Generalized Sparse Matrices and Applications to Decoding and Cryptography

Maxime BROS

University of Limoges (France)
XLIM Research Institute, UMR 7252

maxime.bros@etu.unilim.fr

May 23, 2019

Reminder about Error Correcting Codes

Error correcting codes are used to transmit informations (satellites, DVD, telecommunications, ...) but also for cryptographic purpose.

Reminder about Error Correcting Codes

Error correcting codes are used to transmit informations (satellites, DVD, telecommunications, ...) but also for cryptographic purpose.

Code (definition)

A code $\mathcal{C}$ is vector space of $GF(q)^n$ of dimension k.

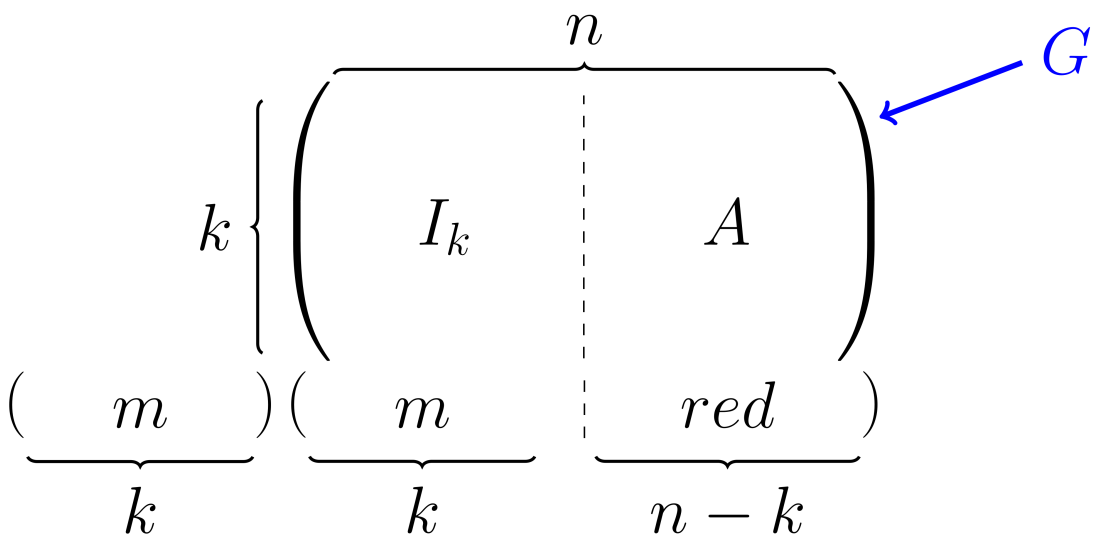
Reminder about Error Correcting Codes

Error correcting codes are used to transmit informations (satellites, DVD, telecommunications, ...) but also for cryptographic purpose.

Code (definition)

A code $\mathcal{C}$ is vector space of $GF(q)^n$ of dimension k.

$$\begin{array}{rcl} \mathcal{E} : GF(q)^k & \longrightarrow & GF(q)^n \\ m & \longmapsto & mG \end{array}$$



Reminder about Error Correcting Codes

Parity Check Matrix

H is a parity check matrix for the code $\mathcal{C}$ if for every word $c \in GF(q)^n$:

$$c \in \mathcal{C} \iff Hc^T = 0_{n-k}$$

Hard Problems in Coding Theory

Decoding Problem (computational)

- Let G be a matrix $k \times n$ over a field $\mathbb{K}$,
  y a vector of length n (with coefficients in $\mathbb{K}$)
  and $\omega \in \mathbb{N}$.
- Find $m \in \mathbb{K}^k$ such that $weight(y - mG)$ for a given metric is smaller or equal to ω.

Syndrome Decoding Problem (computational)

Hard Problems in Coding Theory

Decoding Problem (computational)

- Let G be a matrix $k \times n$ over a field $\mathbb{K}$,
  y a vector of length n (with coefficients in $\mathbb{K}$)
  and $\omega \in \mathbb{N}$.
- Find $m \in \mathbb{K}^k$ such that $weight(y - mG)$ for a given metric is smaller or equal to ω.

Syndrome Decoding Problem (computational)

- Let H be a matrix $(n-k) \times n$ over a field $\mathbb{K}$,
  s a vector of length $(n-k)$ (with coefficients in $\mathbb{K}$)
  and $\omega \in \mathbb{N}$.
- Find $e \in \mathbb{K}^n$ with weight smaller or equal to ω for a given metric such that $He^t = s$?

Hard Problems in Coding Theory

- **These 2 problems are equivalent**.

Hard Problems in Coding Theory

- **These 2 problems are equivalent**.
- Proven NP-complete with Hamming metric in 1978 by Berlekamp, McEliece and Tilborg.

Hard Problems in Coding Theory

- **These 2 problems are equivalent**.
- Proven NP-complete with Hamming metric in 1978 by Berlekamp, McEliece and Tilborg.
- Proven to be probabilistically NP-complete with rank metric in 2017 by Gaborit and Zémor.

Decoding codes with sparse parity check matrix

- We will study two codes (MDPC and LRPC) for which one uses the sparsity of their **parity check matrix** to decode.

Decoding codes with sparse parity check matrix

- We will study two codes (MDPC and LRPC) for which one uses the sparsity of their **parity check matrix** to decode.
- The notion of sparsity one uses depends on the chosen metric.

MDPC

- For MDPC codes sparse means : majority of zeros in the matrix (only $\mathcal{O}(\sqrt{n})$ 1's per row)

MDPC

- For MDPC codes sparse means : majority of zeros in the matrix (only $\mathcal{O}(\sqrt{n})$ 1's per row)
- Only H is sparse $\Longrightarrow$ "Moderate Density Parity Check".

Gallager's algorithm

Reminder : only the error contributes to the syndrome.

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\0\\0
\end{pmatrix}$$

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0
\end{pmatrix}$$

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)=

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1 , 1

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1 , 1 , 3, 2, 2, 2, 3, 1, 1, 1, 1, 2, 2, 0, 1, 2, 1, 1, 2}

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1 , 1 , 3, 2, 2, 2, 3, 1, 1, 1, 1, 2, 2, 0, 1, 2, 1, 1, 2}
Let $\tau = 3$ be our threshold.

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix} y_1\\ y_2\\ y_3\\ y_4\\ y_5\\ y_6\\ y_7\\ y_8\\ y_9\\ y_{10}\\ y_{11}\\ y_{12}\\ y_{13}\\ y_{14}\\ \dots\\ y_{20} \end{pmatrix}
=
\begin{pmatrix} 0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0 \end{pmatrix}$$

#(common 1's)= {1 , 1 , 1 , 3, 2, 2, 2, 3, 1, 1, 1, 1, 2, 2, 0, 1, 2, 1, 1, 2}
Let $\tau = 3$ be our threshold.

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\y_2\\y_3\\y_4+1\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\\dots\\y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1 , 1 , 3, 2, 2, 2, 3, 1, 1, 1, 1, 2, 2, 0, 1, 2, 1, 1, 2}
Let $\tau = 3$ be our threshold.

Gallager's algorithm

Reminder : only the error contributes to the syndrome. More precisely
$Hy^T = \underbrace{Hc^T}_{=0} + He^T$

$$\begin{pmatrix}
1&0&1&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&1\\
1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&1&1&0&1&0&0&0&0&0&1&0&0&0&0&0&1&1\\
0&0&0&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&1&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&1&0&0&1&0&0\\
1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&1&1&0&0&1&0
\end{pmatrix}
\begin{pmatrix}
y_1\\ y_2\\ y_3\\ y_4+1\\ y_5\\ y_6\\ y_7\\ y_8+1\\ y_9\\ y_{10}\\ y_{11}\\ y_{12}\\ y_{13}\\ y_{14}\\ \dots\\ y_{20}
\end{pmatrix}
=
\begin{pmatrix}
0\\1\\0\\1\\1\\1\\0\\1\\1\\1\\0\\0
\end{pmatrix}$$

#(common 1's)= {1 , 1 , 1 , 3, 2, 2, 2, 3, 1, 1, 1, 1, 2, 2, 0, 1, 2, 1, 1, 2}
Let $\tau = 3$ be our threshold.

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$
2. For each of the n columns of H, count the number of common 1's between the syndrome and this column

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$
2. For each of the n columns of H, count the number of common 1's between the syndrome and this column
3. For a given column i, if this number of common 1's is **greater than a threshold** τ, then change the i^{th} bit of y

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$
2. For each of the n columns of H, count the number of common 1's between the syndrome and this column
3. For a given column i, if this number of common 1's is **greater than a threshold** τ, then change the i^{th} bit of y
4. Call this new vector y again, and go back to the first step, until either

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$
2. For each of the n columns of H, count the number of common 1's between the syndrome and this column
3. For a given column i, if this number of common 1's is **greater than a threshold** τ, then change the i^{th} bit of y
4. Call this new vector y again, and go back to the first step, until either
   - $s = 0_{n-k}$ $\Longrightarrow$ **RETURN the last y, which is the codeword**

Gallager's algorithm

One receives a word $y := mG + e = codeword + error$.

Bit-flipping algorithm

1. Compute the syndrome $s = Hy^T$
2. For each of the n columns of H, count the number of common 1's between the syndrome and this column
3. For a given column i, if this number of common 1's is **greater than a threshold** τ, then change the i^{th} bit of y
4. Call this new vector y again, and go back to the first step, until either
   - $s = 0_{n-k}$ $\Longrightarrow$ **RETURN the last y, which is the codeword**
   - **a certain number of iterations is reached** $\Longrightarrow$ **RETURN FAIL**

Rank Decoding Problem

- In 1985, Gabidulin, a Russian researcher, introduced rank-codes over an extension field
  (*the use of rank metric started in 1951 by Hua and then in 1978 by Delsarte who introduced rank distance for matrix-codes*).

Rank Decoding Problem

- In 1985, Gabidulin, a Russian researcher, introduced rank-codes over an extension field
  (*the use of rank metric started in 1951 by Hua and then in 1978 by Delsarte who introduced rank distance for matrix-codes*).

Figure: Ernst Gabidulin (1937 (U.S.S.R) - aged 81 today)

Rank Decoding Problem

- In 1985, Gabidulin, a Russian researcher, introduced rank-codes over an extension field
  (*the use of rank metric started in 1951 by Hua and then in 1978 by Delsarte who introduced rank distance for matrix-codes*).
- Using this metric instead of the Hamming metric :

Figure: Ernst Gabidulin (1937 (U.S.S.R) - aged 81 today)

Rank Decoding Problem

- In 1985, Gabidulin, a Russian researcher, introduced rank-codes over an extension field
  (*the use of rank metric started in 1951 by Hua and then in 1978 by Delsarte who introduced rank distance for matrix-codes*).
- Using this metric instead of the Hamming metric :
  - Decoding Problem $\Longrightarrow$ Rank Decoding Problem (RD)

Figure: Ernst Gabidulin (1937 (U.S.S.R) - aged 81 today)

Rank Decoding Problem

- In 1985, Gabidulin, a Russian researcher, introduced rank-codes over an extension field

  (*the use of rank metric started in 1951 by Hua and then in 1978 by Delsarte who introduced rank distance for matrix-codes*).
- Using this metric instead of the Hamming metric :
  - Decoding Problem $\Longrightarrow$ Rank Decoding Problem (RD)
  - Syndrome Decoding Problem $\Longrightarrow$ Rank Syndrome Decoding Pb. (RSD)

Figure: Ernst Gabidulin (1937 (U.S.S.R) - aged 81 today)

Reminder about finite fields

$GF(q^m)$ $\longleftarrow$ finite field with q^m elements

$|$

$|$ m

$|$

$GF(q)$ $\longleftarrow$ finite field with q elements

Reminder about finite fields

$GF(q^m)$ $\longleftarrow$ finite field with q^m elements

$|$

$|$ m

$|$

$GF(q)$ $\longleftarrow$ finite field with q elements

$GF(q^m)$ is a $GF(q)$-vector space of dimension m

Rank metric

$m = 4$, and let $e \in GF(2^4)^4$

$$e = (1 + \alpha^2, \alpha, 1, \alpha^2)$$

$$M = \begin{matrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{matrix} \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

Rank metric

$m = 4$, and let $e \in GF(2^4)^4$

$$e = (1 + \alpha^2, \alpha, 1, \alpha^2)$$

$$M = \begin{matrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{matrix} \begin{pmatrix} 1 & & \\ 0 & & \\ 1 & & \\ 0 & & \end{pmatrix}$$

Rank metric

$m = 4$, and let $e \in GF(2^4)^4$

$$e = (1 + \alpha^2, \alpha, 1, \alpha^2)$$

$$M = \begin{matrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{matrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Rank metric

$m = 4$, and let $e \in GF(2^4)^4$

$$e = (1 + \alpha^2, \alpha, 1, \alpha^2)$$

$$M = \begin{matrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{matrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

e can be represented by a matrix M of $GF(2)$

Rank metric

$m = 4$, and let $e \in GF(2^4)^4$

$$e = (1 + \alpha^2, \alpha, 1, \alpha^2)$$

$$M = \begin{matrix} 1 \\ \alpha \\ \alpha^2 \\ \alpha^3 \end{matrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

e can be represented by a matrix M of $GF(2)$

We define $Rank(e) = Rank(M) = 3$

Rank metric

Notation : all presentation long, q is a power of a prime p and $m, n \in \mathbb{N}^*$.

Rank of a $GF(q^m)^n$ word

Let $e = (e_1, e_2, \ldots, e_n)$ be a vector of $GF(q^m)^n$.
Given a basis $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m)$ of $GF(q^m)$ over $GF(q)$, one gets :
$\forall i \in \{1, .., n\}, \exists!\ (e_{i,1}, e_{i,2}, \ldots, e_{i,m}) \in GF(q)^m$ such that
$e_i = \sum_{j=1}^{m} e_{i,j} \cdot \mathcal{B}_j$.
The rank of e, noted $Rank(e)$, is defined by the rank of the matrix $(e_{i,j})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}}$.

Rank metric

Notation : all presentation long, q is a power of a prime p and $m, n \in \mathbb{N}^*$.

Rank of a $GF(q^m)^n$ word

Let $e = (e_1, e_2, \ldots, e_n)$ be a vector of $GF(q^m)^n$.
Given a basis $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m)$ of $GF(q^m)$ over $GF(q)$, one gets :
$\forall i \in \{1, .., n\}, \exists!\ (e_{i,1}, e_{i,2}, \ldots, e_{i,m}) \in GF(q)^m$ such that
$e_i = \sum_{j=1}^{m} e_{i,j} \cdot \mathcal{B}_j$.
The rank of e, noted $Rank(e)$, is defined by the rank of the matrix $(e_{i,j})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}}$.

Rank distance

$$\begin{array}{rccc} d : & GF(q^m)^n \times GF(q^m)^n & \rightarrow & \mathbb{N} \\ & (e_1, e_2) & \mapsto & Rank(e_1 - e_2) \end{array}$$

is a distance (or a metric) over $GF(q^m)^n$.

Code properties in Rank metric

What are the differences between code with Hamming and Rank metric ?

Minimal distance of a code with rank metric

Let C be a $[n, k, d]$-code over $GF(q^m)$ for the rank metric
As long as C is additive :

$$d = min\{Rank(c) \mid c \in C\}$$

Code properties in Rank metric

What are the differences between code with Hamming and Rank metric ?

Minimal distance of a code with rank metric

Let C be a $[n, k, d]$-code over $GF(q^m)$ for the rank metric
As long as C is additive :

$$d = min\{Rank(c) \mid c \in C\}$$

Support of a word in C

The support of an element $x = (x_1, \ldots, x_n) \in GF(q^m)^n$ is the $GF(q)$-vector space of $GF(q)^m$ generated by the coordinates x_i of x.

Support of a word

$$e = (1+\alpha^2, \alpha, 1, \alpha^2) \in GF(2^4)^4 \quad \text{and} \quad Rank(e) = 3$$

$$support(e) = < \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} >$$

$$= < \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} >$$

Support of a word

$$e = (1+\alpha^2, \alpha, 1, \alpha^2) \in GF(2^4)^4 \quad \text{and} \quad Rank(e) = 3$$

$$support(e) = < \begin{pmatrix}1\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix} >$$

$$= < \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix} >$$

$support(e)$ is a vector space E of $GF(2)^4$ of dimension 3

LRPC

- In this context, rank metric, sparse means with values in a vector space of small dimension d, as seen before it uses the generalization of the support to rank metric.

LRPC

- In this context, rank metric, sparse means with values in a vector space of small dimension d, as seen before it uses the generalization of the support to rank metric.
- The sparse matrix is still $H \Longrightarrow$ "Low Rank Parity Check".

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

- Let H be the parity check matrix of an LRPC code, with coefficients in $F \subset GF(2)^N$ of dimension d.

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

- Let H be the parity check matrix of an LRPC code, with coefficients in $F \subset GF(2)^N$ of dimension d.
- Let e be the error, with coefficients in $E \subset GF(2)^N$ of dimension r. i.e $supp(e) = E$.

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

- Let H be the parity check matrix of an LRPC code, with coefficients in $F \subset GF(2)^N$ of dimension d.
- Let e be the error, with coefficients in $E \subset GF(2)^N$ of dimension r. i.e $supp(e) = E$.
- From the syndrome $s := Hy^T = He^T$, one gets the vector space $S :=< s_1, \ldots, s_{n-k} >$

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

- Let H be the parity check matrix of an LRPC code, with coefficients in $F \subset GF(2)^N$ of dimension d.
- Let e be the error, with coefficients in $E \subset GF(2)^N$ of dimension r. i.e $supp(e) = E$.
- From the syndrome $s := Hy^T = He^T$, one gets the vector space $S :=< s_1, \ldots, s_{n-k} >$
- This vector space is a subset of the product space $< EF >$ of dimension at most rd.

Decoding LRPC

As for the Hamming metric, the knowledge of **the support** of the error is enough to decode.

- Let H be the parity check matrix of an LRPC code, with coefficients in $F \subset GF(2)^N$ of dimension d.
- Let e be the error, with coefficients in $E \subset GF(2)^N$ of dimension r. i.e $supp(e) = E$.
- From the syndrome $s := Hy^T = He^T$, one gets the vector space $S :=< s_1, \ldots, s_{n-k} >$
- This vector space is a subset of the product space $< EF >$ of dimension at most rd.
- Since $rd < (n-k)$, it is very likely that $S =< EF >$.

Decoding LRPC

Let F_i be the i^{th} element of a basis of F, let's consider the vector space

$$F_i^{-1}S = F_i^{-1} < \begin{matrix} E_1F_1, & E_2F_1, & \dots & E_rF_1, \\ E_1F_2, & E_2F_2, & \dots & E_rF_2, \\ \dots & \dots & \dots & \dots \\ E_1F_i, & E_2F_i, & \dots & E_rF_i, \\ \dots & \dots & \dots & \dots \\ E_1F_d, & E_2F_d, & \dots & E_rF_d, \end{matrix} >$$

Decoding LRPC

Let F_i be the i^{th} element of a basis of F, let's considerer the vector space

$$F_i^{-1}S = < \begin{matrix} F_i^{-1}E_1F_1, & F_i^{-1}E_2F_1, & \dots & F_i^{-1}E_rF_1, \\ F_i^{-1}E_1F_2, & F_i^{-1}E_2F_2, & \dots & F_i^{-1}E_rF_2, \\ \dots & \dots & \dots & \dots \\ F_i^{-1}E_1F_i, & F_i^{-1}E_2F_i, & \dots & F_i^{-1}E_rF_i, \\ \dots & \dots & \dots & \dots \\ F_i^{-1}E_1F_d, & F_i^{-1}E_2F_d, & \dots & F_i^{-1}E_rF_d, \end{matrix} >$$

Decoding LRPC

Let F_i be the i^{th} element of a basis of F, let's considerer the vector space

$$F_i^{-1}S = < \begin{matrix} F_i^{-1}E_1F_1, & F_i^{-1}E_2F_1, & \dots & F_i^{-1}E_rF_1, \\ F_i^{-1}E_1F_2, & F_i^{-1}E_2F_2, & \dots & F_i^{-1}E_rF_2, \\ \dots & \dots & \dots & \dots \\ E_1, & E_2, & \dots & E_r, \\ \dots & \dots & \dots & \dots \\ F_i^{-1}E_1F_d, & F_i^{-1}E_2F_d, & \dots & F_i^{-1}E_rF_d, \end{matrix} >$$

Decoding LRPC

Let F_i be the i^{th} element of a basis of F, let's considerer the vector space

$$F_i^{-1}S = < \begin{matrix} F_i^{-1}E_1F_1, & F_i^{-1}E_2F_1, & \dots & F_i^{-1}E_rF_1, \\ F_i^{-1}E_1F_2, & F_i^{-1}E_2F_2, & \dots & F_i^{-1}E_rF_2, \\ \dots & \dots & \dots & \dots \\ E_1, & E_2, & \dots & E_r, \\ \dots & \dots & \dots & \dots \\ F_i^{-1}E_1F_d, & F_i^{-1}E_2F_d, & \dots & F_i^{-1}E_rF_d, \end{matrix} >$$

So we have $E \subset F_i^{-1}S, \quad \forall\ i \in \{1, \dots, d\}$.

Decoding LRPC

- $E \subset F_i^{-1}S := S_i \quad \forall\, i \in \{1, \ldots, d\}$

Decoding LRPC

- $E \subset F_i^{-1}S := S_i \quad \forall\, i \in \{1, \ldots, d\}$
- The co-space of E in $F_i^{-1}S$ is very likely to be different for $i' \neq i$.

Decoding LRPC

- $E \subset F_i^{-1}S := S_i \quad \forall\, i \in \{1, \ldots, d\}$
- The co-space of E in $F_i^{-1}S$ is very likely to be different for $i' \neq i$.
- And so the key to the decoding algorithm of LRPC is that (with high probability)

$$\bigcap_{i=1}^{d} S_i = E$$

Decoding LRPC

- $E \subset F_i^{-1}S := S_i \quad \forall\, i \in \{1, \ldots, d\}$
- The co-space of E in $F_i^{-1}S$ is very likely to be different for $i' \neq i$.
- And so the key to the decoding algorithm of LRPC is that (with high probability)

$$\bigcap_{i=1}^{d} S_i = E$$

- Knowing the support of the error, one has to solve a linear system with more equations than unknowns to find the error and so to decode y.

Decoding LRPC

- This decoding is probabilistic, but the failure probability decreases exponentially according to the parameters.

Decoding LRPC

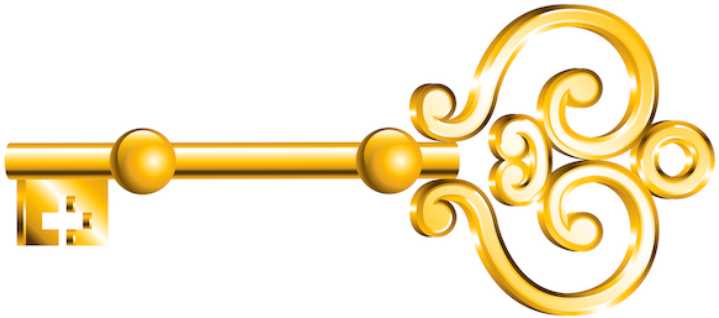
- This decoding is probabilistic, but the failure probability decreases exponentially according to the parameters.
- For instance, with parameters $[n = 94, k = 47, N = 47, d = r = 5]$ for a binary LRPC, $\mathbb{P}(\textit{failure}) = 2^{-23}$.

Asymmetric Cryptography

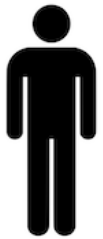
Asymmetric Cryptography (or Public Key Cryptography)

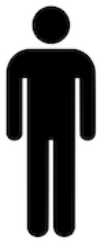
Public Key

Private Key



Asymmetric Cryptography

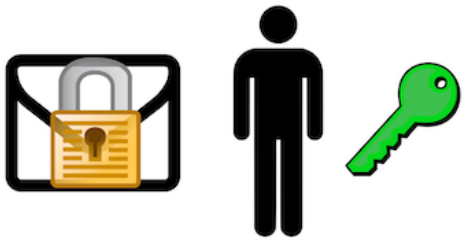




Asymmetric Cryptography

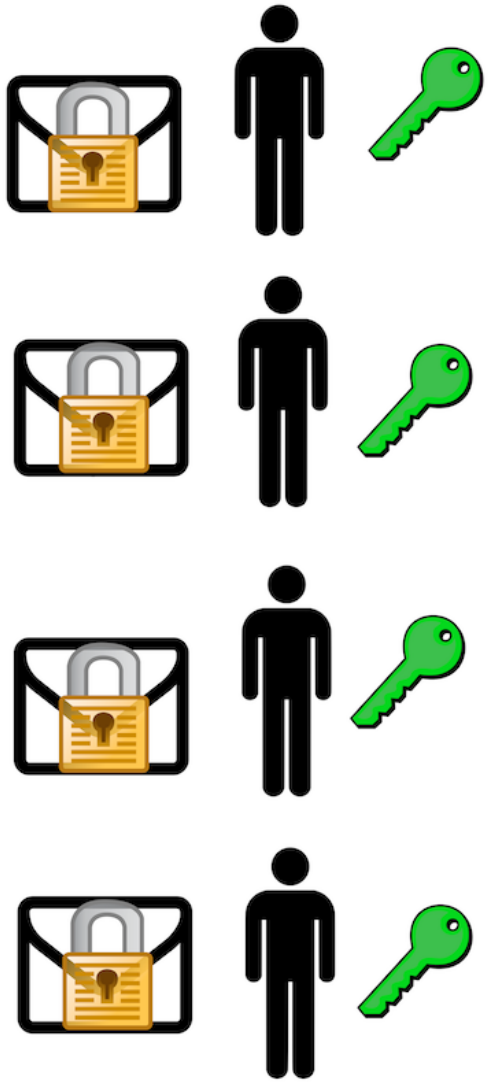
Asymmetric Cryptography

Asymmetric Cryptography

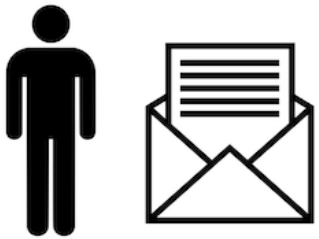


Asymmetric Cryptography

Asymmetric Cryptography



Asymmetric Cryptography



McEliece cryptosystem

Robert McEliece (1942 - 2019) started code-based cryptography in 1978.

McEliece cryptosystem

Robert McEliece (1942 - 2019) started code-based cryptography in 1978.

McEliece cryptosystem

McEliece cryptosystem (or setting)

Let $\mathcal{C}$ be a linear code $[n, k, d]$-code, decodable up to $t = (d-1)/2$ errors in polynomial time with an algorithm $\mathcal{D}$.

Private Key : the generator matrix G of $\mathcal{C}$ (usually in systematic form), S a non-singular $k \times k$ matrix and P a $n \times n$ permutation matrix.
Public Key : $G' = SGP$

Encryption : let m be the plaintext (of length k), the cipher text is $y = mG' + e$ with $w(e) \leq t$ (e random).
Decryption :

- Compute $yP^{-1} = (mS)G + eP^{-1}$
- Decode $mSG = \mathcal{D}(yP^{-1})$
- Recover mS and then m (using S^{-1})

Application to Cryptography : hiding the structure

- Attack : recovering the plaintext
  $\Longrightarrow$ Decoding Random Code Problem **under the assumption that $\mathcal{C}'$ is indistinguishable from a random code**.

Application to Cryptography : hiding the structure

- Attack : recovering the plaintext
  $\Longrightarrow$ Decoding Random Code Problem **under the assumption that $\mathcal{C}'$ is indistinguishable from a random code**.
- This is not the case if $\mathcal{C}$ has a "strong" algebraic structure (Reed-Solomon or Gabidulin for instance).

Application to Cryptography : hiding the structure

- Attack : recovering the plaintext
  $\Longrightarrow$ Decoding Random Code Problem **under the assumption that $\mathcal{C}'$ is indistinguishable from a random code**.
- This is not the case if $\mathcal{C}$ has a "strong" algebraic structure (Reed-Solomon or Gabidulin for instance).
- One wants to get rid of the "scrambling" step.

MDPC cryptosystem

New variant of McEliece cryptosystem [MTSB13] et [GMRZ13]

Let $\mathcal{C}$ be an **MDPC** $[n, k, d]$-code, decodable up to $t = (d-1)/2$ errors in polynomial time with a **probabilistic** decoding algorithm $\mathcal{D}_H$ (using its sparse parity check matrix $H = (h_0|h_1)$).

Private Key : the parity check matrix H of $\mathcal{C}$
Public Key : the generator matrix $G = (I|(h_0 h_1^{-1})^T)$ of $\mathcal{C}$ (systematic form).

Encryption : let m be the plaintext (of length k), the cipher text is $y = mG + e$ with $w(e) \leq t$ (e random).
Decryption :

- Decode $mG = \mathcal{D}_H(y)$
- Recover m (extracting the first k components of mG)

MDPC cryptosystem

New variant of McEliece cryptosystem [MTSB13] et [GMRZ13]

Let $\mathcal{C}$ be an **MDPC** $[n, k, d]$-code, decodable up to $t = (d-1)/2$ errors in polynomial time with a **probabilistic** decoding algorithm $\mathcal{D}_H$ (using its sparse parity check matrix $H = (h_0|h_1)$).

Private Key : the parity check matrix H of $\mathcal{C}$
Public Key : the generator matrix $G = (I|(h_0 h_1^{-1})^T)$ of $\mathcal{C}$ (systematic form).

Encryption : let m be the plaintext (of length k), the cipher text is $y = mG + e$ with $w(e) \leq t$ (e random).
Decryption :

- Decode $mG = \mathcal{D}_H(y)$
- Recover m (extracting the first k components of mG)

Note : LRPC cryptosystem described in 2013 by Gaborit, Murat, Ruatta and Zémor is similar but uses rank metric and the LRPC decoding previously described.

MDPC cryptosystem

New variant of McEliece cryptosystem [MTSB13] et [GMRZ13]

Let $\mathcal{C}$ be an **MDPC** $[n, k, d]$-code, decodable up to $t = (d-1)/2$ errors in polynomial time with a **probabilistic** decoding algorithm $\mathcal{D}_H$ (using its sparse parity check matrix $H = (h_0|h_1)$).

Private Key : the parity check matrix H of $\mathcal{C}$
Public Key : the generator matrix $G = (I|(h_0 h_1^{-1})^T)$ of $\mathcal{C}$ (systematic form).

Encryption : let m be the plaintext (of length k), the cipher text is $y = mG + e$ with $w(e) \leq t$ (e random).
Decryption :

- Decode $mG = \mathcal{D}_H(y)$
- Recover m (extracting the first k components of mG)

Note : LRPC cryptosystem described in 2013 by Gaborit, Murat, Ruatta and Zémor is similar but uses rank metric and the LRPC decoding previously described.
Disclaimer : this system has to be modified to be used in practice, since the systematic form would lead to leaks of the original message.

Hiding the structure

Advantages of MDPC and LRPC :

Hiding the structure

Advantages of MDPC and LRPC :

- Almost no algebraic structure.
  Thus the security doesn't rely on masking the structure but only on the hardness of decoding random codes **under the assumption of indistinguability of MDPC and LRPC**.

Hiding the structure

Advantages of MDPC and LRPC :

- Almost no algebraic structure.
  Thus the security doesn't rely on masking the structure but only on the hardness of decoding random codes **under the assumption of indistinguability of MDPC and LRPC**.
- Thus, the "scrambling" step is replaced by going from the **sparse** H **to the dense** G.

Hiding the structure

Few visual examples : M of size 20×20,
coefficients in $GF(2^{20}) \approx 1$ million elements, $dim(F) = 2$.

```
##################
Matrice M=
[ w^362254         0  w^362254  w^362254         0  w^362254  w^471106  w^362254  w^362254         0         0  w^362254  w^362254         0  w^362254  w^733990  w^733990  w^471106         0         0]
[ w^362254  w^733990  w^471106         0  w^733990  w^362254  w^362254  w^362254  w^471106  w^733990         0  w^362254  w^733990  w^362254  w^733990  w^471106  w^471106         0  w^362254  w^471106]
[         0         0  w^471106  w^733990         0  w^733990  w^733990         0  w^733990  w^471106         0  w^362254         0  w^471106         0  w^362254  w^733990  w^733990  w^471106         0]
[ w^471106  w^471106  w^733990  w^471106  w^362254  w^733990  w^362254         0  w^471106         0  w^733990  w^362254  w^471106         0  w^733990  w^471106  w^471106         0  w^362254  w^362254]
[         0  w^362254  w^471106         0  w^471106  w^733990  w^733990  w^362254  w^471106  w^362254  w^471106         0  w^471106         0  w^362254         0  w^733990  w^471106  w^733990  w^362254]
[ w^362254  w^362254         0  w^471106  w^733990         0  w^362254  w^471106  w^733990  w^471106         0  w^733990         0  w^471106  w^733990         0  w^733990         0  w^733990  w^733990]
[ w^362254  w^733990         0  w^471106  w^471106         0  w^733990  w^733990  w^362254  w^471106  w^733990         0  w^362254  w^471106  w^362254  w^471106  w^733990         0  w^362254]
[ w^733990  w^362254         0  w^471106         0         0         0  w^471106  w^471106         0         0  w^733990  w^733990  w^471106  w^362254         0  w^471106         0         0  w^362254]
[ w^733990         0  w^733990  w^471106         0         0  w^471106  w^471106         0         0  w^362254  w^362254  w^471106  w^362254         0  w^362254         0         0  w^471106  w^733990]
[         0  w^733990  w^471106         0  w^362254  w^471106         0         0         0         0  w^471106  w^471106         0         0  w^362254  w^362254  w^733990         0         0]
[ w^471106         0  w^471106  w^733990  w^471106  w^733990  w^733990  w^471106  w^362254  w^733990         0  w^733990  w^733990         0  w^733990         0  w^471106         0  w^471106         0]
[ w^471106  w^733990  w^733990  w^733990  w^733990  w^362254  w^362254  w^471106  w^733990  w^362254  w^362254         0  w^733990         0  w^733990  w^362254  w^733990  w^471106  w^362254  w^733990]
[         0  w^733990  w^471106  w^362254  w^733990  w^471106  w^733990         0  w^471106  w^471106  w^733990         0         0  w^362254  w^362254  w^733990  w^733990         0  w^471106  w^362254]
[         0  w^471106         0  w^471106  w^733990         0  w^362254  w^362254  w^733990         0  w^471106  w^733990  w^471106  w^362254  w^471106  w^471106         0  w^733990  w^362254  w^733990]
[ w^471106  w^471106  w^471106         0  w^471106  w^471106  w^471106  w^471106  w^733990  w^733990  w^362254  w^471106  w^733990  w^471106  w^362254  w^733990         0  w^362254  w^362254  w^362254]
[ w^471106  w^471106  w^733990  w^733990  w^471106  w^362254  w^471106  w^471106  w^362254  w^733990  w^362254  w^733990  w^362254  w^362254         0  w^733990  w^471106  w^471106         0  w^362254]
[ w^362254  w^362254  w^362254  w^733990  w^471106  w^733990         0  w^471106  w^362254  w^471106  w^362254  w^733990  w^471106  w^471106         0  w^471106  w^471106  w^362254  w^471106  w^362254]
[         0         0         0  w^362254         0  w^471106  w^362254  w^471106  w^733990         0  w^362254  w^362254  w^362254  w^362254  w^733990         0  w^362254  w^471106  w^733990  w^733990]
[ w^362254  w^733990  w^733990         0         0         0  w^733990         0         0  w^733990  w^362254         0         0  w^362254         0         0  w^733990  w^733990  w^733990  w^471106]
[ w^733990  w^471106         0  w^733990  w^471106  w^471106         0  w^733990  w^362254  w^362254         0  w^733990  w^733990  w^733990         0  w^362254  w^733990         0  w^362254  w^362254]
##################
Matrice M^(-1)=
[  w^80448  w^988123  w^863025  w^898705  w^139452  w^623210  w^594037  w^864752  w^105343   w^28264 w^1045033  w^888787  w^617636  w^265974  w^169205   w^94741  w^801759   w^52909  w^607058  w^699552]
[ w^333957  w^942554  w^780678  w^618924 w^1000929  w^440446  w^964460  w^719946 w^1046953  w^959233  w^760287  w^824678  w^776259  w^356585  w^304173  w^637539  w^637281  w^111491  w^774659  w^902573]
[ w^147497  w^781692  w^115424  w^811426  w^649856  w^728662  w^156367  w^142081  w^106927  w^633380   w^63598  w^639272  w^220801  w^862548  w^417167  w^412288  w^982778  w^644957  w^218985  w^704658]
[ w^311167   w^23150  w^296538  w^567087  w^831522  w^221167  w^288209  w^468301  w^297189  w^764454  w^537310  w^720237   w^99700  w^695907  w^253634  w^691839  w^524141  w^551022  w^223229  w^875046]
[ w^993583   w^93992  w^478192  w^810565  w^141922  w^942306  w^117904  w^334543  w^819778    w^5759  w^855073  w^181661  w^888342  w^243147  w^447472  w^465924  w^763151  w^897715  w^278322  w^361146]
[ w^841810  w^756762  w^127936  w^273301  w^547058  w^172200  w^340363  w^344845  w^263618   w^13923  w^595539   w^57782  w^614572  w^312900 w^1005808  w^757103  w^536479  w^283298  w^351206  w^195139]
[ w^590967  w^706720  w^225921  w^381198  w^114839   w^44222  w^812786  w^910129  w^804021  w^155814  w^408812  w^545490  w^158519  w^588939  w^795331  w^495753  w^517364  w^930985  w^340051  w^443794]
[ w^334420  w^692708  w^546219  w^342348  w^443793  w^163160   w^31191  w^596068   w^44804  w^933333  w^234422  w^977827  w^797738  w^777330  w^244160  w^826510  w^425208  w^916518 w^1010002   w^28974]
[ w^727007  w^537798  w^387187  w^622828   w^90354  w^175592  w^578483  w^758515  w^624990 w^1021246  w^715397  w^345528  w^915186  w^455894  w^868470  w^886395  w^913260  w^347761   w^32665   w^99167]
[ w^262635  w^384261  w^266480  w^587969  w^540688  w^268454  w^260662  w^767743  w^566267  w^201909 w^1009597  w^577204  w^549731  w^230250  w^522123  w^491763  w^994584  w^766994  w^652170   w^19346]
[ w^787261  w^787284  w^577188  w^944193   w^76009   w^72114  w^168346  w^188787 w^1007068   w^78632  w^543358  w^734432   w^33820  w^617557  w^755605  w^123401  w^848074  w^717020  w^239070  w^458273]
[ w^653534  w^254962  w^613708 w^1029328  w^731945  w^737731  w^257473  w^402995  w^202381  w^918616  w^295557  w^889336  w^351942  w^380277  w^125269  w^100947   w^12813   w^69492  w^816486 w^1003168]
[ w^820747  w^472469  w^375392  w^928977  w^812964  w^318570  w^966941  w^935842 w^1027964  w^619944  w^850971  w^296038  w^603907  w^423541  w^104752  w^654707  w^379296  w^732234  w^948721  w^361762]
[ w^110971  w^680696  w^923340  w^214480  w^747262  w^926785  w^489888  w^952228  w^445776  w^423943  w^649072   w^22829  w^356036  w^706505  w^939812  w^920048  w^394764 w^1024427  w^670614  w^723510]
[ w^625737  w^796362  w^516193  w^823319  w^407960  w^792252  w^704791  w^442520  w^954309  w^636815  w^499427  w^651800  w^325780  w^982706  w^259759  w^704340  w^738285  w^423883  w^788285  w^864859]
[ w^637454 w^1005277  w^634523  w^597556  w^899547   w^50212  w^771166  w^176848  w^739499 w^1012145  w^894991  w^297408  w^401139  w^445943  w^860647  w^905019  w^586967  w^555279   w^11908  w^393060]
[ w^491731  w^105205  w^838924  w^523104  w^133971  w^621235  w^376245  w^675466  w^253911  w^239901 w^1012958  w^945009  w^722834 w^1047441  w^191877  w^127222  w^163984  w^206531 w^1005897  w^628569]
[ w^525867  w^931830  w^738973  w^215016  w^658662  w^195917  w^891334  w^903857  w^441472  w^344280  w^417181  w^577925  w^705133   w^41732  w^482903  w^837857  w^465912  w^268532  w^478086  w^988553]
[  w^93536  w^751673  w^536066  w^729970  w^123721  w^756989  w^668630  w^636987  w^454234   w^51525  w^574024  w^308632  w^461191  w^137995  w^836411  w^667454  w^403991  w^906525  w^553448  w^160312]
[ w^269807  w^696612  w^755090  w^178200  w^828283  w^733597  w^859384  w^569348  w^146891  w^116635 w^1004028  w^759077  w^155720  w^809492  w^303886 w^1014978  w^790097  w^325774  w^228381  w^278762]
##################
```

Hiding the structure

Few visual examples : M of size 20×20,
coefficients in $GF(2^{20}) \approx 1$ million elements, $dim(F) = 2$.
$dim(support(M^{-1})) = 19$.

```
##################
Matrice M=
[ w^362254        0  w^362254  w^362254        0  w^362254  w^471106  w^362254  w^362254        0        0  w^362254  w^362254        0  w^362254  w^733990  w^733990  w^471106        0        0]
[ w^362254  w^733990  w^471106        0  w^733990  w^362254  w^362254  w^362254  w^471106  w^733990        0  w^362254  w^733990  w^362254  w^733990  w^471106  w^471106        0  w^362254  w^471106]
[       0        0  w^471106  w^733990        0  w^733990  w^733990        0  w^733990  w^471106        0  w^362254        0  w^471106        0  w^362254  w^733990  w^733990  w^471106        0]
[ w^471106  w^471106  w^733990  w^471106  w^362254  w^733990  w^362254        0  w^471106        0  w^733990  w^362254  w^471106        0  w^733990  w^471106  w^471106        0  w^362254  w^362254]
[       0  w^362254  w^471106        0  w^471106  w^733990  w^733990  w^362254  w^471106  w^362254  w^471106        0  w^471106        0  w^362254        0  w^733990  w^471106  w^733990  w^362254]
[ w^362254  w^362254        0  w^471106  w^733990        0  w^362254  w^471106  w^733990  w^471106        0  w^733990        0  w^471106  w^733990        0  w^733990        0  w^733990  w^733990]
[ w^362254  w^733990        0  w^471106  w^471106        0  w^733990  w^362254  w^471106  w^733990        0  w^362254  w^471106  w^362254  w^471106  w^733990        0  w^362254]
[ w^733990  w^362254        0  w^471106        0        0        0  w^471106  w^471106        0        0  w^733990  w^733990  w^471106  w^362254        0  w^471106        0        0  w^362254]
[ w^733990        0  w^733990  w^471106        0        0  w^471106  w^471106        0        0  w^362254  w^362254  w^471106  w^362254        0  w^362254        0        0  w^471106  w^733990]
[       0  w^733990  w^471106        0  w^362254  w^471106        0        0        0        0  w^471106  w^471106        0        0  w^362254  w^362254  w^733990        0        0]
[ w^471106        0  w^471106  w^733990  w^471106  w^733990  w^733990  w^471106  w^362254  w^733990        0  w^733990  w^733990        0  w^733990        0  w^471106        0  w^471106        0]
[ w^471106  w^733990  w^733990  w^733990  w^733990  w^362254  w^362254  w^471106  w^733990  w^362254  w^362254        0  w^733990        0  w^733990  w^362254  w^733990  w^471106  w^362254  w^733990]
[       0  w^733990  w^471106  w^362254  w^733990  w^471106  w^733990        0  w^471106  w^471106  w^733990        0        0  w^362254  w^362254  w^733990  w^733990        0  w^471106  w^362254]
[       0  w^471106        0  w^471106  w^733990        0  w^362254  w^362254  w^733990        0  w^471106  w^733990  w^471106  w^362254  w^471106  w^471106        0  w^733990  w^362254  w^733990]
[ w^471106  w^471106  w^471106        0  w^471106  w^471106  w^471106  w^471106  w^733990  w^733990  w^362254  w^471106  w^733990  w^471106  w^362254  w^733990        0  w^362254  w^362254  w^362254]
[ w^471106  w^471106  w^733990  w^733990  w^471106  w^362254  w^471106  w^471106  w^362254  w^733990  w^362254  w^733990  w^362254  w^362254        0  w^733990  w^471106  w^471106        0  w^362254]
[ w^362254  w^362254  w^362254  w^733990  w^471106  w^733990        0  w^471106  w^362254  w^471106  w^362254  w^733990  w^471106  w^471106        0  w^471106  w^471106  w^362254  w^471106  w^362254]
[       0        0        0  w^362254        0  w^471106  w^362254  w^471106  w^733990        0  w^362254  w^362254  w^362254  w^362254  w^733990        0  w^362254  w^471106  w^733990  w^733990]
[ w^362254  w^733990  w^733990        0        0        0  w^733990        0        0  w^733990  w^362254        0        0  w^362254        0        0  w^733990  w^733990  w^733990  w^471106]
[ w^733990  w^471106        0  w^733990  w^471106  w^471106        0  w^733990  w^362254  w^362254        0  w^733990  w^733990  w^733990        0  w^362254  w^733990        0  w^362254  w^362254]
##################
Matrice M^(-1)=
[  w^80448  w^988123  w^863025  w^898705  w^139452  w^623210  w^594037  w^864752  w^105343   w^28264 w^1045033  w^888787  w^617636  w^265974  w^169205   w^94741  w^801759   w^52909  w^607058  w^699552]
[ w^333957  w^942554  w^780678  w^618924 w^1000929  w^440446  w^964460  w^719946 w^1046953  w^959233  w^760287  w^824678  w^776259  w^356585  w^304173  w^637539  w^637281  w^111491  w^774659  w^902573]
[ w^147497  w^781692  w^115424  w^811426  w^649856  w^728662  w^156367  w^142081  w^106927  w^633380   w^63598  w^639272  w^220801  w^862548  w^417167  w^412288  w^982778  w^644957  w^218985  w^704658]
[ w^311167   w^23150  w^296538  w^567087  w^831522  w^221167  w^288209  w^468301  w^297189  w^764454  w^537310  w^720237   w^99700  w^695907  w^253634  w^691839  w^524141  w^551022  w^223229  w^875046]
[ w^993583   w^93992  w^478192  w^810565  w^141922  w^942306  w^117904  w^334543  w^819778    w^5759  w^855073  w^181661  w^888342  w^243147  w^447472  w^465924  w^763151  w^897715  w^278322  w^361146]
[ w^841810  w^756762  w^127936  w^273301  w^547058  w^172200  w^340363  w^344845  w^263618   w^13923  w^595539   w^57782  w^614572  w^312900 w^1005808  w^757103  w^536479  w^283298  w^351206  w^195139]
[ w^590967  w^706720  w^225921  w^381198  w^114839   w^44222  w^812786  w^910129  w^804021  w^155814  w^408812  w^545490  w^158519  w^588939  w^795331  w^517364  w^930985  w^340051  w^443794]
[ w^334420  w^692708  w^546219  w^342348  w^443793  w^163160   w^31191  w^596068   w^44804  w^933333  w^234422  w^977827  w^797738  w^777330  w^244160  w^826510  w^425208  w^916518 w^1010002   w^28974]
[ w^727007  w^537798  w^387187  w^622828   w^90354  w^175592  w^578483  w^758515  w^624990 w^1021246  w^715397  w^345528  w^915186  w^455894  w^868470  w^886395  w^913260  w^347761   w^32665   w^99167]
[ w^262635  w^384261  w^266480  w^587969  w^540688  w^268454  w^260662  w^767743  w^566267  w^201909 w^1009597  w^577204  w^549731  w^230250  w^522123  w^491763  w^994584  w^766994  w^652170   w^19346]
[ w^787261  w^787284  w^577188  w^944193   w^76009   w^72114  w^168346  w^188787 w^1007068   w^78632  w^543358  w^734432   w^33820  w^617557  w^755605  w^123401  w^848074  w^717020  w^239070  w^458273]
[ w^653534  w^254962  w^613708 w^1029328  w^731945  w^737731  w^257473  w^402995  w^202381  w^918616  w^295557  w^889336  w^351942  w^380277  w^125269  w^100947   w^12813   w^69492  w^816486 w^1003168]
[ w^820747  w^472469  w^375392  w^928977  w^812964  w^318570  w^966941  w^935842 w^1027964  w^619944  w^850971  w^296038  w^603907  w^423541  w^104752  w^654707  w^379296  w^732234  w^948721  w^361762]
[ w^110971  w^680696  w^923340  w^214480  w^747262  w^926785  w^489888  w^952228  w^445776  w^423943  w^649072   w^22829  w^356036  w^706505  w^939812  w^920048  w^394764 w^1024427  w^670614  w^723510]
[ w^625737  w^796362  w^516193  w^823319  w^407960  w^792252  w^704791  w^442520  w^954309  w^636815  w^499427  w^651800  w^325780  w^982706  w^259759  w^704340  w^738285  w^423883  w^788285  w^864859]
[ w^637454 w^1005277  w^634523  w^597556  w^899547   w^50212  w^771166  w^176848  w^739499 w^1012145  w^894991  w^297408  w^401139  w^445943  w^860647  w^905019  w^586967  w^555279   w^11908  w^393060]
[ w^491731  w^105205  w^838924  w^523104  w^133971  w^621235  w^376245  w^675466  w^253911  w^239901 w^1012958  w^945009  w^722834 w^1047441  w^191877  w^127222  w^163984  w^206531 w^1005897  w^628569]
[ w^525867  w^931830  w^738973  w^215016  w^658662  w^195917  w^891334  w^903857  w^441472  w^344280  w^417181  w^577925  w^705133   w^41732  w^482903  w^837857  w^465912  w^268532  w^478086  w^988553]
[  w^93536  w^751673  w^536066  w^729970  w^123721  w^756989  w^668630  w^636987  w^454234   w^51525  w^574024  w^308632  w^461191  w^137995  w^836411  w^667454  w^403991  w^906525  w^553448  w^160312]
[ w^269807  w^696612  w^755090  w^178200  w^828283  w^733597  w^859384  w^569348  w^146891  w^116635 w^1004028  w^759077  w^155720  w^809492  w^303886 w^1014978  w^790097  w^325774  w^228381  w^278762]
##################
```

Hiding the structure

```
mathis-130:beamer bros$ magma
Magma V2.23-1 (STUDENT)   Wed May 22 2019 10:06:37    [Seed = 1608908847]
Type ? for help.  Type <Ctrl>-D to quit.
> load "H.magma";
Loading "H.magma"
n =
250
r =
10
affichage =
0
Trial 1          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 2          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 3          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 4          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 5          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 6          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 7          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 8          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 9          => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
Trial 10         => dimension support(M) = 10, dimension support(M^(-1)) = 250 (max=250)
```

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)
- But a sparse parity check matrix is enough :
  $\Longrightarrow$**the sparser its parity check matrix is, the more a code can decode.**

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)
- But a sparse parity check matrix is enough :
  $\Longrightarrow$**the sparser its parity check matrix is, the more a code can decode.**
- For instance, LDPC (H with $\mathcal{O}(1)$ $1'$ per row) can decode up to $\mathcal{O}(n)$ errors.

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)
- But a sparse parity check matrix is enough :
  $\Longrightarrow$**the sparser its parity check matrix is, the more a code can decode.**
- For instance, LDPC (H with $\mathcal{O}(1)$ $1'$ per row) can decode up to $\mathcal{O}(n)$ errors.
- **Nevertheless, in cryptography, we prefer to use MDPC codes :**

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)
- But a sparse parity check matrix is enough :
  $\Longrightarrow$**the sparser its parity check matrix is, the more a code can decode.**
- For instance, LDPC (H with $\mathcal{O}(1)$ $1'$ per row) can decode up to $\mathcal{O}(n)$ errors.
- **Nevertheless, in cryptography, we prefer to use MDPC codes :**
  - We do not want to correct errors but to increase the complexity of attacks.

Conclusion about sparsity and coding theory

Conclusion about sparsity and coding theory :

- One could think that the more structured a code is, the more it can decode (Reed-Solomon, Reed-Muller, Gabidulin, ...)
- But a sparse parity check matrix is enough :
  $\Longrightarrow$**the sparser its parity check matrix is, the more a code can decode.**
- For instance, LDPC (H with $\mathcal{O}(1)$ $1'$ per row) can decode up to $\mathcal{O}(n)$ errors.
- **Nevertheless, in cryptography, we prefer to use MDPC codes :**
  - We do not want to correct errors but to increase the complexity of attacks.
  - For instance, attacks on LDPC codes would be too easy due to a small private key size.

Algebraic Attack against RSD

As seen previously, the security of rank-based cryptosystems relies on the RSD problem.
From now on, I will consider code over $GF(2^N)$.

Algebraic Attack against RSD

As seen previously, the security of rank-based cryptosystems relies on the RSD problem.
From now on, I will consider code over $GF(2^N)$.

- From $He^T = s$ one gets a system of equations.

Algebraic Attack against RSD

As seen previously, the security of rank-based cryptosystems relies on the RSD problem.
From now on, I will consider code over $GF(2^N)$.

- From $He^T = s$ one gets a system of equations.
- Very particular form : **Quadratic, quasi-bihomogeneous**

Algebraic Attack against RSD

As seen previously, the security of rank-based cryptosystems relies on the RSD problem.
From now on, I will consider code over $GF(2^N)$.

- From $He^T = s$ one gets a system of equations.
- Very particular form : **Quadratic, quasi-bihomogeneous**

```
E_1_5*e_4_1 + E_1_1*e_5_1 + E_1_2*e_5_
E_1_2*e_4_1 + E_1_4*e_4_1 + E_1_2*e_5_
E_1_1*e_4_1 + E_1_3*e_4_1 + E_1_5*e_4_
E_1_6*e_4_1 + E_1_6*e_5_1 + 1,
E_1_1*e_4_1 + E_1_6*e_4_1 + E_1_1*e_5_
E_1_2*e_4_1 + E_1_2*e_5_1 + 1,
E_1_3*e_4_1 + E_1_6*e_4_1 + E_1_3*e_5_
E_1_4*e_4_1 + E_1_6*e_4_1 + E_1_4*e_5_
E_1_5*e_4_1 + E_1_5*e_5_1,
E_1_2*e_4_1 + E_1_3*e_4_1 + E_1_6*e_4_
E_1_2*e_4_1 + E_1_4*e_4_1 + E_1_6*e_4_
E_1_3*e_4_1 + E_1_5*e_4_1 + E_1_1*e_5_
```

Algebraic Attack against RSD

- Take advantage of the small rank of e
  $\Longrightarrow$ **over-constrained** system of equations

Algebraic Attack against RSD

- Take advantage of the small rank of e
  $\Longrightarrow$ **over-constrained** system of equations
- Number of unknowns : Nr+nr. Nr for the basis of E, nr for e.

Algebraic Attack against RSD

- Take advantage of the small rank of e
  $\Longrightarrow$ **over-constrained** system of equations
- Number of unknowns : Nr+nr. Nr for the basis of E, nr for e.
- Number of equations : $(n-k)$ equations in $GF(2^N)$, so $N(n-k)$ equations.

Algebraic Attack against RSD

- Take advantage of the small rank of e
  $\Longrightarrow$ **over-constrained** system of equations
- Number of unknowns : Nr+nr. Nr for the basis of E, nr for e.
- Number of equations : $(n-k)$ equations in $GF(2^N)$, so $N(n-k)$ equations.
- Usually $n = N, k = n/2$ and $r = \mathcal{O}(\sqrt{n})$, so $2n^{3/2}$ unknowns for $(1/2)n^2$ equations.

Algebraic Attack against RSD

- Take advantage of the small rank of e
  $\Longrightarrow$ **over-constrained** system of equations
- Number of unknowns : Nr+nr. Nr for the basis of E, nr for e.
- Number of equations : $(n-k)$ equations in $GF(2^N)$, so $N(n-k)$ equations.
- Usually $n = N, k = n/2$ and $r = \mathcal{O}(\sqrt{n})$, so $2n^{3/2}$ unknowns for $(1/2)n^2$ equations.
- **One way to solve this system is to use Gröbner basis computation, but their complexity is poorly known for system which are not semi-regular.**

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).
- But in our case, the system is **over-constrainted** and it has a strong structure coming from the structure of $GF(2^N)$.

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).
- But in our case, the system is **over-constrainted** and it has a strong structure coming from the structure of $GF(2^N)$.
- **Nowadays, GB computation complexity for over-constrainted systems over $GF(2)$ is still an open question.**

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).
- But in our case, the system is **over-constrainted** and it has a strong structure coming from the structure of $GF(2^N)$.
- **Nowadays, GB computation complexity for over-constrainted systems over $GF(2)$ is still an open question.**
- Conjecture/Heuristic for the case "quadratically" over-constrainted.

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).
- But in our case, the system is **over-constrainted** and it has a strong structure coming from the structure of $GF(2^N)$.
- **Nowadays, GB computation complexity for over-constrainted systems over $GF(2)$ is still an open question.**
- Conjecture/Heuristic for the case "quadratically" over-constrainted.
- Observations on small RSD instances

Algebraic Attack against RSD

- More precisely, if the system were random with $\#\ \textit{unknowns} \approx \#\ \textit{equations}$, we would have some complexity bounds ([Bar04]).
- But in our case, the system is **over-constrainted** and it has a strong structure coming from the structure of $GF(2^N)$.
- **Nowadays, GB computation complexity for over-constrainted systems over $GF(2)$ is still an open question.**
- Conjecture/Heuristic for the case "quadratically" over-constrainted.
- Observations on small RSD instances
  $\Longrightarrow$ drop in complexity in comparison to random systems.

Algebraic Attack against RSD

$[unknowns, equations]$	d_{reg} *random system*	d_{reg} *for RSD systems*
$[48, 48]$	8 (*Bardet*)	$[8, 8, 2, 3] \Longrightarrow 5$
$[48, 48]$	8 (*Bardet*)	$[12, 12, 8, 2] \Longrightarrow 4$
$[58, 70]$	7 *or* 8 $(*)$	$[10, 10, 2, 4] \Longrightarrow 5$

$(*)$ 7 is a prediction using my conjecture for over-constrained system whereas 8 would be the result of Bardet's method for semi-regular system.

:-)

Merci !
Thank you !

References : [McE78], [MTSB13], [GMRZ13], [Bar04].

References

Magali Bardet.

Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.
Theses, Université Pierre et Marie Curie - Paris VI, 2004.

Philippe Gaborit, Gaétan Murat, Olivier Ruatta, and Gilles Zémor.
Low rank parity check codes and their application to cryptography.
In *Proceedings of the Workshop on Coding and Cryptography WCC*, volume 2013, 2013.

Robert J McEliece.
A public-key cryptosystem based on algebraic coding theory.
Coding Thv, 4244:114–116, 1978.

Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo SLM Barreto.
MDPC-McEliece: New McEliece variants from moderate density parity-check codes.
In *2013 IEEE international symposium on information theory*, pages 2069–2073. IEEE, 2013.