On Unequal Data Demand PIR Codes

Martin Puškin, Henk D.L. Hollmann, Ago-Erik Riet

University of Tartu

2022

Table of Contents

Table of Contents

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

A (binary) $[n, k]$ error-correction code C is a k-dimensional subspace of $\mathbb{F}_2^n$.

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

A (binary) $[n, k]$ error-correction code C is a k-dimensional subspace of $\mathbb{F}_2^n$. An *encoder* of the code C is a linear bijection $\epsilon\colon \mathbb{F}_2^k \to C$ which maps the *message word* $m \in \mathbb{F}_2^k$ to a corresponding *code word* $mG \in C$, where $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$. We call G the *generator matrix* for C.

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

A (binary) $[n, k]$ error-correction code C is a k-dimensional subspace of $\mathbb{F}_2^n$. An *encoder* of the code C is a linear bijection $\epsilon \colon \mathbb{F}_2^k \to C$ which maps the *message word* $m \in \mathbb{F}_2^k$ to a corresponding *code word* $mG \in C$, where $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$. We call G the *generator matrix* for C.

An important parameter of C is its distance $d = \min\{wt(u) \mid u \in \mathbb{C} \setminus \{0\}\}$ where $wt(u)$ equals the number of non-zero components of u. If d is known, we also call C an $[n, k, d]$ error-correction code.

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

A (binary) $[n, k]$ error-correction code C is a k-dimensional subspace of $\mathbb{F}_2^n$. An *encoder* of the code C is a linear bijection $\epsilon\colon \mathbb{F}_2^k \to C$ which maps the *message word* $m \in \mathbb{F}_2^k$ to a corresponding *code word* $mG \in C$, where $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$. We call G the *generator matrix* for C.

An important parameter of C is its distance $d = \min\{wt(u) \mid u \in \mathbb{C} \setminus \{0\}\}$ where $wt(u)$ equals the number of non-zero components of u. If d is known, we also call C an $[n, k, d]$ error-correction code.

Example

Let $C = \{(1, 1, 1), (0, 0, 0)\}$. Then C is a $[3, 1, 3]$ error-correction code.

Error-correction codes

Everything in this talk will be over the binary field $\mathbb{F}_2$, unless explicitly stated otherwise.

A (binary) $[n, k]$ error-correction code C is a k-dimensional subspace of $\mathbb{F}_2^n$. An *encoder* of the code C is a linear bijection $\epsilon\colon \mathbb{F}_2^k \to C$ which maps the *message word* $m \in \mathbb{F}_2^k$ to a corresponding *code word* $mG \in C$, where $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$. We call G the *generator matrix* for C.

An important parameter of C is its distance $d = \min\{wt(u) \mid u \in \mathbb{C} \setminus \{0\}\}$ where $wt(u)$ equals the number of non-zero components of u. If d is known, we also call C an $[n, k, d]$ error-correction code.

Example

Let $C = \{(1,1,1),(0,0,0)\}$. Then C is a $[3,1,3]$ error-correction code. If noise in the transmission channel changes one bit in the code word, then we can correct the error.

UEP Codes

Clearly, codes with higher distance have higher error detection and error correction capabilities.

UEP Codes

Clearly, codes with higher distance have higher error detection and error correction capabilities.

However, for regular error-correction codes either the whole message or no part of the message can be recovered.

UEP Codes

Clearly, codes with higher distance have higher error detection and error correction capabilities.

However, for regular error-correction codes either the whole message or no part of the message can be recovered.

An *unequal error protection (UEP) code* is an error-correction code where some bits of the code word may be more protected than others and can sometimes be recovered independently.

Example

We can define the encoder ϵ to map (a, b) to (a, a, a, b). Now, clearly the first coordinate is more protected than the second.

UEP Codes

In 1978, Dunning and Robins introduced the concept of a separation vector to characterize the error protection capability of UEP codes.

In 1978, Dunning and Robins introduced the concept of a separation vector to characterize the error protection capability of UEP codes.

Definition

For a linear $[n, k]$ code C with a generator matrix G we define the *separation vector* $S(G) = (S(G)_1, \dots, S(G)_k)$ by

$$S(G)_i := \min\{wt(mG) \mid m \in \mathbb{F}_2^k, m_i \neq 0\}.$$

UEP Codes

In 1978, Dunning and Robins introduced the concept of a separation vector to characterize the error protection capability of UEP codes.

Definition

For a linear $[n, k]$ code C with a generator matrix G we define the *separation vector* $S(G) = (S(G)_1, \ldots, S(G)_k)$ by

$$S(G)_i := \min\{wt(mG) \mid m \in \mathbb{F}_2^k, m_i \neq 0\}.$$

The higher the value $S(G)_i$ the stronger the protection for the i-th data symbol in the message word m.

The Griesmer Bound

An important lower bound for error-correction codes is the Griesmer bound:

Theorem (Griesmer Bound)

Let C be an $[n, k, d]$ error-correction code. Then

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{d}{2^{i-1}} \right\rceil .$$

The Griesmer Bound

An important lower bound for error-correction codes is the Griesmer bound:

Theorem (Griesmer Bound)

Let C be an $[n, k, d]$ error-correction code. Then

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{d}{2^{i-1}} \right\rceil .$$

In 1998, van Gils generalized the Griesmer bound to UEP codes.

The Griesmer Bound

An important lower bound for error-correction codes is the Griesmer bound:

Theorem (Griesmer Bound)

Let C be an $[n, k, d]$ error-correction code. Then

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{d}{2^{i-1}} \right\rceil .$$

In 1998, van Gils generalized the Griesmer bound to UEP codes.

Theorem (Griesmer Bound for UEP Codes)

Let C be an $[n, k]$ linear code with the separation vector $S = (S_1, \dots, S_k) \in (\mathbb{N} \cup \{0\})^k$ with $S_1 \geq S_2 \geq \dots \geq S_k$. Then

$$n \geq \sum_{i=1}^{k} \left\lceil \frac{S_i}{2^{i-1}} \right\rceil .$$

Table of Contents

PIR Scheme

A private information retrieval (PIR) scheme is any algorithm which allows a user to request data from a system of servers in such a way that no server gets any information about which item was requested.

PIR Scheme

A private information retrieval (PIR) scheme is any algorithm which allows a user to request data from a system of servers in such a way that no server gets any information about which item was requested.

Example

Assume we have two servers S_1 and S_2 which both store the vector $(x_1, \dots, x_n) \in \{0, 1\}^n$ and the user wants to retrieve x_i.

PIR Scheme

A private information retrieval (PIR) scheme is any algorithm which allows a user to request data from a system of servers in such a way that no server gets any information about which item was requested.

Example

Assume we have two servers S_1 and S_2 which both store the vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ and the user wants to retrieve x_i. To this end, they uniformly randomly choose a query vector $q \in \{0,1\}^n$ and send q to S_1 and $q + e_i$ to S_2.

PIR Scheme

A private information retrieval (PIR) scheme is any algorithm which allows a user to request data from a system of servers in such a way that no server gets any information about which item was requested.

Example

Assume we have two servers S_1 and S_2 which both store the vector $(x_1, \dots, x_n) \in \{0,1\}^n$ and the user wants to retrieve x_i. To this end, they uniformly randomly choose a query vector $q \in \{0,1\}^n$ and send q to S_1 and $q + e_i$ to S_2.
Now, both servers calculate the inner product of the query with the data it holds i.e. $r_1 = (x, q)$ and $r_2 = (x, q + e_i)$ and return it to the user.

PIR Scheme

A private information retrieval (PIR) scheme is any algorithm which allows a user to request data from a system of servers in such a way that no server gets any information about which item was requested.

Example

Assume we have two servers S_1 and S_2 which both store the vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ and the user wants to retrieve x_i. To this end, they uniformly randomly choose a query vector $q \in \{0,1\}^n$ and send q to S_1 and $q + e_i$ to S_2.
Now, both servers calculate the inner product of the query with the data it holds i.e. $r_1 = (x, q)$ and $r_2 = (x, q + e_i)$ and return it to the user. The user now calculates the sum of the retrieved data to retrieve x_i:
$r_1 + r_2 = (x, q) + (x, q + e_i) = (x, e_i) = x_i$.

PIR Codes

Classically in PIR schemes, the whole database is replicated among all the servers.

PIR Codes

Classically in PIR schemes, the whole database is replicated among all the servers.
In 2015 Fazeli, Vardy and Yaakobi introduced the concept of PIR codes to get the same result with lower *storage overhead*,

Classically in PIR schemes, the whole database is replicated among all the servers.

In 2015 Fazeli, Vardy and Yaakobi introduced the concept of PIR codes to get the same result with lower *storage overhead*, which is defined as the ratio between the total number of bits stored on all the servers and the number of bits in the database.

PIR codes

Example

We partition the database $x = (x_1, \ldots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3.

PIR codes

Example

We partition the database $x = (x_1, \ldots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3. We store $x^{(1)}$ in T_1, $x^{(2)}$ in T_2 and $x^{(1)} + x^{(2)}$ in T_3 i.e. according to the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Example

We partition the database $x = (x_1, \ldots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3. We store $x^{(1)}$ in T_1, $x^{(2)}$ in T_2 and $x^{(1)} + x^{(2)}$ in T_3 i.e. according to the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Suppose we want to retrieve $x_i = x_j^{(1)}$.

PIR codes

Example

We partition the database $x = (x_1, \ldots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3. We store $x^{(1)}$ in T_1, $x^{(2)}$ in T_2 and $x^{(1)} + x^{(2)}$ in T_3 i.e. according to the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Suppose we want to retrieve $x_i = x_j^{(1)}$. We have two copies of $x^{(1)}$ available: one in T_1 and one by adding the data in T_2 and T_3.

Example

We partition the database $x = (x_1, \dots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3. We store $x^{(1)}$ in T_1, $x^{(2)}$ in T_2 and $x^{(1)} + x^{(2)}$ in T_3 i.e. according to the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Suppose we want to retrieve $x_i = x_j^{(1)}$. We have two copies of $x^{(1)}$ available: one in T_1 and one by adding the data in T_2 and T_3. We send q to T_1 and $q + e_i$ to T_2 and T_3. We retrieve

$$r_1 = (x^{(1)}, q), \quad r_2 = (x^{(2)}, q + e_i), \quad r_3 = (x^{(1)} + x^{(2)}, q + e_i).$$

PIR codes

Example

We partition the database $x = (x_1, \ldots, x_n)$ into two parts: $x = (x^{(1)}, x^{(2)})$ and store these parts in three servers T_1, T_2 and T_3. We store $x^{(1)}$ in T_1, $x^{(2)}$ in T_2 and $x^{(1)} + x^{(2)}$ in T_3 i.e. according to the generator matrix

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

Suppose we want to retrieve $x_i = x_j^{(1)}$. We have two copies of $x^{(1)}$ available: one in T_1 and one by adding the data in T_2 and T_3. We send q to T_1 and $q + e_i$ to T_2 and T_3. We retrieve

$$r_1 = (x^{(1)}, q), \quad r_2 = (x^{(2)}, q + e_i), \quad r_3 = (x^{(1)} + x^{(2)}, q + e_i).$$

Now, $(x^{(1)}, q + e_i) = r_2 + r_3$ and

$$r_1 + (r_2 + r_3) = (x^{(1)}, q) + (x^{(1)}, q + e_i) = (x^{(1)}, e_i) = x_i.$$

PIR Codes

The previous example justifies the following definition:

PIR Codes

The previous example justifies the following definition:

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) (k,t)-*PIR code* if it has the following property:

PIR Codes

The previous example justifies the following definition:

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) (k,t)-*PIR code* if it has the following property:
for every $1 \leq i \leq k$, there exist t disjoint sets of column indices $I_1, ..., I_t \subseteq [n] := \{1, \ldots, n\}$ such that for every $1 \leq j \leq t$, the columns of G with indices from I_j sum up to e_i.

PIR Codes

The previous example justifies the following definition:

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) (k,t)-*PIR code* if it has the following property:
for every $1 \leq i \leq k$, there exist t disjoint sets of column indices $I_1, ..., I_t \subseteq [n] := \{1, \ldots, n\}$ such that for every $1 \leq j \leq t$, the columns of G with indices from I_j sum up to e_i.

Clearly, the higher the value of t, the more accessible the data (e.g. if some servers were to go down or if the data demand is high).

UDD PIR Codes

The previous observation brings us to the concept of *unequal-data-demand (UDD) PIR codes*.

UDD PIR Codes

The previous observation brings us to the concept of *unequal-data-demand (UDD) PIR codes*.

Regular PIR codes are designed for cases where all items in the database are of the same importance

UDD PIR Codes

The previous observation brings us to the concept of *unequal-data-demand (UDD) PIR codes*.

Regular PIR codes are designed for cases where all items in the database are of the same importance

But what if some pieces of data are in much higher demand than others?

UDD PIR Codes

The previous observation brings us to the concept of *unequal-data-demand (UDD) PIR codes*.

Regular PIR codes are designed for cases where all items in the database are of the same importance

But what if some pieces of data are in much higher demand than others?

It would make sense to have more possibilities to recover these pieces of data

UDD PIR Codes

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) $(k,(t_1,\ldots,t_k))$-*UDD PIR code* if it has the following property:

UDD PIR Codes

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) $(k, (t_1, \ldots, t_k))$-*UDD PIR code* if it has the following property:
for every $1 \leq i \leq k$, there exist t_i *disjoint* sets of column indices $I_1, \ldots, I_{t_i} \subseteq [n] := \{1, \ldots, n\}$ such that for every $1 \leq j \leq t_i$, the columns of G with indices from I_j sum up to e_i.

UDD PIR Codes

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) $(k,(t_1,\ldots,t_k))$-*UDD PIR code* if it has the following property:
for every $1 \leq i \leq k$, there exist t_i *disjoint* sets of column indices $I_1, \ldots, I_{t_i} \subseteq [n] := \{1,\ldots,n\}$ such that for every $1 \leq j \leq t_i$, the columns of G with indices from I_j sum up to e_i.

We usually call a $(k,(t_1,\ldots,t_k))$-UDD PIR code simply a $(t_1,\ldots,t_k)$-UDD code.

UDD PIR Codes

Definition

Let C be a k-dimensional linear subspace of $\mathbb{F}_2^n$. We call a generator matrix $G \in \mathrm{Mat}_{k,n}(\mathbb{F}_2)$ of C a (binary) $(k, (t_1, \ldots, t_k))$-*UDD PIR code* if it has the following property:
for every $1 \le i \le k$, there exist t_i *disjoint* sets of column indices $I_1, ..., I_{t_i} \subseteq [n] := \{1, \ldots, n\}$ such that for every $1 \le j \le t_i$, the columns of G with indices from I_j sum up to e_i.

We usually call a $(k, (t_1, \ldots, t_k))$-UDD PIR code simply a $(t_1, \ldots, t_k)$-UDD code. Also, we henceforth always assume $t_1 \geq \ldots \geq t_k$.

UDD PIR Codes

UDD PIR Codes

Example

Consider the generator matrix

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Numbering the columns from left to right, we have the following recovery sets for e_1, e_2 and e_3:

UDD PIR Codes

Example

Consider the generator matrix

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Numbering the columns from left to right, we have the following recovery sets for e_1, e_2 and e_3:

$$\begin{aligned} e_1 &: \{1\}, \{2\}, \{3, 4, 5\} \\ e_2 &: \{3\}, \{1, 4, 5\}, \\ e_3 &: \{4\}, \{1, 3, 5\}. \end{aligned}$$

UDD PIR Codes

Example

Consider the generator matrix

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Numbering the columns from left to right, we have the following recovery sets for e_1, e_2 and e_3:

$$\begin{aligned} e_1 &: \{1\}, \{2\}, \{3, 4, 5\} \\ e_2 &: \{3\}, \{1, 4, 5\}, \\ e_3 &: \{4\}, \{1, 3, 5\}. \end{aligned}$$

Thus, we call G a $(3,(3,2,2))$-UDD PIR code.

UDD PIR Codes

We denote by $P(k, (t_1, \ldots, t_k))$ the shortest possible length of a $(t_1, \ldots, t_k)$-UDD code.

UDD PIR Codes

We denote by $P(k, (t_1, \ldots, t_k))$ the shortest possible length of a $(t_1, \ldots, t_k)$-UDD code.

Problem

Let $k \in \mathbb{N}$ and $T = (t_1, \ldots, t_k)$ with $t_1, \ldots, t_k \in \mathbb{N} \cup \{0\}$ and $t_1 \geq \ldots \geq t_k$. How long is the shortest T-UDD code i.e. what is the value of $P(k, T)$?

Table of Contents

Griesmer bound

It can easily be shown that every t-PIR code has distance at least t. Thus, the following theorem holds:

Griesmer bound

It can easily be shown that every t-PIR code has distance at least t. Thus, the following theorem holds:

Theorem (Griesmer Bound for PIR codes)

Let $k, t \in \mathbb{N}$. *Then*

$$P(k,t) \geq \sum_{i=1}^{k} \left\lceil \frac{t}{2^{i-1}} \right\rceil .$$

Griesmer bound

It can easily be shown that every t-PIR code has distance at least t. Thus, the following theorem holds:

Theorem (Griesmer Bound for PIR codes)

Let $k, t \in \mathbb{N}$. Then

$$P(k,t) \geq \sum_{i=1}^{k} \left\lceil \frac{t}{2^{i-1}} \right\rceil .$$

A similar bound holds for UDD codes:

Griesmer bound

It can easily be shown that every t-PIR code has distance at least t. Thus, the following theorem holds:

Theorem (Griesmer Bound for PIR codes)

Let $k, t \in \mathbb{N}$. Then

$$P(k,t) \geq \sum_{i=1}^{k} \left\lceil \frac{t}{2^{i-1}} \right\rceil .$$

A similar bound holds for UDD codes:

Theorem

Griesmer Bound for UDD PIR codes Let $k \in \mathbb{N}$ and $t_1 \geq \ldots \geq t_k \geq 0$. Then

$$P(k,(t_1,\ldots,t_k)) \geq \sum_{i=1}^{k} \left\lceil \frac{t_i}{2^{i-1}} \right\rceil .$$

Griesmer Bound

Example

In the previous example, we had a $(3,2,2)$-UDD code of length 5, given by the matrix

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

The Griesmer bound says that $P(3,(3,2,2)) \geq 3 + \left\lceil \frac{2}{2} \right\rceil + \left\lceil \frac{2}{4} \right\rceil = 5$, so G is of the minimal possible length.

First Proof of the Griesmer Bound

Lemma

Let $t_1 \geq \ldots \geq t_k \geq 0$ *and* $G = (g_{ij})$ *be a matrix for a* $(t_1, \ldots, t_k)$*-UDD code. Then* G *has a separation vector of* $(S(G)_1, \ldots, S(G)_k)$ *where* $S(G)_i \geq t_i$ *for all* $i \in [k]$*.*

First Proof of the Griesmer Bound

Lemma

Let $t_1 \geq \ldots \geq t_k \geq 0$ *and* $G = (g_{ij})$ *be a matrix for a* $(t_1, \ldots, t_k)$*-UDD code. Then* G *has a separation vector of* $(S(G)_1, \ldots, S(G)_k)$ *where* $S(G)_i \geq t_i$ *for all* $i \in [k]$.

Theorem (Griesmer Bound for UDD PIR codes)

Let $k \in \mathbb{N}$ *and* $t_1 \geq \ldots \geq t_k$. *Then*

$$P(k, (t_1, \ldots, t_k)) \geq \sum_{i=1}^{k} \left\lceil \frac{t_i}{2^{i-1}} \right\rceil .$$

First Proof of the Griesmer Bound

Lemma

Let $t_1 \geq \ldots \geq t_k \geq 0$ and $G = (g_{ij})$ be a matrix for a $(t_1, \ldots, t_k)$-UDD code. Then G has a separation vector of $(S(G)_1, \ldots, S(G)_k)$ where $S(G)_i \geq t_i$ for all $i \in [k]$.

Theorem (Griesmer Bound for UDD PIR codes)

Let $k \in \mathbb{N}$ and $t_1 \geq \ldots \geq t_k$. Then

$$P(k, (t_1, \ldots, t_k)) \geq \sum_{i=1}^{k} \left\lceil \frac{t_i}{2^{i-1}} \right\rceil .$$

Proof. Let G be the matrix form of a $(t_1, \ldots, t_k)$-UDD code. Then G is a generator matrix for an $[n, k]$ linear code with the separation vector $(S(G)_1, \ldots, S(G)_k)$ by the previous lemma. Now the bound follows from the Griesmer bound for UEP codes. □

Second Proof of the Griesmer Bound

Let H_k denote the $(2^k - 1)$-by-$(2^k - 1)$ matrix of scalar products of the non-zero vectors in $\mathbb{F}_2^k$, ordered by their binary representations. Then the following lemma holds.

Second Proof of the Griesmer Bound

Let H_k denote the (2^k-1)-by-(2^k-1) matrix of scalar products of the non-zero vectors in $\mathbb{F}_2^k$, ordered by their binary representations. Then the following lemma holds.

Lemma

1. Adding any two odd-numbered rows of H_k (in $\mathbb{F}_2^{2^k-1}$) gives an even-numbered row of H_k.
2. Let $k \geq 2$ and i be odd. The submatrix which consists of the columns where there is 0 in the i-th row and of all the even-numbered rows of H_k is H_{k-1}.

Second Proof of the Griesmer Bound

Example

$$H_2 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \qquad H_3 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & \color{red}{1} & 1 & \color{red}{0} & 0 & \color{red}{1} & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & \color{red}{0} & 0 & \color{red}{1} & 1 & \color{red}{1} & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & \color{red}{1} & 1 & \color{red}{1} & 1 & \color{red}{0} & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Second Proof of the Griesmer Bound

Theorem

Griesmer Bound for UDD PIR codes Let $k \in \mathbb{N}$ and $t_1 \geq \ldots \geq t_k \geq 0$. Then

$$P(k,(t_1,\ldots,t_k)) \geq \sum_{i=1}^{k} \left\lceil \frac{t_i}{2^{i-1}} \right\rceil .$$

Proof. Let G be a k-by-n matrix corresponding to a $(t_1,\ldots,t_k)$-UDD code, where $t_1 \geq \ldots \geq t_k$.

Second Proof of the Griesmer Bound

Theorem

Griesmer Bound for UDD PIR codes Let $k \in \mathbb{N}$ and $t_1 \geq \ldots \geq t_k \geq 0$. Then

$$P(k, (t_1, \ldots, t_k)) \geq \sum_{i=1}^{k} \left\lceil \frac{t_i}{2^{i-1}} \right\rceil .$$

Proof. Let G be a k-by-n matrix corresponding to a $(t_1, \ldots, t_k)$-UDD code, where $t_1 \geq \ldots \geq t_k$. Suppose that G has a_i columns of the form $\mathrm{i} = (i_{k-1}, \ldots, i_0)^T$ where $i = i_0 + i_1 \cdot 2 + \ldots + i_{k-1}2^{k-1}$, the binary representation of i. Clearly

$$n = \sum_{i=1}^{2^k-1} a_i .$$

Second Proof the Griesmer Bound

By assumption. the unit vector e_j can be written in t_j ways as a sum of columns of G.

Second Proof the Griesmer Bound

By assumption. the unit vector e_j can be written in t_j ways as a sum of columns of G.
Let $h \in \mathbb{F}_2^k \setminus \{0\}$, then $h^\perp \leq \mathbb{F}_2^k$ is a hyperplane.

Second Proof the Griesmer Bound

By assumption. the unit vector e_j can be written in t_j ways as a sum of columns of G.
Let $h \in \mathbb{F}_2^k \setminus \{0\}$, then $h^\perp \leq \mathbb{F}_2^k$ is a hyperplane. Now, if e_j is not an element of the hyperplane $h^\perp$, then each of the t_j recovery sets for e_j must include a column which is *not* it $h^\perp$. This gets us a system of inequalities for $a_1, \ldots, a_{2^k-1}$.

The proof now continues by induction. The bound clearly holds for $k = 1$ with any $t_1 \in \mathbb{N}$. We show the induction step for $k = 3$, the process is identical for a general $k \geq 2$.

The proof now continues by induction. The bound clearly holds for $k = 1$ with any $t_1 \in \mathbb{N}$. We show the induction step for $k = 3$, the process is identical for a general $k \geq 2$. We have the following system of inequalities:

$$
\begin{aligned}
a_1 + a_3 + a_5 + a_7 &\geq t_1 \\
a_2 + a_3 + a_6 + a_7 &\geq t_2 \\
a_1 + a_2 + a_5 + a_6 &\geq t_1 \\
a_4 + a_5 + a_6 + a_7 &\geq t_3 \\
a_1 + a_3 + a_4 + a_6 &\geq t_1 \\
a_2 + a_3 + a_4 + a_7 &\geq t_2 \\
a_1 + a_2 + a_4 + a_7 &\geq t_1
\end{aligned}
$$

The proof now continues by induction. The bound clearly holds for $k = 1$ with any $t_1 \in \mathbb{N}$. We show the induction step for $k = 3$, the process is identical for a general $k \geq 2$. We have the following system of inequalities:

$$
\begin{aligned}
a_1 + a_3 + a_5 + a_7 &\geq t_1 \\
a_2 + a_3 + a_6 + a_7 &\geq t_2 \\
a_1 + a_2 + a_5 + a_6 &\geq t_1 \\
a_4 + a_5 + a_6 + a_7 &\geq t_3 \\
a_1 + a_3 + a_4 + a_6 &\geq t_1 \\
a_2 + a_3 + a_4 + a_7 &\geq t_2 \\
a_1 + a_2 + a_4 + a_7 &\geq t_1
\end{aligned}
$$

The variable a_1 appears only in the inequalities for t_1. If none of these inequalities were tight, we could reduce a_1 by one and still get a solution. We therefore w.l.o.g. assume that one of these inequalities is tight.

The proof now continues by induction. The bound clearly holds for $k = 1$ with any $t_1 \in \mathbb{N}$. We show the induction step for $k = 3$, the process is identical for a general $k \geq 2$. We have the following system of inequalities:

$$\begin{aligned}
a_1 + a_3 + a_5 + a_7 &\geq t_1 \\
a_2 + a_3 + a_6 + a_7 &\geq t_2 \\
a_1 + a_2 + a_5 + a_6 &\geq t_1 \\
a_4 + a_5 + a_6 + a_7 &\geq t_3 \\
a_1 + a_3 + a_4 + a_6 &\geq t_1 \\
a_2 + a_3 + a_4 + a_7 &\geq t_2 \\
a_1 + a_2 + a_4 + a_7 &\geq t_1
\end{aligned}$$

The variable a_1 appears only in the inequalities for t_1. If none of these inequalities were tight, we could reduce a_1 by one and still get a solution. We therefore w.l.o.g. assume that one of these inequalities is tight. We subtract this equality from all the other inequalities with t_1.

First Proof of the Griesmer Bound

In the case where the first inequality is actually an equality, we get the following system:

$$
\begin{aligned}
a_1 + a_3 + a_5 + a_7 &\geq t_1 \\
a_2 + a_3 + a_6 + a_7 &\geq t_2 \\
a_2 - a_3 + a_6 - a_7 &\geq 0 \\
a_4 + a_5 + a_6 + a_7 &\geq t_3 \\
a_4 - a_5 + a_6 - a_7 &\geq 0 \\
a_2 + a_3 + a_4 + a_7 &\geq t_2 \\
a_2 - a_3 + a_4 - a_5 &\geq 0
\end{aligned}
$$

Adding the 2. and 3. inequality, the 4. and 5. inequality and the 6. and 7. inequality, we get the new system with the matrix H_2 (the previous lemma justifies these steps in the general case).

$$2a_2 + 2a_6 \geq t_2$$
$$2a_4 + 2a_6 \geq t_3$$
$$2a_2 + 2a_4 \geq t_2$$

Adding the 2. and 3. inequality, the 4. and 5. inequality and the 6. and 7. inequality, we get the new system with the matrix H_2 (the previous lemma justifies these steps in the general case).

$$2a_2 + 2a_6 \geq t_2$$
$$2a_4 + 2a_6 \geq t_3$$
$$2a_2 + 2a_4 \geq t_2$$

The result now quickly follows from the induction hypothesis. □

Adding the 2. and 3. inequality, the 4. and 5. inequality and the 6. and 7. inequality, we get the new system with the matrix H_2 (the previous lemma justifies these steps in the general case).

$$2a_2 + 2a_6 \geq t_2$$
$$2a_4 + 2a_6 \geq t_3$$
$$2a_2 + 2a_4 \geq t_2$$

The result now quickly follows from the induction hypothesis. □

As every error-correction code is a *UEP* code and every *UEP* code is a *UDD* code, this is a completely new proof for the Griesmer bound for UEP codes and error-correction codes.

Table of Contents

Optimality of the Griesmer Bound for $k \leq 3$

For $k \leq 3$, the Griesmer bound can always be achieved.

Optimality of the Griesmer Bound for $k \leq 3$

For $k \leq 3$, the Griesmer bound can always be achieved.
We show this for $k = 3$, then $k \leq 2$ follows directly.

Optimality of the Griesmer Bound for $k \leq 3$

For $k \leq 3$, the Griesmer bound can always be achieved.
We show this for $k = 3$, then $k \leq 2$ follows directly. In order to do this, we first show optimal constructions for a few special codes:

Optimality of the Griesmer Bound for $k \leq 3$

An optimal $(1, 1, 0)$-UDD code is $G = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$.

An optimal $(1, 1, 1)$-UDD code is $G = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

An optimal $(2, 2, 2)$-UDD code is $G = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$.

An optimal $(3, 3, 3)$-UDD code is $G = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$.

An optimal $(4, 4, 4)$-UDD code is $G = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.
Assume we have a (t_1, t_2, t_3)-UDD code with length $G(t_1, t_2, t_3)$.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.
Assume we have a (t_1, t_2, t_3)-UDD code with length $G(t_1, t_2, t_3)$.

1. As $G(t_1 + 1, t_2, t_3) = G(t_1, t_2, t_3) + 1$, we can get a $(t_1 + 1, t_2, t_3)$-UDD code of length $G(t_1 + 1, t_2, t_3)$ just by adding a column $(1, 0, 0)^T$.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.

Assume we have a (t_1, t_2, t_3)-UDD code with length $G(t_1, t_2, t_3)$.

1. As $G(t_1 + 1, t_2, t_3) = G(t_1, t_2, t_3) + 1$, we can get a $(t_1 + 1, t_2, t_3)$-UDD code of length $G(t_1 + 1, t_2, t_3)$ just by adding a column $(1, 0, 0)^T$.
2. As $G(t_1 + 2, t_2 + 2, t_3) = G(t_1, t_2, t_3) + 3$ we can get a $(t_1 + 2, t_2 + 2, t_3)$-UDD code of length $G(t_1 + 2, t_2 + 2, t_3)$ by adding the columns $(1, 0, 0)^T$, $(0, 1, 0)^T$ and $(1, 1, 0)^T$.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.
Assume we have a (t_1, t_2, t_3)-UDD code with length $G(t_1, t_2, t_3)$.

1. As $G(t_1 + 1, t_2, t_3) = G(t_1, t_2, t_3) + 1$, we can get a $(t_1 + 1, t_2, t_3)$-UDD code of length $G(t_1 + 1, t_2, t_3)$ just by adding a column $(1, 0, 0)^T$.
2. As $G(t_1 + 2, t_2 + 2, t_3) = G(t_1, t_2, t_3) + 3$ we can get a $(t_1 + 2, t_2 + 2, t_3)$-UDD code of length $G(t_1 + 2, t_2 + 2, t_3)$ by adding the columns $(1, 0, 0)^T$, $(0, 1, 0)^T$ and $(1, 1, 0)^T$.
3. As $G(t_1 + 4, t_2 + 4, t_3 + 4) = G(t_1, t_2, t_3) + 7$, we can get a code of length $G(t_1 + 4, t_2 + 4, t_3 + 4)$ for $T = (t_1 + 4, t_2 + 4, t_3 + 4)$ by adding one of each non-zero column.

Optimality of the Griesmer Bound for $k \leq 3$

A code of length $G(t_1, t_2, t_3)$ can be generated from the previous codes for every $t_1 \geq t_2 \geq t_3$.
Assume we have a (t_1, t_2, t_3)-UDD code with length $G(t_1, t_2, t_3)$.

1. As $G(t_1 + 1, t_2, t_3) = G(t_1, t_2, t_3) + 1$, we can get a $(t_1 + 1, t_2, t_3)$-UDD code of length $G(t_1 + 1, t_2, t_3)$ just by adding a column $(1, 0, 0)^T$.
2. As $G(t_1 + 2, t_2 + 2, t_3) = G(t_1, t_2, t_3) + 3$ we can get a $(t_1 + 2, t_2 + 2, t_3)$-UDD code of length $G(t_1 + 2, t_2 + 2, t_3)$ by adding the columns $(1, 0, 0)^T$, $(0, 1, 0)^T$ and $(1, 1, 0)^T$.
3. As $G(t_1 + 4, t_2 + 4, t_3 + 4) = G(t_1, t_2, t_3) + 7$, we can get a code of length $G(t_1 + 4, t_2 + 4, t_3 + 4)$ for $T = (t_1 + 4, t_2 + 4, t_3 + 4)$ by adding one of each non-zero column.
4. As $G(t_1, t_2, 4m) = G(t_1, t_2, 4m - 1) = G(t_1, t_2, 4m - 2) = G(t_1, t_2, 4m - 3)$, we can round t_3 up to $\min\{t_2, 4m\}$ where 4m is the smallest multiple of 4 larger than t_3.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns).

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns). This adds 7 columns to G and adds 4 recovery sets for each unit vector.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns). This adds 7 columns to G and adds 4 recovery sets for each unit vector. We now need a $(6, 5, 1)$-UDD code.

We add $\left\lfloor \frac{t_2' - t_3'}{2} \right\rfloor = 2$ of each vector e_1, e_2 and $e_1 + e_2$.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns). This adds 7 columns to G and adds 4 recovery sets for each unit vector. We now need a $(6, 5, 1)$-UDD code.

We add $\left\lfloor \frac{t_2' - t_3'}{2} \right\rfloor = 2$ of each vector e_1, e_2 and $e_1 + e_2$. This adds 6 columns to G and 4 recovery sets for e_1, e_2.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns). This adds 7 columns to G and adds 4 recovery sets for each unit vector. We now need a $(6, 5, 1)$-UDD code.

We add $\left\lfloor \frac{t_2' - t_3'}{2} \right\rfloor = 2$ of each vector e_1, e_2 and $e_1 + e_2$. This adds 6 columns to G and 4 recovery sets for e_1, e_2. We now need a $(2, 1, 1)$-UDD code.

Optimality of the Griesmer Bound for $k \leq 3$

The previous slide provides an algorithm to generate an optimal matrix for any triple $t_1 \geq t_2 \geq t_3$.

Example

Let $(t_1, t_2, t_3) = (10, 9, 5)$. We construct a $(10, 9, 5)$-UDD code of length $G(10, 9, 5) = 17$:

First we add $\left\lfloor \frac{t_3}{4} \right\rfloor = 1$ simplex code (all non-zero columns). This adds 7 columns to G and adds 4 recovery sets for each unit vector. We now need a $(6, 5, 1)$-UDD code.

We add $\left\lfloor \frac{t_2' - t_3'}{2} \right\rfloor = 2$ of each vector e_1, e_2 and $e_1 + e_2$. This adds 6 columns to G and 4 recovery sets for e_1, e_2. We now need a $(2, 1, 1)$-UDD code.

We add $t_2'' - t_1'' = 1$ of e_1 which reduces us to finding an optimal $(1, 1, 1)$-UDD code. This was done 2 slides ago.

Example

We get the following matrix:

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Thank you!

Questions?