Private information retrieval with codes

Julien Lavauzelle

LAGA, Université Paris 8 Vincennes – Saint-Denis

Seminario de Álgebra, Geometría algebraica y Singularidades
La Laguna

31/10/2023

Outline

1. Private information retrieval

2. PIR schemes with low computation and storage
   Transversal designs and codes
   A PIR scheme with transversal designs
   Collusion-resistant PIR schemes with weighted lifted codes

Outline

1. Private information retrieval

2. PIR schemes with low computation and storage
   - Transversal designs and codes
   - A PIR scheme with transversal designs
   - Collusion-resistant PIR schemes with weighted lifted codes

Private information retrieval (PIR):

Consider a set of files $F_1, \ldots, F_k$, stored on a remote system.

One wants to retrieve one file F_i **privately**, that is, by hiding the value of i to the system.

Is it possible?

Problem statement

Private information retrieval (PIR):

Consider a set of files $F_1, \dots, F_k$, stored on a remote system.

One wants to retrieve one file F_i **privately**, that is, by hiding the value of i to the system.

Is it possible?

Applications: access to medical data, geoprivacy, ...

Private information retrieval (PIR):

Consider a set of files $F_1, \ldots, F_k$, stored on a remote system.

One wants to retrieve one file F_i **privately**, that is, by hiding the value of i to the system.

Is it possible?

Applications: access to medical data, geoprivacy, ...

Trivial solution: download all files.

- perfect privacy: no information
- bad download rate...

Definition of PIR

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

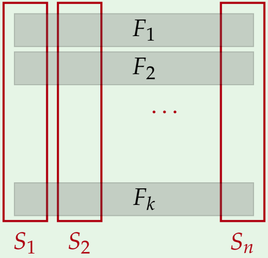
Generally, assume that n servers $S_1, \dots, S_n$ store (in some way) the files $F_1, \dots, F_k$.

Definition of PIR

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

Generally, assume that n servers $S_1, \ldots, S_n$ store (in some way) the files $F_1, \ldots, F_k$.

A **Private Information Retrieval protocol** is a set of algorithms $(\mathcal{Q}, \mathcal{A}, \mathcal{R})$.



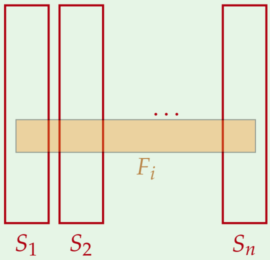
Definition of PIR

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

Generally, assume that n servers $S_1, \dots, S_n$ store (in some way) the files $F_1, \dots, F_k$.

A **Private Information Retrieval protocol** is a set of algorithms $(\mathcal{Q}, \mathcal{A}, \mathcal{R})$.

In order to retrieve file F_i:



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

Generally, assume that n servers $S_1, \dots, S_n$ store (in some way) the files $F_1, \dots, F_k$.

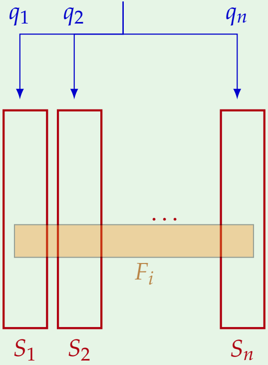
A **Private Information Retrieval protocol** is a set of algorithms $(\mathcal{Q}, \mathcal{A}, \mathcal{R})$.

In order to retrieve file F_i:

1. **Query generation:**

$$\boldsymbol{q} := (q_1, \dots, q_n) \leftarrow \mathcal{Q}(i)$$

Send query q_j to server S_j.



Definition of PIR

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

Generally, assume that n servers $S_1, \dots, S_n$ store (in some way) the files $F_1, \dots, F_k$.

A **Private Information Retrieval protocol** is a set of algorithms $(\mathcal{Q}, \mathcal{A}, \mathcal{R})$.

In order to retrieve file F_i:

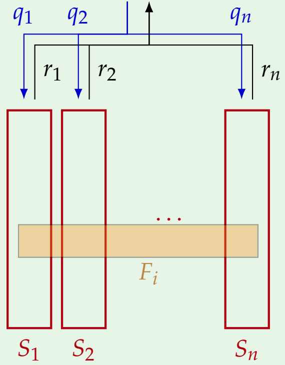
1. **Query generation:**

$$\boldsymbol{q} := (q_1, \dots, q_n) \leftarrow \mathcal{Q}(i)$$

   Send query q_j to server S_j.

2. **Response:** server S_j computes and sends back a response

$$r_j := \mathcal{A}(q_j, F_{|S_j})$$



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

Generally, assume that n servers $S_1, \dots, S_n$ store (in some way) the files $F_1, \dots, F_k$.

A **Private Information Retrieval protocol** is a set of algorithms $(\mathcal{Q}, \mathcal{A}, \mathcal{R})$.

In order to retrieve file F_i:

1. **Query generation:**

$$\boldsymbol{q} := (q_1, \dots, q_n) \leftarrow \mathcal{Q}(i)$$

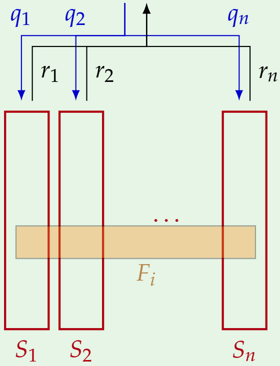
   Send query q_j to server S_j.

2. **Response:** server S_j computes and sends back a response

$$r_j := \mathcal{A}(q_j, F_{|S_j})$$

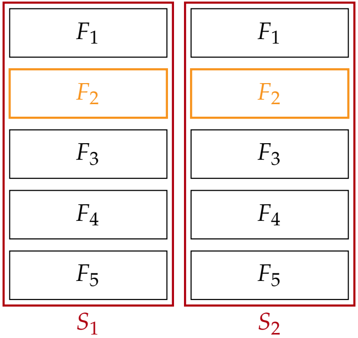
3. **Local reconstruction** of the desired file:

$$F_i = \mathcal{R}(\boldsymbol{q}, \boldsymbol{r}, i)\,.$$



$n = 2$ servers storing a replica of $k = 5$ files $F_1, \ldots, F_5$

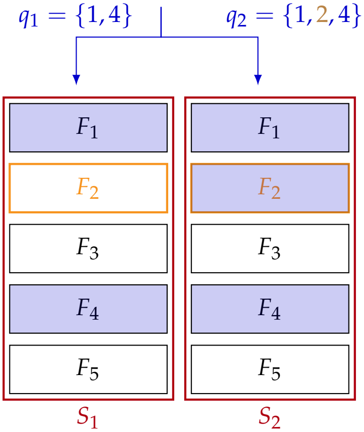
Goal: retrieve file F_2.



$n = 2$ servers storing a replica of $k = 5$ files $F_1, \ldots, F_5$

Goal: retrieve file F_2.

1. **Query generation.** Pick at random a subset $I \subseteq \{1, \ldots, 5\}$, and define:
   - query $q_1 = I$ to server S_1
   - query $q_2 = I \,\Delta\, \{2\}$ to server S_2

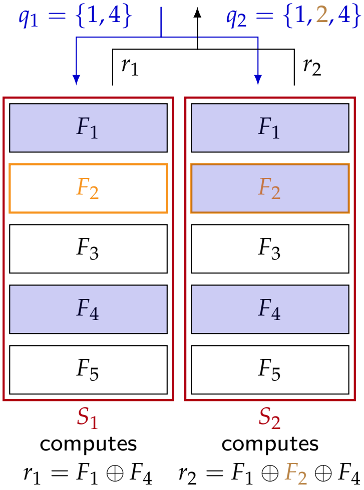


Example

$n = 2$ servers storing a replica of $k = 5$ files $F_1, \dots, F_5$

Goal: retrieve file F_2.

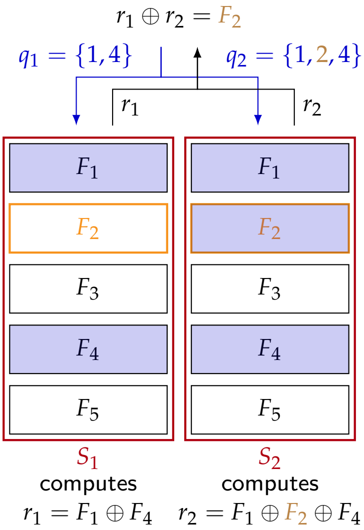
1. **Query generation.** Pick at random a subset $I \subseteq \{1, \dots, 5\}$, and define:
   - query $q_1 = I$ to server S_1
   - query $q_2 = I \,\Delta\, \{2\}$ to server S_2
2. **Responses.** Each server receives a subset $J \subseteq \{1, \dots, 5\}$, and computes the XOR (=bitwise sum) of files indexed by J.



$n = 2$ servers storing a replica of $k = 5$ files $F_1, \ldots, F_5$

Goal: retrieve file F_2.

1. **Query generation.** Pick at random a subset $I \subseteq \{1, \ldots, 5\}$, and define:
   - query $q_1 = I$ to server S_1
   - query $q_2 = I \,\Delta\, \{2\}$ to server S_2
2. **Responses.** Each server receives a subset $J \subseteq \{1, \ldots, 5\}$, and computes the XOR (=bitwise sum) of files indexed by J.
3. **Reconstruction.** One gets file F_2 by XORing the two responses.

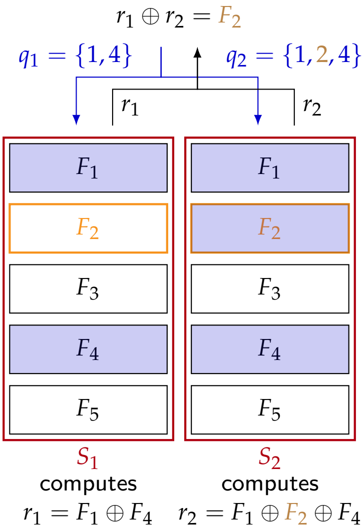


$n = 2$ servers storing a replica of $k = 5$ files $F_1, \ldots, F_5$

Goal: retrieve file F_2.

1. **Query generation.** Pick at random a subset $I \subseteq \{1, \ldots, 5\}$, and define:
   - query $q_1 = I$ to server S_1
   - query $q_2 = I \,\Delta\, \{2\}$ to server S_2
2. **Responses.** Each server receives a subset $J \subseteq \{1, \ldots, 5\}$, and computes the XOR (=bitwise sum) of files indexed by J.
3. **Reconstruction.** One gets file F_2 by XORing the two responses.

Upload: $2 \times 5 = 10$ bits to transmit to the servers
Download: $2|F|$ bits to receive from the servers
Server computation: XOR of $\frac{5}{2}|F|$ bits in average
Client computation: XOR of $2|F|$ bits



Privacy

The adversary: a **collusion of servers** = a subset of servers $\{S_j : j \in T\}$, where $T \subset [1, n]$, which exchange information about queries.

$$t := \max\{|T|, T \subseteq [1, n] \text{ is a collusion}\} \geq 1$$

Privacy

The adversary: a **collusion of servers** = a subset of servers $\{S_j : j \in T\}$, where $T \subset [1, n]$, which exchange information about queries.

$$t := \max\{|T|, T \subseteq [1, n] \text{ is a collusion}\} \geq 1$$

- **Information-theoretic (IT) privacy:**

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

- **Computational privacy:** by varying the index i, distributions of queries $\boldsymbol{q}_{|T} = \mathcal{Q}(i)_{|T}$ are computationally indistinguishable.

Privacy

The adversary: a **collusion of servers** = a subset of servers $\{S_j : j \in T\}$, where $T \subset [1, n]$, which exchange information about queries.

$$t := \max\{|T|, T \subseteq [1, n] \text{ is a collusion}\} \geq 1$$

- **Information-theoretic (IT) privacy:**

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

- **Computational privacy:** by varying the index i, distributions of queries $\boldsymbol{q}_{|T} = \mathcal{Q}(i)_{|T}$ are computationally indistinguishable.

Theorem [CGKS95, CG97]. If $t = n$ (in particular if $n = 1$ server), then:

- for **IT** privacy, **no better solution than full download**,
- **computational** privacy is possible, but remains **expensive** as of now.

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Main parameters of PIR schemes

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Parameters to be taken into account:

– **communication** complexity (upload and download)

Main parameters of PIR schemes

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Parameters to be taken into account:

- **communication** complexity (upload and download)
- **computation** complexity (client and servers)

Main parameters of PIR schemes

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Parameters to be taken into account:

- **communication** complexity (upload and download)
- **computation** complexity (client and servers)
- global **server storage** overhead

Main parameters of PIR schemes

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Parameters to be taken into account:

- **communication** complexity (upload and download)
- **computation** complexity (client and servers)
- global **server storage** overhead
- maximum size of collusions (t)

Main parameters of PIR schemes

We focus on **IT-privacy**
(hence we need $n \geq 2$ servers)

Parameters to be taken into account:

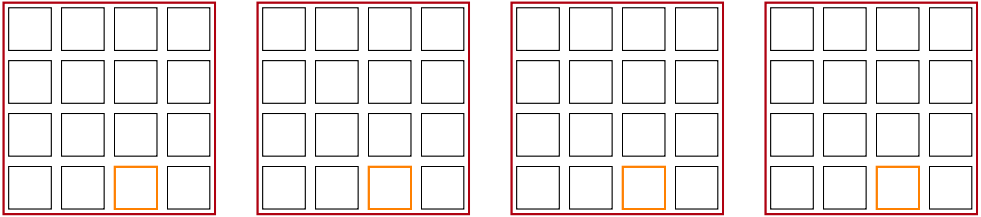
- **communication** complexity (upload and download)
- **computation** complexity (client and servers)
- global **server storage** overhead
- maximum size of collusions (t)

Several possible **settings**:

- **replicated** database vs. **coded** database
- unresponsive or **byzantine** servers
- small entries vs. large entries
- bounded vs. unbounded number of entries in the database
- dynamic database vs. static database

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

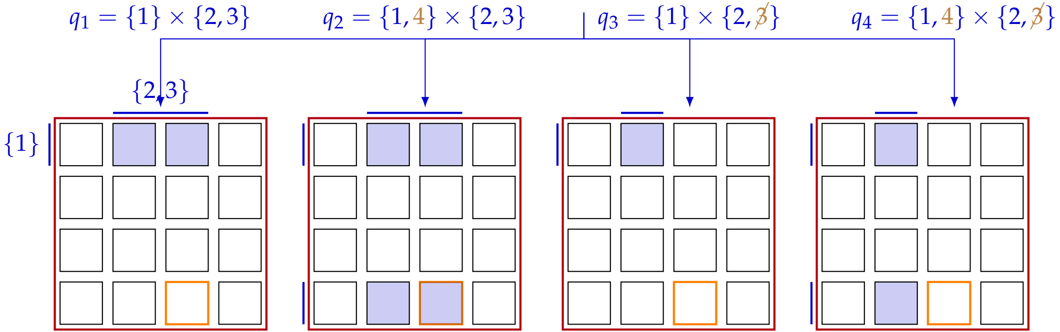
16 files are replicated over 4 servers.
Files are indexed by pairs $(i, j) \in \{1, 2, 3, 4\}^2$
Assume one wants to retrieve file $F_{4,3}$.



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

16 files are replicated over 4 servers.
Files are indexed by pairs $(i, j) \in \{1, 2, 3, 4\}^2$
Assume one wants to retrieve file $F_{4,3}$.

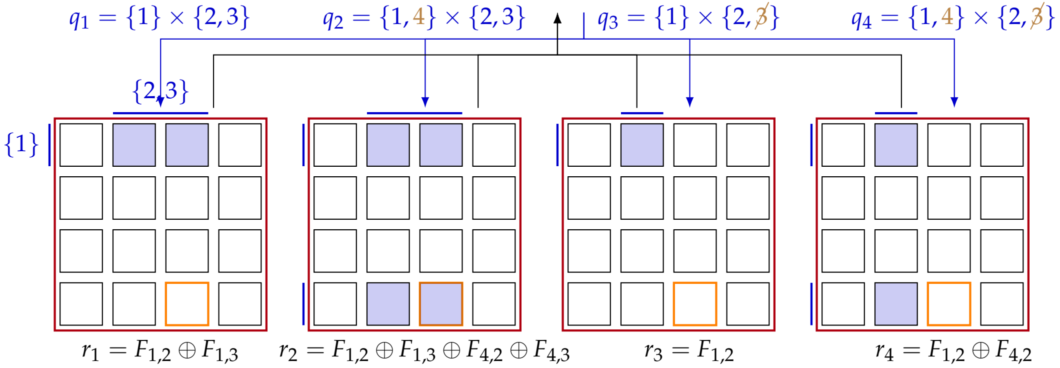
Queries are Cartesian products $I \times J$.
We add/remove indices i and j from I and J depending on the server.



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

16 files are replicated over 4 servers.
Files are indexed by pairs $(i,j) \in \{1,2,3,4\}^2$
Assume one wants to retrieve file $F_{4,3}$.

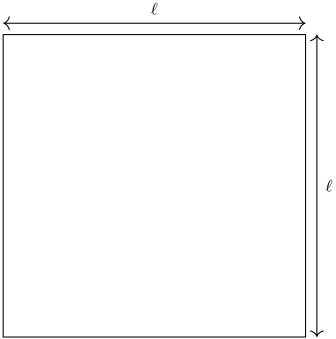
Queries are Cartesian products $I \times J$.
We add/remove indices i and j from I and J depending on the server.



Seminal work [CGKS'95-98]

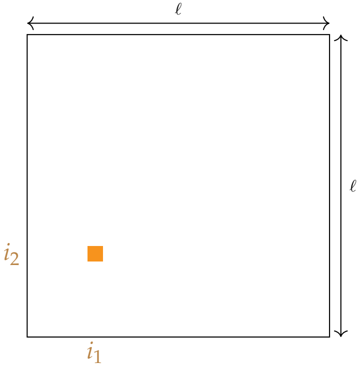
Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.



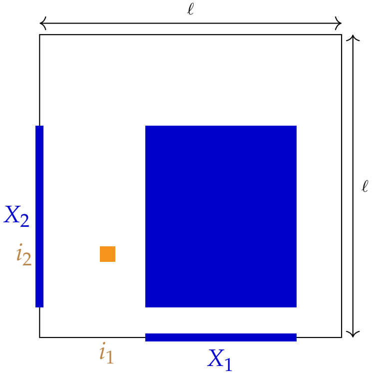
Private Information Retrieval. Chor, Goldreich, Kushilevitz, Sudan. FOCS. **1995**.

1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \leq i_1, i_2 \leq \ell$.



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

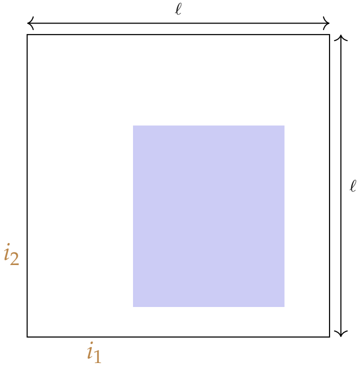
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \le i_1, i_2 \le \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:

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

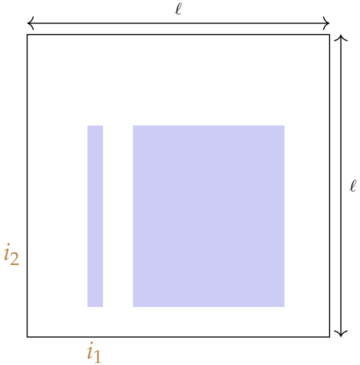
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \leq i_1, i_2 \leq \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1 , X_2) to server S_{00},

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

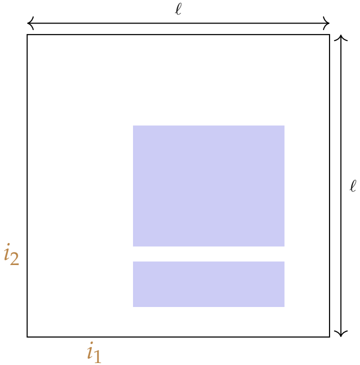
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \leq i_1, i_2 \leq \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1, X_2) to server S_{00},
   - $(X_1 \Delta \{i_1\}, X_2)$ to server S_{10},

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

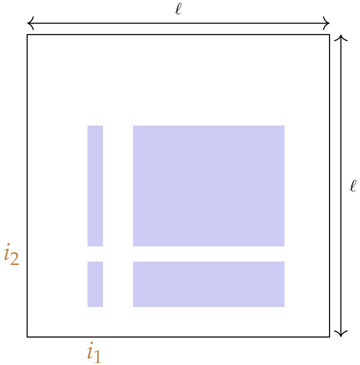
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \leq i_1, i_2 \leq \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1, X_2) to server S_{00},
   - $(X_1 \Delta \{i_1\}, X_2)$ to server S_{10},
   - $(X_1, X_2 \Delta \{i_2\})$ to server S_{01},

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

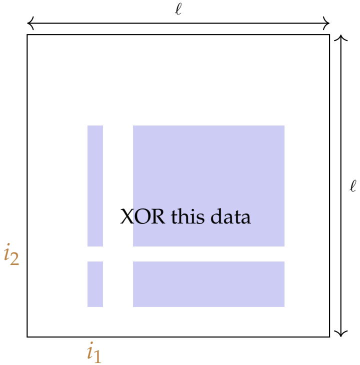
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \le i_1, i_2 \le \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1, X_2) to server S_{00},
   - $(X_1 \Delta \{i_1\}, X_2)$ to server S_{10},
   - $(X_1, X_2 \Delta \{i_2\})$ to server S_{01},
   - $(X_1 \Delta \{i_1\}, X_2 \Delta \{i_2\})$ to server S_{11}.

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

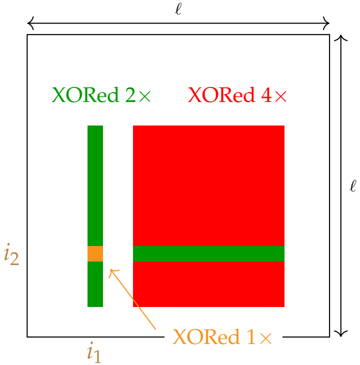
1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \le i_1, i_2 \le \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1, X_2) to server S_{00},
   - $(X_1 \Delta \{i_1\}, X_2)$ to server S_{10},
   - $(X_1, X_2 \Delta \{i_2\})$ to server S_{01},
   - $(X_1 \Delta \{i_1\}, X_2 \Delta \{i_2\})$ to server S_{11}.
2. **Answers:** at reception of (Z_1, Z_2), each server S_j computes $R_j = \bigoplus_{z \in Z_1 \times Z_2} F_z$ and sends back R_j.

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

1st generalization: consider k files $F_{i,j}$ where $(i,j) \in [1,\ell]^2$ and $k = \ell^2$.
Files are replicated over $n = 4$ servers $S_{00}, S_{01}, S_{10}, S_{11}$.
Goal: retrieve F_{i_1,i_2}, for $1 \leq i_1, i_2 \leq \ell$.



1. **Query generation:** pick at random two subsets X_1, X_2 of $[1,\ell]$. Then send:
   - (X_1, X_2) to server S_{00},
   - $(X_1\Delta\{i_1\}, X_2)$ to server S_{10},
   - $(X_1, X_2\Delta\{i_2\})$ to server S_{01},
   - $(X_1\Delta\{i_1\}, X_2\Delta\{i_2\})$ to server S_{11}.
2. **Answers:** at reception of (Z_1, Z_2), each server S_j computes $R_j = \bigoplus_{z \in Z_1 \times Z_2} F_z$ and sends back R_j.
3. **Reconstruction:** compute the XOR of the 4 files R_j and retrieves F_{i_1,i_2}.

Features of the PIR scheme in [CGKS'95-98]

Correct, and **secure** if no collusion.

Features of the PIR scheme in [CGKS'95-98]

Correct, and **secure** if no collusion.

Quantitative results. Assume the k files have same size $|F|$.
With $n = 4$ servers,

- **Communication:** $8\sqrt{k}$ uploaded bits, $4|F|$ downloaded bits,
- **Storage:** replication of all files over 4 servers,
- **Complexity:**

  for each server: in average, XOR of $(\ell/2)^2 = k/4$ files
  for the user: XOR of 4 files.

Features of the PIR scheme in [CGKS'95-98]

Correct, and **secure** if no collusion.

Quantitative results. Assume the k files have same size $|F|$.
With $n = 4$ servers,

- **Communication:** $8\sqrt{k}$ uploaded bits, $4|F|$ downloaded bits,
- **Storage:** replication of all files over 4 servers,
- **Complexity:**

  for each server: in average, XOR of $(\ell/2)^2 = k/4$ files
  for the user: XOR of 4 files.

Generalizable to $n = 2^b$ servers:

- **Communication:** $b2^b k^{1/b} = n \log(n) k^{1/\log(n)}$ uploaded bits, $n|F|$ downloaded bits,
- **Storage:** replication of all files over n servers,
- **Complexity:**
  - for each server: in average, XOR of k/n files
  - for the user: XOR of n files.

Timeline

- 1995: first definition [CGKS95]
- 2000: reduction from smooth locally decodable codes [KT00]
- 2000-10's: many improvements
  - PIR with 3 servers and subpolynomial communication [Yek08, Efr09]
  - PIR with 2 servers and subpolynomial communication [DG16]
  - lower storage overhead with *PIR codes* [FVY15]
- 2016-now: capacity-achieving schemes, schemes dedicated to storage systems
  - capacity of PIR [SJ17, BU18]
  - (nearly) capacity-achieving schemes [SRR14, CHY15, TR16, ...]

Outline

1. Private information retrieval

2. PIR schemes with low computation and storage
   - Transversal designs and codes
   - A PIR scheme with transversal designs
   - Collusion-resistant PIR schemes with weighted lifted codes

Previous scheme:

- moderate communication complexity
- computationally inefficient (linear in $|F|$)
- huge storage overhead (replicas of $|F|$)

Previous scheme:

- moderate communication complexity
- computationally inefficient (linear in $|F|$)
- huge storage overhead (replicas of $|F|$)

Our goal:

- moderate communication complexity
- **optimal computation** (one read for each server)
- **smaller storage overhead** thanks to a pre-encoding and a distribution of the database

Previous scheme:

- moderate communication complexity
- computationally inefficient (linear in $|F|$)
- huge storage overhead (replicas of $|F|$)

Our goal:

- moderate communication complexity
- **optimal computation** (one read for each server)
- **smaller storage overhead** thanks to a pre-encoding and a distribution of the database

Tools: coding theory and combinatorics

- transversal designs and associated codes,
- "lifted" codes.

Outline

1. Private information retrieval

2. PIR schemes with low computation and storage

Transversal designs and codes

A PIR scheme with transversal designs

Collusion-resistant PIR schemes with weighted lifted codes

A (linear) **code** $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

Any code admits **parity-check matrices** $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$ such that

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

Coding theory

A (linear) **code** $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

Any code admits **parity-check matrices** $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$ such that

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

Terminology:

- $\boldsymbol{h} \in \mathrm{RowSpan}(\boldsymbol{H})$ is a **parity-check equation** for $\mathcal{C}$.
- **Support:** $\mathrm{supp}(\boldsymbol{h}) := \{i \in \{1,\dots,n\}, h_i = 0\}$
- **Weight:** $\mathrm{wt}(\boldsymbol{h}) := |\mathrm{supp}(\boldsymbol{h})|$.

A (linear) **code** $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}_q^n$.

Any code admits **parity-check matrices** $\boldsymbol{H} \in \mathbb{F}_q^{(n-k)\times n}$ such that

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

Terminology:

- $\boldsymbol{h} \in \mathrm{RowSpan}(\boldsymbol{H})$ is a **parity-check equation** for $\mathcal{C}$.
- **Support:** $\mathrm{supp}(\boldsymbol{h}) := \{i \in \{1,\dots,n\}, h_i = 0\}$
- **Weight:** $\mathrm{wt}(\boldsymbol{h}) := |\mathrm{supp}(\boldsymbol{h})|$.

Important remark. If $r = \mathrm{wt}(\boldsymbol{h})$ and $i \in \mathrm{supp}(\boldsymbol{h})$, then for every $\boldsymbol{c} \in \mathcal{C}$, one can recover $c_i \in \mathbb{F}_q$ by accessing at most $r-1$ other coordinates c_j of the codeword $\boldsymbol{c}$:

$$c_i = -\frac{1}{h_i} \sum_{j \in \mathrm{supp}(\boldsymbol{h}) \setminus \{i\}} h_j c_j$$

In that case we call $\mathrm{supp}(\boldsymbol{h}) \setminus \{i\}$ a **helper set** for i.

Goal

Our goal. Design a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ such that, for every $i \in \{1, \dots, n\}$, there exists a set of helper sets which uniformly covers $\{1, \dots, n\}$.

Our goal. Design a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ such that, for every $i \in \{1, \ldots, n\}$, there exists a set of helper sets which uniformly covers $\{1, \ldots, n\}$.

- This will provide a way to recover c_i by accessing uniformly at random other coordinates of $\boldsymbol{c}$.
- Querying these "random coordinates" will leak no information about i to the servers.

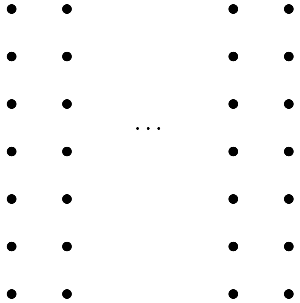
Our goal. Design a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ such that, for every $i \in \{1, \ldots, n\}$, there exists a set of helper sets which uniformly covers $\{1, \ldots, n\}$.

- This will provide a way to recover c_i by accessing uniformly at random other coordinates of $\boldsymbol{c}$.
- Querying these "random coordinates" will leak no information about i to the servers.

Let's do this with **combinatorics**.

A **transversal design** $\mathrm{TD}(n, s) = (X, \mathcal{B}, \mathcal{G})$ is given by:

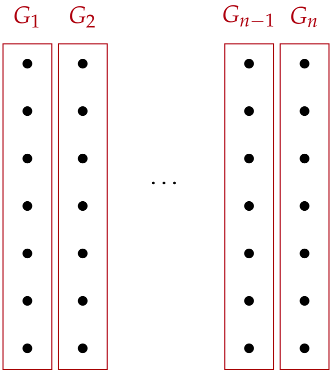
- X a set of *points*, $|X| = N = ns$,



A **transversal design** $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ is given by:

- X a set of *points*, $|X| = N = ns$,
- a partition of X into subsets $\mathcal{G} = \{G_j\}_{1 \le j \le n}$ called *groups*:

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

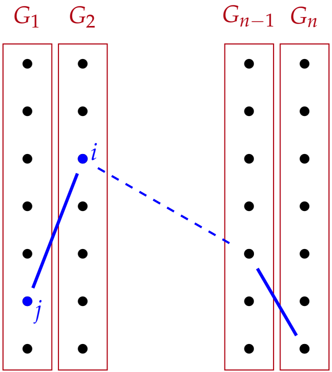


A **transversal design** $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ is given by:

- X a set of *points*, $|X| = N = ns$,
- a partition of X into subsets $\mathcal{G} = \{G_j\}_{1 \le j \le n}$ called *groups*:

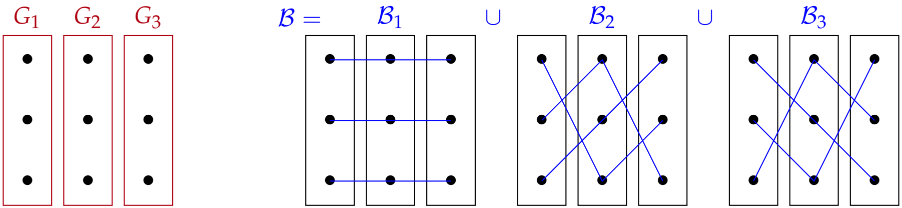
$$X = \coprod_{j=1}^{n} G_j \text{ and } |G_j| = s\,,$$

- a set of subsets of X, "incident" to $\mathcal{G}$, called *blocks* $B \in \mathcal{B}$:
  - $B \subset X$ and $|B| = n$
  - for all $\{i,j\} \subset X$, the pair $\{i,j\}$ lie
    either in a single group $G \in \mathcal{G}$,
    or in a unique block $B \in \mathcal{B}$



An example for a $\mathrm{TD}(3,3)$:

- $ns = 9$ points
- $s = 3$ groups G_1, G_2, G_3 of size 3
- $ns = 9$ blocks of $n = 3$ points, partitionned into 3 parallel classes $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$



Codes from designs

Let $\mathcal{T}$ be a transversal design $\mathrm{TD}(n,s) = (X, \mathcal{B}, \mathcal{G})$ with points $X = \{x_1, \dots, x_{ns}\}$, blocks $B = \{B_1, \dots, B_{ns}\}$ and groups $\mathcal{G} = \{G_1, \dots, G_n\}$.

Its **incidence matrix** $\boldsymbol{M}$, of size $|\mathcal{B}| \times |X| = ns \times ns$, is defined by:

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

Let $\mathcal{T}$ be a transversal design $\mathrm{TD}(n, s) = (X, \mathcal{B}, \mathcal{G})$ with points $X = \{x_1, \dots, x_{ns}\}$, blocks $B = \{B_1, \dots, B_{ns}\}$ and groups $\mathcal{G} = \{G_1, \dots, G_n\}$.

Its **incidence matrix** $\boldsymbol{M}$, of size $|\mathcal{B}| \times |X| = ns \times ns$, is defined by:

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

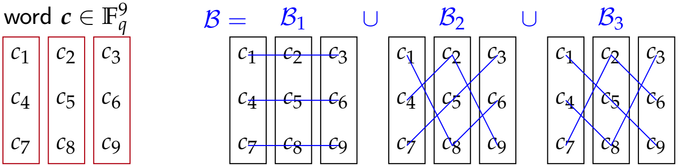
Definition. The **linear code** $\mathcal{C}$ **based on** $\mathcal{T}$ **over** $\mathbb{F}_q$ is the $\mathbb{F}_q$-linear code having $\boldsymbol{M}$ as a parity-check matrix (*i.e.* $\mathcal{C}^\perp$ is generated by $\boldsymbol{M}$).

- $\mathrm{length}(\mathcal{C}) = |X| = ns$,
- $\dim(\mathcal{C}) = \dim(\ker \boldsymbol{M})$,
- every block $B \in \mathcal{B}$ gives a parity-check equation $\boldsymbol{h} \in \mathcal{C}^\perp$, such that

$$\mathrm{supp}(\boldsymbol{h}) = B \qquad \text{and} \qquad \mathrm{wt}(\boldsymbol{h}_{|G_j}) = 1, \quad \forall j = 1, \dots, n$$

Example

The transversal design $\mathrm{TD}(3,3)$ represented by:



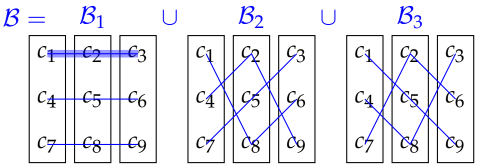
gives a code with the following parity-check matrix:

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

The transversal design $\mathrm{TD}(3,3)$ represented by:

word $\boldsymbol{c} \in \mathbb{F}_q^9$

$$\begin{array}{|c|c|c|} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{array}$$

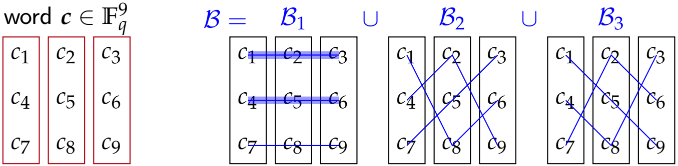


gives a code with the following parity-check matrix:

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

Example

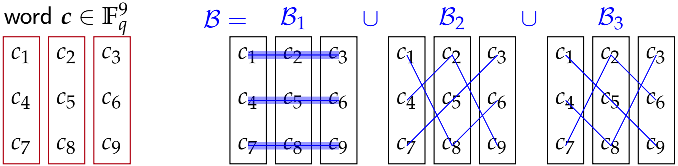
The transversal design $\mathrm{TD}(3,3)$ represented by:



gives a code with the following parity-check matrix:

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

The transversal design $\mathrm{TD}(3,3)$ represented by:

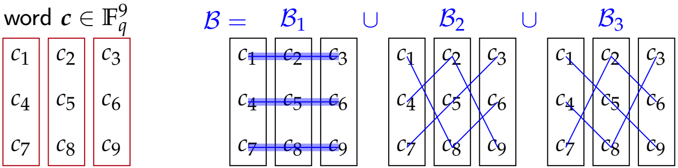


gives a code with the following parity-check matrix:

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

Example

The transversal design TD$(3,3)$ represented by:



gives a code with the following parity-check matrix:

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

Dimension of the code?

- depends on the characteristic,
- for instance, over $\mathbb{F}_3$, we have $\mathrm{rk}(H) = 6 \implies \dim(\mathcal{C}) = 3$.

1. Private information retrieval

2. PIR schemes with low computation and storage

Transversal designs and codes

A PIR scheme with transversal designs

Collusion-resistant PIR schemes with weighted lifted codes

The PIR scheme

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

Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a $\mathrm{TD}(n, s)$, with $N = ns$.

The PIR scheme

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

Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a $\mathrm{TD}(n, s)$, with $N = ns$.

- **Initialisation.** Encode files $(F_1, \ldots, F_k) \mapsto \boldsymbol{c} \in \mathcal{C}$, and upload $\boldsymbol{c}_{|G_j}$ on server S_j.

G_1	G_2	G_3
c_1	c_4	c_7
c_2	c_5	c_8
c_3	c_6	c_9

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

Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a $\mathrm{TD}(n, s)$, with $N = ns$.

- **Initialisation.** Encode files $(F_1, \dots, F_k) \mapsto \boldsymbol{c} \in \mathcal{C}$, and upload $\boldsymbol{c}_{|G_j}$ on server S_j.
- **To recover** $F_i = c_i$, with $i \in X$:

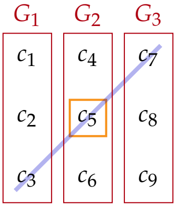
G_1	G_2	G_3
c_1	c_4	c_7
c_2	$\boxed{c_5}$	c_8
c_3	c_6	c_9

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

Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a $\mathrm{TD}(n,s)$, with $N = ns$.

- **Initialisation.** Encode files $(F_1, \dots, F_k) \mapsto \boldsymbol{c} \in \mathcal{C}$, and upload $\boldsymbol{c}_{|G_j}$ on server S_j.
- **To recover** $F_i = c_i$, with $i \in X$:
  1. User randomly picks a block $B \in \mathcal{B}$ containing i.
     Then, user defines queries:

$$q_j = \mathcal{Q}(i)_j := \begin{cases} \text{unique point in } B \cap G_j & \text{if } i \notin G_j \\ \text{a random point in } G_j & \text{otherwise.} \end{cases}$$



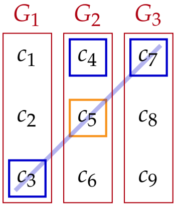
The PIR scheme

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

Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a $\mathrm{TD}(n, s)$, with $N = ns$.

- **Initialisation.** Encode files $(F_1, \dots, F_k) \mapsto \boldsymbol{c} \in \mathcal{C}$, and upload $\boldsymbol{c}_{|G_j}$ on server S_j.
- **To recover** $F_i = c_i$, with $i \in X$:
  1. User randomly picks a block $B \in \mathcal{B}$ containing i.
     Then, user defines queries:

$$q_j = \mathcal{Q}(i)_j := \begin{cases} \text{unique point in } B \cap G_j & \text{if } i \notin G_j \\ \text{a random point in } G_j & \text{otherwise.} \end{cases}$$



The PIR scheme

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

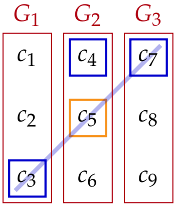
Let $\mathcal{C} \subseteq \mathbb{F}_q^N$ be a code based on a TD(n, s), with $N = ns$.

- **Initialisation.** Encode files $(F_1, \dots, F_k) \mapsto \boldsymbol{c} \in \mathcal{C}$, and upload $\boldsymbol{c}_{|G_j}$ on server S_j.
- **To recover** $F_i = c_i$, with $i \in X$:
  1. User randomly picks a block $B \in \mathcal{B}$ containing i.
     Then, user defines queries:

     $$q_j = \mathcal{Q}(i)_j := \begin{cases} \text{unique point in } B \cap G_j & \text{if } i \notin G_j \\ \text{a random point in } G_j & \text{otherwise.} \end{cases}$$

  2. Each server S_j sends back c_{q_j}
  3. User recovers

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



Theorem. This PIR protocol is information-theoretically private.

Proof:

- the only server which holds F_i received a random query;
- for each other server S_j, query q_j gives no information on the block B which has been picked $\Rightarrow$ no information leaks on i.

Theorem. This PIR protocol is information-theoretically private.

Proof:

- the only server which holds F_i received a random query;
- for each other server S_j, query q_j gives no information on the block B which has been picked $\Rightarrow$ no information leaks on i.

Features.

- communication complexity: $n \log s$ uploaded bits, $n \log q$ downloaded bits

Privacy and parameters

Theorem. This PIR protocol is information-theoretically private.

Proof:

- the only server which holds F_i received a random query;
- for each other server S_j, query q_j gives no information on the block B which has been picked $\Rightarrow$ no information leaks on i.

Features.

- communication complexity: $n \log s$ uploaded bits, $n \log q$ downloaded bits
- computational complexity:
  - **only 1 read for each server** (optimal)
  - $\leq n$ additions over $\mathbb{F}_q$ for the user

Theorem. This PIR protocol is information-theoretically private.

Proof:

– the only server which holds F_i received a random query;

– for each other server S_j, query q_j gives no information on the block B which has been picked $\Rightarrow$ no information leaks on i.

Features.

- communication complexity: $n \log s$ uploaded bits, $n \log q$ downloaded bits
- computational complexity:
  - **only 1 read for each server** (optimal)
  - $\leq n$ additions over $\mathbb{F}_q$ for the user
- storage overhead: $(ns - k) \log q$ bits, where $k = \dim(\mathcal{C})$

Theorem. This PIR protocol is information-theoretically private.

Proof:

- the only server which holds F_i received a random query;
- for each other server S_j, query q_j gives no information on the block B which has been picked $\Rightarrow$ no information leaks on i.

Features.

- communication complexity: $n \log s$ uploaded bits, $n \log q$ downloaded bits
- computational complexity:
  - **only 1 read for each server** (optimal)
  - $\leq n$ additions over $\mathbb{F}_q$ for the user
- storage overhead: $(ns - k) \log q$ bits, where $k = \dim(\mathcal{C})$

Question: transversal designs leading to large dimension codes?

Instances with geometric designs

An example: the **classical affine transversal design**:

- $X = \mathbb{F}_q^m$ for $m \geq 2$,
- $\mathcal{G}$ a partition of X into q hyperplanes $G_1, \dots, G_q$,
- $\mathcal{B} = \{\text{affine lines } L \text{ secant to each } G_j\}$.

The code has:

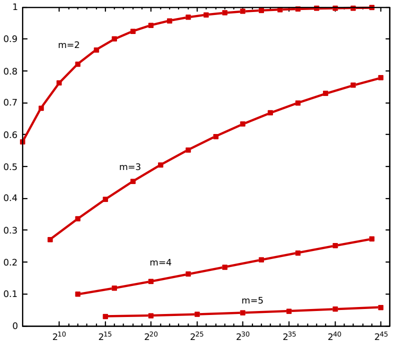
- length $ns = q^m$,
- "locality" $n = q$.

An example: the **classical affine transversal design**:

- $X = \mathbb{F}_q^m$ for $m \geq 2$,
- $\mathcal{G}$ a partition of X into q hyperplanes $G_1, \ldots, G_q$,
- $\mathcal{B} = \{$affine lines L secant to each $G_j\}$.

The code has:

- length $ns = q^m$,
- "locality" $n = q$.



Information rate of affine geometry TD-based code depending on the length.
Each curve corresponds to a value of $m \in \{2, 3, 4, 5\}$.

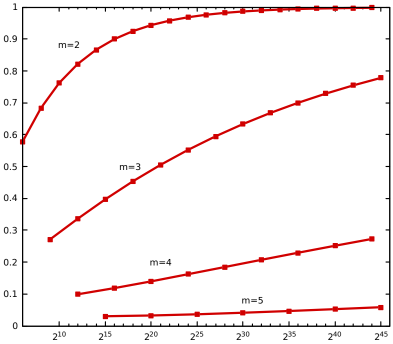
An example: the **classical affine transversal design**:

- $X = \mathbb{F}_q^m$ for $m \geq 2$,
- $\mathcal{G}$ a partition of X into q hyperplanes $G_1, \dots, G_q$,
- $\mathcal{B} = \{$affine lines L secant to each $G_j\}$.

The code has:

- length $ns = q^m$,
- "locality" $n = q$.

Question: how to deal with collusions and errors?



Information rate of affine geometry TD-based code depending on the length.
Each curve corresponds to a value of $m \in \{2, 3, 4, 5\}$.

1. Private information retrieval

2. PIR schemes with low computation and storage

- Transversal designs and codes
- A PIR scheme with transversal designs
- Collusion-resistant PIR schemes with weighted lifted codes

An algebraic construction

We have seen a **combinatorial** construction of codes for PIR, using **transversal designs**.

Let's now see what we can do **algebraically**.

Definition. The (full-length) **Reed–Solomon code** of dimension k over $\mathbb{F}_q$ is:

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_{\mathbb{A}^1}(f) := (f(x_1), \ldots, f(x_q)) \mid \deg(f) \le k-1\}\,.$$

Definition. The (full-length) **Reed–Solomon code** of dimension k over $\mathbb{F}_q$ is:

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_{\mathbb{A}^1}(f) := (f(x_1), \dots, f(x_q)) \mid \deg(f) \le k-1\}\,.$$

The code $\mathcal{C} = \mathrm{RS}_q(k)$ is **MDS**: every codeword $\boldsymbol{c} \in \mathcal{C}$ can be reconstructed from any k-subset of coordinates of $\boldsymbol{c}$.

Definition. The (full-length) **Reed–Solomon code** of dimension k over $\mathbb{F}_q$ is:

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_{\mathbb{A}^1}(f) := (f(x_1), \dots, f(x_q)) \mid \deg(f) \leq k-1\} .$$

The code $\mathcal{C} = \mathrm{RS}_q(k)$ is **MDS**: every codeword $\boldsymbol{c} \in \mathcal{C}$ can be reconstructed from any k-subset of coordinates of $\boldsymbol{c}$.

Definition. The **Reed–Muller code** of order m and degree r over $\mathbb{F}_q$ is:

$$\mathrm{RM}_q(m, r) := \{\mathrm{ev}_{\mathbb{A}^m}(f) \mid f \in \mathbb{F}_q[\boldsymbol{X}] \text{ and } \deg(f) \leq r\} .$$

Definition. The (full-length) **Reed–Solomon code** of dimension k over $\mathbb{F}_q$ is:

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_{\mathbb{A}^1}(f) := (f(x_1), \dots, f(x_q)) \mid \deg(f) \leq k-1\}\,.$$

The code $\mathcal{C} = \mathrm{RS}_q(k)$ is **MDS**: every codeword $\boldsymbol{c} \in \mathcal{C}$ can be reconstructed from any k-subset of coordinates of $\boldsymbol{c}$.

Definition. The **Reed–Muller code** of order m and degree r over $\mathbb{F}_q$ is:

$$\mathrm{RM}_q(m,r) := \{\mathrm{ev}_{\mathbb{A}^m}(f) \mid f \in \mathbb{F}_q[\boldsymbol{X}] \text{ and } \deg(f) \leq r\}\,.$$

Reed–Muller codes have the following property:

$$\forall \boldsymbol{c} = \mathrm{ev}_{\mathbb{A}^m}(f) \in \mathrm{RM}_q(m,r) \quad \text{and} \quad \forall \text{ affine line } L \subset \mathbb{A}^m,$$
$$\mathrm{ev}_{\mathbb{A}^1}(f_{|L}) \in \mathrm{RS}_q(r+1)\,.$$

(where $f_{|L}$ is the lowest-degree univariate polynomial interpolating f over L)

Reed–Solomon and Reed–Muller codes

Definition. The (full-length) **Reed–Solomon code** of dimension k over $\mathbb{F}_q$ is:

$$\mathrm{RS}_q(k) := \{\mathrm{ev}_{\mathbb{A}^1}(f) := (f(x_1), \dots, f(x_q)) \mid \deg(f) \le k-1\}\,.$$

The code $\mathcal{C} = \mathrm{RS}_q(k)$ is **MDS**: every codeword $\boldsymbol{c} \in \mathcal{C}$ can be reconstructed from any k-subset of coordinates of $\boldsymbol{c}$.

Definition. The **Reed–Muller code** of order m and degree r over $\mathbb{F}_q$ is:

$$\mathrm{RM}_q(m,r) := \{\mathrm{ev}_{\mathbb{A}^m}(f) \mid f \in \mathbb{F}_q[\boldsymbol{X}] \text{ and } \deg(f) \le r\}\,.$$

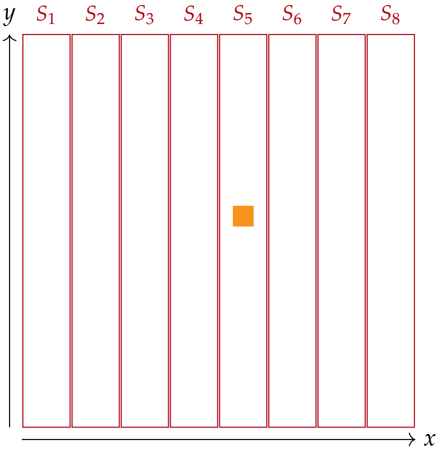
Reed–Muller codes have the following property:

$$\forall \boldsymbol{c} = \mathrm{ev}_{\mathbb{A}^m}(f) \in \mathrm{RM}_q(m,r) \quad \text{and} \quad \forall \text{ affine line } L \subset \mathbb{A}^m,$$
$$\mathrm{ev}_{\mathbb{A}^1}(f_{|L}) \in \mathrm{RS}_q(r+1)\,.$$

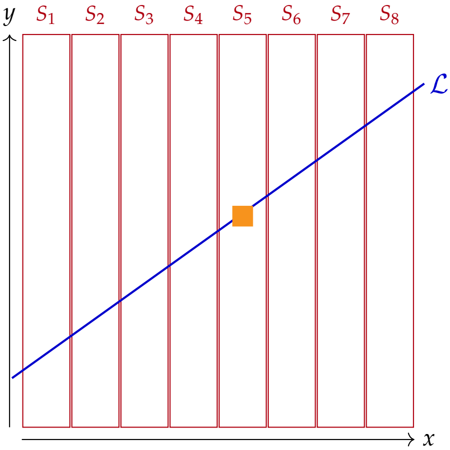
(where $f_{|L}$ is the lowest-degree univariate polynomial interpolating f over L)

In particular, if $r \le q-2$, then $c_i = f(P_i)$ can be reconstructed by interpolating a polynomial of degree r on $q-1$ other points of **any line** passing through P_i.

A PIR scheme based on Reed–Muller codes

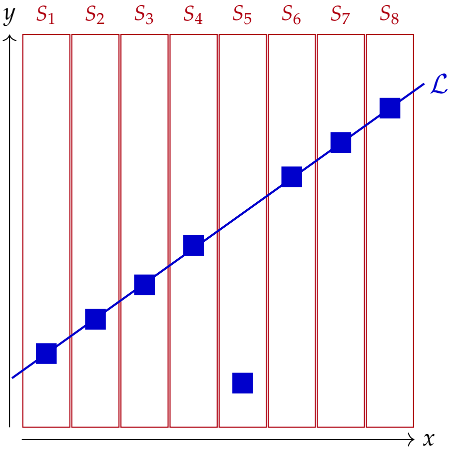


Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers



Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers

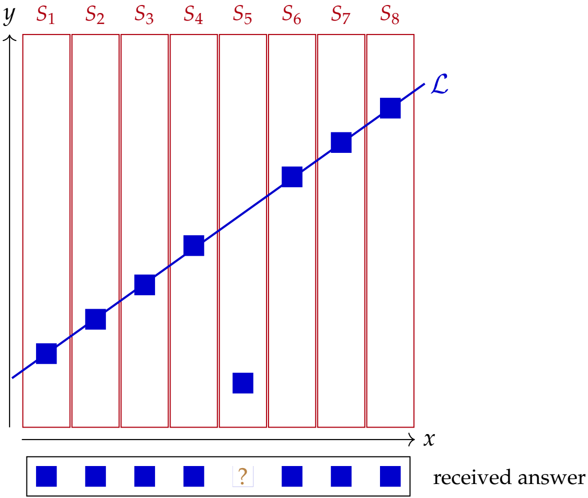
Assume one wants to extract file F_i.



Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i,

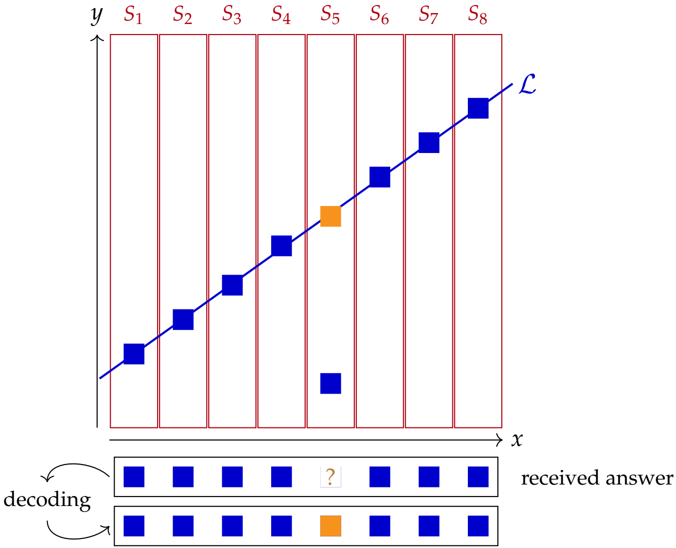


Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i, query all files F_j except F_i,

A PIR scheme based on Reed–Muller codes

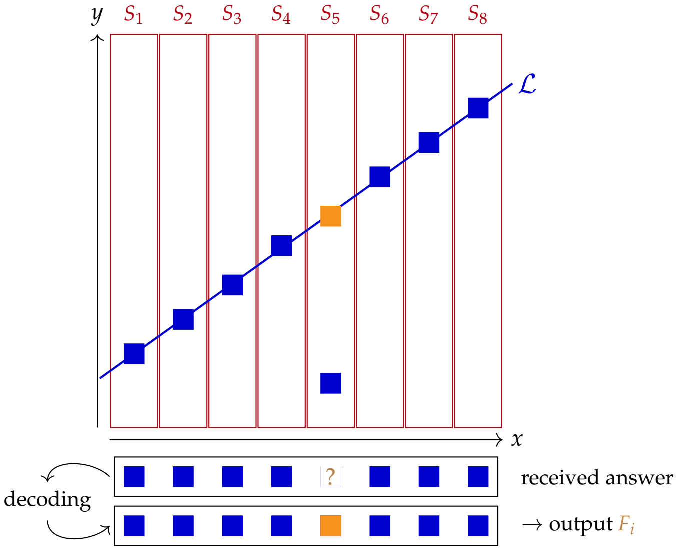


Database F is encoded with $\mathrm{RM}_q(m,r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i, query all files F_j except F_i, and interpolate the corresponding univariate polynomial.

A PIR scheme based on Reed–Muller codes

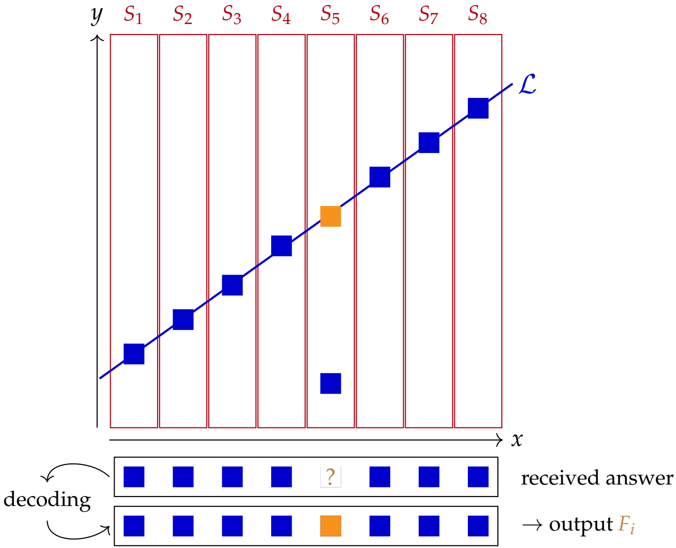


Database F is encoded with $\mathrm{RM}_q(m,r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i, query all files F_j except F_i, and interpolate the corresponding univariate polynomial.

A PIR scheme based on Reed–Muller codes



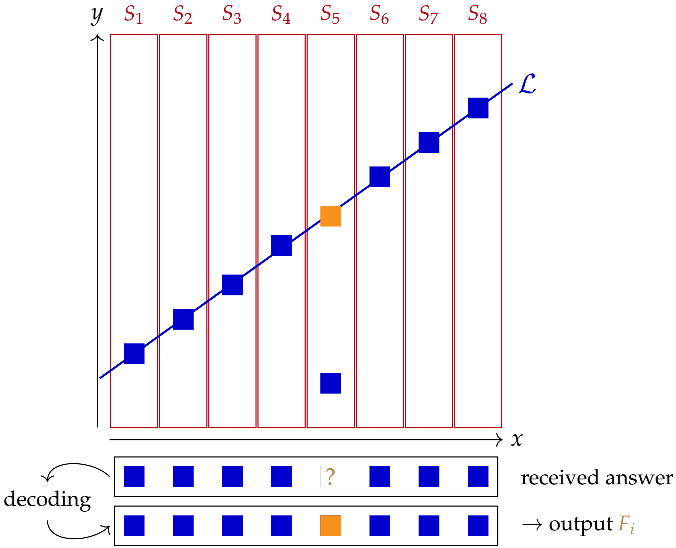
Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i, query all files F_j except F_i, and interpolate the corresponding univariate polynomial.

$\mathcal{L}$ is an affine line

$\implies$ no individual server can find where is



Database F is encoded with $\mathrm{RM}_q(m, r)$, then distributed across the servers

Assume one wants to extract file F_i.

Pick a line L through i, query all files F_j except F_i, and interpolate the corresponding univariate polynomial.

$\mathcal{L}$ is an affine line

$\Longrightarrow$ no individual server can find where is

$\Longrightarrow$ the PIR scheme is private

Features with $\mathrm{RM}_q(m, r)$ of length q^m.

- communication complexity: $(m-1)q \log q$ uploaded bits, $q \log q$ downloaded bits
- computational complexity:
  - **only 1 read for each server** (optimal)
  - a decoding procedure for $\mathrm{RS}_q(r)$ for the user
- storage overhead: the **information rate** of $\mathrm{RM}_q(m, r)$ with $r \leq q-1$ is

$$\simeq \frac{(r/q)^m}{m!} \leq \frac{1}{m!} \ldots$$

$\Longrightarrow$ We need codes with the same "recovery properties", and with larger dimension.

Lifted codes

"Lifted" codes are the largest codes having the same property as Reed–Muller codes.

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

Lifted codes

"Lifted" codes are the largest codes having the same property as Reed–Muller codes.

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

Definition. The **m-th lifted Reed-Solomon code** of degree r over $\mathbb{F}_q$ is:

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

Lifted codes

"Lifted" codes are the largest codes having the same property as Reed–Muller codes.

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

Definition. The **m-th lifted Reed-Solomon code** of degree r over $\mathbb{F}_q$ is:

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

Lifted codes contain Reed-Muller codes, **sometimes properly**.

"Lifted" codes are the largest codes having the same property as Reed–Muller codes.

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

Definition. The **m-th lifted Reed-Solomon code** of degree r over $\mathbb{F}_q$ is:

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

Lifted codes contain Reed-Muller codes, **sometimes properly**.

Example. For $q = 4$, $m = 2$, $r = 2$, consider $f(X,Y) = X^2Y^2$ and an affine line L with equation $(aT+b, cT+d)$. Then,

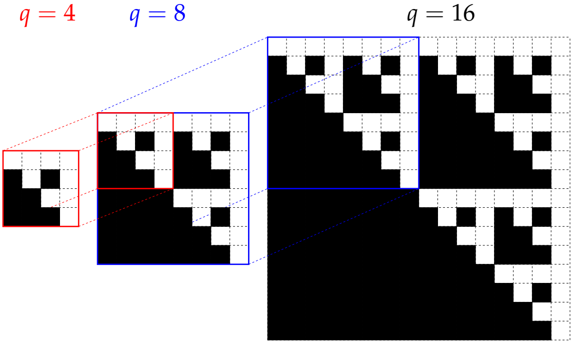
$$f(aT+b, cT+d) \equiv (a^2d^2 + b^2c^2)T^2 + a^2c^2T + b^2d^2 \mod (T^4 - T)$$

corresponds to a degree–2 polynomial in T.
Hence,

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

Theorem (Guo, Kopparty, Sudan '13). For every fixed m, and for growing alphabet and length, lifted codes reach arbitrarily large information rates.

Black squares: pairs (i,j) such that $\mathrm{ev}(X^iY^j) \in \mathrm{Lift}_q(m=2, r=q-2)$.



Corollary. For a sufficiently large number of servers, we have PIR with storage overhead $\to 0$.

Question: how to deal with collusions and byzantine errors?

Question: how to deal with collusions and byzantine errors?

For convenience, here $m = 2$.

Definition. A t-curve is:

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

Question: how to deal with collusions and byzantine errors?

For convenience, here $m = 2$.

Definition. A t-curve is:

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

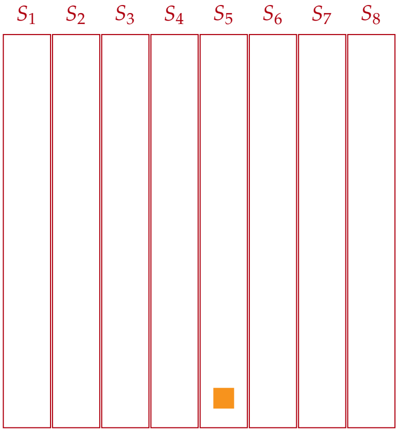
Definition. The **weighted lifted Reed-Solomon code** of degree r and weight t over $\mathbb{F}_q$ is:

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

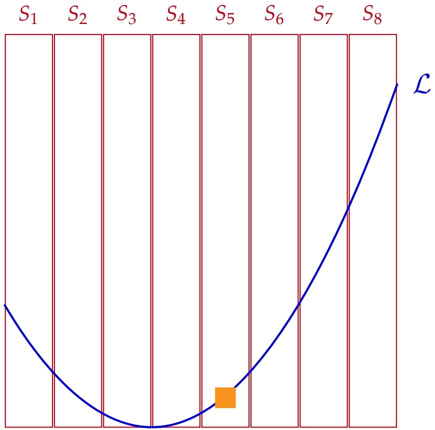
Consequence: for every codeword $\boldsymbol{c} \in \mathrm{WLift}_q(t, r)$ and every t-curve $\mathcal{L}$, we have:

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

A PIR scheme based on weighted lifted codes

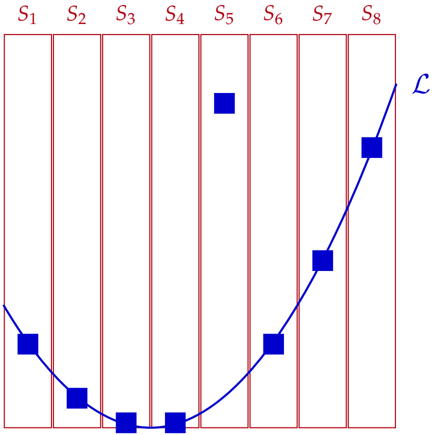


Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers

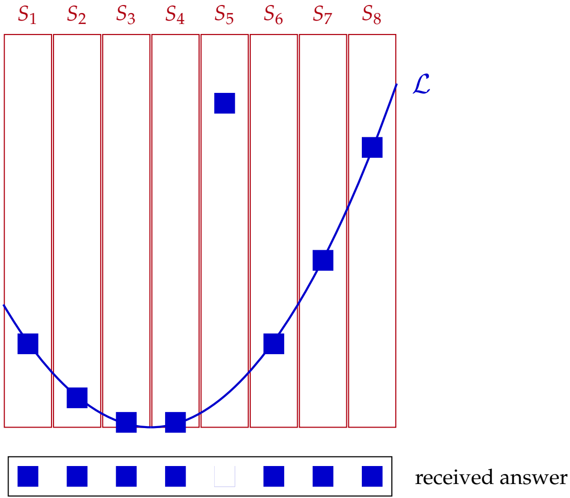


Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers

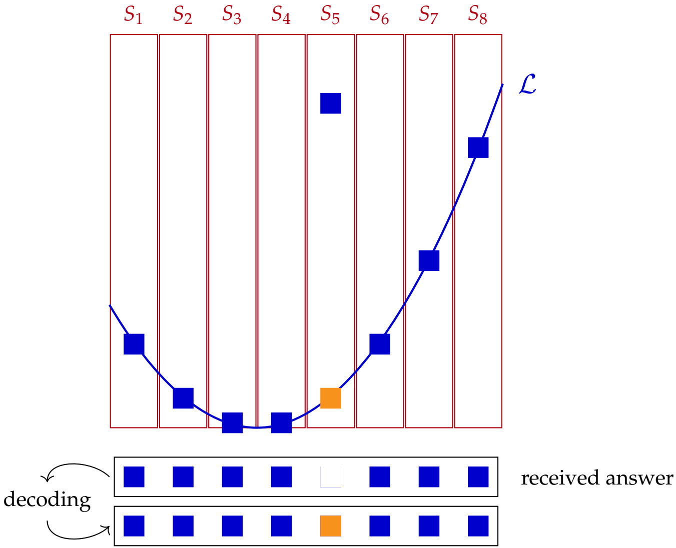
A PIR scheme based on weighted lifted codes



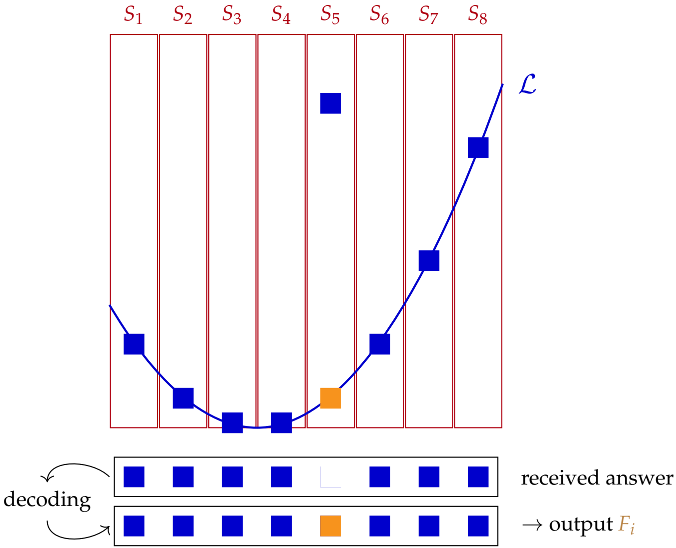
Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers



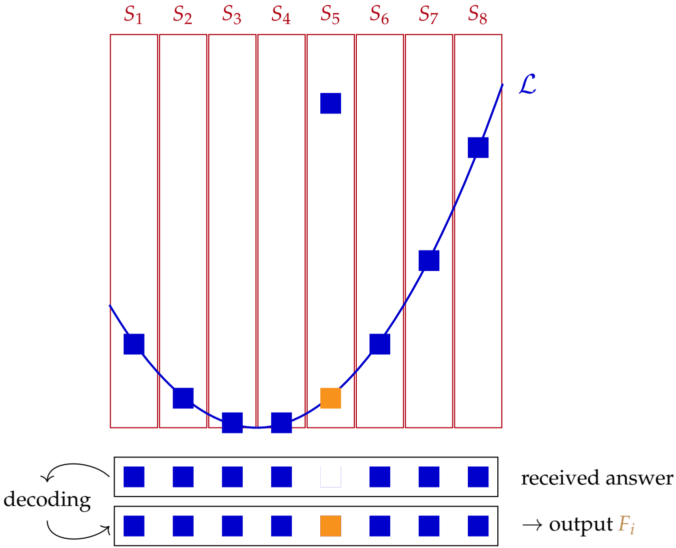
Database F in encoded with $\mathrm{WLift}_q(t,r)$, then distributed across q servers



Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers



Database F in encoded with $\mathrm{WLift}_q(t,r)$, then distributed across q servers

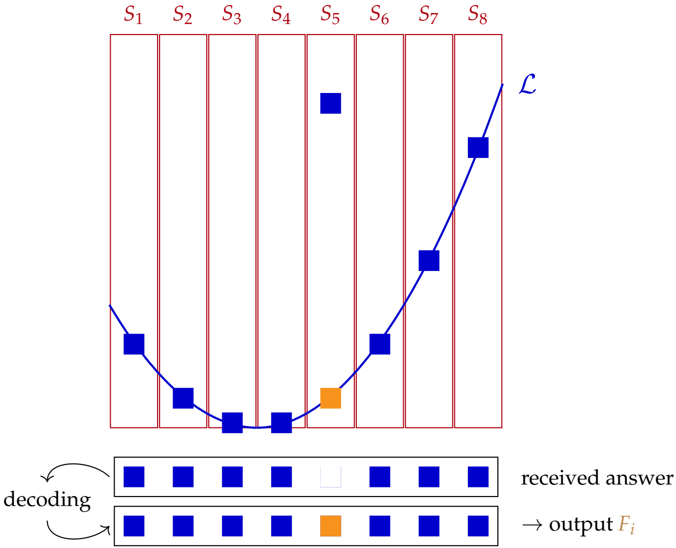


Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers

$\mathcal{L}$ is a t-curve

$\Longrightarrow$ no t-set of servers can find where is

A PIR scheme based on weighted lifted codes



Database F in encoded with $\mathrm{WLift}_q(t, r)$, then distributed across q servers

$\mathcal{L}$ is a t-curve

$\implies$ no t-set of servers can find where is ■

$\implies$ the PIR scheme is t-private

Parameters

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

Theorem. Let p be a prime number, $t \geq 1$ and $\alpha \geq 2$ be fixed integers. Then, the information rate $\mathrm{WLift}_{p^e}(t, p^e - \alpha)$ grows to 1 when $e \to \infty$.

Corollary: we get PIR schemes with relative storage overhead $\to 0$, for a constant number of adversaries.

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

Theorem. Let p be a prime number, $t \geq 1$ and $\alpha \geq 2$ be fixed integers. Then, the information rate $\mathrm{WLift}_{p^e}(t, p^e - \alpha)$ grows to 1 when $e \to \infty$.

Corollary: we get PIR schemes with relative storage overhead $\to 0$, for a constant number of adversaries.

Theorem. Let p be a prime number, $t \geq 1$ and $c \geq 1$ be fixed integers. Let $\gamma = 1 - p^{-c}$ and $\mathcal{C}_e = \mathrm{WLift}_{p^e}(t, \gamma p^e)$. Then, the information rate R_e of $\mathcal{C}_e$ satisfies:

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

Corollary: we get PIR schemes with **constant relative storage overhead**, for a **constant number of collusions** and a **constant fraction of errors**.

Other constructions

Other works. PIR has been a hot topic during for few years.

- Notion of **PIR capacity**: achievable bounds on the download rate of PIR schemes.
  $\rightarrow$ requires lot of comuttation of the server side
- **Optimal** constructions over given distributed storage systems:
  $\rightarrow$ data is already encoded by the storage system
  $\rightarrow$ we can avoid re-encoding and still do PIR

[SJ17] *The Capacity of Private Information Retrieval*. Sun, Jafar. IEEE-TIT. **2017**.

[TGR18] *PIR from MDS Coded Data in Distributed Storage Systems*. Tajeddine, Gnilke, El Rouayheb. IEEE-TIT. **2018**.

[TGKFH18] *Robust PIR from Coded Systems with Byzantine and Colluding Servers*. Tajeddine, Gnilke, Karpuk, Freij-Hollanti, Hollanti. ISIT. **2018**.

Private Information Retrieval Schemes With Product-Matrix MBR Codes. **L.**, Tajeddine, Freij-Hollanti, Hollanti. IEEE IFS. **2021**.

Conclusion

Open questions / future works.

1. Combinatorial bounds on the parameters.
2. Updatable files?
3. Extension to peer-to-peer storage systems (codes on random graphs).

Conclusion

Open questions / future works.

1. Combinatorial bounds on the parameters.
2. Updatable files?
3. Extension to peer-to-peer storage systems (codes on random graphs).

Thank you for your attention!