Nicolas Sendrier

Majeure d'informatique

Introduction la théorie de l'information

Cours n°10

Codes LDPC

Codes à matrice de parité creuse (1)

$$H = \begin{pmatrix}
0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0 \\
0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&0 \\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0 \\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0 \\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&1&0&0 \\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0 \\
0&0&0&0&0&0&0&1&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0 \\
0&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0 \\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&1&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0 \\
0&1&0&0&1&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0 \\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0 \\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1 \\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&1&0&0&0&0&0&0 \\
1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0 \\
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\
0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0 \\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\
0&0&0&0&1&1&0&1&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&1&0&1&0&0&0 \\
0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1 \\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0
\end{pmatrix}$$

Codes à matrice de parité creuse (2)

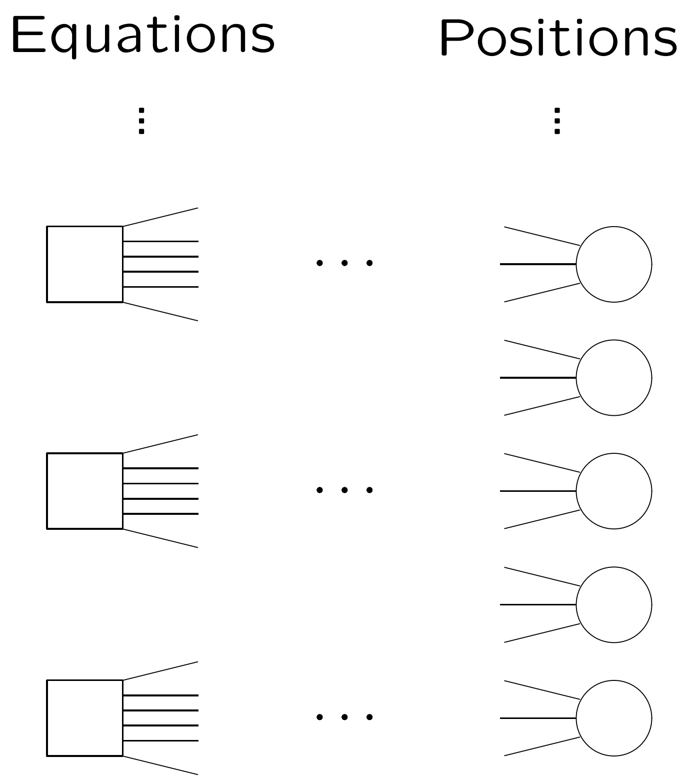
La matrice H du transparent précédent est de taille 23×46 et contient exactement 6 '1' par ligne et 3 '1' par colonne. Nous noterons $\mathcal{C}$ le code, r le nombre d'équations et n la longueur.

Elle définit un code LDPC (*Low Density Parity Check code*). Les LDPC $(3,6)$ sont parmi les plus classiques, les longueurs utilisées vont jusqu'à $100\,000$.

En pratique, on préférera une représentation creuse, et la i-ième équation sera un sous-ensemble E_i de $[0,n[$

$$((x_0,\ldots,x_{n-1})\in\mathcal{C}) \Longleftrightarrow \left(\forall i, \sum_{j\in E_i} x_j = 0\right)$$

Représentation par un graphe d'un LDPC(3,6)



Le graphe possède deux types de sommets, les équations et les positions. Les équations sont d'arité 6 et les position d'arité 3.

Il y a autant d'arètes partant des équations que d'arètes partant des sommets. Il faut les mettre en correspondance, sachant qu'une équation ne peut pas utiliser plusieurs fois la même position (et réciproquement).

Le code sera meilleur s'il n'existe pas de cycles de longueur 4, ou même 6.

Décodage – Décision dure

On a reçu un vecteur $y = (y_0, \ldots, y_{n-1})$. Si y est un mot de code, les y_i vérifient toutes les équations de parité.

Sinon certaines sont fausses et on itère le processus suivant :
- on évalue chaque équation i, $0 \leq i < r$, on obtient $s_i = 0$ si elle est juste et $s_i = 1$ si elle est fausse,
- pour chaque position j, $0 \leq j < n$, on change la valeur de y_j si une majorité de ses équations ne sont pas vérifiées.

On ne modifie les y_j à chaque itération qu'après évaluation de tous les s_i.

On s'arrête, soit au bout d'un nombre prédéterminé d'itérations (plusieurs dizaines en pratique), soit lorsque y ne varie plus.

Décodage – Décision souple

On a reçu un vecteur $y = (y_0, \ldots, y_{n-1}) \in \{-1, +1\}$, où -1 correspond à 0 et $+1$ correspond à 1.

Pour tout i, $0 \leq i < r$, on note E_i la liste de ses positions.

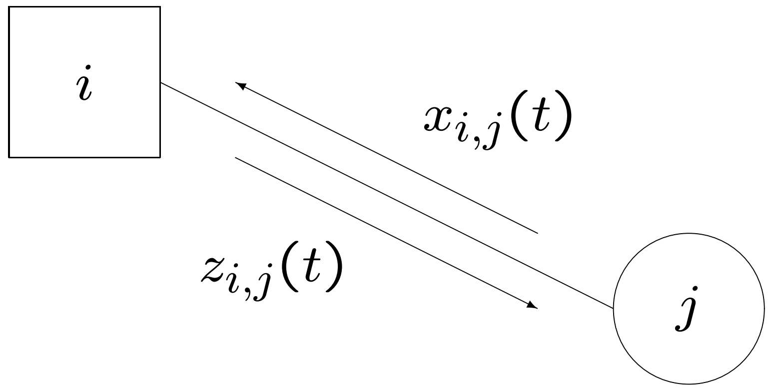
Pour tout (i, j) tel que $j \in E_i$, on initialise

$$x_{i,j}(0) = y_j \text{ et } z_{i,j}(0) = 0$$

et pour tout entier $t > 0$, on définit

$$\begin{aligned} z_{i,j}(t) &= \min_{j' \in E_i \setminus \{j\}} (|x_{i,j}(t-1)|) \times signe\left(\prod_{j' \in E_i \setminus \{j\}} x_{i,j'}(t-1)\right) \\ x_{i,j}(t) &= y_j + \sum_{j \in E_{i'}, i' \neq i} z_{i,j}(t) \end{aligned}$$

Décodage – Décision souple, interprétation graphique



À chaque unité de temps, une information de vraisemblance provenant des équations est fournie aux positions

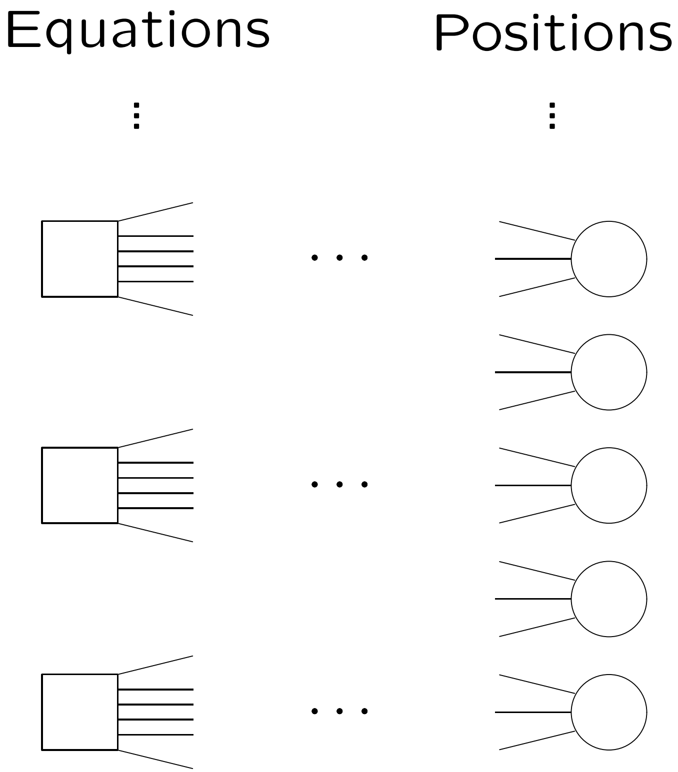
puis

une information de vraisemblance provenant des positions est fournie aux équations

On effectue un nombre prédéterminé d'itérations puis la valeur finale de chaque bit dépendra du signe de

$$y_j + \sum_{j \in E_i} z_{i,j}(t)$$

Génération des équations



On sextuple les indices des équations $\rightarrow u$ entre 0 et $6r-1$.

On triple les indices des positions $\rightarrow v$ entre 0 et $3n-1=6r-1$.

On génère une permutation aléatoire de $3n$ éléments.

Chaque fois qu'un cycle de petite taille (2, 4, éventullement 6) est détecté, on croise l'une des arètes du cycle avec une autre choisie au hasard.