}(t) = \frac{1}{2} \cdot \cos(\pi t)$$

# 5 Compression

**Exercice 21 (printemps 2020-2021).**

**a)**

| **algo_rec** |
|---|
| entrée : *listes $L_1$, $L_2$, chacune de taille $n$*<br>sortie : *oui/non* |
| $L_2 \longleftarrow$ **tri par fusion**$(L_2, n)$<br>***Pour*** *$i$ allant de 1 à $n$*<br>  ***Si*** **recherche dichotomique**$(L_2, n, L_1(i)) =$ *oui*<br>  ***sortir :*** *oui*<br>***sortir :*** *non* |

**b)**

Vu que toutes les lettres de $L_1$ sont différentes, $H(L_1) = \log_2(n)$; même chose pour $L_2$. (1 point)

Vu également que **algo**$(L_1, L_2, n) =$ non, $L_1 \cup L_2$ est une liste de $2n$ lettres qui sont toutes différentes, et donc $H(L_1 \cup L_2) = \log_2(2n)$. (1 point)

Et on vérifie que $(*)$ $1 + \frac{\log_2(n)+\log_2(n)}{2} = 1 + \log_2(n) = \log_2(2) + \log_2(n) = \log_2(2n)$. (2 points)

*Note:* Cette égalité se généralise au cas où les listes $L_1, L_2$ sont des listes quelconques de lettres vérifiant juste la propriété **algo**$(L_1, L_2, n) =$ non.

**c)** L'égalité se transforme en une inégalité stricte:

$$H(L_1 \cup L_2) < 1 + \frac{H(L_1) + H(L_2)}{2}$$

Justifications possibles:

- Avec un exemple, le plus simple étant $L_1 = L_2$, menant à

$$H(L_1 \cup L_2) = H(L_1) < 1 + \frac{H(L_1) + H(L_1)}{2} = 1 + H(L_1)$$

mais on peut aussi penser à deux listes ne partageant qu'une seule lettre en commun.

- Avec un calcul général (difficile)

**Exercice 22 (automne 2021-2022).**

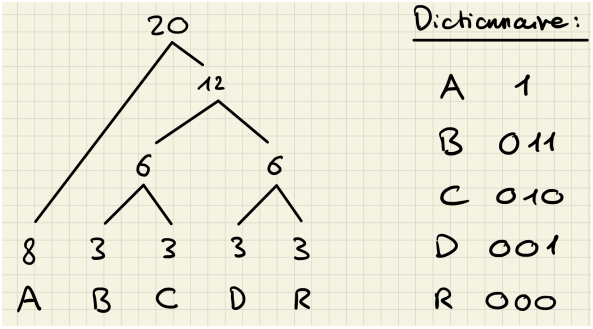
**a)** La réponse est indépendante de la longueur du texte, car seules les probabilités d'apparition des lettres comptent. On peut supposer par exemple que chaque lettre apparaît une fois et que l'espace apparaît 16 fois. En créant l'arbre de Huffman, on trouve que l'espace est représenté par 1 bit et que chaque lettre est représentée par 5 bits, donc en moyenne: $\frac{1}{2}\,1 + \frac{1}{2}\,5 = 3$ bits par caractère.

**b)** Dans ce cas, on peut supposer que chaque lettre apparaît 2 fois et que l'espace apparaît 16 fois. En reconstruisant l'arbre de Huffman dans ce cas, on voit que la représentation prcécédente utilisant 1 bit pour l'espace et 5 bits pour les autres lettres est toujours optimale. On obtient alors en moyenne $\frac{1}{3}\,1 + \frac{2}{3}\,5 = \frac{11}{3} \simeq 3,67$ bits par caractère.

**Exercice 23 (printemps 2021-2022).**

**a)** $H(X) = \frac{4}{5} - \frac{3}{5}\log_2(3) + \log_2(5) \simeq 2.14$

**b)** Avec l'algorithme de Huffman, on obtient (l'algorithme de Shannon-Fano est sous-optimal ici) :




**c)** Le nombre moyen de bits utilisés par lettre vaut $44/20 = 2.2$. C'est cohérent avec le théorème de Shannon qui dit que ce nombre est toujours plus grand ou égal à l'entropie de la séquence, qui vaut ici 2.14.

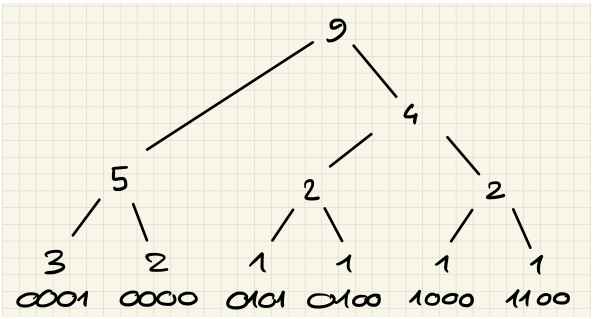
**d)** Méthode I: 80 bits. Méthode II: 120 bits. Méthode III: 77 bits.

**e)** Les trois méthodes permettent de détecter ces erreurs (car la distance minimale de chacun de ces systèmes d'encodage est respectivement: 2, 3, 3, et au plus une erreur survient tous les 8 bits).

**f)** Seules les méthodes II et III permettent de corriger ces erreurs, pour les mêmes raisons que ci-dessus.

**Exercice 24 (automne 2022-2023).**

- En regroupant les bits par 4, on obtient l'arbre suivant avec l'algorithme de Huffman:




ce qui donne le dictionnaire:

| symbole | 0001 | 0000 | 0101 | 0100 | 1000 | 1100 |
|---|---|---|---|---|---|---|
| nb apparitions | 3 | 2 | 1 | 1 | 1 | 1 |
| mot de code | 11 | 10 | 011 | 010 | 001 | 000 |

Et donc le nombre total de bits utilisés dans ce cas vaut $5 \cdot 2 + 4 \cdot 3 = 22$.

- En regroupant les bits par 3, les symboles 000, 001, 010 et 100 apparaissent chacun 3 fois dans la séquence, et le codage de Huffman utilise 2 bits pour chaque symbole, donc $4 \cdot 3 \cdot 2 = 24$ bits au total. La première méthode est donc plus efficace.

**Exercice 25 (printemps 2022-2023).**

**a)** Une séquence de 8 lettres possible est ABCCDDDD.

**b)** $\frac{2}{8} \cdot 3 + \frac{2}{8} \cdot 2 + \frac{4}{8} \cdot 1 = \frac{6+4+4}{8} = \frac{14}{8} = \frac{7}{4}$ $(= 1.75)$

**c)** Une réponse possible est:

| A | B | C | D | E |
|---|---|---|---|---|
| 1111 | 1110 | 110 | 10 | 0 |

**d)** $\frac{2}{16} \cdot 4 + \frac{2}{16} \cdot 3 + \frac{4}{16} \cdot 2 + \frac{8}{16} \cdot 1 = \frac{8+6+8+8}{16} = \frac{30}{16} = \frac{15}{8}$ $(= 1.875)$

**Exercice 26 (automne 2023-2024).**

**a)** Une réponse possible:

| A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|
| 0 | 1 | 20 | 21 | 220 | 221 | 222 |

**b)** En tout, $18 \times 1 + 6 \times 2 + 3 \times 3 = 18 + 12 + 9 = 39$ trits sont utilisés, ce qui donne $\frac{39}{27} = \frac{13}{9}$ trits par lettre en moyenne.

**c)** Vu qu'il y a 7 symboles différents, et que $3 < 8 \leq 9 = 3^2$, 2 trits par lettre seraient nécessaires dans ce cas.

**Exercice 27 (printemps 2023-2024).**

**a)** Les probabilités d'apparition sont les suivantes: espace: 3/8, A: 1/4, B: 1/4, C: 1/8. Dans ce as, il y a deux possibilités pour l'arbre et l'encodage de Huffman (et encore plusieurs possibilités équivalentes):

soit: espace: 11, A: 10, B: 01, C: 00

soit: espace: 1, A: 01, B: 001, C: 000

**b)** Dans les deux cas ci-dessus, le nombre moyen de bits par lettre utilisé vaut 2.

**c)** L'entropie vaut

$$\begin{aligned} H &= \frac{3}{8} \cdot \log_2\left(\frac{8}{3}\right) + 2 \cdot \frac{1}{4} \cdot \log_2(4) + \frac{1}{8} \cdot \log_2(8) \\ &= \frac{3}{8} \cdot 3 - \frac{3}{8} \cdot \log_2(3) + 2 \cdot \frac{1}{2} + \frac{1}{8} \cdot 3 = 2.5 - \frac{3}{8} \cdot \log_2(3) \end{aligned}$$

Avec l'approximation indiquée, on obtient

$$H \simeq 2.5 - \frac{4.8}{8} = 2.5 - 0.6 = 1.9$$

**Exercice 28 (automne 2024-2025).**

**a)** $H(\text{HONO}) = 1,5$ $\qquad$ $H(\text{LULU}) = 1$ $\qquad$ $H(\text{HONOLULU}) = 2,25$

**b)** (voir question c) ci-dessous pour une réponse alternative à cette question)

Soient $p_1, \ldots, p_k$ les probabilités d'apparition des $k$ lettres différentes qui forment la séquence $X$ et $q_1, \ldots, q_m$ les probabilités d'apparition des $m$ lettres différentes qui forment la séquence $Y$. Vu que les deux séquences sont de même longueur $(n)$, les probabilités d'apparition de toutes ces lettres dans la séquence $XY$ sont donc données par

$$\frac{p_1}{2}, \ldots, \frac{p_k}{2}, \frac{q_1}{2}, \ldots, \frac{q_m}{2}$$

et donc l'entropie vaut

$$\begin{aligned} H(XY) &= \sum_{i=1}^{k} \frac{p_i}{2} \log_2\left(\frac{2}{p_i}\right) + \sum_{j=1}^{m} \frac{q_j}{2} \log_2\left(\frac{2}{q_j}\right) \\ &= \frac{1}{2}\left(\log_2(2) + \sum_{i=1}^{k} p_i \log_2\left(\frac{1}{p_i}\right)\right) + \frac{1}{2}\left(\log_2(2) + \sum_{j=1}^{k} q_j \log_2\left(\frac{1}{q_j}\right)\right) \\ &= \frac{1}{2}\left(1 + H(X) + 1 + H(Y)\right) = \frac{H(X) + H(Y)}{2} + 1 \end{aligned}$$

comme on peut le vérifier dans l'exemple HONOLULU de la question a) : $2,25 = \frac{1,5+1}{2} + 1$

**c)** En général, pour obtenir un code sans préfixe pour la séquence $XY$, on rajoute un 0 au début de chaque mot de code des lettres de la séquence $X$, et un 1 au début de chaque mot de code des lettres de la séquence $Y$.

Ainsi par exemple, si pour encoder la séquence HONO, on utilise le dictionnaire $H : 00$, $N : 01$, $O : 1$ et pour la séquence LULU, on utilise le dictionnaire $L : 0$, $U : 1$, alors la méthode décrite ci-dessus donnera le dictionnaire suivant:

$$H : 000, \quad N : 001, \quad O : 01, \quad L : 10, \quad U : 11$$

qui est bien un dictionnaire sans préfixe.

*Retour à la question b):* Comme les dictionnaires utilisés ci-dessus sont optimaux et que leurs longueurs moyennes correspondent aux entropies des différentes séquences, on trouve que

$$L(C_{XY}) = \frac{1}{2}\left(\left(1 + L(C_X)\right) + \left(1 + L(C_Y)\right)\right) = \frac{L(C_X) + L(C_Y)}{2} + 1$$

ce qui nous redonne bien la réponse de la question b).

# 6 Correction d'erreurs

**Exercice 29 (automne 2021-2022).**

**a)** Si on pense aux paquets $X_1, X_2, X_3, X_4, X_5$ comme à des bits isolés, on voit que les mots de code possiblement envoyés sont:

$$00000 \qquad 10101 \qquad 01011 \qquad \text{et} \qquad 11110$$

La distance minimale de ce code correcteur d'erreur vaut $d = 3$: il est donc possible de corriger 2 effacements, dans le pire des cas.

*Note:* Dans certains cas, il est possible de corriger jusqu'à 3 effacements (si par exemple les paquets $X_3, X_4, X_5$ sont effacés), mais pas dans le pire des cas.

**b)** Il suffit que la clé $K$ soit d'une longueur de 2048 bits. En effet, divisons cette clé en deux parties égales $K_1$ et $K_2$. Alice calcule $Y_1 = X_1 \oplus K_1$ et $Y_2 = X_2 \oplus K_2$ et transmet:

$$Y_1, Y_2, Y_1, Y_2, Y_1 \oplus Y_2$$

Le même niveau de correction d'erreurs est ainsi assuré, et même si Eve intercepte tous les paquets, elle ne peut déchiffrer ni $X_1$, ni $X_2$.

**Exercice 30 (printemps 2022-2023).**

**a)** La distance minimale $d = 4$: considérons pour ça deux mots de code $x$ et $y$:

- si $x_1, x_2, x_3$ et $y_1, y_2, y_3$ diffèrent en 1 position, alors $x_4, x_5, x_6, x_7$ et $y_4, y_5, y_6, y_7$ diffèrent en 3 positions.

- si $x_1, x_2, x_3$ et $y_1, y_2, y_3$ diffèrent en 2 positions, alors $x_4, x_5, x_6, x_7$ et $y_4, y_5, y_6, y_7$ diffèrent en 2 positions.

- si $x_1, x_2, x_3$ et $y_1, y_2, y_3$ diffèrent en 3 positions, alors $x_4, x_5, x_6, x_7$ et $y_4, y_5, y_6, y_7$ diffèrent en 1 position.

A noter que dans ce code particulier, *tous* les mots de code sont exactement à distance 4 l'un de l'autre (ex: 0000000 et 1110001).

**b)** Vu que la distance minimale $d = 4$, ce code peut corriger 3 effacements et 1 erreur.

**c)** Non, dans ce cas, la distance du message reçu avec un mot de code (1001101, 0101011 ou 0010111) est au minimum 2, et il n'est donc pas possible de corriger l'erreur.

**Exercice 31 (automne 2023-2024).**

**a)** Considérons deux messages de 4 bits $x_1, x_2, x_3, x_4$ et $y_1, y_2, y_3, y_4$:

- si ceux-ci diffèrent en 1 position, alors les bits de parité correspondants diffèrent en 3 positions;

- si ceux-ci diffèrent en 2 positions, alors les bits de parité correspondants diffèrent en 2 positions;

- si ceux-ci diffèrent en 3 positions, alors les bits de parité correspondants diffèrent en 1 position;

(à noter que comme tout est symétrique, il suffit chaque fois de considérer p. ex. les cas 0000 et 1000, puis 0000 et 1100, et enfin 0000 et 1110).

Donc la distance minimale de ce code vaut 4.

**b)** Ce code permet donc de corriger $d - 1 = 3$ effacements ou $\lfloor \frac{d-1}{2} \rfloor = 1$ erreur.

**c)** Cette distance vaut 8, avec par exemple les mots de code 00000000 et 11111111.

**Exercice 32 (printemps 2023-2024).**

**a)** La distance minimale $d = 2$:

Si les séquences $x_1, x_2, x_3, x_4$ et $y_1, y_2, y_3, y_4$ diffèrent en une position seulement, alors nécessairement au moins un des bits de parité (le 5 ou le 6) sera différent. Donc la distance minimale $d \geq 2$.

D'autre part, 0000000 et 1100000 sont des mots de code à distance 2, donc la distance minimale $d = 2$.

**b)** 0 erreur et 1 effacement

**c)** Les mots de code envoyés peuvent être 1110011 ou 1101011.

**Exercice 33 (automne 2024-2025).**

**a)** $d = 2$; c'est la distance entre les mots de code 11111 et 10101, par exemple; toutes les autres distances sont plus grandes ou égales à 2. Donc ce code corrige au plus un effacement, et aucune erreur.

**b)** Oui. La distance minimale de code valant 2, si seulement un bit est modifié, il sera toujours possible de détecter une telle erreur.

**c)** Oui, car le seul mot de code qui diffère de la séquence reçue en seul bit est le mot de code 00000 (correspondant au message "il fait beau").

*Note:* Ici, il est donc possible de corriger l'erreur (mais on n'est pas dans le pire des cas).