The Generalized Covering Radii of Codes

Dor Elimelech

Ben-Gurion University of the Negev, Israel

Joint work with:
Moshe Schwartz (BGU), Marcelo Firer (Unicamp), Hengjia Wei (Peng Cheng Laboratory)

Motivation

Linear Data Querying

Linear Data Querying

- A common query type in database systems involves a linear combination of the database items with coefficients supplied by the user.

Linear Data Querying

Linear Data Querying

- A common query type in database systems involves a linear combination of the database items with coefficients supplied by the user.
- The database server - m items, $x_1, \ldots, x_m \in \mathbb{F}_{q^\ell}$.

Linear Data Querying

Linear Data Querying

- A common query type in database systems involves a linear combination of the database items with coefficients supplied by the user.
- The database server - m items, $x_1, \ldots, x_m \in \mathbb{F}_{q^\ell}$.
- The user queries the database by providing $s_1, \ldots, s_m \in \mathbb{F}_q$.

Linear Data Querying

Linear Data Querying

- A common query type in database systems involves a linear combination of the database items with coefficients supplied by the user.
- The database server - m items, $x_1, \ldots, x_m \in \mathbb{F}_{q^\ell}$.
- The user queries the database by providing $s_1, \ldots, s_m \in \mathbb{F}_q$.
- The database returns the linear combination $\sum_{i=1}^{m} s_i x_i$.

Linear Data Querying

Linear Data Querying

- A common query type in database systems involves a linear combination of the database items with coefficients supplied by the user.
- The database server - m items, $x_1, \ldots, x_m \in \mathbb{F}_{q^\ell}$.
- The user queries the database by providing $s_1, \ldots, s_m \in \mathbb{F}_q$.
- The database returns the linear combination $\sum_{i=1}^m s_i x_i$.

Examples

- Private information retrieval (PIR).

— see **Chor, Goldreich, Kushilevitz and Sudan**, *IEEE Trans. on Inform. Theory*, 1995.

- Partial-sum queries.

— see **Chazelle and Rosenberg**, *Proceedings of the fifth annual symposium on Computational geometry,*, 1989.

Linear Data Querying (Cont.)

Aspects in need of optimization

- Amount of storage at the server.
- The required bandwidth for the querying protocol.
- Access Complexity - the time required to access the elements of the database needed to compute the answer to a user query.

Linear Data Querying (Cont.)

Aspects in need of optimization

- Amount of storage at the server.
- The required bandwidth for the querying protocol.
- Access Complexity - the time required to access the elements of the database needed to compute the answer to a user query.

Access Complexity

- The time required is proportional to the number of non-zero coefficients among $s_1, \ldots, s_m$.
- In the PIR scheme the coefficients are uniform i.i.d variables over $\mathbb{F}_q$ $\implies$ The expected number of non-zero coefficients is $\left(1 - \frac{1}{q}\right)m$.

Access Complexity and Storage

Reducing Access Complexity

- Design a set of linear combinations to be pre-computed and stored by the server.
- Instead of storing $\overline{x} = (x_1, \ldots, x_m)$ as is, the server stores $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, where each $\overline{h}_i \in \mathbb{F}_q^m$ describes a linear combination.

Access Complexity and Storage

Reducing Access Complexity

- Design a set of linear combinations to be pre-computed and stored by the server.
- Instead of storing $\overline{x} = (x_1, \ldots, x_m)$ as is, the server stores $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, where each $\overline{h}_i \in \mathbb{F}_q^m$ describes a linear combination.

Example

- If $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$ are all the possible linear combinations, the access complexity drops to just one item per query.

Access Complexity and Storage

Reducing Access Complexity

- Design a set of linear combinations to be pre-computed and stored by the server.
- Instead of storing $\overline{x} = (x_1, \ldots, x_m)$ as is, the server stores $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, where each $\overline{h}_i \in \mathbb{F}_q^m$ describes a linear combination.

Example

- If $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$ are all the possible linear combinations, the access complexity drops to just one item per query.
- Problem: The required storage amount is q^m instead of m.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored
- The combinations $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_1 + \overline{x}_3$ are queried.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored
- The combinations $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_1 + \overline{x}_3$ are queried.
- If we answer each query separately - we need 1 access for $\overline{x}_1 + \overline{x}_2$ and 2 elements of $\overline{x}_1 + \overline{x}_3$.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored
- The combinations $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_1 + \overline{x}_3$ are queried.
- If we answer each query separately - we need 1 access for $\overline{x}_1 + \overline{x}_2$ and 2 elements of $\overline{x}_1 + \overline{x}_3$. Overall - access 3 database elements.

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored
- The combinations $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_1 + \overline{x}_3$ are queried.
- If we answer each query separately - we need 1 access for $\overline{x}_1 + \overline{x}_2$ and 2 elements of $\overline{x}_1 + \overline{x}_3$. Overall - access 3 database elements.
- If we answer both queries together - by accessing $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_3 - \overline{x}_2$, we may obtain

$$\overline{x}_3 + \overline{x}_1 = (\overline{x}_1 + \overline{x}_2) + (\overline{x}_3 - \overline{x}_2).$$

Access Complexity-Storage-Latency Trade-Off

Reducing Access Complexity

- **Group Queries** - instead of answering one query at the time - accesses the combinations required for $t \geqslant 1$ queries together.
- Problem: This causes latency.

Example

- Assume that we have the combinations $\overline{x}_1$,$\overline{x}_2$,$\overline{x}_3$,$\overline{x}_1 + \overline{x}_2$, $\overline{x}_3 - \overline{x}_2$ pre-computed and stored
- The combinations $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_1 + \overline{x}_3$ are queried.
- If we answer each query separately - we need 1 access for $\overline{x}_1 + \overline{x}_2$ and 2 elements of $\overline{x}_1 + \overline{x}_3$. Overall - access 3 database elements.
- If we answer both queries together - by accessing $\overline{x}_1 + \overline{x}_2$ and $\overline{x}_3 - \overline{x}_2$, we may obtain

$$\overline{x}_3 + \overline{x}_1 = (\overline{x}_1 + \overline{x}_2) + (\overline{x}_3 - \overline{x}_2).$$

Save 1 access to the database.

Access Complexity-Storage-Latency Trade-Off

Question

- Having the pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, how many pre-computed combinations needs to be accessed in order to answer t linear queries together?

Access Complexity-Storage-Latency Trade-Off

Question

- Having the pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, how many pre-computed combinations needs to be accessed in order to answer t linear queries together?
- Namely, what is the minimal r such that any t coefficient vectors $\overline{s}_1, \ldots, \overline{s}_t$ can be computed as a linear combination of $r \leqslant m$ of $\overline{h}_1, \ldots, \overline{h}_n$. That is, there exits $\ell_1, \ldots, \ell_r$, such that $\overline{s}_j = \sum_{i=1}^{r} \alpha_i \overline{h}_{\ell_i}$? in that case, $\overline{s}_j \cdot \overline{x} = \sum_{i=1}^{r} \alpha_i (\overline{h}_{\ell_i} \cdot \overline{x})$

Access Complexity-Storage-Latency Trade-Off

Question

- Having the pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, how many pre-computed combinations needs to be accessed in order to answer t linear queries together?
- Namely, what is the minimal r such that any t coefficient vectors $\overline{s}_1, \ldots, \overline{s}_t$ can be computed as a linear combination of $r \leqslant m$ of $\overline{h}_1, \ldots, \overline{h}_n$. That is, there exits $\ell_1, \ldots, \ell_r$, such that $\overline{s}_j = \sum_{i=1}^{r} \alpha_i \overline{h}_{\ell_i}$? in that case, $\overline{s}_j \cdot \overline{x} = \sum_{i=1}^{r} \alpha_i (\overline{h}_{\ell_i} \cdot \overline{x})$
- We define this number to be the t-th generalized covering radius of the $[n, n-m=k]_q$ linear code generated by the parity matrix H whose columns are $\overline{h}_1, \ldots, \overline{h}_n$.

The Generalized Covering Radius Definition

The Generalized Covering Radius Definition

We may consider the vectors $\overline{h}_1, \ldots, \overline{h}_n$ as the columns of a parity-check matrix H of some $[n, n-m=k]_q$ code C.

The Generalized Covering Radius Definition

We may consider the vectors $\overline{h}_1, \ldots, \overline{h}_n$ as the columns of a parity-check matrix H of some $[n, n-m=k]_q$ code C.

Definition

Let C be an $[n,k]_q$ linear code over $\mathbb{F}_q$, given by an $(n-k) \times n$ parity-check matrix $H \in \mathbb{F}_q^{(n-k)\times n}$ with columns $\overline{h}_1, \ldots, \overline{h}_n$. For every $t \in \mathbb{N}$ we define the *t-th generalized covering radius*, $R_t(C)$, to be the minimal integer $r \in \mathbb{N}$ such that for every set of vectors $S = \{\overline{s}_1, \ldots, \overline{s}_t\} \subseteq \mathbb{F}_q^{n-k}$, there exists $I \subseteq \mathbb{F}_q^{n-k}$, $|I| = r$ such that $S \subseteq \operatorname{span}\left\{\overline{h}_i \ : \ i \in I\right\}$.

The Generalized Covering Radius Definition

We may consider the vectors $\overline{h}_1, \ldots, \overline{h}_n$ as the columns of a parity-check matrix H of some $[n, n-m=k]_q$ code C.

Definition

Let C be an $[n, k]_q$ linear code over $\mathbb{F}_q$, given by an $(n-k) \times n$ parity-check matrix $H \in \mathbb{F}_q^{(n-k)\times n}$ with columns $\overline{h}_1, \ldots, \overline{h}_n$. For every $t \in \mathbb{N}$ we define the t-th generalized covering radius, $R_t(C)$, to be the minimal integer $r \in \mathbb{N}$ such that for every set of vectors $S = \{\overline{s}_1, \ldots, \overline{s}_t\} \subseteq \mathbb{F}_q^{n-k}$, there exists $I \subseteq \mathbb{F}_q^{n-k}$, $|I| = r$ such that $S \subseteq \mathrm{span}\left\{\overline{h}_i \ : \ i \in I\right\}$.

The Trade-Off as A Covering Problem

The smallest possible number of pre-computed combinations for answering a group of t queries accessing r database elements is lower bounded by the smallest possible length of a linear code with t-covering radius r and redundancy m over $\mathbb{F}_q$.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.
- For all $1 \leqslant t \leqslant n - k$, $t \leqslant R_t(C)$.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.
- For all $1 \leqslant t \leqslant n - k$, $t \leqslant R_t(C)$.
- Monotonicity - $R_1(C) \leqslant R_2(C) \cdots \leqslant R_{n-k}(C) = n - k$.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.
- For all $1 \leqslant t \leqslant n-k$, $t \leqslant R_t(C)$.
- Monotonicity - $R_1(C) \leqslant R_2(C) \cdots \leqslant R_{n-k}(C) = n-k$.

Example

Let C be the binary $[2^m-1, 2^m-m-1, 3]$ Hamming code. The columns of the standard parity check matrix H are all the binary non-zero vectors of length m. What is $R_t(C)$ for $1 \leqslant t \leqslant m$?

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.
- For all $1 \leqslant t \leqslant n-k$, $t \leqslant R_t(C)$.
- Monotonicity - $R_1(C) \leqslant R_2(C) \cdots \leqslant R_{n-k}(C) = n-k$.

Example

Let C be the binary $[2^m-1, 2^m-m-1, 3]$ Hamming code. The columns of the standard parity check matrix H are all the binary non-zero vectors of length m. What is $R_t(C)$ for $1 \leqslant t \leqslant m$?
Given t non-zero vectors in $\mathbb{F}_2^m$, they are all columns of H. Thus, $R_t(C) \leqslant t$.

The Generalized Covering Radius (Cont.)

Remarks

- For $t = 1$ we get the regular covering radius of a linear code.
- This definition does not depend on the choice of the parity check matrix.
- For all $1 \leqslant t \leqslant n - k$, $t \leqslant R_t(C)$.
- Monotonicity - $R_1(C) \leqslant R_2(C) \cdots \leqslant R_{n-k}(C) = n - k$.

Example

Let C be the binary $[2^m - 1, 2^m - m - 1, 3]$ Hamming code. The columns of the standard parity check matrix H are all the binary non-zero vectors of length m. What is $R_t(C)$ for $1 \leqslant t \leqslant m$?
Given t non-zero vectors in $\mathbb{F}_2^m$, they are all columns of H. Thus, $R_t(C) \leqslant t$. By the remark - $R_t(C) = t$.

Equivalent Definitions

t-Weights and t-distance

- For $t \in \mathbb{N}$ we define the t-weights on $\mathbb{F}_q^{t\times n}$. For a matrix $\mathbf{v} \in \mathbb{F}_q^{t\times n}$ with row vectors $\overline{v}_1, \ldots, \overline{v}_t$ we define

$$\mathrm{wt}^{(t)}(\mathbf{v}) \triangleq \left| \bigcup_{1\leqslant i\leqslant t} \mathrm{supp}(\overline{v}_i) \right|, \quad d^{(t)}(\mathbf{v}_1, \mathbf{v}_2) \triangleq \mathrm{wt}^{(t)}(\mathbf{v}_1 - \mathbf{v}_2).$$

- For a matrix $\mathbf{v} \in \mathbb{F}_q^{t\times n}$ we denote the ball or radius r centred in $\mathbf{v}$ (with respect to $d^{(t)}$) by $B_r^{(t)}(\mathbf{v})$, and it volume by $V_{r,n,q}^{(t)}$.
- If $C \subseteq \mathbb{F}_q^n$ is a linear code, we define C^t to be the set of all matrices in $\mathbb{F}_q^{t\times n}$ such that their rows belongs in C.

Equivalent Definitions(Cont.)

Geometric Definition

Let C be an $[n,k]_q$ code, we consider $C^t \subseteq \mathbb{F}_q^{t\times n}$. Then $R_t(C)$ is the regular covering radius of C^t with respect to $d^{(t)}$:

$$R_t(C) = \min\left\{ r \in \mathbb{N} \ : \ \bigcup_{\mathbf{c}\in C^t} B_r^{(t)}(\mathbf{c}) = \mathbb{F}_q^{t\times n} \right\}.$$

Equivalent Definitions(Cont.)

Geometric Definition

Let C be an $[n,k]_q$ code, we consider $C^t \subseteq \mathbb{F}_q^{t\times n}$. Then $R_t(C)$ is the regular covering radius of C^t with respect to $d^{(t)}$:

$$R_t(C) = \min\left\{ r \in \mathbb{N} \ : \ \bigcup_{\mathbf{c}\in C^t} B_r^{(t)}(\mathbf{c}) = \mathbb{F}_q^{t\times n} \right\}.$$

Remark: This definition extends to general (perhaps non-linear) codes.

Equivalent Definitions(Cont.)

Geometric Definition

Let C be an $[n,k]_q$ code, we consider $C^t \subseteq \mathbb{F}_q^{t\times n}$. Then $R_t(C)$ is the regular covering radius of C^t with respect to $d^{(t)}$:

$$R_t(C) = \min\left\{ r \in \mathbb{N} \ : \ \bigcup_{\mathbf{c}\in C^t} B_r^{(t)}(\mathbf{c}) = \mathbb{F}_q^{t\times n} \right\}.$$

Remark: This definition extends to general (perhaps non-linear) codes.

Algebraic Definition

Let C be an $[n,k]_q$ linear code. Assume $G \in \mathbb{F}_q^{k\times n}$ is a generator matrix for C. Let C_t be the linear code over $\mathbb{F}_{q^t}$ generated by the same matrix G. That is, $C_t = \left\{ \overline{u}G \ : \ \overline{u} \in \mathbb{F}_{q^t}^k \right\}$. Then $R_t(C) = R_1(C_t)$.

The algebraic definition is related to the work of Helleseth on extension codes (1979).

Asymptotic Results

Bounds

Definition

Let $k_t(n, r, q)$ denote the smallest dimension of a linear code C over $\mathbb{F}_q$ with length n and t-covering radius $R_t(C) \leqslant r$.

Bounds

Definition

Let $k_t(n, r, q)$ denote the smallest dimension of a linear code C over $\mathbb{F}_q$ with length n and t-covering radius $R_t(C) \leqslant r$.
For a normalized covering radius $0 \leqslant \rho \leqslant 1$, the minimal rate achieving ρ is defined to be

$$\kappa_t(\rho, q) \triangleq \liminf_{n\to\infty} \frac{k_t(n, \rho n, q)}{n}.$$

Bounds

Definition

Let $k_t(n, r, q)$ denote the smallest dimension of a linear code C over $\mathbb{F}_q$ with length n and t-covering radius $R_t(C) \leqslant r$.
For a normalized covering radius $0 \leqslant \rho \leqslant 1$, the minimal rate achieving ρ is defined to be

$$\kappa_t(\rho, q) \triangleq \liminf_{n\to\infty} \frac{k_t(n, \rho n, q)}{n}.$$

Lower Bound (Ball-Covering)

For $\rho \in \left(0, 1 - \frac{1}{q^t}\right)$,

$$\kappa_t(\rho, q) \geqslant 1 - H_{q^t}(\rho).$$

A Naive Upper Bound

Theorem (A Naive Upper Bound)

For $\rho \in [0, 1]$
$$\kappa_t(\rho, q) \leqslant \kappa_1\left(\frac{\rho}{t}, q\right) = 1 - H_q\left(\frac{\rho}{t}\right)$$

A Naive Upper Bound

Theorem (A Naive Upper Bound)

For $\rho \in [0, 1]$

$$\kappa_t(\rho, q) \leqslant \kappa_1\left(\frac{\rho}{t}, q\right) = 1 - H_q\left(\frac{\rho}{t}\right)$$

Proof Sketch

We use a sub-additivity property:

For an $[n, k]_q$ linear code C and $t_1, t_2 \in \mathbb{N}$,

$$R_{t_1+t_2}(C) \leqslant R_{t_1}(C) + R_{t_2}(C).$$

As a consequence $R_t(C) \leqslant t \cdot R_1(C)$.

We combine with a result by Cohen and Frankl (1985)

$$\kappa_1(\rho, q) = 1 - H_q(\rho).$$

Improved Upper Bound for the binary case where $t = 2$

Theorem

For any $0 < \rho \leqslant 1$,

$$\kappa_2(\rho, 2) \leqslant \begin{cases} 1 - (4H_4(\rho) - f(\rho)) & 0 \leqslant \rho < \frac{3}{4}, \\ 0 & \frac{3}{4} \leqslant \rho \leqslant 1, \end{cases}$$

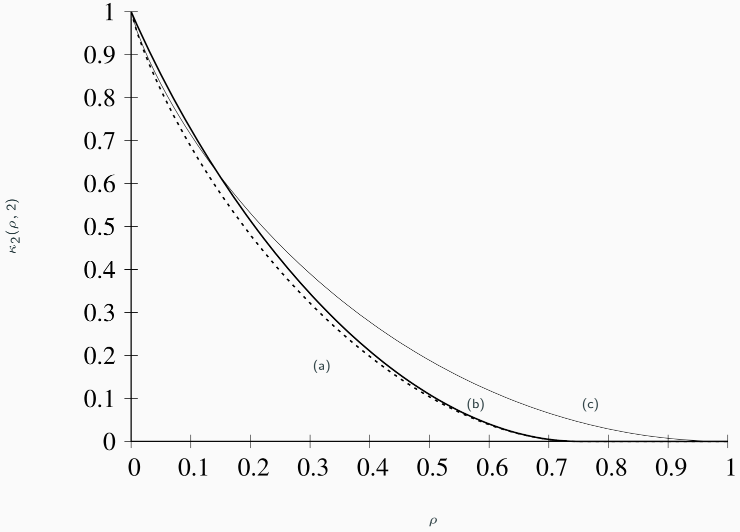
where

$$f(\rho) = \begin{cases} H_2(s(\rho)) + 2s(\rho) + 2(1 - s(\rho))H_2\left(\frac{\rho - s(\rho)}{1 - s(\rho)}\right) & 0 \leqslant \rho < \frac{3}{4}, \\ 3 & \frac{3}{4} \leqslant \rho \leqslant 1. \end{cases}$$

and

$$s(\rho) \triangleq \frac{1}{10}\left(1 + 8\rho - \sqrt{1 + 16\rho - 16\rho^2}\right).$$

A Comparison of The Bounds



A comparison of the bounds on $\kappa_2(\rho, 2)$: (a) the ball-covering lower bound, (b) the improved upper bound, and (c) the naive upper bound.

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

We use the probabilistic method.

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

We use the probabilistic method. Assume $G \in \mathbb{F}_2^{k \times n}$ is a uniformly random matrix, and consider the code $C^2 = \left\{ \mathbf{u}G \ : \ \mathbf{u} \in \mathbb{F}_2^{2 \times k} \right\}$.

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

We use the probabilistic method. Assume $G \in \mathbb{F}_2^{k \times n}$ is a uniformly random matrix, and consider the code $C^2 = \left\{ \mathbf{u}G \ : \ \mathbf{u} \in \mathbb{F}_2^{2 \times k} \right\}$. For any $\mathbf{v} \in \mathbb{F}_2^{2 \times n}$, we count the number of times it is covered by balls of radius r around codewords of C^2 that are generated by full-rank matrices:

$$X_{\mathbf{v}} \triangleq \sum_{\substack{\mathbf{u} \in \mathbb{F}_2^{2 \times k} \\ \text{rank}(\mathbf{u}) = 2}} \mathbb{I}_{\left\{ \mathbf{v} \in B_r^{(2)}(\mathbf{u}G) \right\}}.$$

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

We use the probabilistic method. Assume $G \in \mathbb{F}_2^{k\times n}$ is a uniformly random matrix, and consider the code $C^2 = \left\{\mathbf{u}G \ : \ \mathbf{u} \in \mathbb{F}_2^{2\times k}\right\}$. For any $\mathbf{v} \in \mathbb{F}_2^{2\times n}$, we count the number of times it is covered by balls of radius r around codewords of C^2 that are generated by full-rank matrices:

$$X_{\mathbf{v}} \triangleq \sum_{\substack{\mathbf{u}\in\mathbb{F}_2^{2\times k} \\ \mathrm{rank}(\mathbf{u})=2}} \mathbb{I}_{\left\{\mathbf{v}\in B_r^{(2)}(\mathbf{u}G)\right\}}.$$

Using a careful analysis we can determine that

$$\begin{aligned} V_{r,n,2}^{(2)} \cdot 2^{2k-1-2n} < \mathrm{E}[X_{\mathbf{v}}] < V_{r,n,2}^{(2)} \cdot 2^{2k-2n}, \\ \mathrm{Var}(X_{\mathbf{v}}) \leqslant 7\,\mathrm{E}[X_{\mathbf{v}}] + 2^{3(k-n)+n(f(\rho)+o(1))}. \end{aligned}$$

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

The code is then constructed in two stages:

1. We choose

$$k = \lceil n(1 - 4H_4(\rho) + f(\rho)) + \log_2(n) \rceil.$$

   We then show the code obtained when G has k rows already covers a large enough portion of the space.

Improved Upper Bound for the binary case where $t = 2$

Proof Sketch

The code is then constructed in two stages:

1. We choose
$$k = \lceil n(1 - 4H_4(\rho) + f(\rho)) + \log_2(n) \rceil.$$
We then show the code obtained when G has k rows already covers a large enough portion of the space.
2. We then successively add $2\lceil \log_2(n) \rceil + 2$ rows to G that to guarantee the coverage of the entire space.

Let us place this in the context of PIR

- Consider the binary case, and assume we allow a latency of $t = 2$, namely, the server waits until two queries arrive and then handles them both.

Let us place this in the context of PIR

- Consider the binary case, and assume we allow a latency of $t = 2$, namely, the server waits until two queries arrive and then handles them both.
- $r = \rho n$ represents the number of database elements we allow to access for answering 2 queries, $k = n - m$ represents the overhead.

Let us place this in the context of PIR

- Consider the binary case, and assume we allow a latency of $t = 2$, namely, the server waits until two queries arrive and then handles them both.
- $r = \rho n$ represents the number of database elements we allow to access for answering 2 queries, $k = n - m$ represents the overhead.
- Further assume, that to handle the two queries we allow the server to access at most $\frac{1}{2}$ of its storage. Stated alternatively, the average access per query is a $\frac{1}{4}$ of the storage.

Let us place this in the context of PIR

- Consider the binary case, and assume we allow a latency of $t = 2$, namely, the server waits until two queries arrive and then handles them both.
- $r = \rho n$ represents the number of database elements we allow to access for answering 2 queries, $k = n - m$ represents the overhead.
- Further assume, that to handle the two queries we allow the server to access at most $\frac{1}{2}$ of its storage. Stated alternatively, the average access per query is a $\frac{1}{4}$ of the storage.
- A naive approach, using $\kappa_1(\frac{1}{4}, 2) \approx 0.19$, implies the storage may contain only 81% user information and 19% overhead.

Let us place this in the context of PIR

- Consider the binary case, and assume we allow a latency of $t = 2$, namely, the server waits until two queries arrive and then handles them both.
- $r = \rho n$ represents the number of database elements we allow to access for answering 2 queries, $k = n - m$ represents the overhead.
- Further assume, that to handle the two queries we allow the server to access at most $\frac{1}{2}$ of its storage. Stated alternatively, the average access per query is a $\frac{1}{4}$ of the storage.
- A naive approach, using $\kappa_1(\frac{1}{4}, 2) \approx 0.19$, implies the storage may contain only 81% user information and 19% overhead.
- Since $\kappa_2(\frac{1}{2}, 2) \leqslant 0.11$, there exists a code allowing 89% of the server storage for user information and only 11% overhead.

A step towards the calculation of $\kappa_t(\rho, q)$

Conjecture

We conjecture that the lower bound meets the true value of $\kappa_t(\rho, q)$. That is, For any $0 \leqslant \rho \leqslant 1$,

$$\kappa_t(\rho, q) = \begin{cases} 1 - H_{q^t}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^t}, \\ 0 & 1 - \frac{1}{q^t} \leqslant \rho \leqslant 1. \end{cases}$$

We know that it is true for $t = 1$(Cohen and Frankl 1985)

A step towards the calculation of $\kappa_t(\rho, q)$

Conjecture

We conjecture that the lower bound meets the true value of $\kappa_t(\rho, q)$. That is, For any $0 \leqslant \rho \leqslant 1$,

$$\kappa_t(\rho, q) = \begin{cases} 1 - H_{q^t}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^t}, \\ 0 & 1 - \frac{1}{q^t} \leqslant \rho \leqslant 1. \end{cases}$$

We know that it is true for $t = 1$(Cohen and Frankl 1985)

General codes

We recall that by the geometric definition, R_t is defined for any code - $R_t(C)$ is the covering radius of $C^t \subseteq \mathbb{F}_q^{t \times n}$ with respect to $d^{(t)}$.

A step towards the calculation of $\kappa_t(\rho, q)$

Conjecture

We conjecture that the lower bound meets the true value of $\kappa_t(\rho, q)$. That is, For any $0 \leqslant \rho \leqslant 1$,

$$\kappa_t(\rho, q) = \begin{cases} 1 - H_{q^t}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^t}, \\ 0 & 1 - \frac{1}{q^t} \leqslant \rho \leqslant 1. \end{cases}$$

We know that it is true for $t = 1$(Cohen and Frankl 1985)

General codes

We recall that by the geometric definition, R_t is defined for any code - $R_t(C)$ is the covering radius of $C^t \subseteq \mathbb{F}_q^{t \times n}$ with respect to $d^{(t)}$.

We define $\hat{k}_t(n, r, q)$ to be the minimal dimension of a **general** code of length n with t-covering radius at most r and $\hat{\kappa}_t(\rho, q) \triangleq \liminf_{n \to \infty} \frac{\hat{k}_t(n, \rho n, q)}{n}$.

A step towards the calculation of $\kappa_t(\rho, q)$

Conjecture

We conjecture that the lower bound meets the true value of $\kappa_t(\rho, q)$. That is, For any $0 \leqslant \rho \leqslant 1$,

$$\kappa_t(\rho, q) = \begin{cases} 1 - H_{q^t}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^t}, \\ 0 & 1 - \frac{1}{q^t} \leqslant \rho \leqslant 1. \end{cases}$$

We know that it is true for $t = 1$(Cohen and Frankl 1985)

General codes

We recall that by the geometric definition, R_t is defined for any code - $R_t(C)$ is the covering radius of $C^t \subseteq \mathbb{F}_q^{t \times n}$ with respect to $d^{(t)}$.

We define $\hat{k}_t(n, r, q)$ to be the minimal dimension of a **general** code of length n with t-covering radius at most r and $\hat{\kappa}_t(\rho, q) \triangleq \liminf_{n \to \infty} \frac{\hat{k}_t(n, \rho n, q)}{n}$.

We know that for any ρ and q, $\hat{\kappa}_1(\rho, q) = \kappa_1(\rho, q)$.

A step towards the calculation of $\kappa_t(\rho, q)$

Theorem

1. For $t = 2$ and any $q \in \mathbb{N}$ and $0 \leqslant \rho \leqslant 1$,

$$\hat{\kappa}_2(\rho, q) = \begin{cases} 1 - H_{q^2}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^2}, \\ 0 & 1 - \frac{1}{q^2} \leqslant \rho \leqslant 1. \end{cases}$$

A step towards the calculation of $\kappa_t(\rho, q)$

Theorem

1. For $t = 2$ and any $q \in \mathbb{N}$ and $0 \leqslant \rho \leqslant 1$,

$$\hat{\kappa}_2(\rho, q) = \begin{cases} 1 - H_{q^2}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^2}, \\ 0 & 1 - \frac{1}{q^2} \leqslant \rho \leqslant 1. \end{cases}$$

2. In fact - if $\alpha_n(\varepsilon)$ fraction of codes with $R_2(C) \leqslant \rho n$ among codes of length n and size $\lfloor q^{n(\hat{\kappa}_2(\rho,q)+\varepsilon)} \rfloor$, then $\alpha_n(\varepsilon) \to 1$ as $n \to \infty$.

A step towards the calculation of $\kappa_t(\rho, q)$

Theorem

1. For $t = 2$ and any $q \in \mathbb{N}$ and $0 \leqslant \rho \leqslant 1$,

$$\hat{\kappa}_2(\rho, q) = \begin{cases} 1 - H_{q^2}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^2}, \\ 0 & 1 - \frac{1}{q^2} \leqslant \rho \leqslant 1. \end{cases}$$

2. In fact - if $\alpha_n(\varepsilon)$ fraction of codes with $R_2(C) \leqslant \rho n$ among codes of length n and size $\lfloor q^{n(\hat{\kappa_2}(\rho,q)+\varepsilon)} \rfloor$, then $\alpha_n(\varepsilon) \to 1$ as $n \to \infty$.

Proof idea - construct a sequence of code codes using the probabilistic method.

A step towards the calculation of $\kappa_t(\rho, q)$

Theorem

1. For $t = 2$ and any $q \in \mathbb{N}$ and $0 \leqslant \rho \leqslant 1$,

$$\hat{\kappa}_2(\rho, q) = \begin{cases} 1 - H_{q^2}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^2}, \\ 0 & 1 - \frac{1}{q^2} \leqslant \rho \leqslant 1. \end{cases}$$

2. In fact - if $\alpha_n(\varepsilon)$ fraction of codes with $R_2(C) \leqslant \rho n$ among codes of length n and size $\lfloor q^{n(\hat{\kappa}_2(\rho,q)+\varepsilon)} \rfloor$, then $\alpha_n(\varepsilon) \to 1$ as $n \to \infty$.

Proof idea - construct a sequence of code codes using the probabilistic method.

The Next Step

We know that for any ρ and q, $\hat{\kappa}_1(\rho, q) = \kappa_1(\rho, q)$.

A step towards the calculation of $\kappa_t(\rho, q)$

Theorem

1. For $t = 2$ and any $q \in \mathbb{N}$ and $0 \leqslant \rho \leqslant 1$,

$$\hat{\kappa}_2(\rho, q) = \begin{cases} 1 - H_{q^2}(\rho) & 0 \leqslant \rho < 1 - \frac{1}{q^2}, \\ 0 & 1 - \frac{1}{q^2} \leqslant \rho \leqslant 1. \end{cases}$$

2. In fact - if $\alpha_n(\varepsilon)$ fraction of codes with $R_2(C) \leqslant \rho n$ among codes of length n and size $\lfloor q^{n(\hat{\kappa}_2(\rho,q)+\varepsilon)} \rfloor$, then $\alpha_n(\varepsilon) \to 1$ as $n \to \infty$.

Proof idea - construct a sequence of code codes using the probabilistic method.

The Next Step

We know that for any ρ and q, $\hat{\kappa}_1(\rho, q) = \kappa_1(\rho, q)$. The next would be to prove that:

$$\hat{\kappa}_2(\rho, q) = \kappa_2(\rho, q).$$

The Generalized Covering Radii of Some Known Codes

Let's start with the Hamming code

Theorem

Let C be the $[n = \frac{q^m-1}{q-1}, k = \frac{q^m-1}{q-1} - m, 3]$ Hamming code over $\mathbb{F}_q$. Then for all $1 \leqslant t \leqslant m$,

$$R_t(C) = t.$$

Let's start with the Hamming code

Theorem

Let C be the $[n = \frac{q^m-1}{q-1}, k = \frac{q^m-1}{q-1} - m, 3]$ Hamming code over $\mathbb{F}_q$. Then for all $1 \leqslant t \leqslant m$,

$$R_t(C) = t.$$

Theorem

Let C be an $[n, k \geqslant 1, d \geqslant 3]$ linear code over $\mathbb{F}_q$. Then

$$R_1 < R_2 < \ldots < R_{n-k} = n - k,$$

if and only if C is the q-ary Hamming code.

Let's start with the Hamming code

Theorem

Let C be the $[n = \frac{q^m-1}{q-1}, k = \frac{q^m-1}{q-1} - m, 3]$ *Hamming code over* $\mathbb{F}_q$*. Then for all* $1 \leqslant t \leqslant m$,

$$R_t(C) = t.$$

Theorem

Let C be an $[n, k \geqslant 1, d \geqslant 3]$ *linear code over* $\mathbb{F}_q$*. Then*

$$R_1 < R_2 < \ldots < R_{n-k} = n - k,$$

if and only if C is the q-ary Hamming code.

Proof.

Since $R_{n-k} = n - k$, we must have $R_1 = 1$. But a linear code with parameters $[n, k, d \geqslant 3]$ with covering radius $R_1 = 1$ is 1-perfect and it must be the q-ary Hamming code. □

MDS codes have a very narrow hierarchy

Theorem (Gabidulin and Kløve, *ITW*, 1998)

Let C be a q-ary $[n, k, n-k+1]$ MDS code. Then its (regular) covering radius is

$$R_1(C) = n - k \qquad \text{or} \qquad R_1(C) = n - k - 1.$$

MDS codes have a very narrow hierarchy

Theorem (Gabidulin and Kløve, *ITW*, 1998)

Let C be a q-ary $[n, k, n-k+1]$ MDS code. Then its (regular) covering radius is

$$R_1(C) = n - k \qquad \text{or} \qquad R_1(C) = n - k - 1.$$

Corollary

Let C be a q-ary $[n, k, n-k+1]$ MDS code. Then for all $1 \leqslant t \leqslant n-k$,

$$R_t(C) = n - k \qquad \text{or} \qquad R_t(C) = n - k - 1.$$

Proof.

Immediate given the monotonicity of the generalized covering radii. □

Reed-Muller codes might be interesting

Pros:

- They have a wide range of parameters.
- They have many equivalent definitions.
- They are useful in communications (attain capacity of symmetric and erasure channels, and are connected to locally decodable codes, probabilistic proof systems) and cryptography (connected to the study of Boolean functions and sequence design).

— see **Reeves and Pfister**, *arXiv*, 2021, **Kudekar et al.**, *IEEE Trans. on Inform. Theory*, 2017, **Yekhanin**, *Now*, 2012, **Abbe et al.**, *IEEE Trans. on Inform. Theory*, 2015, **Kurosawa et al.**, *IEEE Trans. on Inform. Theory*, 2004, **Schmidt**, *IEEE Trans. on Inform. Theory*, 2007

Reed-Muller codes might be interesting

Pros:

- They have a wide range of parameters.
- They have many equivalent definitions.
- They are useful in communications (attain capacity of symmetric and erasure channels, and are connected to locally decodable codes, probabilistic proof systems) and cryptography (connected to the study of Boolean functions and sequence design).

— see **Reeves and Pfister**, *arXiv*, 2021, **Kudekar et al.**, *IEEE Trans. on Inform. Theory*, 2017, **Yekhanin**, *Now*, 2012, **Abbe et al.**, *IEEE Trans. on Inform. Theory*, 2015, **Kurosawa et al.**, *IEEE Trans. on Inform. Theory*, 2004, **Schmidt**, *IEEE Trans. on Inform. Theory*, 2007

Cons:

We don't even know their exact covering radius!

Here's a brief reminder

We shall find it convenient to use the following recursive construction of Reed-Muller codes:

Assume C_1 and C_2 are $[n, k_1]_q$ and $[n, k_2]_q$ codes, respectively. The $(u, u+v)$ construction uses C_1 and C_2 to produce a code

$$C = \{(\overline{u}, \overline{u}+\overline{v}) \mid \overline{u} \in C_1, \overline{v} \in C_2\}.$$

Here's a brief reminder

We shall find it convenient to use the following recursive construction of Reed-Muller codes:

Assume C_1 and C_2 are $[n, k_1]_q$ and $[n, k_2]_q$ codes, respectively. The $(u, u+v)$ construction uses C_1 and C_2 to produce a code

$$C = \{(\overline{u}, \overline{u} + \overline{v}) \,|\, \overline{u} \in C_1, \overline{v} \in C_2\}.$$

Reed-Muller Code Construction

- $\mathrm{RM}(0, m) \triangleq \left\{\overline{0}, \overline{1}\right\} \subseteq \mathbb{F}_2^{2^m}$

Here's a brief reminder

We shall find it convenient to use the following recursive construction of Reed-Muller codes:

Assume C_1 and C_2 are $[n, k_1]_q$ and $[n, k_2]_q$ codes, respectively. The $(u, u+v)$ construction uses C_1 and C_2 to produce a code

$$C = \{(\overline{u}, \overline{u} + \overline{v}) \,|\, \overline{u} \in C_1, \overline{v} \in C_2\}.$$

Reed-Muller Code Construction

- $\mathrm{RM}(0, m) \triangleq \{\overline{0}, \overline{1}\} \subseteq \mathbb{F}_2^{2^m}$
- $\mathrm{RM}(m, m) \triangleq \mathbb{F}_2^{2^m}$

Here's a brief reminder

We shall find it convenient to use the following recursive construction of Reed-Muller codes:

Assume C_1 and C_2 are $[n, k_1]_q$ and $[n, k_2]_q$ codes, respectively. The $(u, u+v)$ construction uses C_1 and C_2 to produce a code

$$C = \{(\overline{u}, \overline{u} + \overline{v}) \mid \overline{u} \in C_1, \overline{v} \in C_2\}.$$

Reed-Muller Code Construction

- $\mathrm{RM}(0, m) \triangleq \{\overline{0}, \overline{1}\} \subseteq \mathbb{F}_2^{2^m}$
- $\mathrm{RM}(m, m) \triangleq \mathbb{F}_2^{2^m}$
- For $1 \leqslant r \leqslant m-1$, we define $\mathrm{RM}(r, m)$ to be the code produced by the $(u, u+v)$ construction using $\mathrm{RM}(r, m-1)$ and $\mathrm{RM}(r-1, m-1)$.

Results

Exact values

We have calculated exact t-th covering radii of $\mathrm{RM}(r, m)$ (denoted by $R_t(r, m)$) in some extreme cases:

Results

Exact values

We have calculated exact t-th covering radii of $\mathrm{RM}(r, m)$ (denoted by $R_t(r, m)$) in some extreme cases:

$R_t(0, m)$	$2^m - \lceil 2^{m-t} \rceil$	Repetitions code
$R_t(m-2, m)$	$\min\{t, m\} + 1$	Extended Hamming code
$R_t(m-1, m)$	1	Parity code
$R_t(m, m)$	0	Full code

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. $\mathrm{RM}(r, m)$ where r is constant - the rate tends to 0 as $m \to \infty$.

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. $\mathrm{RM}(r, m)$ where r is constant - the rate tends to 0 as $m \to \infty$.
2. $\mathrm{RM}(m - s, m)$ where s is constant - the rate tends to 1 as $m \to \infty$.

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. RM(r, m) where r is constant - the rate tends to 0 as $m \to \infty$.
2. RM$(m - s, m)$ where s is constant - the rate tends to 1 as $m \to \infty$.
3. RM$(\alpha m, m)$ where α is constant:

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. RM(r, m) where r is constant - the rate tends to 0 as $m \to \infty$.
2. RM$(m - s, m)$ where s is constant - the rate tends to 1 as $m \to \infty$.
3. RM$(\alpha m, m)$ where α is constant:
    - $\alpha < \frac{1}{2}$ - the rate tends to 0 as $m \to \infty$

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. $\mathrm{RM}(r, m)$ where r is constant - the rate tends to 0 as $m \to \infty$.
2. $\mathrm{RM}(m - s, m)$ where s is constant - the rate tends to 1 as $m \to \infty$.
3. $\mathrm{RM}(\alpha m, m)$ where α is constant:
    - $\alpha < \frac{1}{2}$ - the rate tends to 0 as $m \to \infty$
    - $\alpha > \frac{1}{2}$ - the rate tends to 1 as $m \to \infty$.

Results (cont.)

Bounds

We provide lower and upper bounds on $R_t(r, m)$ in various scenarios, where $m \to \infty$:

1. RM(r, m) where r is constant - the rate tends to 0 as $m \to \infty$.
2. RM$(m - s, m)$ where s is constant - the rate tends to 1 as $m \to \infty$.
3. RM$(\alpha m, m)$ where α is constant:
   - $\alpha < \frac{1}{2}$ - the rate tends to 0 as $m \to \infty$
   - $\alpha > \frac{1}{2}$ - the rate tends to 1 as $m \to \infty$.
4. RM(r, m) where $r = \frac{1}{2}m + \Theta(\sqrt{m})$ - the rate can converge to any constant number in $[0, 1]$.

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!

with pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, $R_t(C)$ is the minimal of number of pre-computed combinations that has to be accessed in order to answer t linear queries.

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!

with pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, $R_t(C)$ is the minimal of number of pre-computed combinations that has to be accessed in order to answer t linear queries.

Observations

- Given t linear queries - we can always answer each one separately.

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!
with pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, $R_t(C)$ is the minimal of number of pre-computed combinations that has to be accessed in order to answer t linear queries.

Observations

- Given t linear queries - we can always answer each one separately. $\implies R_t(C) \leqslant t \cdot R_1(C)$ - and this linear relation holds when we do not gain from grouping queries.

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!

with pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, $R_t(C)$ is the minimal of number of pre-computed combinations that has to be accessed in order to answer t linear queries.

Observations

- Given t linear queries - we can always answer each one separately. $\implies R_t(C) \leqslant t \cdot R_1(C)$ - and this linear relation holds when we do not gain from grouping queries.
- Given a fixed number of database elemnts - m, the dimension of the corresponding code is $k = n - m$, where n is number of pre-computed linear combinations.

Results (cont.)

A small reminder before we look at the bounds

It all started from linear data querying!

with pre-computed linear combinations $\overline{h}_1 \cdot \overline{x}, \ldots, \overline{h}_n \cdot \overline{x}$, $R_t(C)$ is the minimal of number of pre-computed combinations that has to be accessed in order to answer t linear queries.

Observations

- Given t linear queries - we can always answer each one separately. $\implies R_t(C) \leqslant t \cdot R_1(C)$ - and this linear relation holds when we do not gain from grouping queries.
- Given a fixed number of database elemnts - m, the dimension of the corresponding code is $k = n - m$, where n is number of pre-computed linear combinations. $\implies$ Since we want to reduce the number of pre-computed linear combinations - codes with low rate are desirable!

A summary of the bounds

$R_t(r,m)$	$\leqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2^t-1}}{2^t}(1+\sqrt{2})^{r-1}2^{m/2} + O(m^{r-2})$	
	$\geqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2t(2^t-1)\ln 2}}{2^t\sqrt{r!}}m^{r/2}2^{m/2}(1+o(1))$	
$R_t(m-s,m)$	$\leqslant \frac{t}{(s-2)!}m^{s-2} + O(m^{s-3})$	
	$\geqslant \frac{t}{(s-1)!}m^{s-2} + O(m^{s-3}\log(m))$	
$R_t(\alpha m,m)$	$\leqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2^t-1}}{2^t(2+\sqrt{2})}2^{m\left(\frac{1}{2}+\alpha\log_2(1+\sqrt{2})\right)}(1+o(1))$	$0<\alpha<1-\frac{1}{\sqrt{2}}$
	$\leqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2^t-1}}{2^t}\cdot\frac{1}{\sqrt{8m\alpha(1-\alpha)}}\cdot 2^{mH_2(\alpha)}$	$1-\frac{1}{\sqrt{2}}\leqslant\alpha<\frac{1}{2}$
	$\leqslant t\cdot 4^{H_2(\alpha)}\cdot 2^{mH_2(\alpha)}\cdot(1+o(1))$	$\frac{1}{2}<\alpha<1$
	$\geqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2t(2^t-1)\ln 2}}{2^t}\cdot 2^{\frac{m}{2}(1+H_2(\alpha))}\cdot(1+o(1))$	$0<\alpha<\frac{1}{2}$
	$\geqslant t\cdot\sqrt{\frac{1-\alpha}{8(\alpha m)^3}}\cdot 2^{mH_2(\alpha)}\cdot(1+o(1))$	$\frac{1}{2}<\alpha<1$
$R_t(r,m)$	$\leqslant \left(1-\frac{1}{2^t}\right)2^m - \frac{\sqrt{2^t-1}}{2^t}\frac{2^m}{\sqrt{\frac{1}{2}m\pi}}e^{-\frac{(m-2r)^2}{2m}}(1+o(1))$	$\sum_{i=0}^{r}\binom{m}{i}=\kappa 2^m$
	$\geqslant H_{2^t}^{-1}(1-\kappa)2^m(1+o(1))$	

Proofs Ideas - Lower Bounds

Algebraic definition of the generalized covering radius

Let C be an $[n, k]_q$ linear code. Assume $G \in \mathbb{F}_q^{k \times n}$ is a generator matrix for C. Let C_t be the linear code over $\mathbb{F}_{q^t}$ generated by the same matrix G. That is, $C_t = \left\{ \overline{u}G \ : \ \overline{u} \in \mathbb{F}_{q^t}^k \right\}$. Then $R_t(C) = R_1(C_t)$.

Proofs Ideas - Lower Bounds

Algebraic definition of the generalized covering radius

Let C be an $[n,k]_q$ linear code. Assume $G \in \mathbb{F}_q^{k\times n}$ is a generator matrix for C. Let C_t be the linear code over $\mathbb{F}_{q^t}$ generated by the same matrix G. That is, $C_t = \left\{\overline{u}G \ : \ \overline{u} \in \mathbb{F}_{q^t}^k\right\}$. Then $R_t(C) = R_1(C_t)$.

Lemma - ball-covering argument

For an $[n,k]_q$ code C and

$$\log_q\left(V_{q,n,R_1(C)}\right) \geqslant n-k,$$

where $V_{q,n,r}$ denotes the volume of the Hamming ball of radius r in $\mathbb{F}_q^n$.

Proofs Ideas - Lower Bounds

Algebraic definition of the generalized covering radius

Let C be an $[n,k]_q$ linear code. Assume $G \in \mathbb{F}_q^{k\times n}$ is a generator matrix for C. Let C_t be the linear code over $\mathbb{F}_{q^t}$ generated by the same matrix G. That is, $C_t = \left\{\overline{u}G \ : \ \overline{u} \in \mathbb{F}_{q^t}^k\right\}$. Then $R_t(C) = R_1(C_t)$.

Lemma - ball-covering argument

For an $[n,k]_q$ code C and

$$\log_q\left(V_{q,n,R_1(C)}\right) \geqslant n-k,$$

where $V_{q,n,r}$ denotes the volume of the Hamming ball of radius r in $\mathbb{F}_q^n$.

Corollary

Applying on the ball-covering argument on C_t:

$$\log_{q^t}\left(V_{q^t,n,R_t(C)}\right) = \log_{q^t}\left(V_{q^t,n,R_1(C_t)}\right) \geqslant n-k.$$

Proofs Ideas - Upper Bounds

Lemma

if a code C is produced using the $(u, u+v)$ construction with C_1 and C_2, then

$$R_t(C) \leqslant R_t(C_1) + R_t(C_2).$$

Proofs Ideas - Upper Bounds

Lemma

if a code C is produced using the $(u, u+v)$ construction with C_1 and C_2, then

$$R_t(C) \leqslant R_t(C_1) + R_t(C_2).$$

Corollary

Since Reed-Muller codes are obtained from the $(u, u+v)$ construction - for any $m \geqslant 2$ and $1 \leqslant r \leqslant m-1$

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

Proofs Ideas - Upper Bounds

Lemma

if a code C is produced using the $(u, u+v)$ construction with C_1 and C_2, then

$$R_t(C) \leqslant R_t(C_1) + R_t(C_2).$$

Corollary

Since Reed-Muller codes are obtained from the $(u, u+v)$ construction - for any $m \geqslant 2$ and $1 \leqslant r \leqslant m-1$

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

Strategy

We upper bound the extreme cases and recursively apply the $(u, u+v)$ bounds. By careful analysis, we obtain our bounds in the considered scenarios.

Let us go back to our motivating scenario

Assume a server stores linear combinations of the database items according to the columns of a parity-check matrix H of $\mathrm{RM}(r, m)$.

Let us go back to our motivating scenario

Assume a server stores linear combinations of the database items according to the columns of a parity-check matrix H of $\mathrm{RM}(r, m)$.

After translating the problem to a coding theoretic one:

Goal

Given $t \times 2^m$ binary matrix in $\mathbb{F}_2^{t \times 2^m}$, how can we find $t \times 2^m$ codeword in $(\mathrm{RM}(r, m))^t$ which is a "small" $d^{(t)}$ distance away from the given matrix?

Let us go back to our motivating scenario

Assume a server stores linear combinations of the database items according to the columns of a parity-check matrix H of $\mathrm{RM}(r, m)$.

After translating the problem to a coding theoretic one:

Goal

Given $t \times 2^m$ binary matrix in $\mathbb{F}_2^{t \times 2^m}$, how can we find $t \times 2^m$ codeword in $(\mathrm{RM}(r, m))^t$ which is a "small" $d^{(t)}$ distance away from the given matrix?

The optimal solution requires $R_t(r, m)$ distance.

Let us go back to our motivating scenario

Assume a server stores linear combinations of the database items according to the columns of a parity-check matrix H of $\mathrm{RM}(r, m)$.

After translating the problem to a coding theoretic one:

Goal

Given $t \times 2^m$ binary matrix in $\mathbb{F}_2^{t \times 2^m}$, how can we find $t \times 2^m$ codeword in $(\mathrm{RM}(r, m))^t$ which is a "small" $d^{(t)}$ distance away from the given matrix?

The optimal solution requires $R_t(r, m)$ distance.

Question

Can we find a solution that requires a distance that is no more than the upper bounds we presented on $R_t(r, m)$?

Yes, we can!

Main Idea

- Our bounds are based on subadditivity:

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

Yes, we can!

Main Idea

- Our bounds are based on subadditivity:

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

- When receiving a $t \times 2^m$ matrix in the space, recursively find nearby codeword using $(\mathrm{RM}(r, m-1))^t$, and then use $(\mathrm{RM}(r-1, m-1))^t$ for the right half.

Yes, we can!

Main Idea

- Our bounds are based on subadditivity:

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

- When receiving a $t \times 2^m$ matrix in the space, recursively find nearby codeword using $(\mathrm{RM}(r, m-1))^t$, and then use $(\mathrm{RM}(r-1, m-1))^t$ for the right half.
- The recursion is grounded in two base cases:
  1. $\mathrm{RM}(m, m)$ is the entire space, and hence the nearest codeword are the received matrix.
  2. $(\mathrm{RM}(1, m))^t$ is small enough to use a brute-force search to find the best codeword.

Yes, we can!

Main Idea

- Our bounds are based on subadditivity:

$$R_t(r, m) \leqslant R_t(r-1, m-1) + R_t(r, m-1).$$

- When receiving a $t \times 2^m$ matrix in the space, recursively find nearby codeword using $(\mathrm{RM}(r, m-1))^t$, and then use $(\mathrm{RM}(r-1, m-1))^t$ for the right half.
- The recursion is grounded in two base cases:
  1. $\mathrm{RM}(m, m)$ is the entire space, and hence the nearest codeword are the received matrix.
  2. $(\mathrm{RM}(1, m))^t$ is small enough to use a brute-force search to find the best codeword.
- The time complexity: $O(t2^t(2^{t+1})^{m+1}(2^{t+1}-1)^{-r} + tm2^m)$.

The algorithm is simple

Algorithm 1: A t-covering algorithm for $\mathrm{RM}(r, m)$ with radius $U_t(r, m)$

Function `recursive`($\mathbf{v}$, r)

Input : $\mathbf{v} \in \mathbb{F}_2^{t \times 2^m}$, $r \in \mathbb{N}$, $1 \leqslant r \leqslant m$

`// Check edge cases`

if $r = m$ **then** **return** $\mathbf{v}$

if $r = 1$ **then** **return** $\mathrm{argmin}_{\mathbf{c} \in \mathrm{RM}(1,m)^t} d^{(t)}(\mathbf{v}, \mathbf{c})$

`// Use the` $(u, u + v)$ `recursion`

Let $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_2^{t \times 2^{m-1}}$ s.t. $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2)$

$\mathbf{c}_1 \leftarrow \texttt{recursive}(\mathbf{v}_1, r)$

$\mathbf{c}_2 \leftarrow \texttt{recursive}(\mathbf{v}_2 - \mathbf{c}_1, r - 1)$

return $(\mathbf{c}_1, \mathbf{c}_1 + \mathbf{c}_2)$

Open Questions and Future Work

The generalized covering radius of linear codes is a completely new topic and many interesting directions for future research remain. For example:

Open Questions and Future Work

The generalized covering radius of linear codes is a completely new topic and many interesting directions for future research remain. For example:

- Computing the generalized covering radius for known families of codes.

Open Questions and Future Work

The generalized covering radius of linear codes is a completely new topic and many interesting directions for future research remain. For example:

- Computing the generalized covering radius for known families of codes.
- Improving the asymptotic bounds, and generalizing them.

Open Questions and Future Work

The generalized covering radius of linear codes is a completely new topic and many interesting directions for future research remain. For example:

- Computing the generalized covering radius for known families of codes.
- Improving the asymptotic bounds, and generalizing them.
- Constructing codes with a given t-covering radius.

Thank you for your attention!