Des codes correcteurs pour le retrait confidentiel d'information

Julien Lavauzelle

équipe AGC3, LAGA, Université Paris 8 Vincennes – Saint-Denis

Séminaire d'ouverture aux informatiques et de formation

31/05/2022

Outline

Outline

Motivation

Accroissement de l'accès à des données stockées à distance :

- données publiques (cartographie, informations & évènements)
- données personnelles (services de stockage en ligne, données médicales)

Motivation

Accroissement de l'accès à des données stockées à distance :

- données publiques (cartographie, informations & évènements)
- données personnelles (services de stockage en ligne, données médicales)

Désir d'un accès **confidentiel à ces données**, par

- l'**anonymat** : l'identité de l'entité qui accède à l'information est inconnue

Motivation

Accroissement de l'accès à des données stockées à distance :

- données publiques (cartographie, informations & évènements)
- données personnelles (services de stockage en ligne, données médicales)

Désir d'un accès **confidentiel à ces données**, par

- l'**anonymat** : l'identité de l'entité qui accède à l'information est inconnue
- le **chiffrement** : la valeur de l'information accédée est inconnue

Motivation

Accroissement de l'accès à des données stockées à distance :

- données publiques (cartographie, informations & évènements)
- données personnelles (services de stockage en ligne, données médicales)

Désir d'un accès **confidentiel à ces données**, par

- l'**anonymat** : l'identité de l'entité qui accède à l'information est inconnue
- le **chiffrement** : la valeur de l'information accédée est inconnue
- le **retrait confidentiel** : l'identité de l'information accédée est inconnue

Un peu de formalisme

Retrait confidentiel d'information (*private information retrieval*, PIR) :

Soit une base de données distante $F \in \Sigma^M$ et un indice $i \in [1, M]$,
peut-on **extraire** l'entrée F_i de la base,
sans révéler aucune information sur l'indice i ?

Un peu de formalisme

Retrait confidentiel d'information (*private information retrieval*, PIR) :

Soit une base de données distante $F \in \Sigma^M$ et un indice $i \in [1, M]$,
peut-on **extraire** l'entrée F_i de la base,
sans révéler aucune information sur l'indice i ?

Une **solution triviale :** télécharger complètement la base de données.

PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

Un protocole de **retrait confidentiel d'information** est un ensemble de trois algorithmes $(\mathsf{Query}, \mathsf{Answer}, \mathsf{Reconstruct})$. Pour extraire F_i :

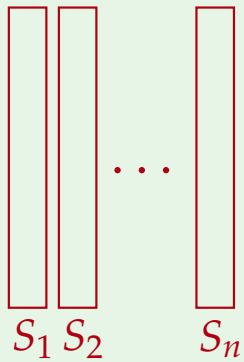
PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

Un protocole de **retrait confidentiel d'information** est un ensemble de trois algorithmes $(\mathsf{Query}, \mathsf{Answer}, \mathsf{Reconstruct})$. Pour extraire F_i :

User



PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

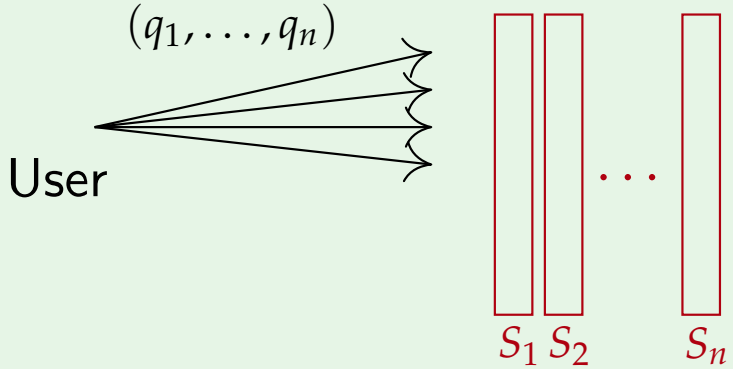
Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

Un protocole de **retrait confidentiel d'information** est un ensemble de trois algorithmes $(\mathsf{Query}, \mathsf{Answer}, \mathsf{Reconstruct})$. Pour extraire F_i :

1. User engendre un vecteur de requêtes

$$\boldsymbol{q} := (q_1, \ldots, q_n) \leftarrow \mathsf{Query}(i)$$

et envoie q_j au serveur S_j.



PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

Un protocole de **retrait confidentiel d'information** est un ensemble de trois algorithmes $(\mathsf{Query}, \mathsf{Answer}, \mathsf{Reconstruct})$. Pour extraire F_i :

1. User engendre un vecteur de requêtes

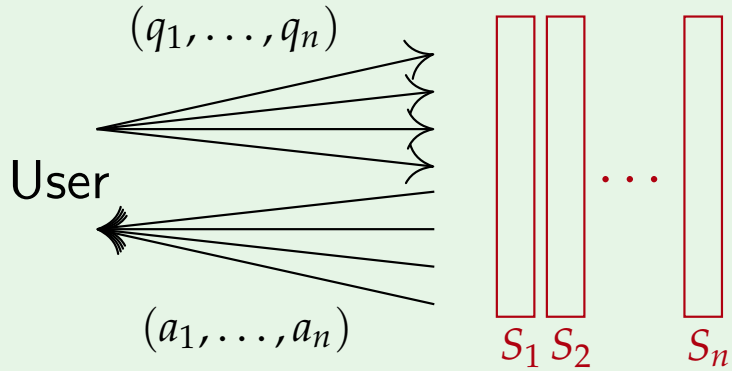
$$\boldsymbol{q} := (q_1, \ldots, q_n) \leftarrow \mathsf{Query}(i)$$

et envoie q_j au serveur S_j.

2. Chaque serveur S_j calcule

$$a_j := \mathsf{Answer}(q_j, F_{|S_j})$$

et retourne la valeur a_j à User.



PIR : définition

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Un vecteur d'entrées $F \in \Sigma^M$ est stocké sur n serveurs $S_1, \ldots, S_n$.
L'entité User désire extraire l'entrée F_i de manière confidentielle.

Un protocole de **retrait confidentiel d'information** est un ensemble de trois algorithmes $(\mathsf{Query}, \mathsf{Answer}, \mathsf{Reconstruct})$. Pour extraire F_i :

1. User engendre un vecteur de requêtes

   $$\boldsymbol{q} := (q_1, \ldots, q_n) \leftarrow \mathsf{Query}(i)$$

   et envoie q_j au serveur S_j.

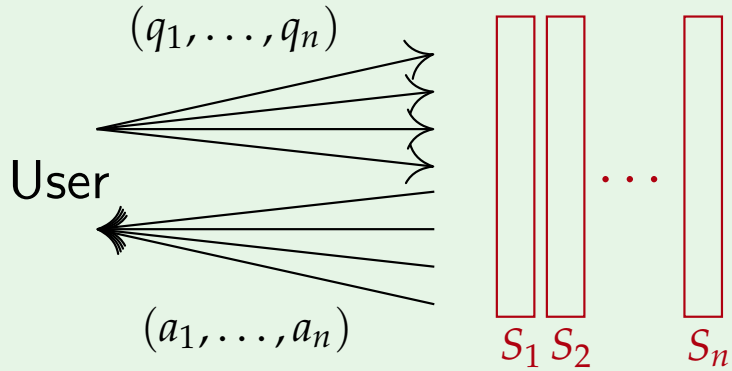
2. Chaque serveur S_j calcule

   $$a_j := \mathsf{Answer}(q_j, F_{|S_j})$$

   et retourne la valeur a_j à User.

3. User reconstitue l'information désirée par :

   $$F_i = \mathsf{Reconstruct}(\boldsymbol{q}, \boldsymbol{a}, i)\,.$$



Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Deux modèles de sécurité :

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Deux modèles de sécurité :

- **Au sens de la théorie de l'information (IT-PIR) :**

Si $q = \mathsf{Query}(i)$, on souhaite que :

$$\mathrm{I}(i\,; q_{|T}) = 0, \quad \forall T \subseteq [1, n], |T| \leq t.$$

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Deux modèles de sécurité :

- **Au sens de la théorie de l'information (IT-PIR) :**
Si $q = \mathsf{Query}(i)$, on souhaite que :

$$\mathrm{I}(i\,; q_{|T}) = 0, \quad \forall T \subseteq [1, n], |T| \leq t.$$

- **Au sens calculatoire (C-PIR) :**
À T fixé et i variable, les distributions de requêtes $q_{|T} = \mathsf{Query}(i)_{|T}$ sont calculatoirement indistinguables.

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Deux modèles de sécurité :

• **Au sens de la théorie de l'information (IT-PIR) :**
Si $q = \mathsf{Query}(i)$, on souhaite que :

$$\mathrm{I}(i\,; q_{|T}) = 0, \quad \forall T \subseteq [1, n], |T| \leq t.$$

• **Au sens calculatoire (C-PIR) :**
À T fixé et i variable, les distributions de requêtes $q_{|T} = \mathsf{Query}(i)_{|T}$ sont calculatoirement indistinguables.

Théorème [CGKS95, CG97]. Si $t = n$ (en particulier pour $n = 1$ serveur), alors IT-PIR implique de télécharger entièrement la base de données.

Quel modèle de sécurité ?

L'**adversaire** : des **coalitions de serveurs** = des sous-ensembles de $\{S_j : j \in T\}$, avec $T \subset [1, n]$, qui échangent de l'information sur les requêtes reçues.

On note t la taille de la plus grande coalition.

Deux modèles de sécurité :

- **Au sens de la théorie de l'information (IT-PIR) :**
Si $q = \mathsf{Query}(i)$, on souhaite que :

$$\mathrm{I}(i\,; q_{|T}) = 0, \quad \forall T \subseteq [1, n], |T| \leq t.$$

- **Au sens calculatoire (C-PIR) :**
À T fixé et i variable, les distributions de requêtes $q_{|T} = \mathsf{Query}(i)_{|T}$ sont calculatoirement indistinguables.

Théorème [CGKS95, CG97]. Si $t = n$ (en particulier pour $n = 1$ serveur), alors IT-PIR implique de télécharger entièrement la base de données.

Remarque. C-PIR est possible avec 1 serveur, mais reste trop coûteux en pratique.

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)
- coût de **stockage** supplémentaire

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)
- coût de **stockage** supplémentaire
- taille maximale des coalitions d'adversaires

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)
- coût de **stockage** supplémentaire
- taille maximale des coalitions d'adversaires

Pour différents **contextes**, suivant :

- si la base de données est **répliquée** ou **ré-encodée**

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)
- coût de **stockage** supplémentaire
- taille maximale des coalitions d'adversaires

Pour différents **contextes**, suivant :

- si la base de données est **répliquée** ou **ré-encodée**
- la présence de serveurs **sans réponse**, **malicieux**

Critères pour le meilleur du PIR

On s'intéresse à **IT-PIR** (donc avec $n \geq 2$ serveurs)

Paramètres à prendre en compte :

- complexité de **communication** (*up*load et *down*load)
- complexité de **calcul** (client et serveur)
- coût de **stockage** supplémentaire
- taille maximale des coalitions d'adversaires

Pour différents **contextes**, suivant :

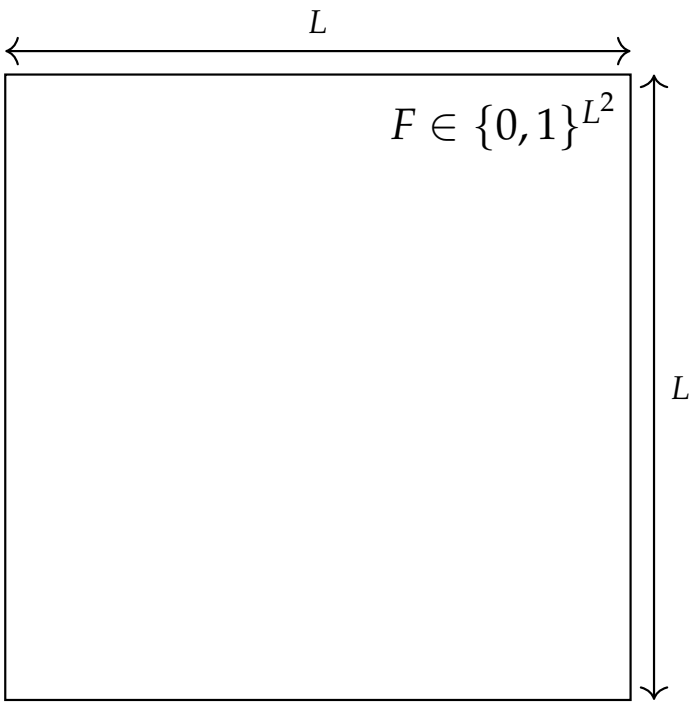
- si la base de données est **répliquée** ou **ré-encodée**
- la présence de serveurs **sans réponse**, **malicieux**
- la taille des entrées (clés vs. fichiers vidéos)
- le nombre d'entrées
- la statiticité (mise à jour, ajout d'entrées, etc.)

La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.



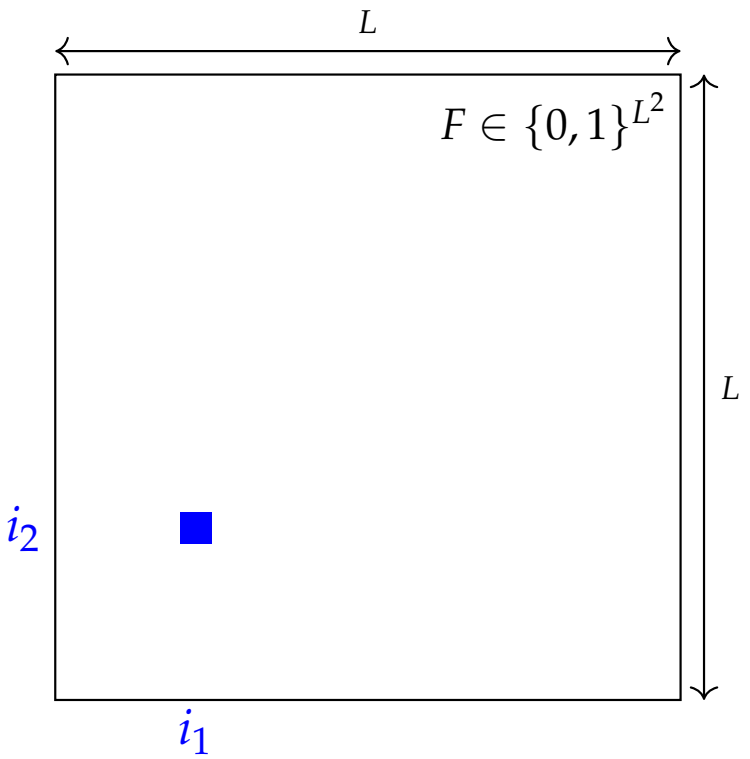
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



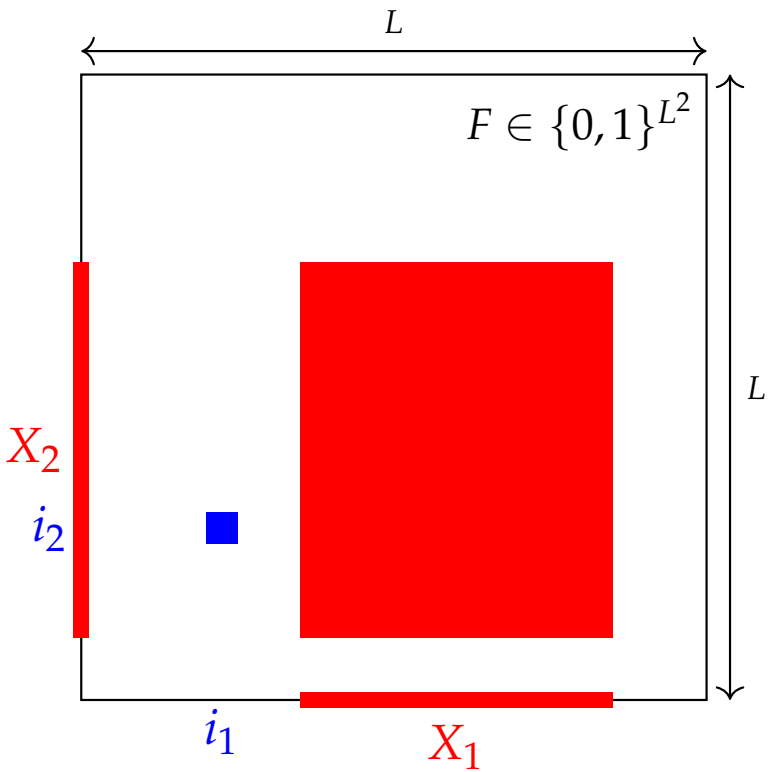
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1,i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :

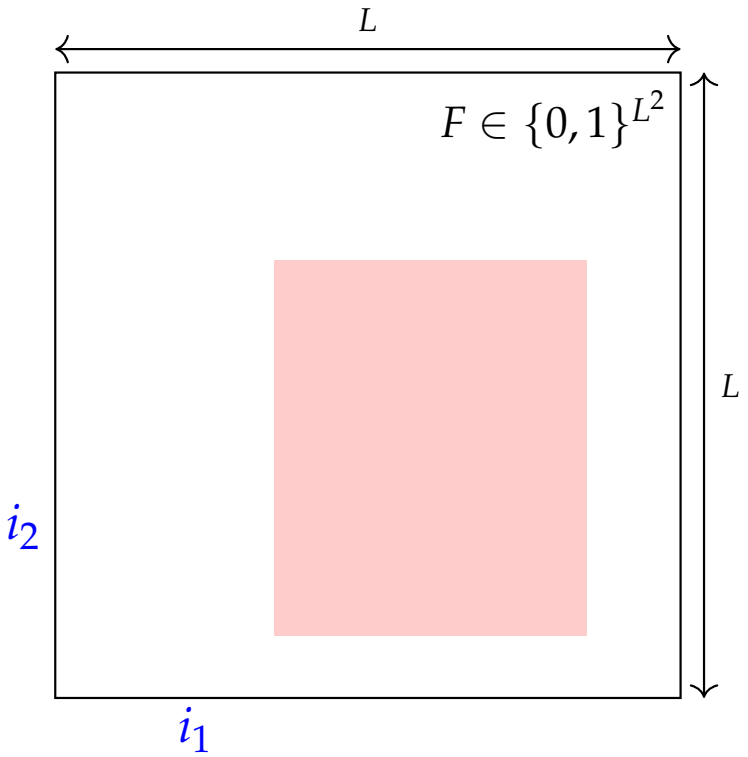
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},

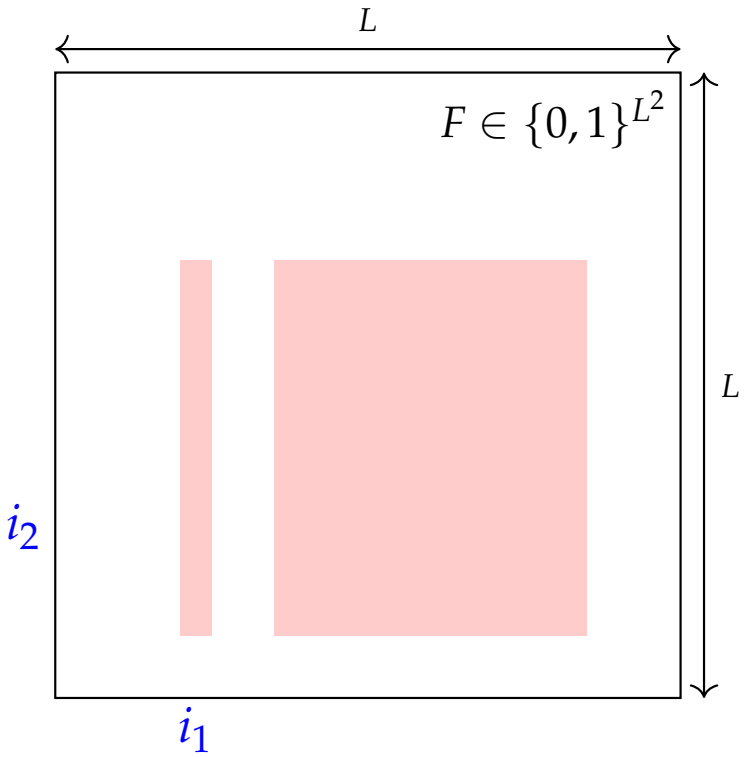
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},
   - $\mathsf{Query}(\boldsymbol{i})_{10} = (X_1 \Delta \{i_1\}, \quad X_2 \quad)$ à S_{10},

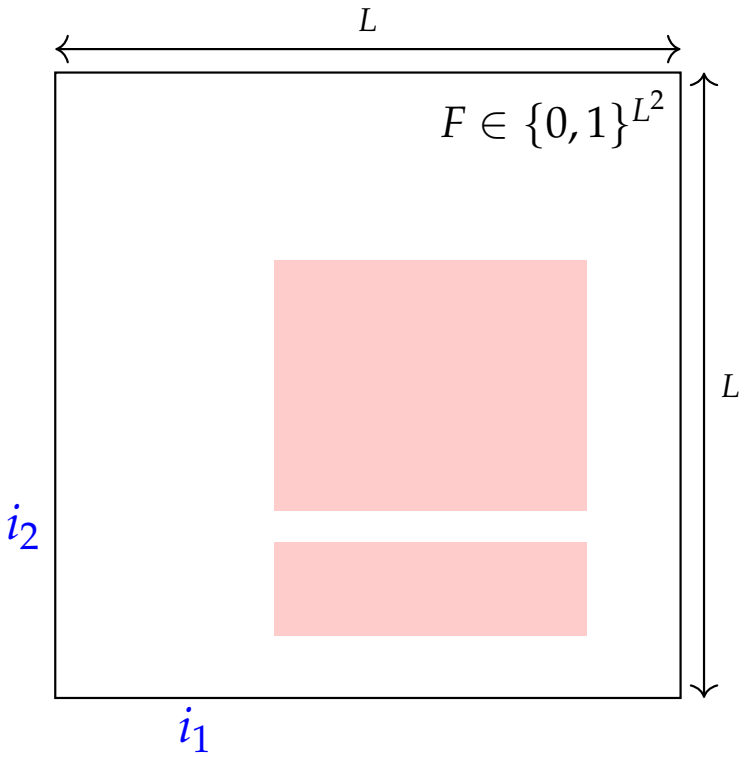
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},
   - $\mathsf{Query}(\boldsymbol{i})_{10} = (X_1 \Delta \{i_1\}, \quad X_2 \quad)$ à S_{10},
   - $\mathsf{Query}(\boldsymbol{i})_{01} = (\quad X_1 \quad , X_2 \Delta \{i_2\})$ à S_{01},

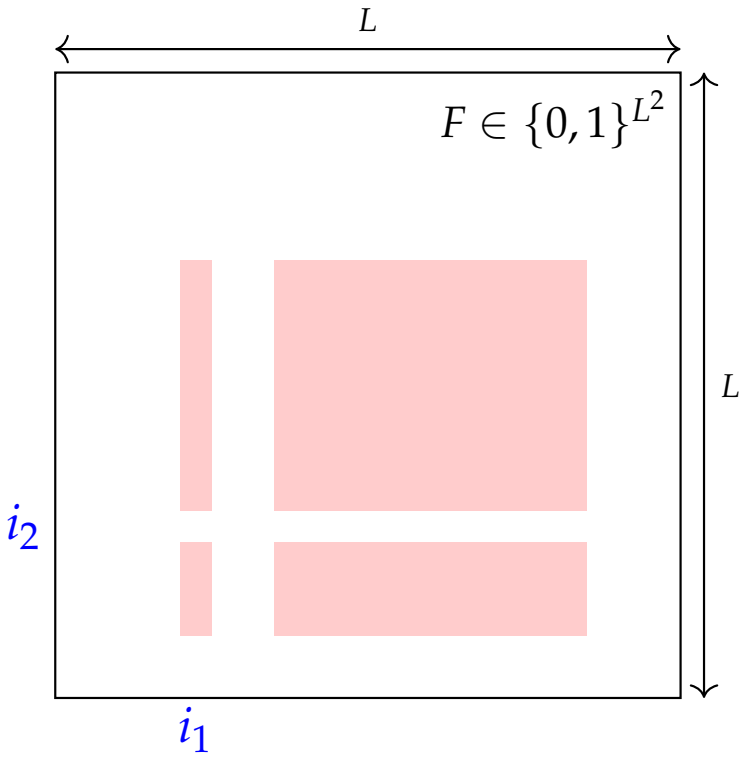
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},
   - $\mathsf{Query}(\boldsymbol{i})_{10} = (X_1 \Delta \{i_1\}, \quad X_2 \quad)$ à S_{10},
   - $\mathsf{Query}(\boldsymbol{i})_{01} = (\quad X_1 \quad , X_2 \Delta \{i_2\})$ à S_{01},
   - $\mathsf{Query}(\boldsymbol{i})_{11} = (X_1 \Delta \{i_1\}, X_2 \Delta \{i_2\})$ à S_{11}.

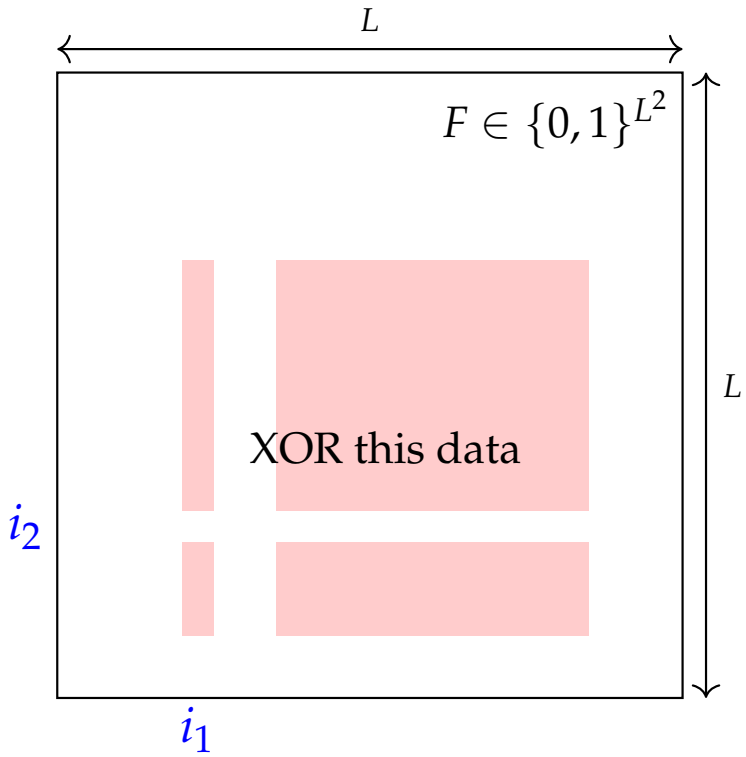
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1,i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},
   - $\mathsf{Query}(\boldsymbol{i})_{10} = (X_1 \Delta \{i_1\}, \quad X_2 \quad)$ à S_{10},
   - $\mathsf{Query}(\boldsymbol{i})_{01} = (\quad X_1 \quad , X_2 \Delta \{i_2\})$ à S_{01},
   - $\mathsf{Query}(\boldsymbol{i})_{11} = (X_1 \Delta \{i_1\}, X_2 \Delta \{i_2\})$ à S_{11}.
2. Sur réception de (Z_1, Z_2), chaque serveur calcule et retourne $a = \bigoplus_{z \in Z_1 \times Z_2} F_z$.

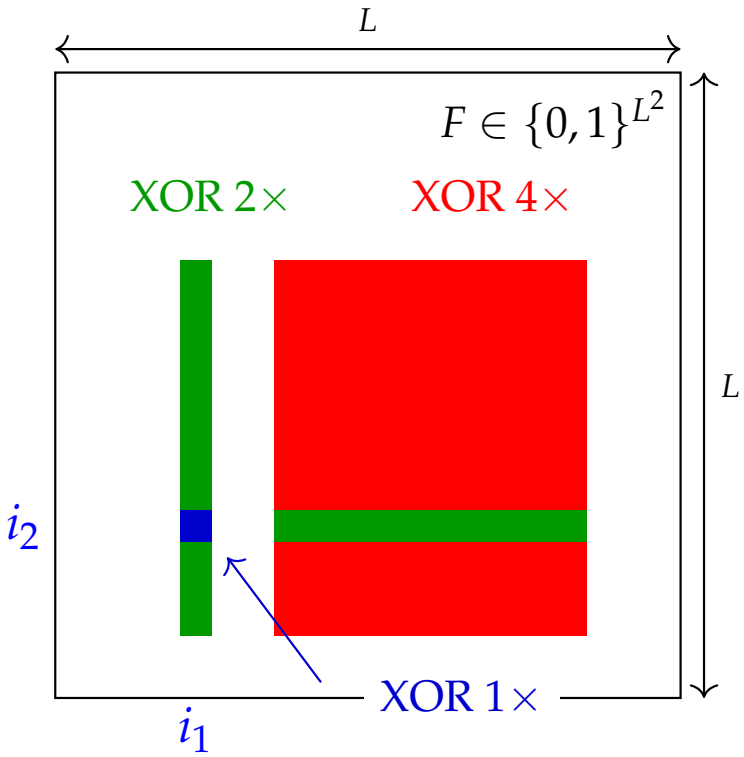
La première construction [CGKS'95-98]

Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

Contexte : base de données F répliquée sur n serveurs, avec :

- $|F| = M$ entrées de même taille (*e.g.* 1 bit), avec $M = L^2$ et $[1, M] \simeq [1, L]^2$.
- $n = 4$ serveurs $S_{00}, S_{01}, S_{10}, S_{11}$, qui stockent chacun une copie de F.

Objectif : extraire $F_{\boldsymbol{i}} = F_{(i_1, i_2)}$ pour $1 \leq i_1, i_2 \leq L$.



1. User engendre aléatoirement deux sous-ensembles X_1, X_2 de $[1, L]$. Ensuite, User envoie :
   - $\mathsf{Query}(\boldsymbol{i})_{00} = (\quad X_1 \quad , \quad X_2 \quad)$ à S_{00},
   - $\mathsf{Query}(\boldsymbol{i})_{10} = (X_1 \Delta \{i_1\}, \quad X_2 \quad)$ à S_{10},
   - $\mathsf{Query}(\boldsymbol{i})_{01} = (\quad X_1 \quad , X_2 \Delta \{i_2\})$ à S_{01},
   - $\mathsf{Query}(\boldsymbol{i})_{11} = (X_1 \Delta \{i_1\}, X_2 \Delta \{i_2\})$ à S_{11}.
2. Sur réception de (Z_1, Z_2), chaque serveur calcule et retourne $a = \bigoplus_{z \in Z_1 \times Z_2} F_z$.
3. User XOR les 4 bits reçus et obtient $F_{\boldsymbol{i}}$.

Propriétés du protocole de PIR de [CGKS'95-98]

Le protocole est **correct** et **confidentiel** si $t = 1$ (pas de coalition).

Propriétés du protocole de PIR de [CGKS'95-98]

Le protocole est **correct** et **confidentiel** si $t = 1$ (pas de coalition).

- **Communication :** $8\sqrt{M}$ bits envoyés, 4 bits reçus,
- **Stockage :** copie de F sur $n = 4$ serveurs,
- **Calcul :**
  - pour chaque serveur : en moyenne, XOR de $(L/2)^2 = M/4$ bits
  - pour le client : XOR de $n = 4$ bits.

Propriétés du protocole de PIR de [CGKS'95-98]

Le protocole est **correct** et **confidentiel** si $t = 1$ (pas de coalition).

- **Communication :** $8\sqrt{M}$ bits envoyés, 4 bits reçus,
- **Stockage :** copie de F sur $n = 4$ serveurs,
- **Calcul :**
  - pour chaque serveur : en moyenne, XOR de $(L/2)^2 = M/4$ bits
  - pour le client : XOR de $n = 4$ bits.

Géneralisable à $n = 2^b$ serveurs :

- **Communication :** $n \log(n) M^{1/\log(n)}$ bits envoyés, n bits reçus,
- **Stockage :** copie de F sur n servers,
- **Calcul :**
  - pour chaque serveur : en moyenne, XOR de M/n bits
  - pour l'utilisateur : XOR de n bits.

Pistes d'amélioration

Dans le protocole précédent :

- complexité de communication sous-linéaire, mais "moyenne"
- calcul lourd pour les serveurs (linéaire en la taille de F)
- nécessite de copier la base de données sur plusieurs serveurs

Pistes d'amélioration

Dans le protocole précédent :

- complexité de communication sous-linéaire, mais "moyenne"
- calcul lourd pour les serveurs (linéaire en la taille de F)
- nécessite de copier la base de données sur plusieurs serveurs

Objectifs :

- meilleure complexité de communication
- **complexité de calcul optimale**
- plus faible redondance de stockage via un **encodage** et une distribution de la base de données

Pistes d'amélioration

Dans le protocole précédent :

- complexité de communication sous-linéaire, mais "moyenne"
- calcul lourd pour les serveurs (linéaire en la taille de F)
- nécessite de copier la base de données sur plusieurs serveurs

Objectifs :

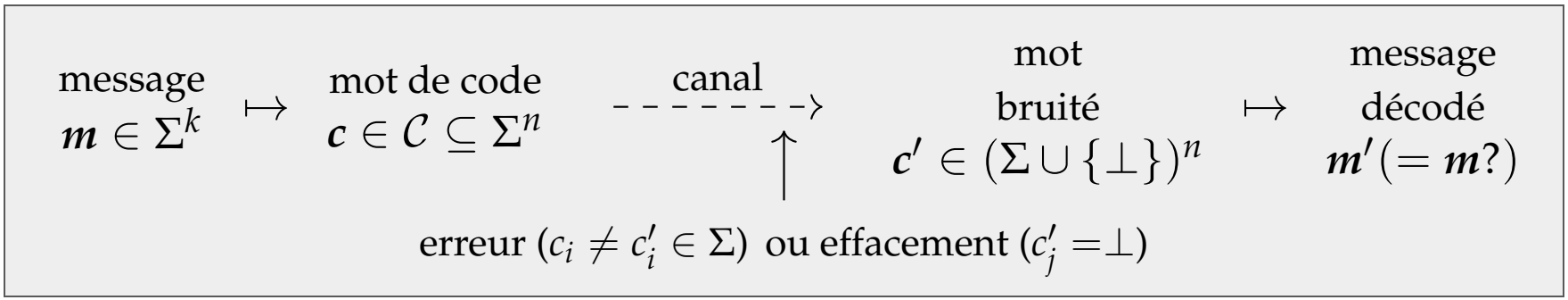
- meilleure complexité de communication
- **complexité de calcul optimale**
- plus faible redondance de stockage via un **encodage** et une distribution de la base de données

Outils : théorie des codes et combinatoire

Outline

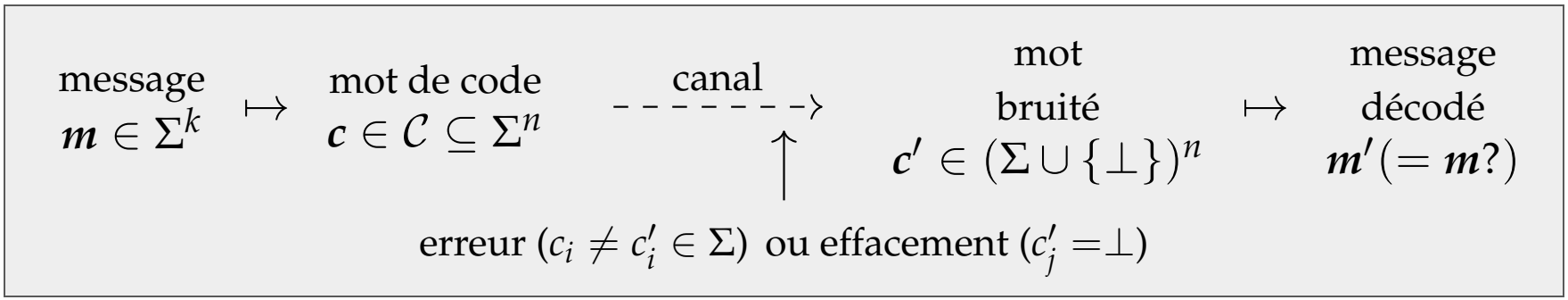
Motivation originelle

Transmission d'information en présence de bruit.



Motivation originelle

Transmission d'information en présence de bruit.

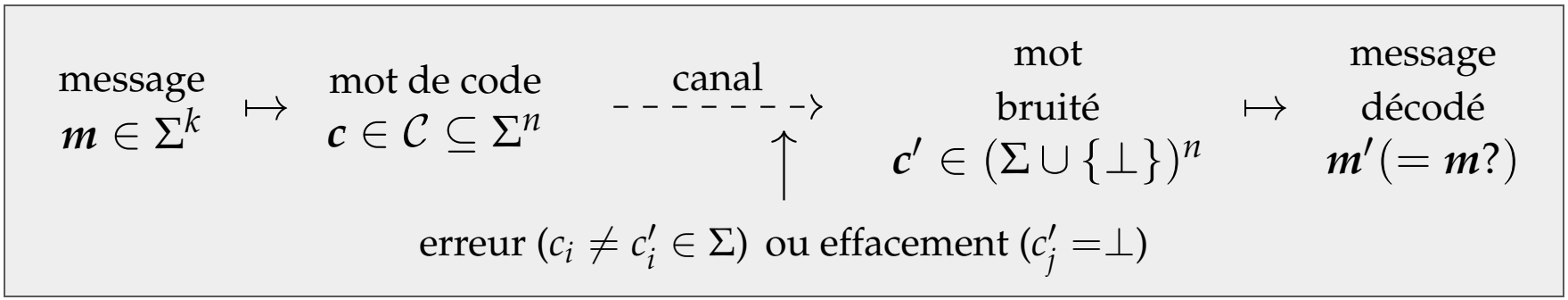


Remarques :

- ici tous les mots de code c ont même longueur n
- l'application d'encodage $\phi : m \mapsto c$ doit être injective

Motivation originelle

Transmission d'information en présence de bruit.



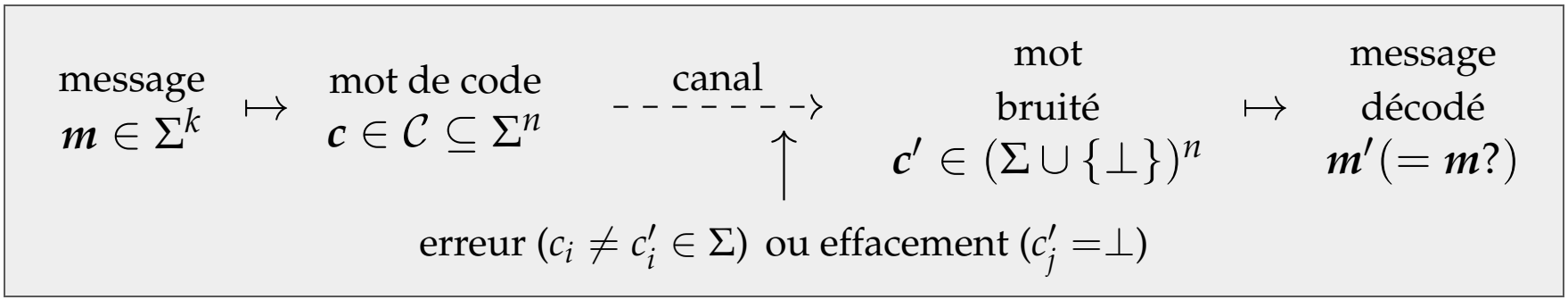
Remarques :

- ici tous les mots de code c ont même longueur n
- l'application d'encodage $\phi : m \mapsto c$ doit être injective

L'ensemble $\mathcal{C}$ des mots de code (pour $m \in \Sigma^k$) est appelé **code correcteur**.

Motivation originelle

Transmission d'information en présence de bruit.



Remarques :

- ici tous les mots de code $\boldsymbol{c}$ ont même longueur n
- l'application d'encodage $\phi : \boldsymbol{m} \mapsto \boldsymbol{c}$ doit être injective

L'ensemble $\mathcal{C}$ des mots de code (pour $\boldsymbol{m} \in \Sigma^k$) est appelé **code correcteur**.

Pour des aspects **pratiques** :

- $\Sigma = \mathbb{F}$ est un corps $\quad$ (*e.g.* $\{0,1\}, \mathbb{Z}/p\mathbb{Z}, \mathbb{F}_q$),
- ϕ est $\mathbb{F}$-linéaire $\implies$ le code $\mathcal{C}$ est un espace vectoriel, on note $k = \dim_{\mathbb{F}} \mathcal{C}$.

Codes linéaires

Soit $\mathcal{C} \subseteq \mathbb{F}^n$ un code linéaire de dimension k.

Définition. Toute matrice $\boldsymbol{G} \in \mathbb{F}^{k \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{mG} \mid \boldsymbol{m} \in \mathbb{F}^k\}$$

est une **matrice génératrice** de $\mathcal{C}$.

Codes linéaires

Soit $\mathcal{C} \subseteq \mathbb{F}^n$ un code linéaire de dimension k.

Définition. Toute matrice $\boldsymbol{G} \in \mathbb{F}^{k \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{m}\boldsymbol{G} \mid \boldsymbol{m} \in \mathbb{F}^k\}$$

est une **matrice génératrice** de $\mathcal{C}$.

On appelle **équation de parité** tout vecteur $\boldsymbol{h} \in \mathbb{F}^n$ tel que

$$\forall \boldsymbol{c} \in \mathcal{C},\ \langle \boldsymbol{h}, \boldsymbol{c} \rangle := \sum_{i=1}^{n} h_i c_i = 0$$

On note $\mathcal{C}^\perp$ l'ensemble des équations de parité de $\mathcal{C}$.

Codes linéaires

Soit $\mathcal{C} \subseteq \mathbb{F}^n$ un code linéaire de dimension k.

Définition. Toute matrice $\boldsymbol{G} \in \mathbb{F}^{k \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{mG} \mid \boldsymbol{m} \in \mathbb{F}^k\}$$

est une **matrice génératrice** de $\mathcal{C}$.

On appelle **équation de parité** tout vecteur $\boldsymbol{h} \in \mathbb{F}^n$ tel que

$$\forall \boldsymbol{c} \in \mathcal{C},\ \langle \boldsymbol{h}, \boldsymbol{c} \rangle := \sum_{i=1}^{n} h_i c_i = 0$$

On note $\mathcal{C}^\perp$ l'ensemble des équations de parité de $\mathcal{C}$.

Définition. Toute matrice $\boldsymbol{H} \in \mathbb{F}^{m \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{c} \in \mathbb{F}^n \mid \boldsymbol{H}\boldsymbol{c}^\top = \boldsymbol{0}\}$$

est une **matrice de parité** de $\mathcal{C}$.

Codes linéaires

Soit $\mathcal{C} \subseteq \mathbb{F}^n$ un code linéaire de dimension k.

Définition. Toute matrice $\boldsymbol{G} \in \mathbb{F}^{k \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{m}\boldsymbol{G} \mid \boldsymbol{m} \in \mathbb{F}^k\}$$

est une **matrice génératrice** de $\mathcal{C}$.

On appelle **équation de parité** tout vecteur $\boldsymbol{h} \in \mathbb{F}^n$ tel que

$$\forall \boldsymbol{c} \in \mathcal{C},\ \langle \boldsymbol{h}, \boldsymbol{c} \rangle := \sum_{i=1}^{n} h_i c_i = 0$$

On note $\mathcal{C}^\perp$ l'ensemble des équations de parité de $\mathcal{C}$.

Définition. Toute matrice $\boldsymbol{H} \in \mathbb{F}^{m \times n}$ telle que

$$\mathcal{C} = \{\boldsymbol{c} \in \mathbb{F}^n \mid \boldsymbol{H}\boldsymbol{c}^\top = \boldsymbol{0}\}$$

est une **matrice de parité** de $\mathcal{C}$.

On a alors $\dim_{\mathbb{F}}(\mathcal{C}) = n - \mathrm{rk}_{\mathbb{F}}(\boldsymbol{H})$.

Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Soit $\boldsymbol{c} \in \mathcal{C}$, dégradé par un effacecement à l'indice i : $c_i = \perp$.

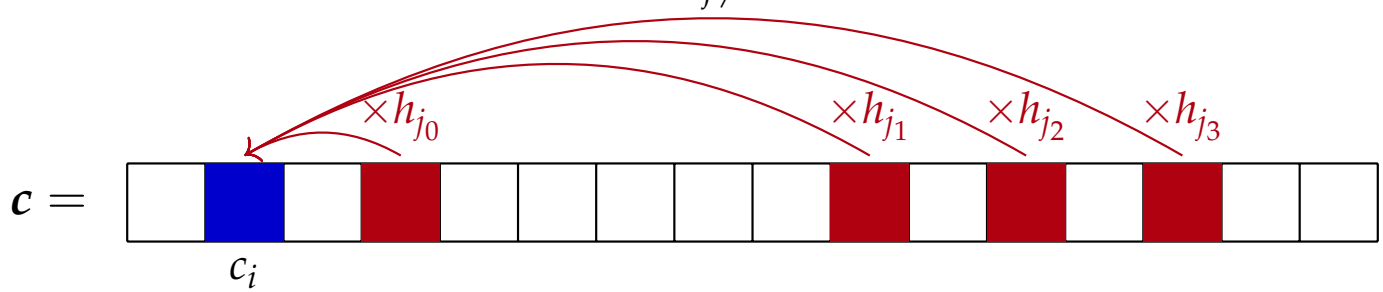
Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Soit $\boldsymbol{c} \in \mathcal{C}$, dégradé par un effacecement à l'indice i : $\quad c_i = \perp$.

S'il existe une équation de parité $\boldsymbol{h} \in \mathcal{C}^\perp$ telle que $h_i \neq 0$, alors :

$$c_i = \frac{-1}{h_i} \sum_{j \neq i} h_j c_j \,.$$



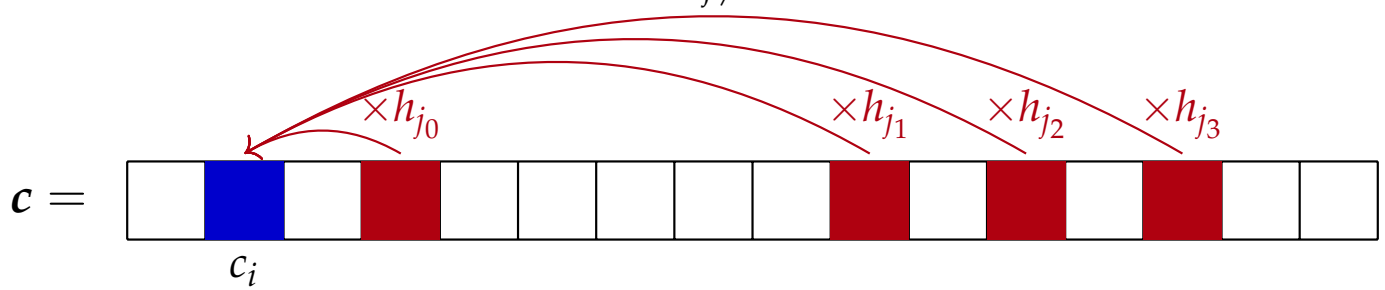
Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Soit $\boldsymbol{c} \in \mathcal{C}$, dégradé par un effacecement à l'indice i : $\quad c_i = \perp$.

S'il existe une équation de parité $\boldsymbol{h} \in \mathcal{C}^\perp$ telle que $h_i \neq 0$, alors :

$$c_i = \frac{-1}{h_i} \sum_{j \neq i} h_j c_j \,.$$



Notons $\mathrm{supp}(\boldsymbol{h}) = \{j \in [1, n] \mid h_j \neq 0\}$ le **support** de $\boldsymbol{h}$.

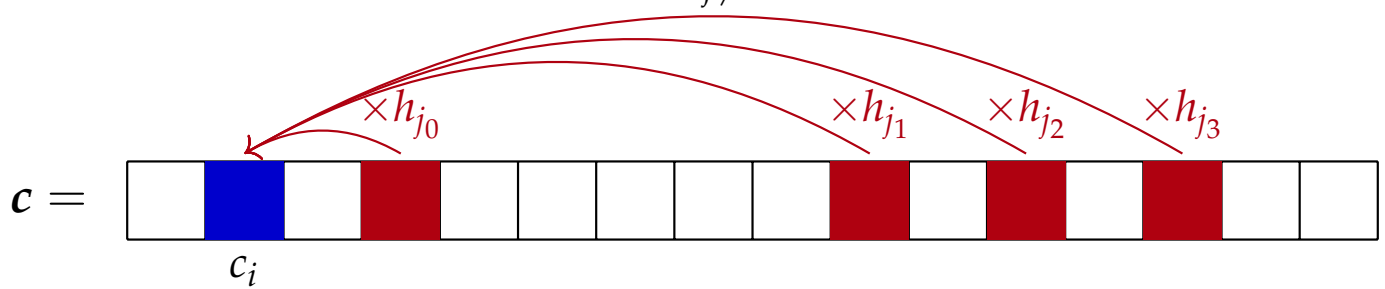
Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Soit $\boldsymbol{c} \in \mathcal{C}$, dégradé par un effacecement à l'indice i : $\quad c_i = \perp$.

S'il existe une équation de parité $\boldsymbol{h} \in \mathcal{C}^\perp$ telle que $h_i \neq 0$, alors :

$$c_i = \frac{-1}{h_i} \sum_{j \neq i} h_j c_j \,.$$



Notons $\mathrm{supp}(\boldsymbol{h}) = \{j \in [1, n] \mid h_j \neq 0\}$ le **support** de $\boldsymbol{h}$.

- on peut réparer la coordonnée c_i s'il n'y a pas d'erreur/effacement sur $\mathrm{supp}(\boldsymbol{h}) \setminus \{i\}$;

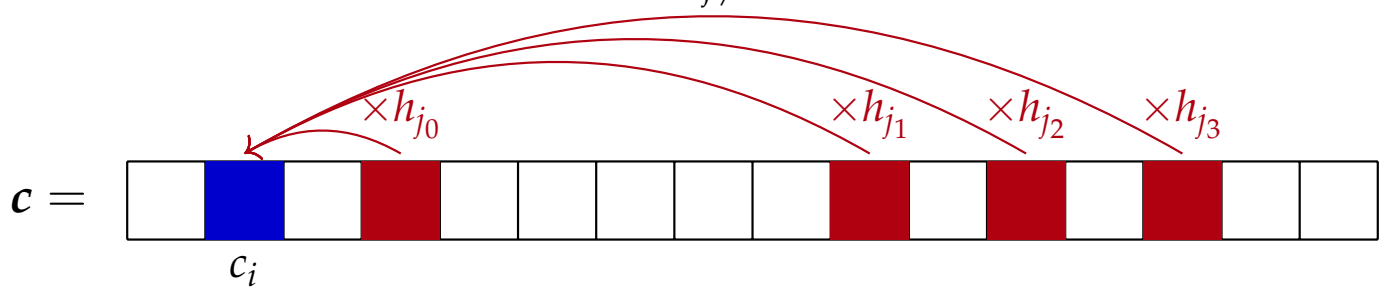
Correction d'effacement

Un **effacement**, représenté par $\perp$, est une coordonnée d'un mot que l'on **sait** fausse.

Soit $\boldsymbol{c} \in \mathcal{C}$, dégradé par un effacecement à l'indice i : $\quad c_i = \perp$.

S'il existe une équation de parité $\boldsymbol{h} \in \mathcal{C}^\perp$ telle que $h_i \neq 0$, alors :

$$c_i = \frac{-1}{h_i} \sum_{j \neq i} h_j c_j \,.$$



Notons $\mathrm{supp}(\boldsymbol{h}) = \{j \in [1,n] \mid h_j \neq 0\}$ le **support** de $\boldsymbol{h}$.

- on peut réparer la coordonnée c_i s'il n'y a pas d'erreur/effacement sur $\mathrm{supp}(\boldsymbol{h}) \setminus \{i\}$;
- pour une réparation efficace, ou souhaite $|\mathrm{supp}(\boldsymbol{h})|$ le plus petit possible.

Correction d'erreurs

Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Correction d'erreurs

Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Correction d'erreurs

Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Correction d'erreurs

Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.

Correction d'erreurs

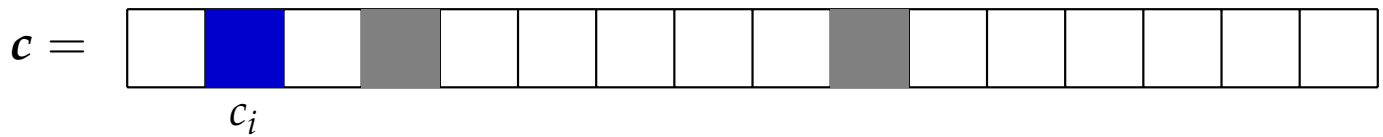
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



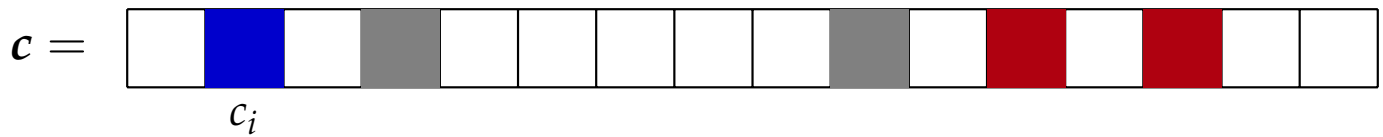
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



Correction d'erreurs

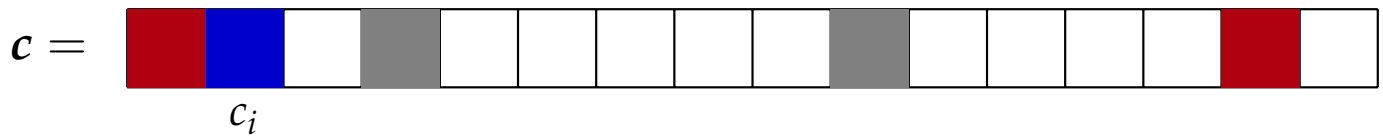
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



Correction d'erreurs

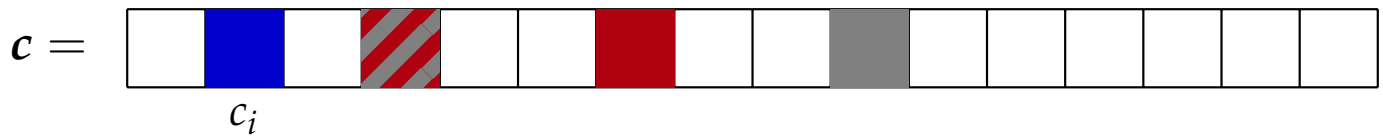
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



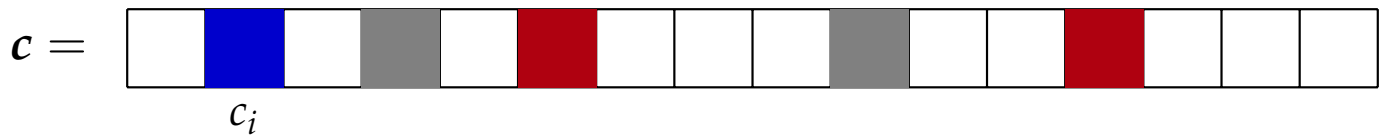
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



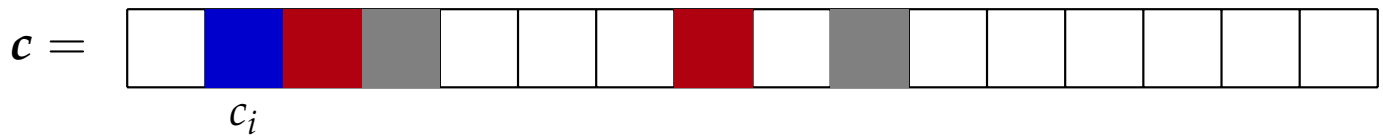
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



Correction d'erreurs

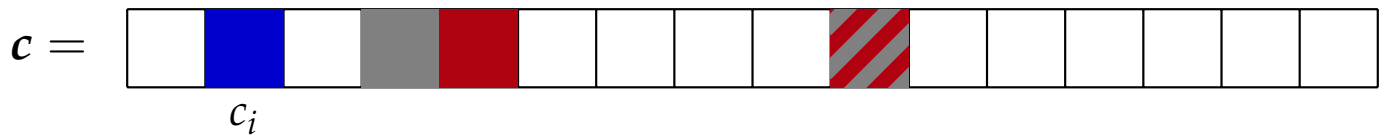
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



Correction d'erreurs

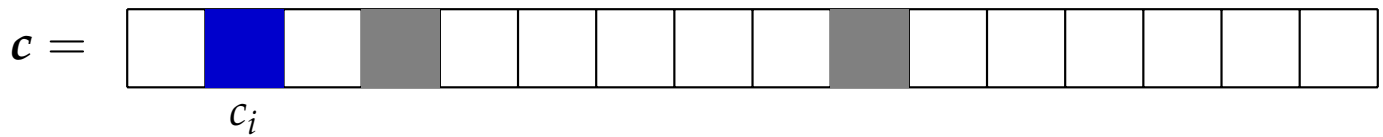
Une **erreur** est un symbole dégradé d'un mot dont on ne connait pas la position.

Certains codes sont munis d'**algorithmes de décodage** efficaces.

Les algorithmes de décodage classiques nécessitent, pour corriger une erreur, de **lire entièrement** le mot erroné $\implies$ complexité $\Omega(n)$.

Question légitime : décodage (probabiliste) d'erreur en complexité sous-linéaire ?

Idée (informelle) : construire un code avec beaucoup d'équations de parité de même petit poids $\ell \ll n$, **réparties uniformément** sur $[1, n]$. Puis, tirer au hasard l'une d'entre elles et l'utiliser pour réparer c_i.



Résultats : avec probabilité de réussite constante,

- on peut le faire en temps $O_n(1)$ ($\simeq$ théorème PCP), mais alors $n = \exp(k)$,
- on peut le faire en temps $O_n(2^{\sqrt{\log n \log\log n}})$, avec $n = O(k)$

Un exemple : le code de Hadamard

Messages $\boldsymbol{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$.

Un exemple : le code de Hadamard

Messages $\boldsymbol{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$.

Soit $\{0,1\}^k =: \{\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(2^k)}\}$

Un exemple : le code de Hadamard

Messages $\boldsymbol{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$.

Soit $\{0,1\}^k =: \{\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(2^k)}\}$

Définition. Le code de Hadamard est défini par l'encodage

$$\boldsymbol{m} \mapsto \boldsymbol{c} \in \{0,1\}^{2^k} \quad \text{où} \quad c_i = \sum_{j=1}^{k} a_j^{(i)} m_j$$

C'est un code de dimension k et de longueur $n = 2^k$.

Un exemple : le code de Hadamard

Messages $\boldsymbol{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$.

Soit $\{0,1\}^k =: \{\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(2^k)}\}$

Définition. Le code de Hadamard est défini par l'encodage

$$\boldsymbol{m} \mapsto \boldsymbol{c} \in \{0,1\}^{2^k} \quad \text{où} \quad c_i = \sum_{j=1}^{k} a_j^{(i)} m_j$$

C'est un code de dimension k et de longueur $n = 2^k$.

Propriété importante :

$$\text{si} \quad \boldsymbol{a}^{(i_1)} = \boldsymbol{a}^{(i_2)} + \boldsymbol{a}^{(i_3)}, \quad \text{alors} \quad c_{i_1} = c_{i_2} + c_{i_3}$$

Un exemple : le code de Hadamard

Messages $\boldsymbol{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$.

Soit $\{0,1\}^k =: \{\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(2^k)}\}$

Définition. Le code de Hadamard est défini par l'encodage

$$\boldsymbol{m} \mapsto \boldsymbol{c} \in \{0,1\}^{2^k} \quad \text{où} \quad c_i = \sum_{j=1}^{k} a_j^{(i)} m_j$$

C'est un code de dimension k et de longueur $n = 2^k$.

Propriété importante :

$$\text{si} \quad \boldsymbol{a}^{(i_1)} = \boldsymbol{a}^{(i_2)} + \boldsymbol{a}^{(i_3)}, \quad \text{alors} \quad c_{i_1} = c_{i_2} + c_{i_3}$$

Conséquence : il existe des équations de parité de poids $\ell = 3$, **réparties uniformément** sur $[1, n]$.

Outline

Outline

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

Initialisation : la base de donnée F est copiée sur $\ell - 1$ serveurs qui ne coopèrent pas.

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

Initialisation : la base de donnée F est copiée sur $\ell - 1$ serveurs qui ne coopèrent pas.

Protocole pour extraire $F_i = c_i$.

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

Initialisation : la base de donnée F est copiée sur $\ell - 1$ serveurs qui ne coopèrent pas.

Protocole pour extraire $F_i = c_i$.

1. Pour définir $\mathsf{Query}(i)$:
   - on tire aléatoirement $\boldsymbol{h} \leftarrow \mathcal{H}^{(i)}$, avec $\mathrm{supp}(\boldsymbol{h}) = \{h_i, h_{m_1}, \ldots, h_{m_{\ell-1}}\}$,
   - on définit $\mathsf{Query}(i)_j = m_j$.

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

Initialisation : la base de donnée F est copiée sur $\ell - 1$ serveurs qui ne coopèrent pas.

Protocole pour extraire $F_i = c_i$.

1. Pour définir $\mathsf{Query}(i)$:
   - on tire aléatoirement $\boldsymbol{h} \leftarrow \mathcal{H}^{(i)}$, avec $\mathrm{supp}(\boldsymbol{h}) = \{h_i, h_{m_1}, \ldots, h_{m_{\ell-1}}\}$,
   - on définit $\mathsf{Query}(i)_j = m_j$.
2. Chaque serveur S_j retourne $a_j = \mathrm{Enc}_{\mathcal{C}}(F)_{m_j}$.

PIR à partir de code localement décodable

Paramètres globaux :

- $\mathcal{C} \subseteq \mathbb{F}^n$ un code localement décodable de localité ℓ
- $\mathcal{H} = \bigcup_{i=1}^{n} \mathcal{H}^{(i)} \subseteq \mathcal{C}^{\perp}$, où $\mathcal{H}^{(i)}$ est un sous-ensemble d'équations de parité
  de poids ℓ,
  dont le support contient i,
  réparties uniformément sur $[1, n]$.

Initialisation : la base de donnée F est copiée sur $\ell - 1$ serveurs qui ne coopèrent pas.

Protocole pour extraire $F_i = c_i$.

1. Pour définir $\mathsf{Query}(i)$:
   - on tire aléatoirement $\boldsymbol{h} \leftarrow \mathcal{H}^{(i)}$, avec $\mathrm{supp}(\boldsymbol{h}) = \{h_i, h_{m_1}, \ldots, h_{m_{\ell-1}}\}$,
   - on définit $\mathsf{Query}(i)_j = m_j$.
2. Chaque serveur S_j retourne $a_j = \mathrm{Enc}_{\mathcal{C}}(F)_{m_j}$.
3. User reconstruit $F_i = -\frac{1}{h_i} \sum_j h_{m_j} a_j$.

Résultats

Théorème (informel, Katz-Trevisan 00). Tout code localement décodable de localité ℓ induit un protocole de PIR à $\ell - 1$ serveurs.

Résultats

Théorème (informel, Katz-Trevisan 00). Tout code localement décodable de localité ℓ induit un protocole de PIR à $\ell - 1$ serveurs.

Grandeurs.

- la base de données est copiée sur les $\ell - 1$ serveurs (coûteux)
- à chaque exécution, chaque serveur doit recalculer le symbole $\mathrm{Enc}_{\mathcal{C}}(F)_{m_j}$ (très coûteux)

Résultats

Théorème (informel, Katz-Trevisan 00). Tout code localement décodable de localité ℓ induit un protocole de PIR à $\ell - 1$ serveurs.

Grandeurs.

- la base de données est copiée sur les $\ell - 1$ serveurs (coûteux)
- à chaque exécution, chaque serveur doit recalculer le symbole $\mathrm{Enc}_{\mathcal{C}}(F)_{m_j}$ (très coûteux)

Solution "classique" : compromis temps/mémoire par un précalcul

- le coût supplémentaire de stockage dépend de la dimension du code
- souvent très coputeux en mémoire

Résultats

Théorème (informel, Katz-Trevisan 00). Tout code localement décodable de localité ℓ induit un protocole de PIR à $\ell - 1$ serveurs.

Grandeurs.

- la base de données est copiée sur les $\ell - 1$ serveurs (coûteux)
- à chaque exécution, chaque serveur doit recalculer le symbole $\mathrm{Enc}_{\mathcal{C}}(F)_{m_j}$ (très coûteux)

Solution "classique" : compromis temps/mémoire par un précalcul

- le coût supplémentaire de stockage dépend de la dimension du code
- souvent très coputeux en mémoire

Nouvelle idée (initiée par Augot, Levy-dit-Vehel, Shikfa, 2014) :

- partitionner l'encodage de la base de données sur les serveurs,
- nécessite de concevoir un code $\mathcal{C}$ dont l'ensemble d'équations de parité de petits poids $\mathcal{H}$ puisse correspondre à cette partition.

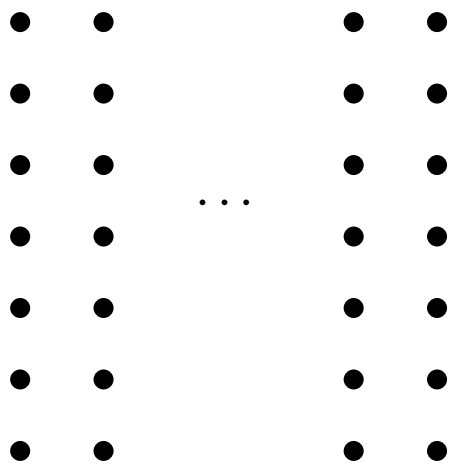
Outline

Designs transversaux

Pour concevoir ces codes, un peu de combinatoire...

Un **design transversal** $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ est donné par :

- X un ensemble de *points*, $|X| = N = ns$,



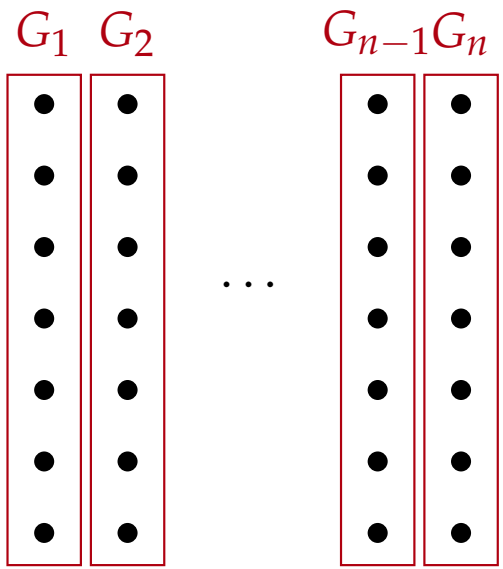
Designs transversaux

Pour concevoir ces codes, un peu de combinatoire...

Un **design transversal** $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ est donné par :

- X un ensemble de *points*, $|X| = N = ns$,
- une partition de X en *groupes* $\mathcal{G} = \{G_j\}_{1 \leq j \leq n}$

$$X = \coprod_{j=1}^{n} G_j \text{ et } |G_j| = s\,,$$



Designs transversaux

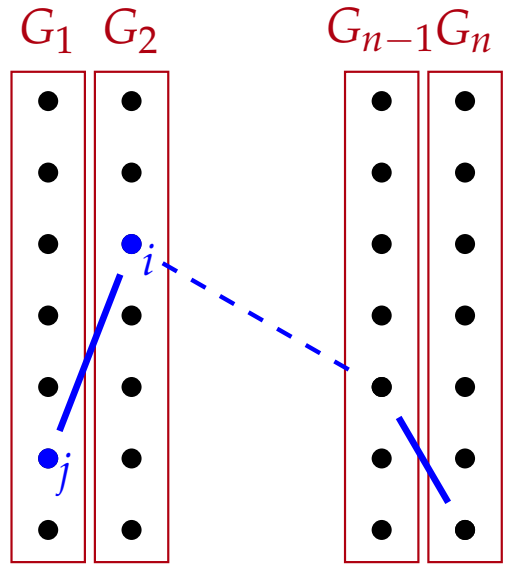
Pour concevoir ces codes, un peu de combinatoire...

Un **design transversal** $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ est donné par :

- X un ensemble de *points*, $|X| = N = ns$,
- une partition de X en *groupes* $\mathcal{G} = \{G_j\}_{1 \le j \le n}$

$$X = \coprod_{j=1}^{n} G_j \text{ et } |G_j| = s\,,$$

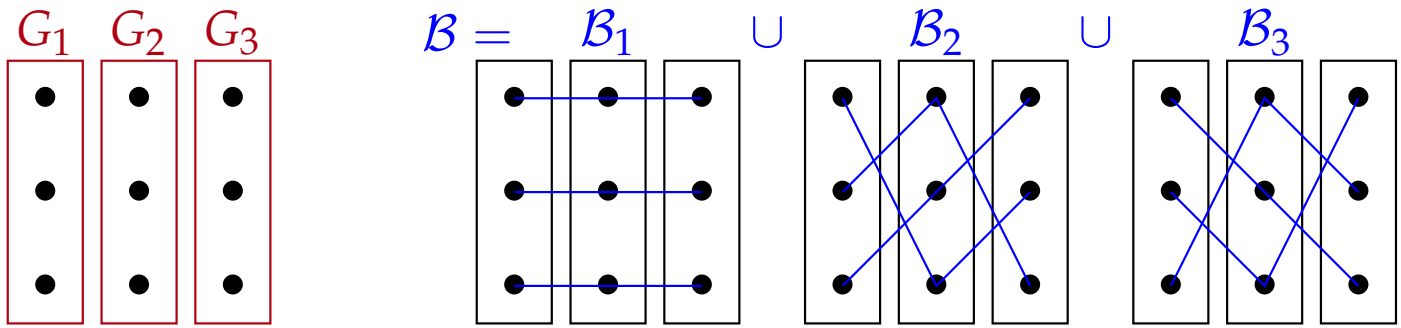
- un ensemble de *blocs* $B \in \mathcal{B}$ tels que
  - $B \subset X$ et $|B| = n$;
  - toute paire $\{i,j\} \subset X$ est incluse :
    ou bien dans un unique groupe $G \in \mathcal{G}$,
    ou bien dans un unique bloc $B \in \mathcal{B}$



Un exemple de design transversal

Exemple pour TD$(3,3)$:

- $ns = 9$ points
- $s = 3$ groupes G_1, G_2, G_3 de cardinal 3
- $ns = 9$ blocs de $n = 3$ points (ici présentées en 3 classes $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$)



Des codes à partir de designs

Soit $\mathcal{T}$ un design transversal $\mathrm{TD}(n, s) = (X, \mathcal{B}, \mathcal{G})$.

Sa **matrice d'incidence** $\boldsymbol{M}$, de taille $|\mathcal{B}| \times |X| = ns \times ns$, est définie comme :

$$M_{i,j} = \begin{cases} 1 & \text{if } x_j \in B_i \\ 0 & \text{otherwise.} \end{cases}$$

Des codes à partir de designs

Soit $\mathcal{T}$ un design transversal $\mathrm{TD}(n, s) = (X, \mathcal{B}, \mathcal{G})$.

Sa **matrice d'incidence** $\boldsymbol{M}$, de taille $|\mathcal{B}| \times |X| = ns \times ns$, est définie comme :

$$M_{i,j} = \begin{cases} 1 & \text{if } x_j \in B_i \\ 0 & \text{otherwise.} \end{cases}$$

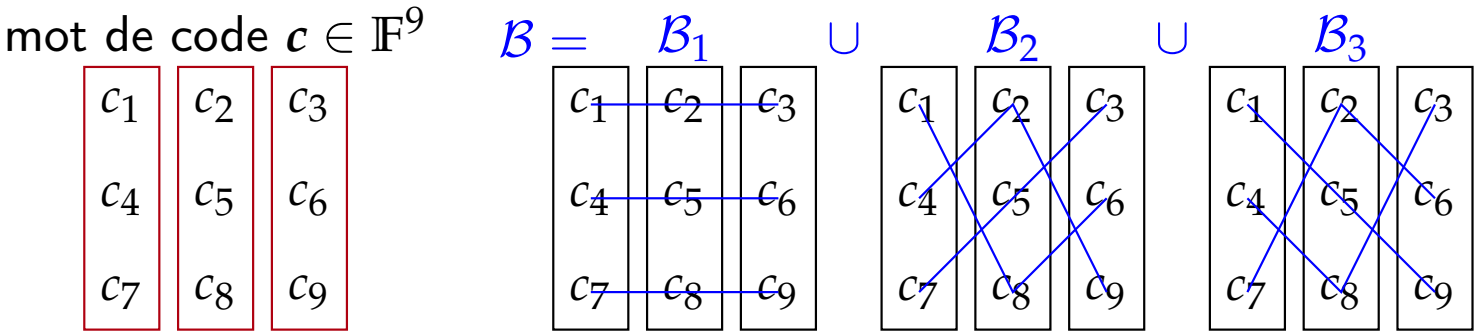
Définition. Le **code $\mathcal{C}$ construit sur le design $\mathcal{T}$ sur le corps $\mathbb{F}$** est le code $\mathbb{F}$-linéaire qui admet $\boldsymbol{M}$ comme matrice de parité.

- la longueur du code est $\mathrm{length}(\mathcal{C}) = |X| = ns$,
- la dimension du code est $\dim(\mathcal{C}) = \dim(\ker \boldsymbol{M})$,
- chaque bloc $B \in \mathcal{B}$ produit une équation de parité $\boldsymbol{h} \in \mathbb{F}^{ns}$, telle que

$$\mathrm{wt}(\boldsymbol{h}_{|G_j}) = 1, \quad \forall j = 1, \ldots, n$$

Example

On reprend TD$(3,3)$:

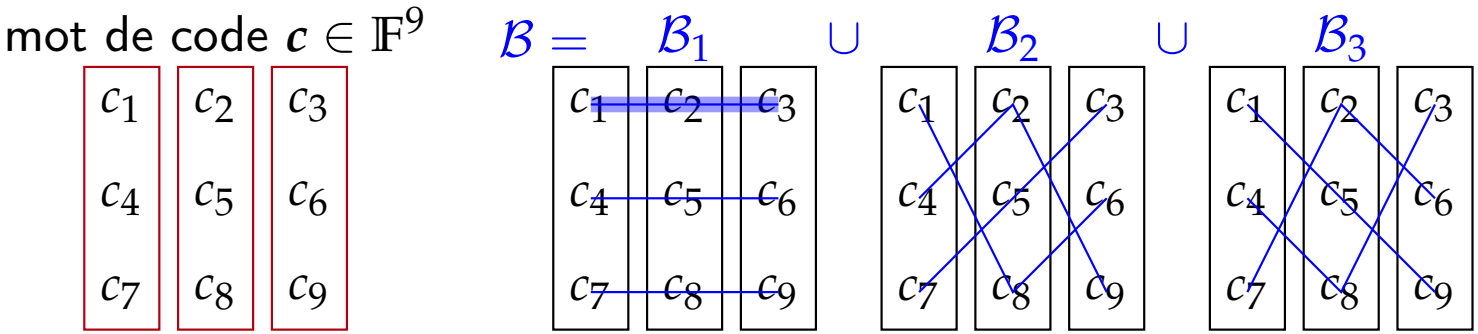


La matrice d'incidence du design (= matrice de parité du code) est :

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

Example

On reprend TD$(3,3)$:

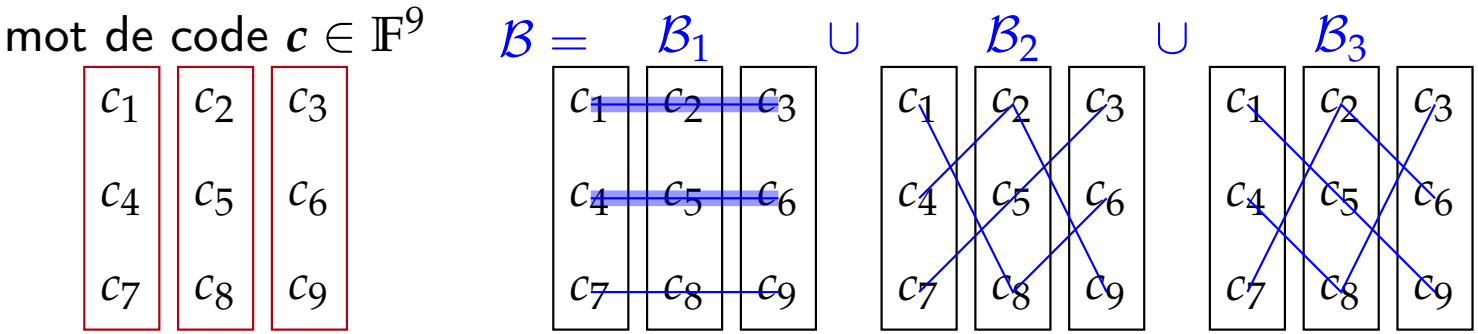


La matrice d'incidence du design (= matrice de parité du code) est :

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

Example

On reprend TD$(3,3)$:



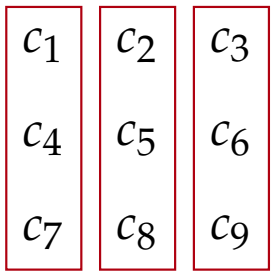
La matrice d'incidence du design (= matrice de parité du code) est :

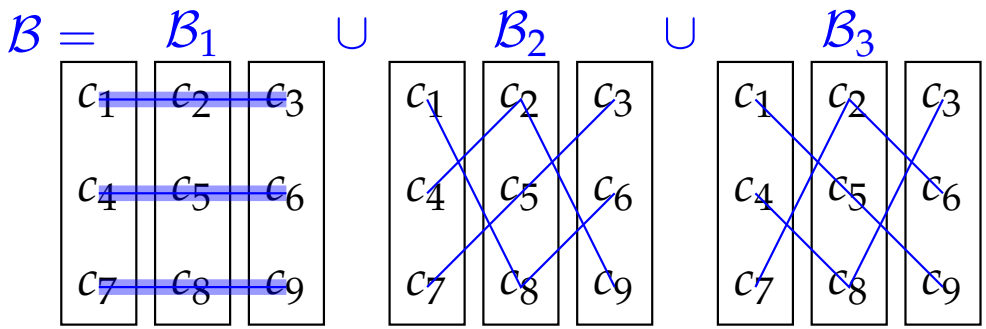
$$\boldsymbol{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 &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

Example

On reprend TD$(3,3)$:

mot de code $\boldsymbol{c} \in \mathbb{F}^9$





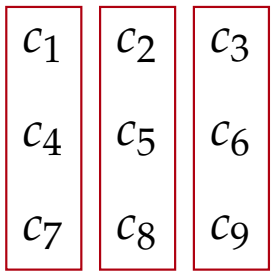
La matrice d'incidence du design (= matrice de parité du code) est :

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

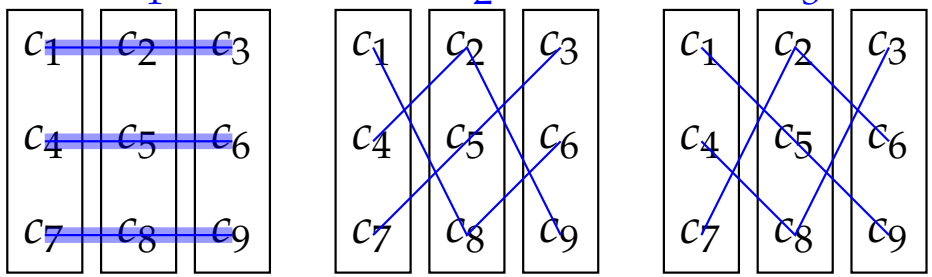
Example

On reprend TD$(3,3)$:

mot de code $\boldsymbol{c} \in \mathbb{F}^9$



$\mathcal{B} =$



La matrice d'incidence du design (= matrice de parité du code) est :

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

Le noyau de la matrice dépend du corps $\mathbb{F}$ considéré.

Cas le plus favorable : sur $\mathbb{F}_3 := \mathbb{Z}/3\mathbb{Z}$, on a $\mathrm{rk}(\boldsymbol{H}) = 6 \quad \Longrightarrow \quad \dim_{\mathbb{F}_3}(\mathcal{C}) = 3.$

Le protocole

Private Information Retrieval from Transversal Designs. **L.**. IEEE-TIT. **2019**.

Soit $\mathcal{C} \subseteq \mathbb{F}_q^{ns}$ un code construit sur un $\mathrm{TD}(n, s)$.

Le protocole

Private Information Retrieval from Transversal Designs. **L.**. IEEE-TIT. **2019**.

Soit $\mathcal{C} \subseteq \mathbb{F}_q^{ns}$ un code construit sur un TD(n, s).

• **Initialisation.** User encode le message $\boldsymbol{m} = F$ en un mot du code $\boldsymbol{c} \in \mathcal{C}$. Puis, User distribue $\boldsymbol{c}_{|G_j}$ au serveur S_j.

Le protocole

Private Information Retrieval from Transversal Designs. **L.**. IEEE-TIT. **2019**.

Soit $\mathcal{C} \subseteq \mathbb{F}_q^{ns}$ un code construit sur un TD(n, s).

• **Initialisation.** User encode le message $\boldsymbol{m} = F$ en un mot du code $\boldsymbol{c} \in \mathcal{C}$.
Puis, User distribue $\boldsymbol{c}_{|G_j}$ au serveur S_j.

• **Pour extraire** $F_i = c_i$ où $i \in X$:

1. User tire aléatoirement un bloc $B \in \mathcal{B}$ qui contient i.
   Puis, User définit la requête :

$$q_j = \mathsf{Query}(i)_j := \begin{cases} \text{unique élément de } B \cap G_j & \text{si } i \notin G_j \\ \text{un point aléatoire de } G_j & \text{sinon.} \end{cases}$$

Le protocole

Private Information Retrieval from Transversal Designs. **L.**. IEEE-TIT. **2019**.

Soit $\mathcal{C} \subseteq \mathbb{F}_q^{ns}$ un code construit sur un TD(n, s).

• **Initialisation.** User encode le message $\boldsymbol{m} = F$ en un mot du code $\boldsymbol{c} \in \mathcal{C}$.
Puis, User distribue $\boldsymbol{c}_{|G_j}$ au serveur S_j.

• **Pour extraire** $F_i = c_i$ où $i \in X$:

1. User tire aléatoirement un bloc $B \in \mathcal{B}$ qui contient i.
   Puis, User définit la requête :

$$q_j = \mathsf{Query}(i)_j := \begin{cases} \text{unique élément de } B \cap G_j & \text{si } i \notin G_j \\ \text{un point aléatoire de } G_j & \text{sinon.} \end{cases}$$

2. Chaque serveur S_j retourne c_{q_j}

Le protocole

Private Information Retrieval from Transversal Designs. **L.**. IEEE-TIT. **2019**.

Soit $\mathcal{C} \subseteq \mathbb{F}_q^{ns}$ un code construit sur un TD(n, s).

• **Initialisation.** User encode le message $\boldsymbol{m} = F$ en un mot du code $\boldsymbol{c} \in \mathcal{C}$.
Puis, User distribue $\boldsymbol{c}_{|G_j}$ au serveur S_j.

• **Pour extraire** $F_i = c_i$ où $i \in X$:

1. User tire aléatoirement un bloc $B \in \mathcal{B}$ qui contient i.
   Puis, User définit la requête :

$$q_j = \mathsf{Query}(i)_j := \begin{cases} \text{unique élément de } B \cap G_j & \text{si } i \notin G_j \\ \text{un point aléatoire de } G_j & \text{sinon.} \end{cases}$$

2. Chaque serveur S_j retourne c_{q_j}
3. User reconstruit

$$c_i = -\sum_{j:\, i \notin G_j} c_{q_j} = -\sum_{b \in B \setminus \{i\}} c_b$$

Confidentialité et paramètres du protocole

Théorème. Ce protocole PIR est correct et sûr du point de vue IT.

Preuve :

- le seul serveur qui détient F_i reçoit une requête aléatoire ;
- pour tout autre serveur S_j, la requête q_j ne fournit aucune information sur le bloc B qui a été choisi par User $\Rightarrow$ aucune information sur i n'est révélée.

Confidentialité et paramètres du protocole

Théorème. Ce protocole PIR est correct et sûr du point de vue IT.

Preuve :

- le seul serveur qui détient F_i reçoit une requête aléatoire ;
- pour tout autre serveur S_j, la requête q_j ne fournit aucune information sur le bloc B qui a été choisi par User $\Rightarrow$ aucune information sur i n'est révélée.

Caractéristiques.

- complexité de communication : $n \log s$ bits envoyés, $n \log q$ bits reçus

Confidentialité et paramètres du protocole

Théorème. Ce protocole PIR est correct et sûr du point de vue IT.

Preuve :

- le seul serveur qui détient F_i reçoit une requête aléatoire ;
- pour tout autre serveur S_j, la requête q_j ne fournit aucune information sur le bloc B qui a été choisi par User $\Rightarrow$ aucune information sur i n'est révélée.

Caractéristiques.

- complexité de communication : $n \log s$ bits envoyés, $n \log q$ bits reçus
- complexité de calcul :
  - **seulement 1 lecture pour chaque serveur** (optimal)
  - $\leq n$ additions sur $\mathbb{F}$ pour User

Confidentialité et paramètres du protocole

Théorème. Ce protocole PIR est correct et sûr du point de vue IT.

Preuve :

- le seul serveur qui détient F_i reçoit une requête aléatoire ;
- pour tout autre serveur S_j, la requête q_j ne fournit aucune information sur le bloc B qui a été choisi par User $\Rightarrow$ aucune information sur i n'est révélée.

Caractéristiques.

- complexité de communication : $n \log s$ bits envoyés, $n \log q$ bits reçus
- complexité de calcul :
  - **seulement 1 lecture pour chaque serveur** (optimal)
  - $\leq n$ additions sur $\mathbb{F}$ pour User
- stockage : redondance de $(ns - k) \log q$ bits, où $k = \dim_{\mathbb{F}}(\mathcal{C})$

Confidentialité et paramètres du protocole

Théorème. Ce protocole PIR est correct et sûr du point de vue IT.

Preuve :

- le seul serveur qui détient F_i reçoit une requête aléatoire ;
- pour tout autre serveur S_j, la requête q_j ne fournit aucune information sur le bloc B qui a été choisi par User $\Rightarrow$ aucune information sur i n'est révélée.

Caractéristiques.

- complexité de communication : $n \log s$ bits envoyés, $n \log q$ bits reçus
- complexité de calcul :
  - **seulement 1 lecture pour chaque serveur** (optimal)
  - $\leq n$ additions sur $\mathbb{F}$ pour User
- stockage : redondance de $(ns - k) \log q$ bits, où $k = \dim_{\mathbb{F}}(\mathcal{C})$

Question : quels designs transversaux mènent à des codes de grande dimension ?

De bonnes instances avec des designs géométriques

Design issu de la **géométrie affine** sur le corps $\mathbb{F} = \mathbb{F}_q$.

- $X = \mathbb{F}_q^m$ pour $m \geq 2$ fixé,
- $\mathcal{G}$ une partition de X en q hyperplans $G_1, \ldots, G_q$,
- $\mathcal{B} = \{\text{droites affines } L \text{ sécantes à chaque hyperplan } G_j\}$.

De bonnes instances avec des designs géométriques

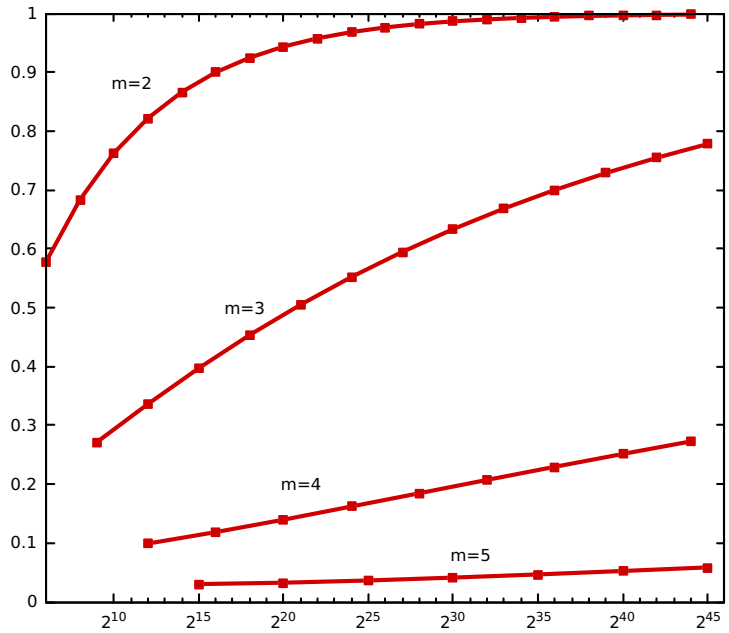
Design issu de la **géométrie affine** sur le corps $\mathbb{F} = \mathbb{F}_q$.

- $X = \mathbb{F}_q^m$ pour $m \geq 2$ fixé,
- $\mathcal{G}$ une partition de X en q hyperplans $G_1, \ldots, G_q$,
- $\mathcal{B} = \{$droites affines L sécantes à chaque hyperplan $G_j\}$.

Ce code a :

- longueur $ns = q^m$,
- "localité" $n = q$.

taux d'information k/N



longueur $N = ns = 2^{em}$

Outline

Codes de Reed–Solomon et de Reed–Muller

Définition. Le code de **Reed–Solomon** de dimension k sur le corps fini $\mathbb{F}_q$ est

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_1(f) := (f(x_1), \ldots, f(x_q)) \mid f \in \mathbb{F}_q[X] \text{ et } \deg(f) \leq k-1\}\,.$$

Codes de Reed–Solomon et de Reed–Muller

Définition. Le code de **Reed–Solomon** de dimension k sur le corps fini $\mathbb{F}_q$ est

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_1(f) := (f(x_1), \ldots, f(x_q)) \mid f \in \mathbb{F}_q[X] \text{ et } \deg(f) \leq k-1\}\,.$$

Propriété remarquable : tout symbole c_i d'un mot de code $\boldsymbol{c} \in \mathrm{RS}_q(k)$ peut être reconstruit à partir de n'importe quel sous-ensemble de k coordonnées de $\boldsymbol{c}$.

Codes de Reed–Solomon et de Reed–Muller

Définition. Le code de **Reed–Solomon** de dimension k sur le corps fini $\mathbb{F}_q$ est

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_1(f) := (f(x_1), \ldots, f(x_q)) \mid f \in \mathbb{F}_q[X] \text{ et } \deg(f) \leq k-1\}\,.$$

Propriété remarquable : tout symbole c_i d'un mot de code $\boldsymbol{c} \in \mathrm{RS}_q(k)$ peut être reconstruit à partir de n'importe quel sous-ensemble de k coordonnées de $\boldsymbol{c}$.

Définition. Le code de **Reed–Muller** d'ordre m et de degré r sur $\mathbb{F}_q$ est

$$\mathrm{RM}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \deg(f) \leq r\}\,.$$

Définition. Le code de **Reed–Solomon** de dimension k sur le corps fini $\mathbb{F}_q$ est

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_1(f) := (f(x_1), \ldots, f(x_q)) \mid f \in \mathbb{F}_q[X] \text{ et } \deg(f) \leq k-1\}\,.$$

Propriété remarquable : tout symbole c_i d'un mot de code $\boldsymbol{c} \in \mathrm{RS}_q(k)$ peut être reconstruit à partir de n'importe quel sous-ensemble de k coordonnées de $\boldsymbol{c}$.

Définition. Le code de **Reed–Muller** d'ordre m et de degré r sur $\mathbb{F}_q$ est

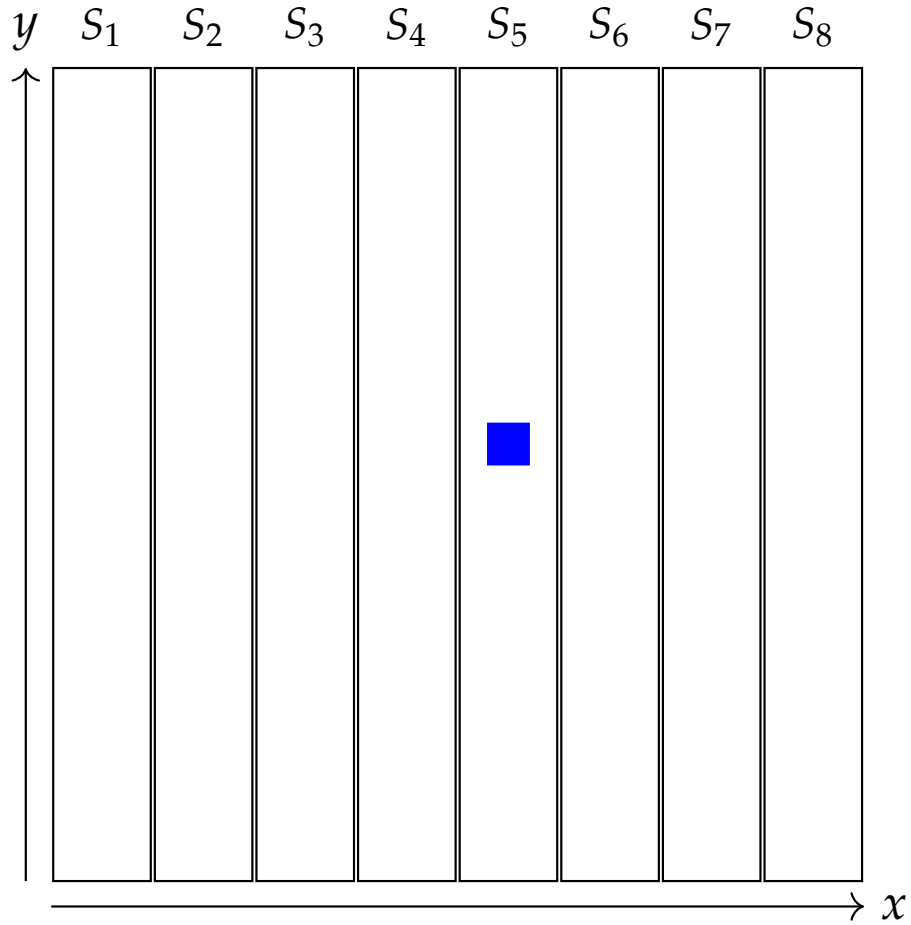
$$\mathrm{RM}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \deg(f) \leq r\}\,.$$

Propriété remarquable :

$$\forall \boldsymbol{c} = \mathrm{ev}_m(f) \in \mathrm{RM}_q(m,r) \quad \text{et} \quad \forall \text{ droite affine } L \subset \mathbb{F}_q^m,$$
$$\mathrm{ev}_1(f_{|L}) \in \mathrm{RS}_q(r+1)\,.$$

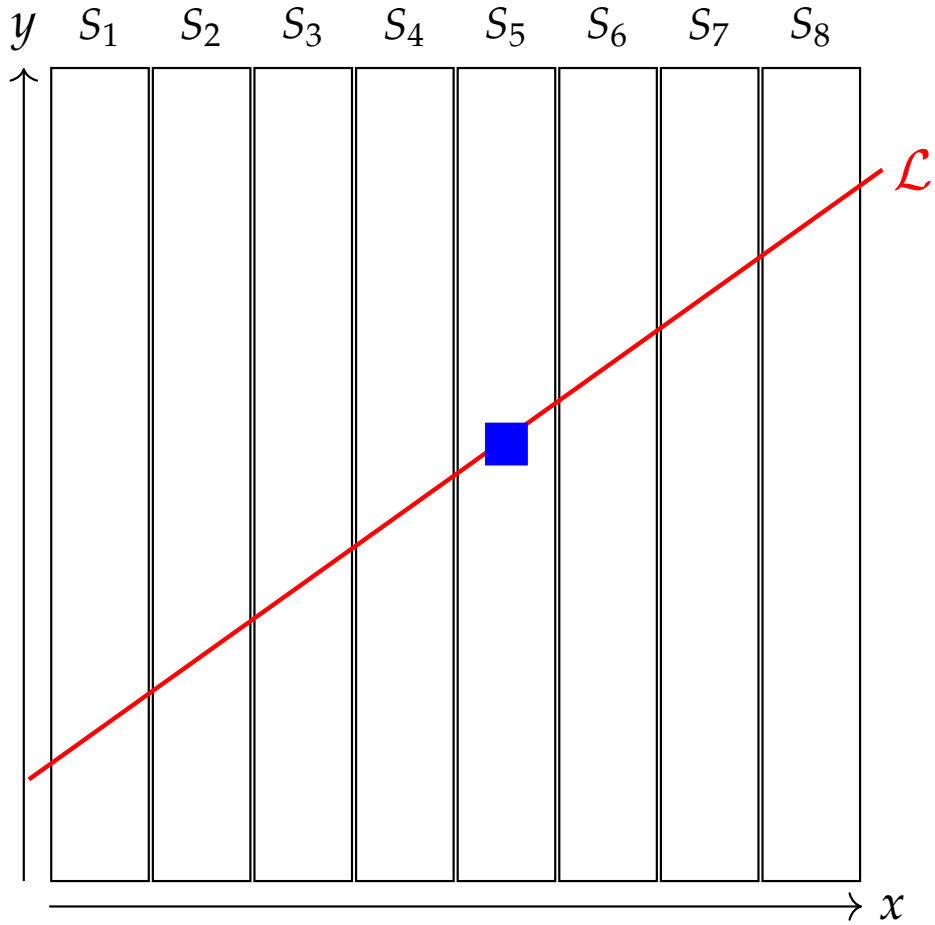
(où $f_{|L}$ est le polynôme interpolateur de f sur L)

Un protocol de PIR construit sur les codes de Reed–Muller



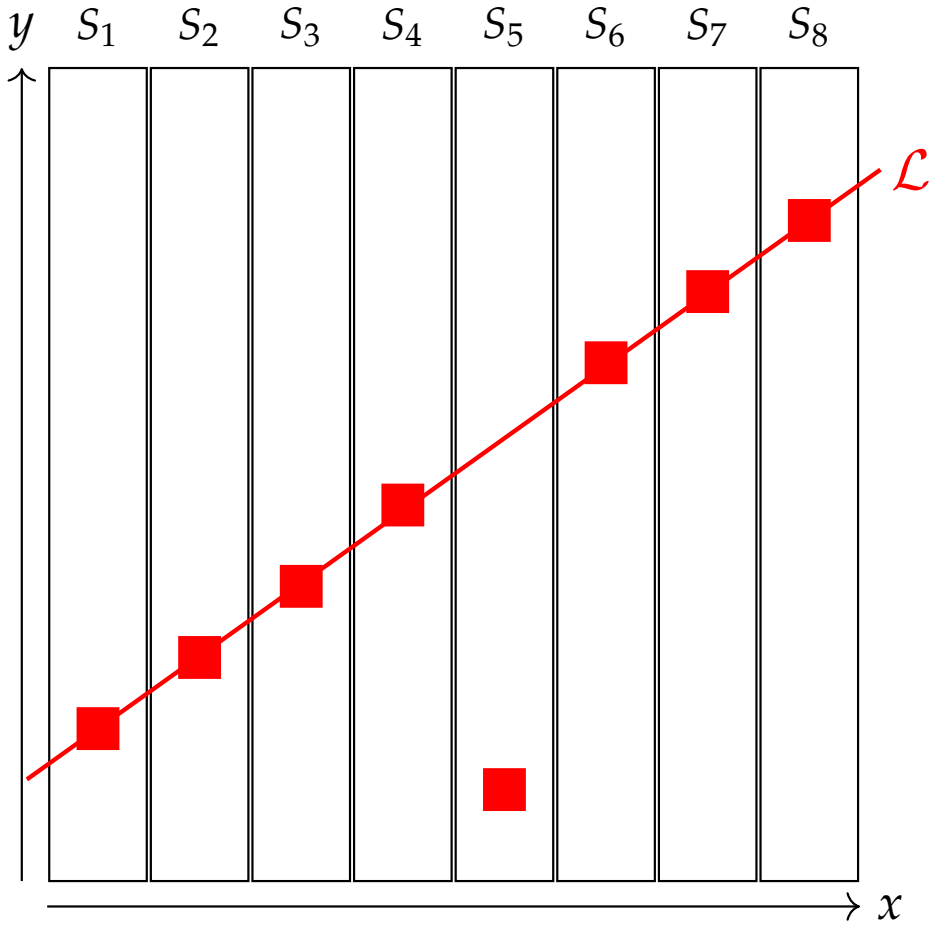
La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller



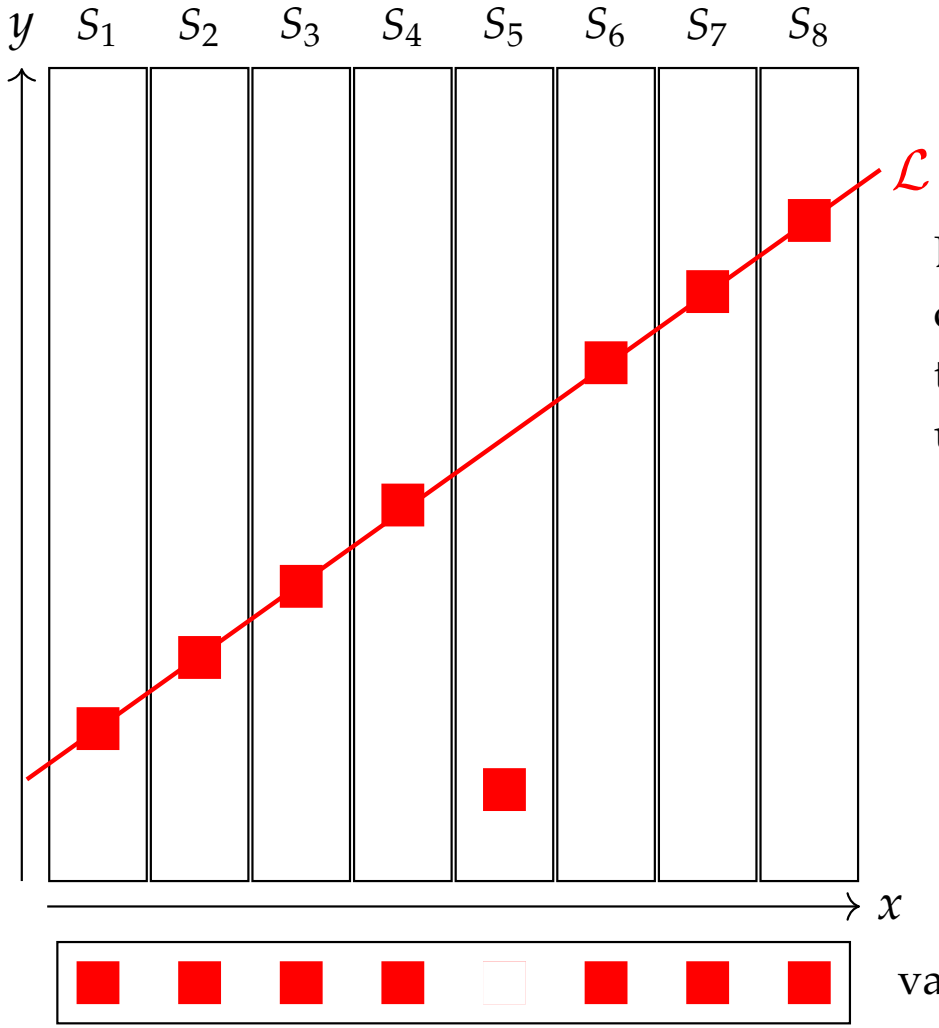
La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller



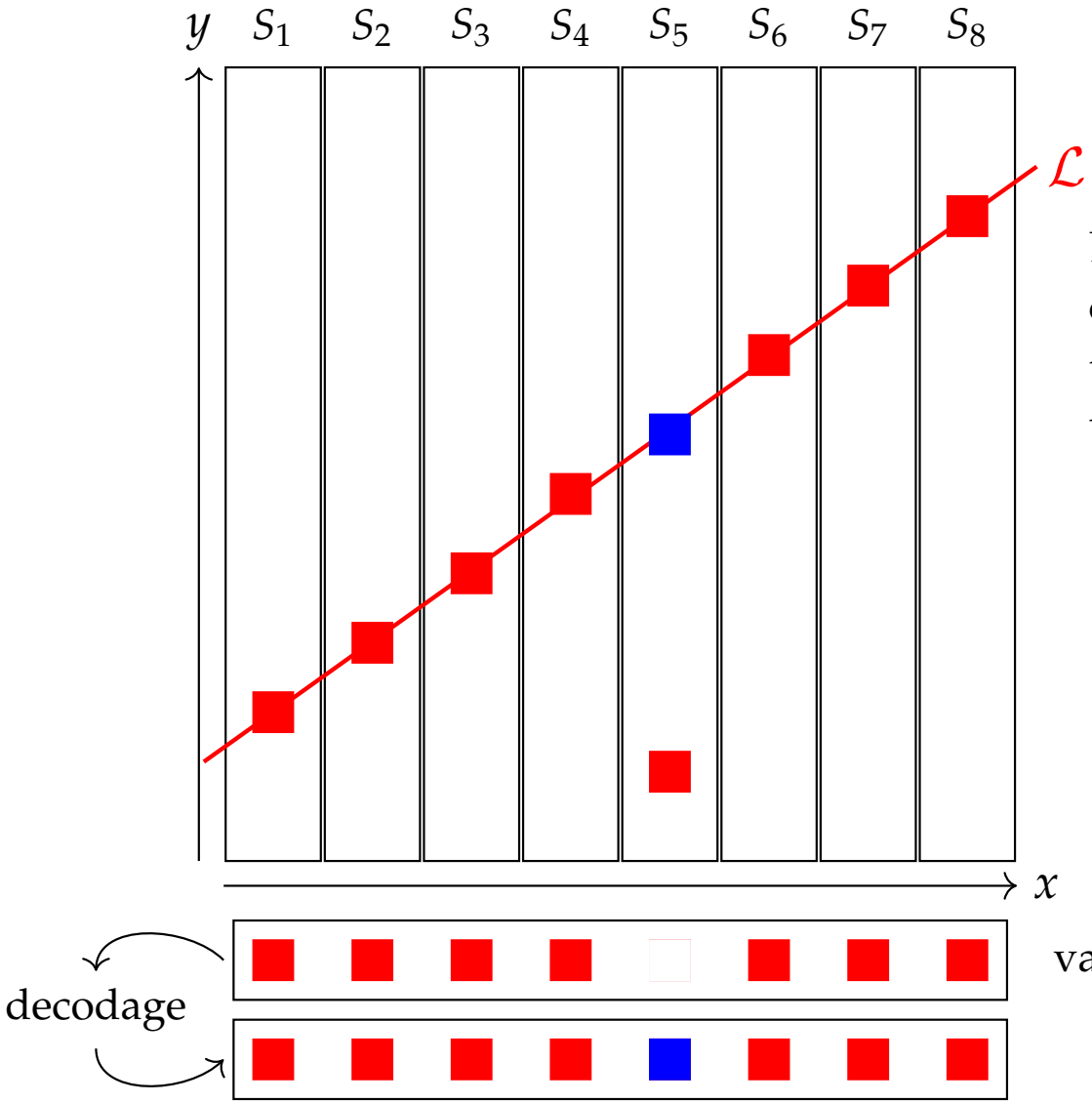
La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller



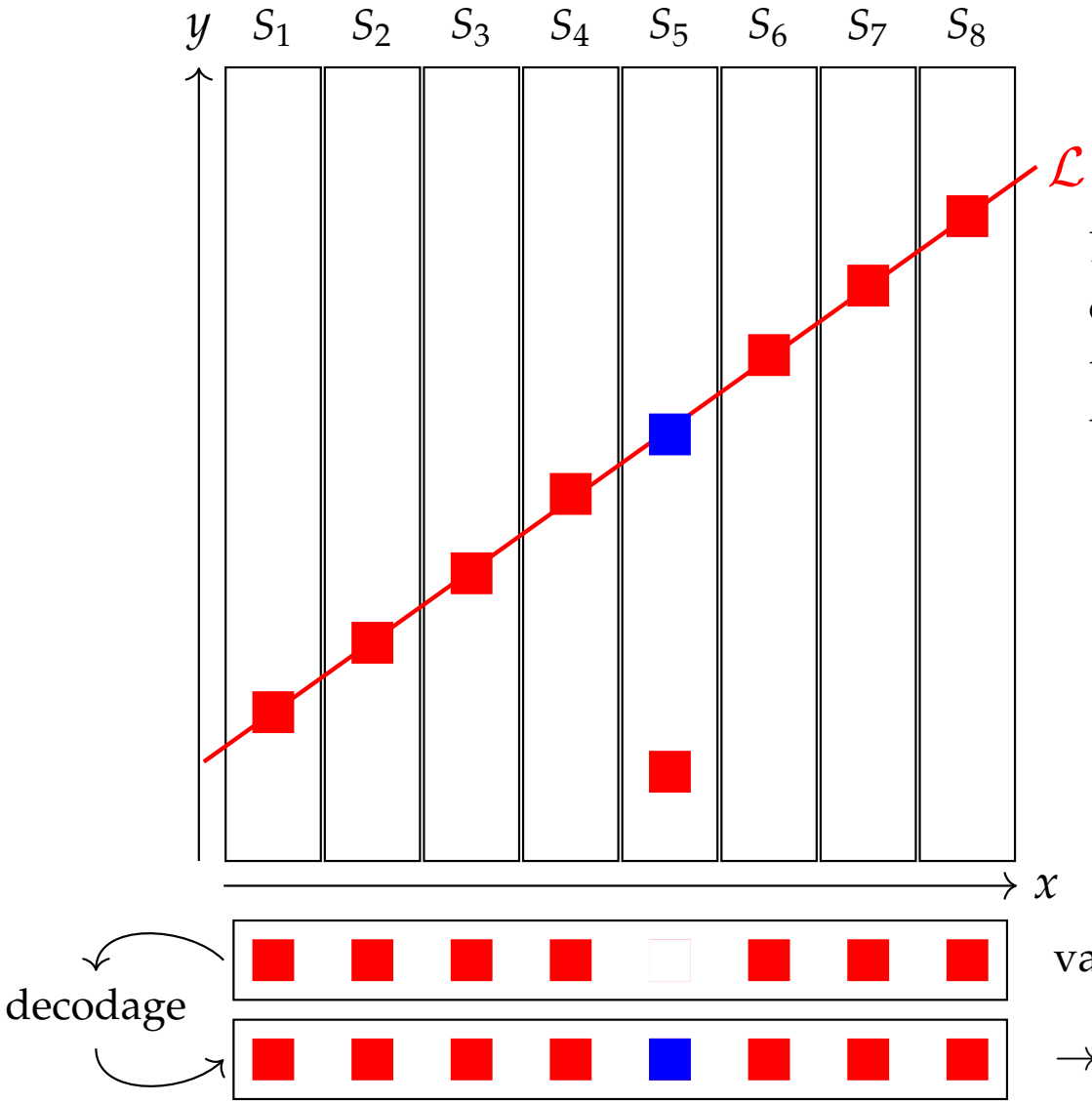
La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller



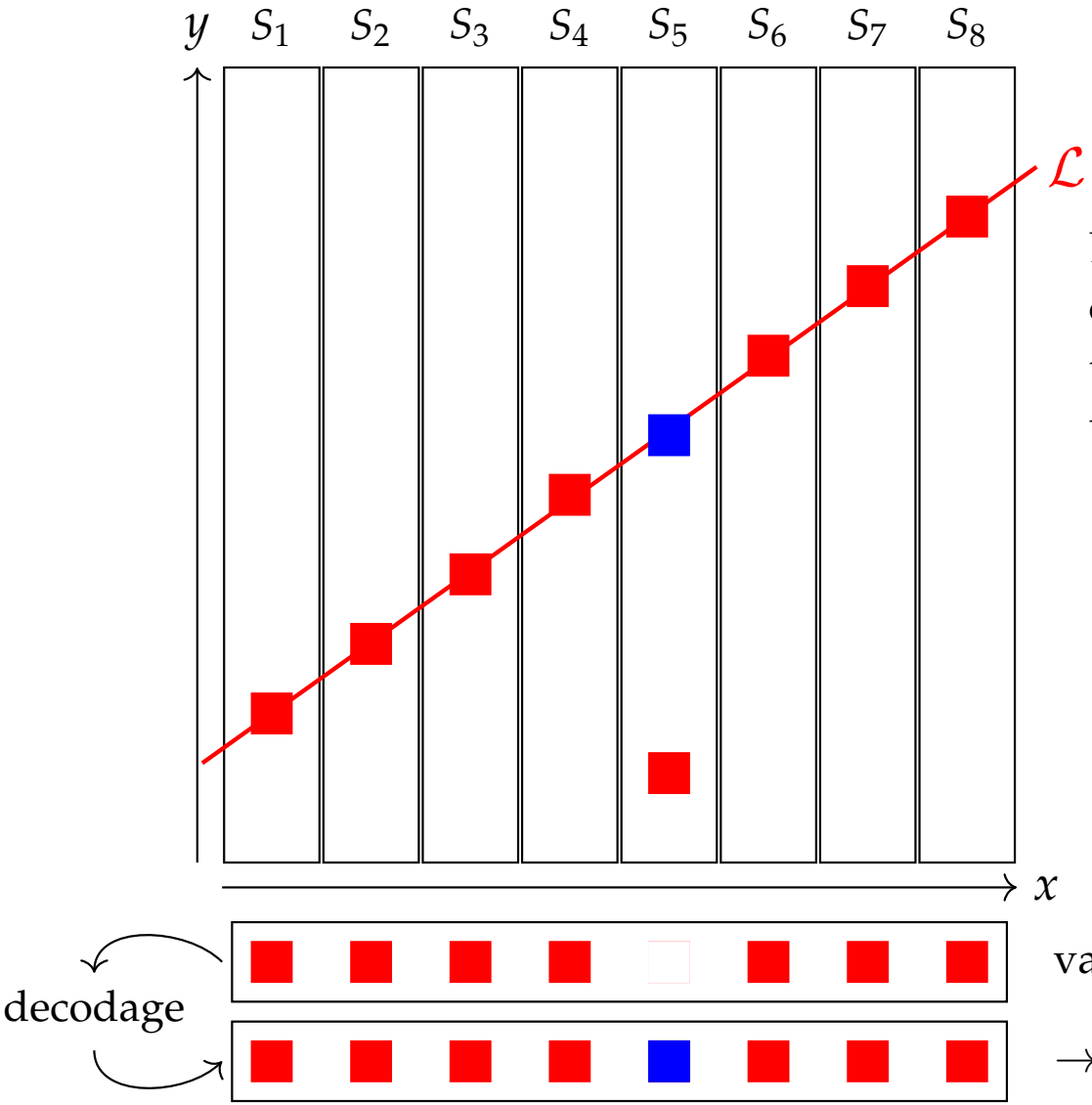
La base de données F est encodée avec $\mathrm{RM}_q(m,r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller



La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

Un protocol de PIR construit sur les codes de Reed–Muller

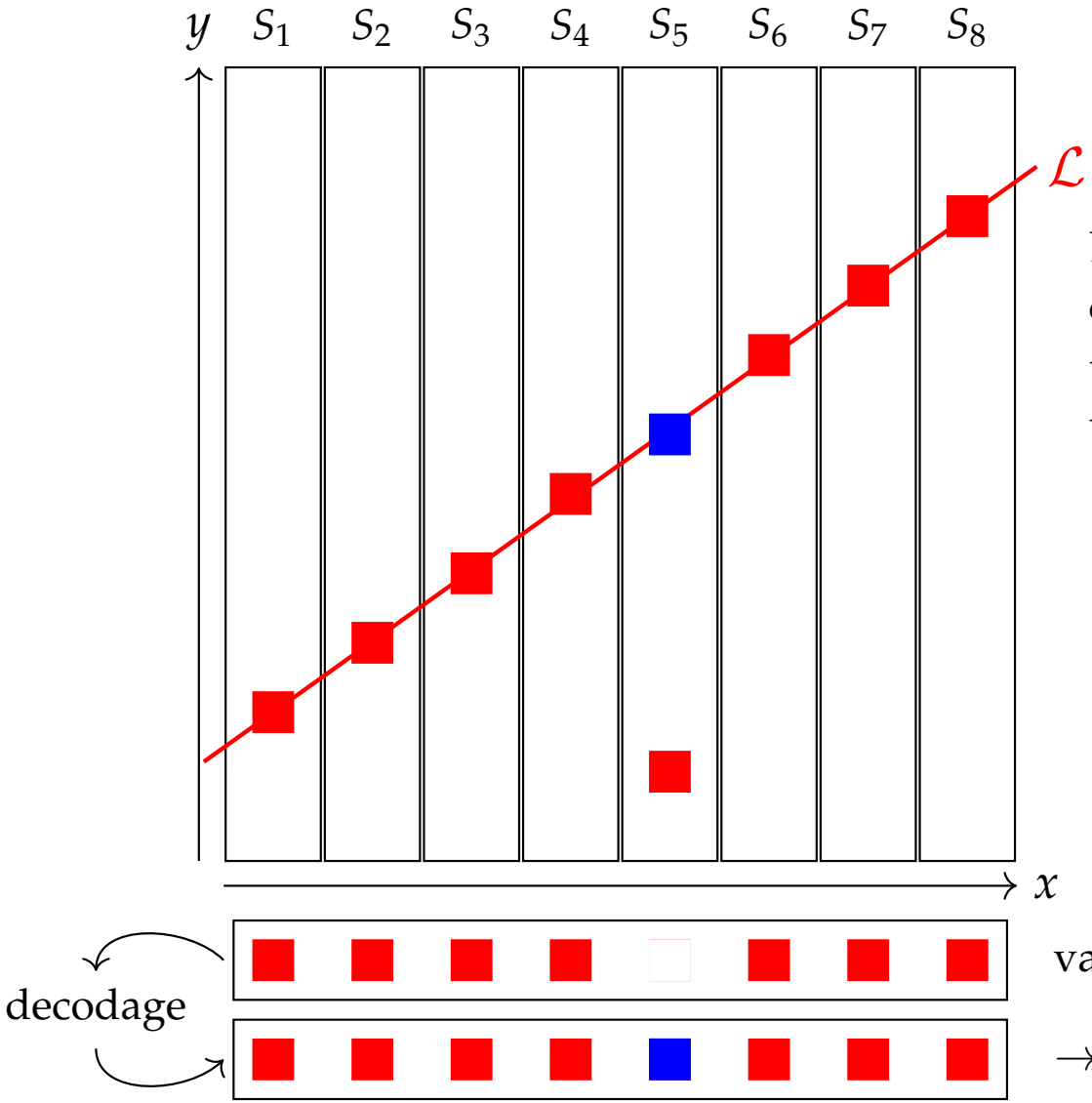


La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

$\mathcal{L}$ est une droite affine

$\Longrightarrow$ individuellement, aucun serveur ne sait où est

Un protocol de PIR construit sur les codes de Reed–Muller



La base de données F est encodée avec $\mathrm{RM}_q(m, r)$, puis distribuée sur les serveurs selon une partition en hyperplans.

$\mathcal{L}$ est une droite affine

$\Longrightarrow$ individuellement, aucun serveur ne sait où est ■

$\Longrightarrow$ le protocole est confidentiel

Paramètres du protocole

Pour le code $\mathrm{RM}_q(m,r)$ de longueur $n = q^m$.

- communication : $(m-1)q \log q$ bits envoyés, $q \log q$ bits reçus
- calcul :
  - **1 lecture pour chaque serveur** (optimal)
  - un décodage pour l'utilisateur
- stockage : le ratio k/n du code $\mathrm{RM}_q(m,r)$ avec $r \leq q-1$ est

$$\simeq \frac{(r/q)^m}{m!} \ldots$$

$\Longrightarrow$ On voudrait un code avec des propriétés similaires, mais une meilleure dimension.

Relèvements de code

Les “codes relevés” sont les plus grands codes ayant les même propriétés locales que les codes de Reed–Muller.

New affine-invariant codes from lifting. Guo, Kopparty, Sudan. ITCS. **2013**.

Relèvements de code

Les “codes relevés” sont les plus grands codes ayant les même propriétés locales que les codes de Reed–Muller.

New affine-invariant codes from lifting. Guo, Kopparty, Sudan. ITCS. **2013**.

Définition. Le code de Reed–Solomon **relevé** m **fois** de degré r sur $\mathbb{F}_q$ est :

$$\mathrm{Lift}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \forall \text{ droite affine } L \subset \mathbb{A}^m, \deg(f_{|L}) \leq r\}\,.$$

Relèvements de code

Les “codes relevés” sont les plus grands codes ayant les même propriétés locales que les codes de Reed–Muller.

New affine-invariant codes from lifting. Guo, Kopparty, Sudan. ITCS. **2013**.

Définition. Le code de Reed–Solomon **relevé** m **fois** de degré r sur $\mathbb{F}_q$ est :

$$\mathrm{Lift}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \forall \text{ droite affine } L \subset \mathbb{A}^m, \deg(f_{|L}) \leq r\}\,.$$

Ces codes sont parfois bien plus grands que les codes de Reed-Muller codes (c’est surprenant !).

Relèvements de code

Les “codes relevés” sont les plus grands codes ayant les même propriétés locales que les codes de Reed–Muller.

New affine-invariant codes from lifting. Guo, Kopparty, Sudan. ITCS. **2013**.

Définition. Le code de Reed–Solomon **relevé m fois** de degré r sur $\mathbb{F}_q$ est :

$$\mathrm{Lift}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \forall \text{ droite affine } L \subset \mathbb{A}^m, \deg(f_{|L}) \leq r\}\,.$$

Ces codes sont parfois bien plus grands que les codes de Reed-Muller codes (c’est surprenant !).

Exemple. Pour $m = 2$, $r = 2$, on considère $f(X,Y) = X^2Y^2$ sur $\mathbb{F}_4$.

$$f(aT+b, cT+d) \equiv (a^2d^2 + b^2c^2)T^2 + a^2c^2T + b^2d^2 \mod (T^4 - T)$$

Alors,

$$\mathrm{ev}_2(X^2Y^2) \in \mathrm{Lift}_4(2,2) \quad \text{mais} \quad \mathrm{ev}_2(X^2Y^2) \notin \mathrm{RM}_4(2,2)\,.$$

Relèvements de code

Les "codes relevés" sont les plus grands codes ayant les même propriétés locales que les codes de Reed–Muller.

New affine-invariant codes from lifting. Guo, Kopparty, Sudan. ITCS. **2013**.

Définition. Le code de Reed–Solomon **relevé** m **fois** de degré r sur $\mathbb{F}_q$ est :

$$\mathrm{Lift}_q(m,r) := \{\mathrm{ev}_m(f) \mid f \in \mathbb{F}_q[\mathbf{X}] \text{ et } \forall \text{ droite affine } L \subset \mathbb{A}^m, \deg(f_{|L}) \leq r\}\,.$$

Ces codes sont parfois bien plus grands que les codes de Reed-Muller codes (c'est surprenant !).

Exemple. Pour $m = 2$, $r = 2$, on considère $f(X,Y) = X^2Y^2$ sur $\mathbb{F}_4$.

$$f(aT+b, cT+d) \equiv (a^2d^2 + b^2c^2)T^2 + a^2c^2T + b^2d^2 \mod (T^4 - T)$$

Alors,

$$\mathrm{ev}_2(X^2Y^2) \in \mathrm{Lift}_4(2,2) \quad \text{mais} \quad \mathrm{ev}_2(X^2Y^2) \notin \mathrm{RM}_4(2,2)\,.$$

Prop. Pour tout m fixé, les codes relevés peuvent atteindre un ratio k/n arbitrairement proche de 1.

Relèvement à poids

Question : comment gérer les coalitions de serveurs ?

Relèvement à poids

Question : comment gérer les coalitions de serveurs ?

Simplifions et prenons $m = 2$.

Définition. Une t-courbe est :

$$\mathcal{L} = \{(x, g(x)) \in \mathbb{F}_q^2 \mid g \in \mathbb{F}_q[X], \deg(g) \leq t\}$$

Relèvement à poids

Question : comment gérer les coalitions de serveurs ?

Simplifions et prenons $m = 2$.

Définition. Une t-courbe est :

$$\mathcal{L} = \{(x, g(x)) \in \mathbb{F}_q^2 \mid g \in \mathbb{F}_q[X], \deg(g) \leq t\}$$

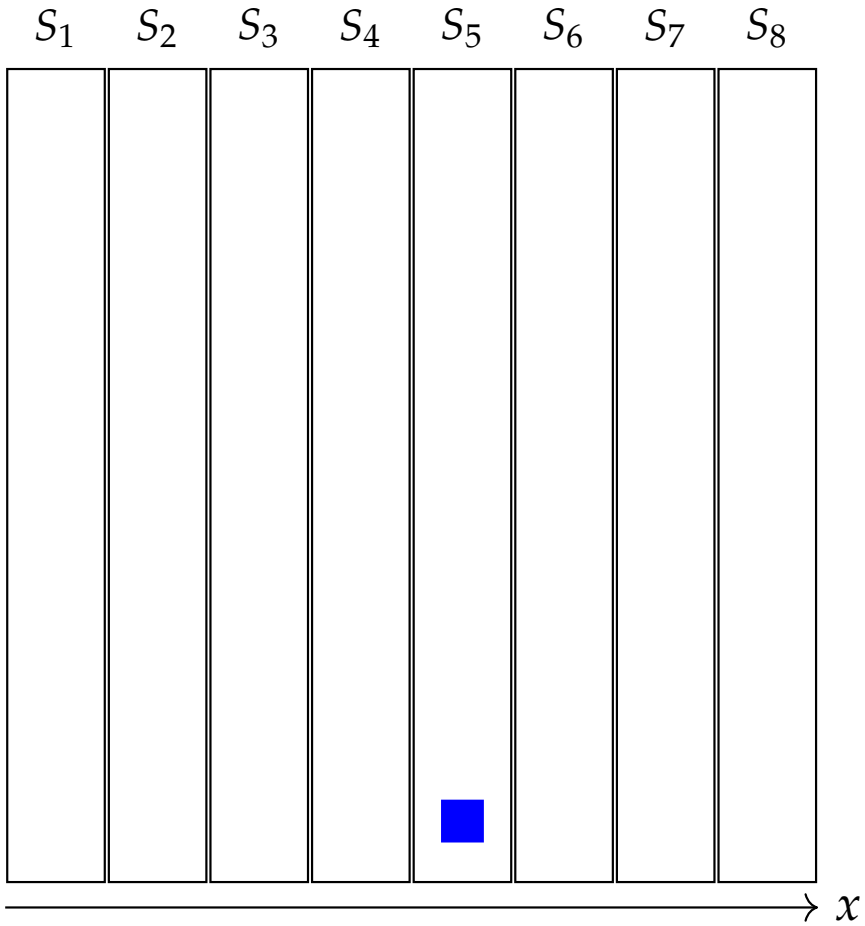
Définition. Le **relèvement pondéré du code de Reed–Solomon code** de degré r et de poids t sur $\mathbb{F}_q$ est :

$$\mathrm{WLift}_q(t, r) := \{\mathrm{ev}_2(f) \mid f \in \mathbb{F}_q[X, Y] \text{ et } \forall\, t\text{-courbe } \mathcal{L} \subset \mathbb{A}^2, \deg(f_{|\mathcal{L}}) \leq r\}$$

Conséquence : pour tout mot $\boldsymbol{c} \in \mathrm{WLift}_q(t, r)$ et pour toute t-courbe $\mathcal{L}$, on a :

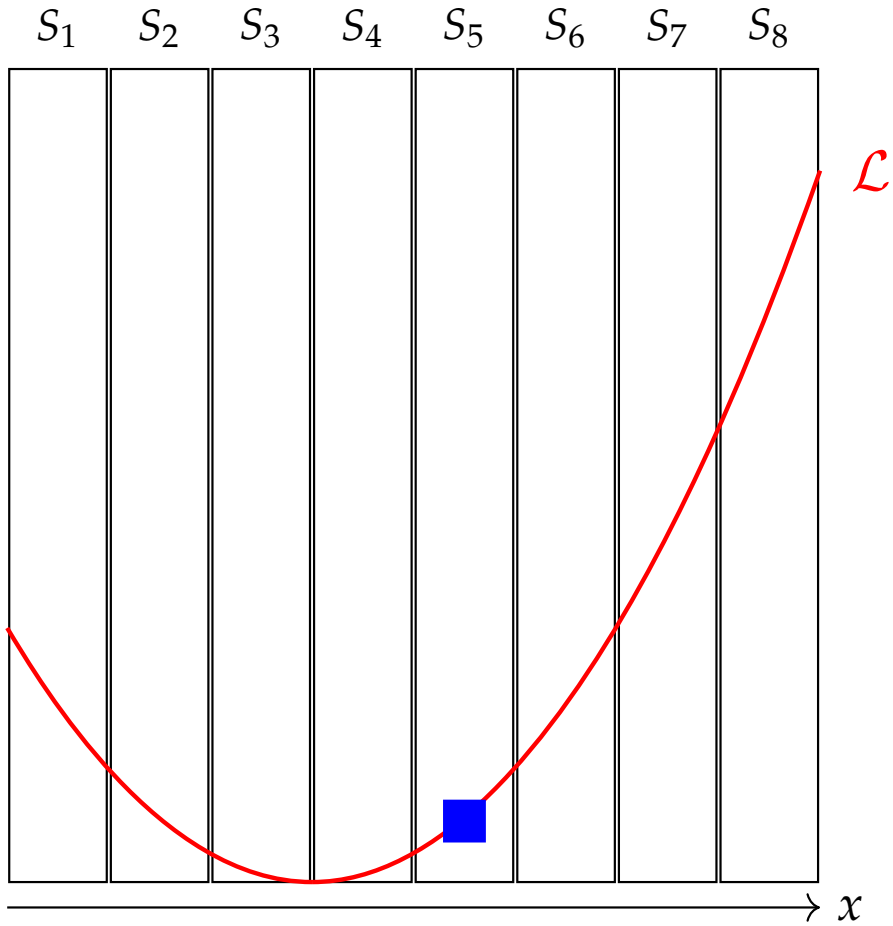
$$\boldsymbol{c}_{|\mathcal{L}} \in \mathrm{RS}_q(r + 1)\,.$$

Un protocole de PIR construit sur des relèvements pondérés



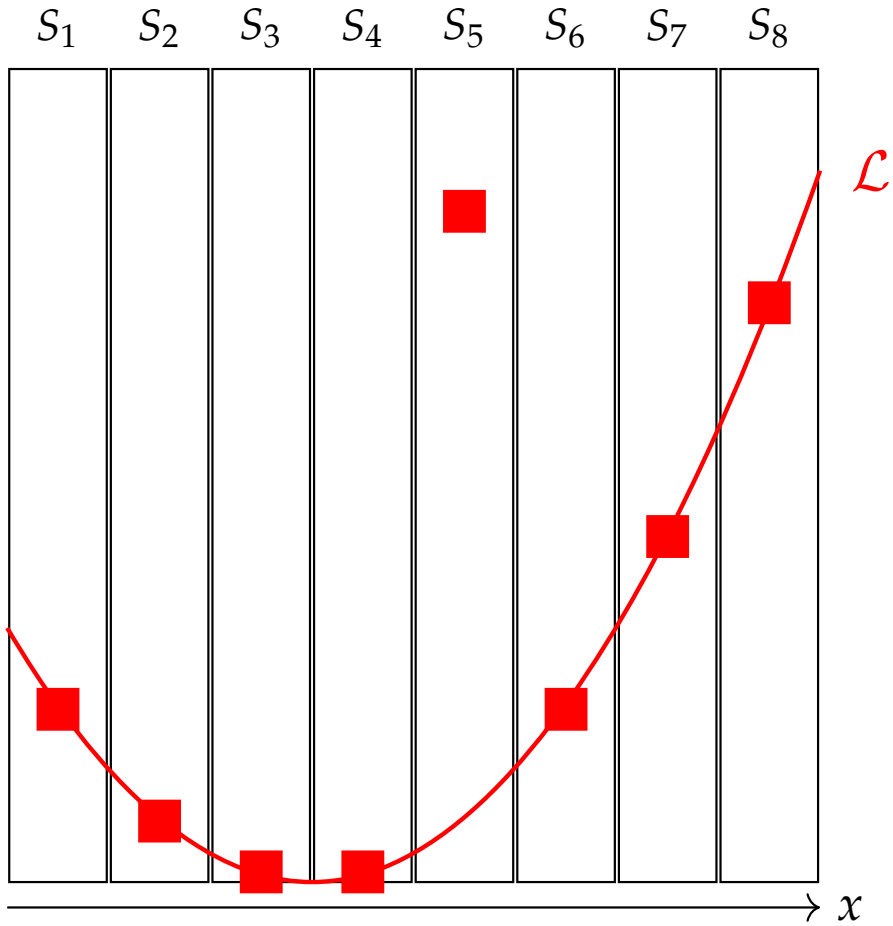
La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



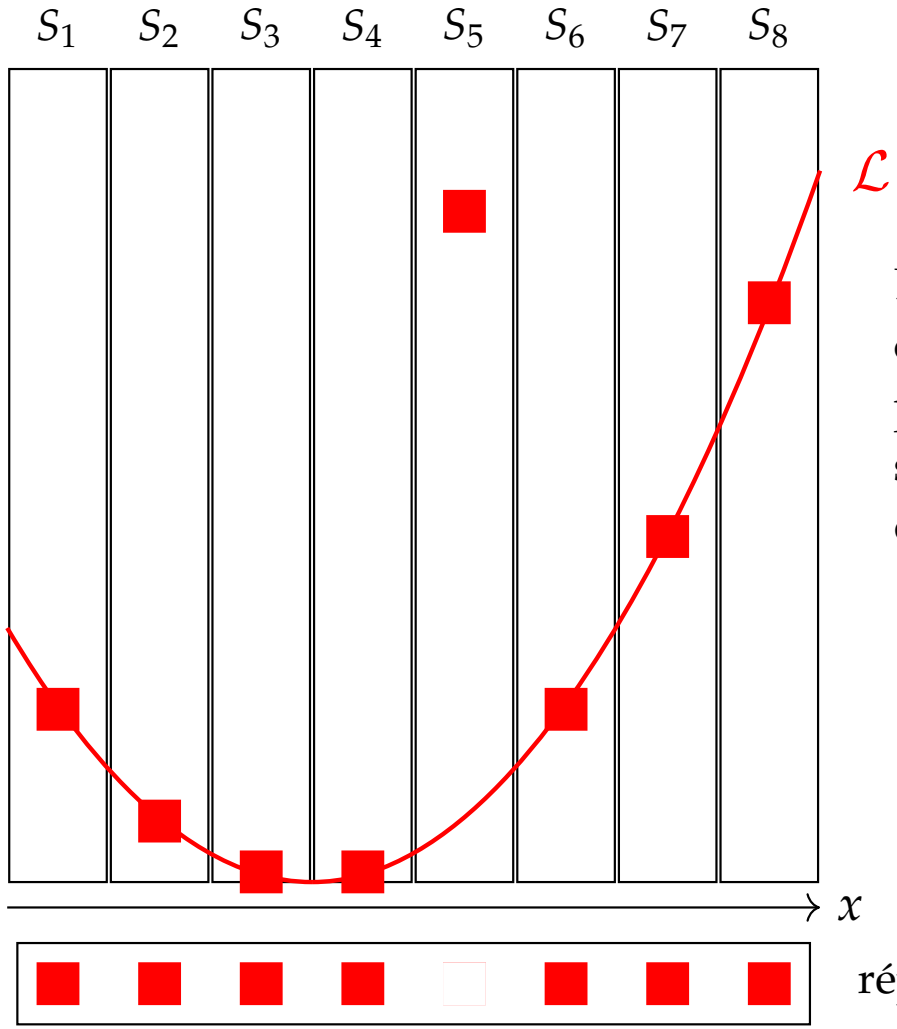
La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



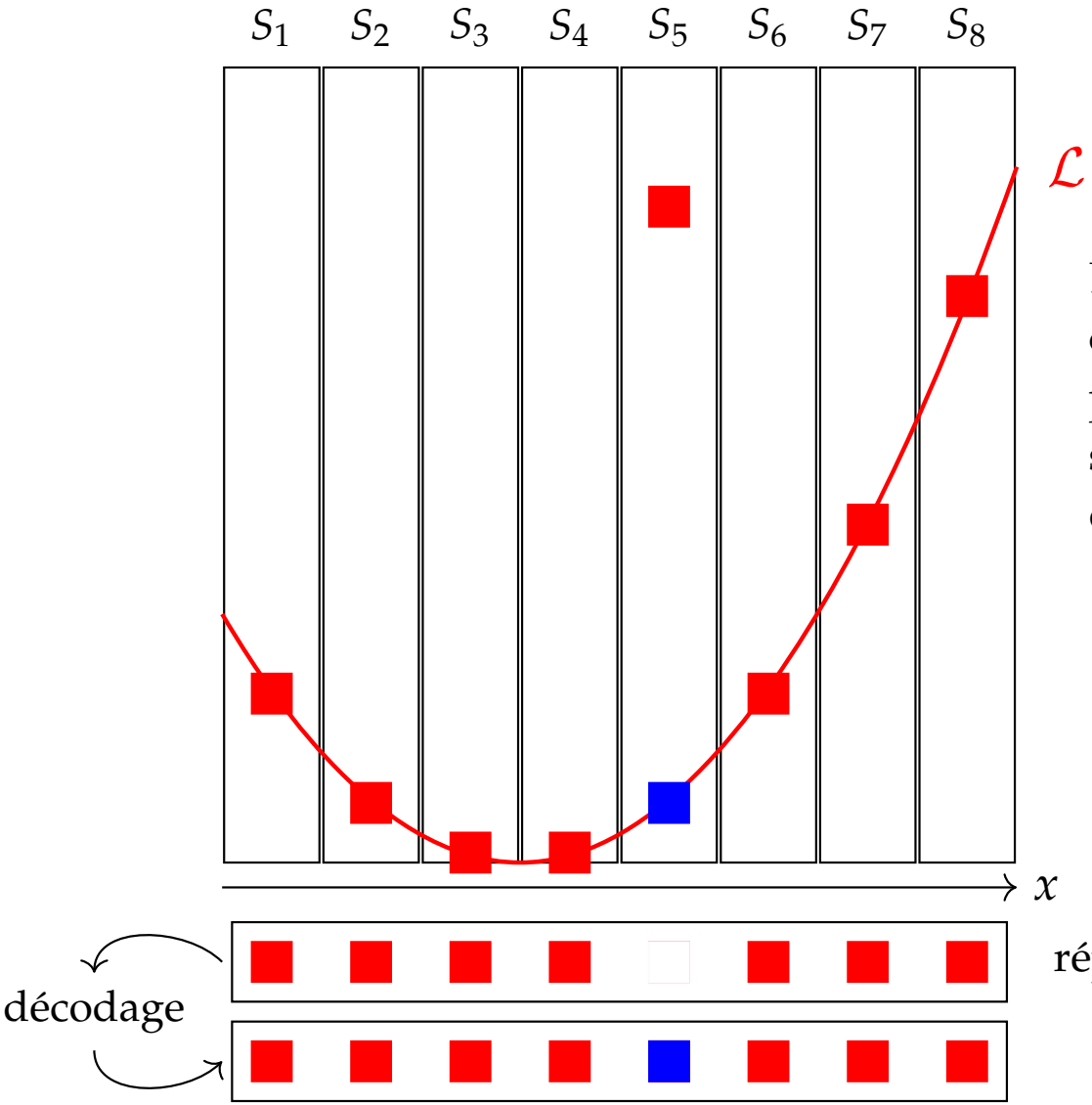
La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



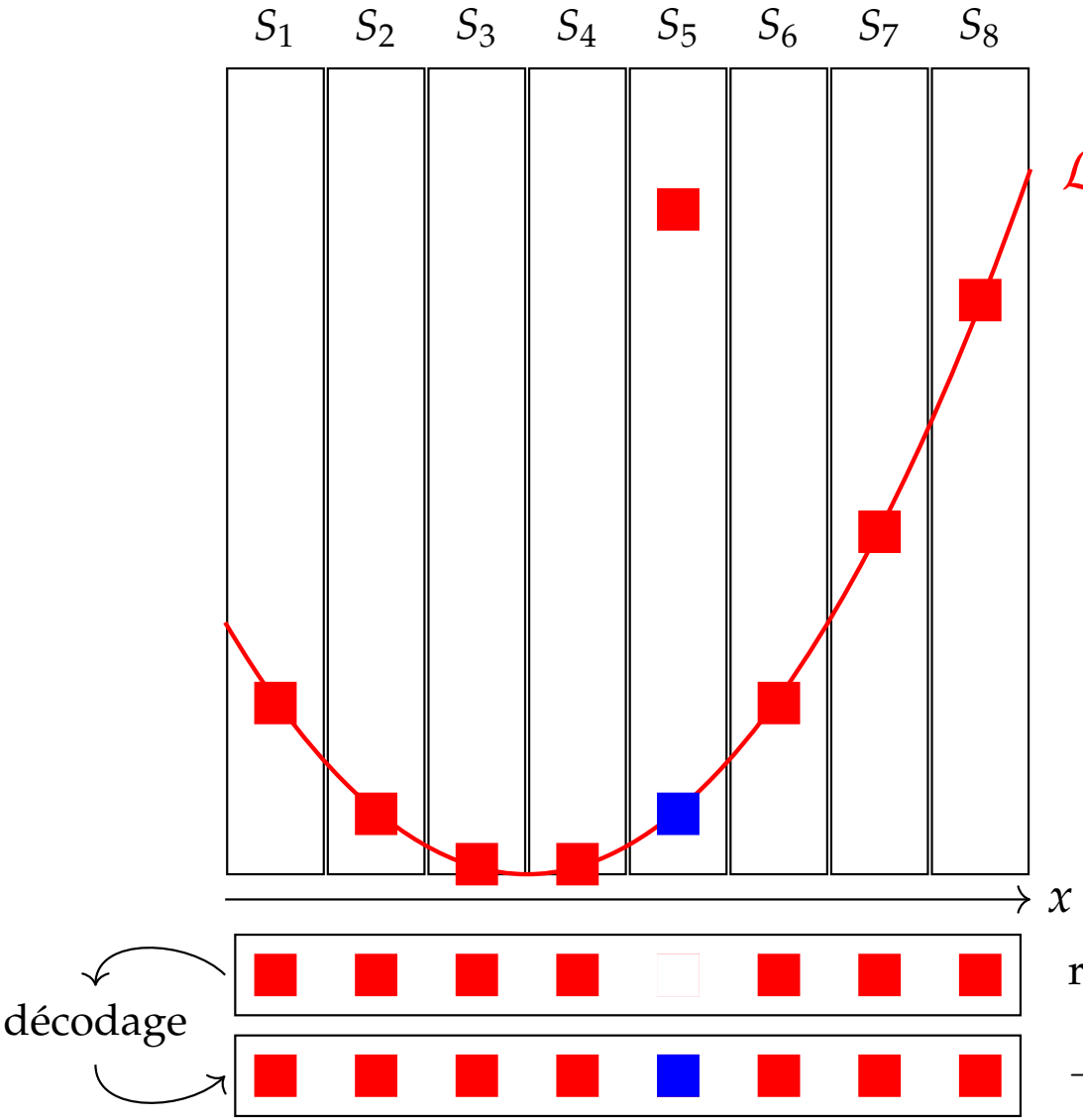
La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



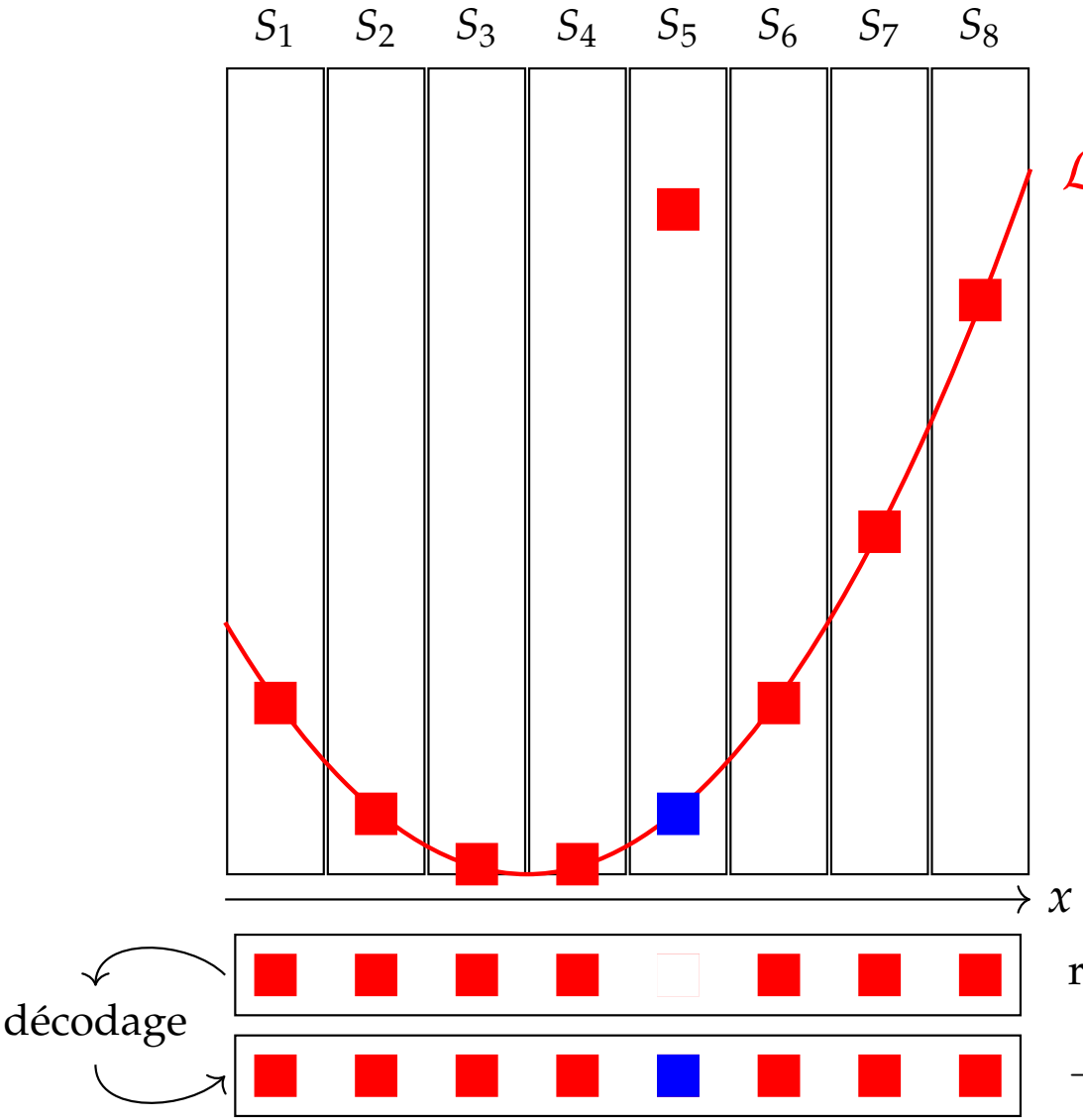
La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

Un protocole de PIR construit sur des relèvements pondérés



La base de données F est encodée avec le code $\mathrm{WLift}_q(t, r)$, puis distribuée parmi les serveurs selon une partition en droites verticales

$\mathcal{L}$ est une t-courbe

$\Longrightarrow$ aucune coalition de t serveurs ne peut deviner où est ■

Quelques résultats

Weighted Lifted Codes: Local Correctabilities and Application to Robust Private Information Retrieval. **L.**, Nardi. IEEE TIT. **2021**.

Théorème. Fixons p un nombre premier, $t \geq 1$ et $\alpha \geq 2$ des entiers. Alors, le ratio k/n du code $\mathrm{WLift}_{p^e}(t, p^e - \alpha)$ tend vers 1 pour $e \to \infty$.

Corollaire : pour un nombre de serveurs grandissant mais des tailles de coalitions constantes, on obtient des protocoles de PIR avec une redondance de stockage qui tend vers 0.

Quelques résultats

Weighted Lifted Codes: Local Correctabilities and Application to Robust Private Information Retrieval. **L.**, Nardi. IEEE TIT. **2021**.

Théorème. Fixons p un nombre premier, $t \geq 1$ et $\alpha \geq 2$ des entiers. Alors, le ratio k/n du code $\mathrm{WLift}_{p^e}(t, p^e - \alpha)$ tend vers 1 pour $e \to \infty$.

Corollaire : pour un nombre de serveurs grandissant mais des tailles de coalitions constantes, on obtient des protocoles de PIR avec une redondance de stockage qui tend vers 0.

Théorème. Fixons p un nombre premier, $t \geq 1$ et $c \geq 1$ des entiers. Notons $\gamma = 1 - p^{-c}$ et $\mathcal{C}_e = \mathrm{WLift}_{p^e}(t, \gamma p^e)$. Alors, le ratio $R_e = k/n$ du code $\mathcal{C}_e$ vérifie

$$\lim_{e \to \infty} R_e = K_{t,p,c} > 0.$$

Corollaire : pour un nombre de serveurs grandissant avec une fraction constante de serveurs en coalition et/ou en échec, on obtient des protocoles de PIR avec une redondance de stockage bornée par une constante.

Conclusion

Autres constructions de protocoles de PIR :

- sur des bases de données dont l'encodage n'est pas maîtrisé par l'utilisateur (RAID, Hadoop, Azure etc.),
- avec des patterns de coalitions particuliers,
- symétrique : l'utilisateur n'apprend pas plus que F_i,
- etc.

Conclusion

Autres constructions de protocoles de PIR :

- sur des bases de données dont l'encodage n'est pas maîtrisé par l'utilisateur (RAID, Hadoop, Azure etc.),
- avec des patterns de coalitions particuliers,
- symétrique : l'utilisateur n'apprend pas plus que F_i,
- etc.

Autres applications de la notion de localité pour des protocoles cryptographiques :

- preuves d'extractibilité d'une donnée,
- preuves (non-)interactives d'intégrité,
- etc.