Exponential Lower Bound for 2-Query LDCs via a Quantum Argument

Ronald de Wolf

CWI Amsterdam

STOC 03, joint with Iordanis Kerenidis (LRI Paris)

Error-correcting codes

Error-correcting codes

- Encoding $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$

Error-correcting codes

- Encoding $C : \{0,1\}^n \rightarrow \{0,1\}^m$, $m \geq n$
- Even if $C(x)$ is corrupted in δm positions, we can still recover the whole x

Error-correcting codes

- Encoding $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$
- Even if $C(x)$ is corrupted in δm positions, we can still recover the whole x
- We can achieve this with $m = O(n)$, linear-time encoding and decoding.

Error-correcting codes

- Encoding $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$
- Even if $C(x)$ is corrupted in δm positions, we can still recover the whole x
- We can achieve this with $m = O(n)$, linear-time encoding and decoding.
- $O(1)$ time per bit!

Error-correcting codes

- Encoding $C : \{0,1\}^n \to \{0,1\}^m$, $m \geq n$
- Even if $C(x)$ is corrupted in δm positions, we can still recover the whole x
- We can achieve this with $m = O(n)$, linear-time encoding and decoding.
- $O(1)$ time per bit!
- Disadvantage: if you only want one bit x_i, you still need to decode the whole $C(x)$

Locally decodable codes (LDCs)

Locally decodable codes (LDCs)

- Recover x_i with high probability,
  looking only at a few positions in the codeword

Locally decodable codes (LDCs)

- Recover x_i with high probability,
  looking only at a few positions in the codeword
- $C : \{0,1\}^n \to \{0,1\}^m$ is a (q, δ, ε)-*locally decodable code*

Locally decodable codes (LDCs)

- Recover x_i with high probability,
  looking only at a few positions in the codeword

- $C : \{0,1\}^n \to \{0,1\}^m$ is a (q, δ, ε)-*locally decodable code*

  if there exists a randomized decoder A
  such that for every $y \in \{0,1\}^m$ and $i \in [n]$

Locally decodable codes (LDCs)

- Recover x_i with high probability,
  looking only at a few positions in the codeword

- $C : \{0,1\}^n \to \{0,1\}^m$ is a (q, δ, ε)-*locally decodable code*

  if there exists a randomized decoder A
  such that for every $y \in \{0,1\}^m$ and $i \in [n]$

  1. $A^y(i)$ makes $\leq q$ queries to bits of y (non-adaptively)

Locally decodable codes (LDCs)

- Recover x_i with high probability,
  looking only at a few positions in the codeword

- $C : \{0,1\}^n \to \{0,1\}^m$ is a (q, δ, ε)-*locally decodable code*

  if there exists a randomized decoder A
  such that for every $y \in \{0,1\}^m$ and $i \in [n]$

  1. $A^y(i)$ makes $\leq q$ queries to bits of y (non-adaptively)

  2. $d(y, C(x)) \leq \delta m \Rightarrow \Pr[A^y(i) = x_i] \geq 1/2 + \varepsilon$

Example: Hadamard code

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$
- Decoding x_i

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$
  2. query j and $j \oplus e_i$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$
  2. query j and $j \oplus e_i$
  3. output $y_j \oplus y_{j \oplus e_i}$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$
  2. query j and $j \oplus e_i$
  3. output $y_j \oplus y_{j \oplus e_i}$

- Works perfectly if $y = C(x)$ (no noise)

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$
  2. query j and $j \oplus e_i$
  3. output $y_j \oplus y_{j \oplus e_i}$

- Works perfectly if $y = C(x)$ (no noise)
- δ-corruption hits $C(x)_j$ or $C(x)_{j \oplus e_i}$ with probability $\leq 2\delta$

Example: Hadamard code

- Define $C(x)_j = j \cdot x \bmod 2$ for all $j \in \{0,1\}^n$, so $m = 2^n$
- Example: $C(11) = 0110$

- Decoding x_i
  1. pick random $j \in \{0,1\}^n$
  2. query j and $j \oplus e_i$
  3. output $y_j \oplus y_{j \oplus e_i}$

- Works perfectly if $y = C(x)$ (no noise)
- δ-corruption hits $C(x)_j$ or $C(x)_{j \oplus e_i}$ with probability $\leq 2\delta$

$$\Pr[A^y(i) = x_i] \geq 1 - 2\delta$$

What was known about LDCs

What was known about LDCs

Main question: tradeoff between q and m

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

$$\begin{aligned} q = m & \quad \Rightarrow \quad m \leq O(n) \ \text{(standard ECC)} \\ q = (\log n)^2 & \quad \Rightarrow \quad m \leq poly(n) \ \text{(BFLS 92)} \\ \text{constant } q & \quad \Rightarrow \quad m \leq 2^{n^{c(q)}} \ \text{(from PIR)} \end{aligned}$$

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

$$\begin{aligned} q = m & \quad \Rightarrow \quad m \leq O(n) \ \ (\text{standard ECC}) \\ q = (\log n)^2 & \quad \Rightarrow \quad m \leq poly(n) \ (\text{BFLS 92}) \\ \text{constant } q & \quad \Rightarrow \quad m \leq 2^{n^{c(q)}} \ (\text{from PIR}) \end{aligned}$$

- Lower bounds:

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

$$\begin{aligned} q = m & \Rightarrow m \leq O(n) \text{ (standard ECC)} \\ q = (\log n)^2 & \Rightarrow m \leq poly(n) \text{ (BFLS 92)} \\ \text{constant } q & \Rightarrow m \leq 2^{n^{c(q)}} \text{ (from PIR)} \end{aligned}$$

- Lower bounds:

$$\begin{aligned} q = 1 & \Rightarrow \text{LDCs don't exist (KT 00)} \\ q > 1 & \Rightarrow m \geq n^{1+\frac{1}{q-1}} \text{ (KT 00)} \\ q = 2, \text{linear } C & \Rightarrow m \geq 2^{\Omega(n)}, c = \delta\varepsilon/8 \text{ (GKST 02)} \end{aligned}$$

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

$$\begin{aligned} q = m & \Rightarrow m \leq O(n) \text{ (standard ECC)} \\ q = (\log n)^2 & \Rightarrow m \leq poly(n) \text{ (BFLS 92)} \\ \text{constant } q & \Rightarrow m \leq 2^{n^{c(q)}} \text{ (from PIR)} \end{aligned}$$

- Lower bounds:

$$\begin{aligned} q = 1 & \Rightarrow \text{LDCs don}'\text{t exist (KT 00)} \\ q > 1 & \Rightarrow m \geq n^{1+\frac{1}{q-1}} \text{ (KT 00)} \\ q = 2, \text{linear } C & \Rightarrow m \geq 2^{\Omega(n)}, c = \delta\varepsilon/8 \text{ (GKST 02)} \end{aligned}$$

- Our result:

What was known about LDCs

Main question: tradeoff between q and m

- Upper bounds:

$$q = m \quad \Rightarrow \quad m \leq O(n) \ \text{(standard ECC)}$$
$$q = (\log n)^2 \quad \Rightarrow \quad m \leq poly(n) \ \text{(BFLS 92)}$$
$$\text{constant } q \quad \Rightarrow \quad m \leq 2^{n^{c(q)}} \ \text{(from PIR)}$$

- Lower bounds:

$$q = 1 \quad \Rightarrow \quad \text{LDCs don't exist (KT 00)}$$
$$q > 1 \quad \Rightarrow \quad m \geq n^{1+\frac{1}{q-1}} \ \text{(KT 00)}$$
$$q = 2, \text{linear } C \quad \Rightarrow \quad m \geq 2^{\Omega(n)}, c = \delta\varepsilon/8 \ \text{(GKST 02)}$$

- Our result:

$$q = 2 \quad \Rightarrow \quad m \geq 2^{\Omega(n)} \text{ for all LDCs}$$

Our proof uses quantum!

Our proof uses quantum!

- Step 1:

  2-query LDCs can be decoded with 1 quantum query:

Our proof uses quantum!

- Step 1:

  2-query LDCs can be decoded with 1 quantum query:
  $(2, \delta, \varepsilon)$-LDC is $(1, \delta, 4\varepsilon/7)$-LQDC

Our proof uses quantum!

- Step 1:

  2-query LDCs can be decoded with 1 quantum query:
  $(2, \delta, \varepsilon)$-LDC is $(1, \delta, 4\varepsilon/7)$-LQDC (example: Hadamard)

Our proof uses quantum!

- Step 1:

  2-query LDCs can be decoded with 1 quantum query: $(2, \delta, \varepsilon)$-LDC is $(1, \delta, 4\varepsilon/7)$-LQDC (example: Hadamard)

- Step 2:

  $(1, \delta, \varepsilon)$-LQDC needs length $m \geq 2^{\Omega(n)}$, because it implies a $\log m$-qubit random access code for x

Locally Quantum-Decodable Codes

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

$$U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$$

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

  1. Query superposition of $j = 1$ and $j = 2$:

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

  1. Query superposition of $j = 1$ and $j = 2$:

     $$(-1)^{y_1} |1\rangle + (-1)^{y_2} |2\rangle$$

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

  1. Query superposition of $j = 1$ and $j = 2$:

$$(-1)^{y_1} |1\rangle + (-1)^{y_2} |2\rangle = (-1)^{y_1} \left(|1\rangle + (-1)^{y_1 \oplus y_2} |2\rangle \right)$$

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j} |j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

  1. Query superposition of $j = 1$ and $j = 2$:

  $$(-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle = (-1)^{y_1}\left(|1\rangle + (-1)^{y_1 \oplus y_2}|2\rangle\right)$$

  2. Measuring in Hadamard basis gives $y_1 \oplus y_2$

Locally Quantum-Decodable Codes

- Code is classical, but the q-query decoder is quantum
- Can query m-bit string y in superposition:

  $U_y : |j\rangle \rightarrow (-1)^{y_j}|j\rangle$
- Example: computing $y_1 \oplus y_2$ with 1 query:

  1. Query superposition of $j = 1$ and $j = 2$:

     $$(-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle = (-1)^{y_1}\left(|1\rangle + (-1)^{y_1 \oplus y_2}|2\rangle\right)$$

  2. Measuring in Hadamard basis gives $y_1 \oplus y_2$
- Hence Hadamard code is a 1-query LQDC!

Step 1: From 2-LDC to 1-LQDC

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with
1 quantum query and success probability exactly $11/14$:

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query and success probability exactly $11/14$:

1. Query $|\phi\rangle = |0\rangle + (-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle$

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query and success probability exactly $11/14$:

1. Query $|\phi\rangle = |0\rangle + (-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle$
2. Measure in 4-element basis
   $|\psi_{b_1 b_2}\rangle = |0\rangle + (-1)^{b_1}|1\rangle + (-1)^{b_2}|2\rangle + (-1)^{b_1+b_2}|3\rangle$
3. $\Pr[b_1 b_2 = y_1 y_2] = |\langle\phi|\psi_{y_1 y_2}\rangle|^2 = 3/4$

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query and success probability exactly $11/14$:

1. Query $|\phi\rangle = |0\rangle + (-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle$
2. Measure in 4-element basis
   $|\psi_{b_1 b_2}\rangle = |0\rangle + (-1)^{b_1}|1\rangle + (-1)^{b_2}|2\rangle + (-1)^{b_1+b_2}|3\rangle$
3. $\Pr[b_1 b_2 = y_1 y_2] = |\langle\phi|\psi_{y_1 y_2}\rangle|^2 = 3/4$
4. $b_1 b_2 +$ truth table of $f \Rightarrow$ output

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query and success probability exactly $11/14$:

1. Query $|\phi\rangle = |0\rangle + (-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle$
2. Measure in 4-element basis
   $|\psi_{b_1 b_2}\rangle = |0\rangle + (-1)^{b_1}|1\rangle + (-1)^{b_2}|2\rangle + (-1)^{b_1+b_2}|3\rangle$
3. $\Pr[b_1 b_2 = y_1 y_2] = |\langle\phi|\psi_{y_1 y_2}\rangle|^2 = 3/4$
4. $b_1 b_2 +$ truth table of $f \Rightarrow$ output

For classical 2-query decoder with success probability $p = 1/2 + \varepsilon$, 1 quantum query gives

Step 1: From 2-LDC to 1-LQDC

We can compute any Boolean function $f(y_1, y_2)$ with 1 quantum query and success probability exactly $11/14$:

1. Query $|\phi\rangle = |0\rangle + (-1)^{y_1}|1\rangle + (-1)^{y_2}|2\rangle$
2. Measure in 4-element basis
   $|\psi_{b_1 b_2}\rangle = |0\rangle + (-1)^{b_1}|1\rangle + (-1)^{b_2}|2\rangle + (-1)^{b_1+b_2}|3\rangle$
3. $\Pr[b_1 b_2 = y_1 y_2] = |\langle\phi|\psi_{y_1 y_2}\rangle|^2 = 3/4$
4. $b_1 b_2 +$ truth table of $f \Rightarrow$ output

For classical 2-query decoder with success probability $p = 1/2 + \varepsilon$, 1 quantum query gives

$$\frac{11}{14}p + \frac{3}{14}(1-p) = \frac{1}{2} + \frac{4\varepsilon}{7}$$

Step 2: Lower bound for 1-LQDC

Step 2: Lower bound for 1-LQDC

- Quantum decoder predicts x_i by measuring query state $\sum_{j=1}^{m}(-1)^{C(x)_j}\alpha_{ij}|j\rangle$

Step 2: Lower bound for 1-LQDC

- Quantum decoder predicts x_i by measuring query state $\sum_{j=1}^{m}(-1)^{C(x)_j}\alpha_{ij}|j\rangle$
- This can tolerate up to δm phase-errors

Step 2: Lower bound for 1-LQDC

- Quantum decoder predicts x_i by measuring query state $\sum_{j=1}^{m}(-1)^{C(x)_j}\alpha_{ij}|j\rangle$
- This can tolerate up to δm phase-errors
- Set of small amplitudes $A_i = \{j : \alpha_{ij} \leq 1/\sqrt{\delta m}\}$ misses at most δm indices

Step 2: Lower bound for 1-LQDC

- Quantum decoder predicts x_i by measuring

  query state $\sum_{j=1}^{m}(-1)^{C(x)_j}\alpha_{ij}|j\rangle$

- This can tolerate up to δm phase-errors
- Set of small amplitudes $A_i = \{j : \alpha_{ij} \leq 1/\sqrt{\delta m}\}$ misses at most δm indices
- Given $|A_i(x)\rangle = \sum_{j\in A_i}(-1)^{C(x)_j}\alpha_{ij}|j\rangle$,

  we can predict x_i with good bias $\approx \varepsilon$

Step 2: get $|A_i(x)\rangle$ from uniform state

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:

  1. Measure $|U(x)\rangle$ with operators $M_i^* M_i$, $I - M_i^* M_i$, where $M_i = \sqrt{\delta m} \sum_{j \in A_i} \alpha_{ij} |j\rangle\langle j|$

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:

  1. Measure $|U(x)\rangle$ with operators $M_i^* M_i$, $I - M_i^* M_i$, where $M_i = \sqrt{\delta m} \sum_{j \in A_i} \alpha_{ij} |j\rangle\langle j|$
  2. With prob $\approx \delta$: $M_i : |U(x)\rangle \mapsto |A_i(x)\rangle$, then we can predict x_i with bias $\approx \varepsilon$

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:
  1. Measure $|U(x)\rangle$ with operators $M_i^* M_i$, $I - M_i^* M_i$, where $M_i = \sqrt{\delta m} \sum_{j \in A_i} \alpha_{ij} |j\rangle\langle j|$
  2. With prob $\approx \delta$: $M_i : |U(x)\rangle \mapsto |A_i(x)\rangle$, then we can predict x_i with bias $\approx \varepsilon$
  3. This gives x_i with prob $p \approx 1/2 + \delta\varepsilon$
- $|U(x)\rangle$ is a random access code for x!

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:
  1. Measure $|U(x)\rangle$ with operators $M_i^* M_i$, $I - M_i^* M_i$, where $M_i = \sqrt{\delta m} \sum_{j \in A_i} \alpha_{ij} |j\rangle\langle j|$
  2. With prob $\approx \delta$: $M_i : |U(x)\rangle \mapsto |A_i(x)\rangle$, then we can predict x_i with bias $\approx \varepsilon$
  3. This gives x_i with prob $p \approx 1/2 + \delta\varepsilon$
- $|U(x)\rangle$ is a random access code for x!

$$\underbrace{\log m}_{\#\text{qubits of } U(x)}$$

Step 2: get $|A_i(x)\rangle$ from uniform state

- Predict x_i from $|U(x)\rangle = \sum_{j=1}^{m} (-1)^{C(x)_j} |j\rangle$:
  1. Measure $|U(x)\rangle$ with operators $M_i^* M_i$, $I - M_i^* M_i$, where $M_i = \sqrt{\delta m} \sum_{j \in A_i} \alpha_{ij} |j\rangle\langle j|$
  2. With prob $\approx \delta$: $M_i : |U(x)\rangle \mapsto |A_i(x)\rangle$, then we can predict x_i with bias $\approx \varepsilon$
  3. This gives x_i with prob $p \approx 1/2 + \delta\varepsilon$
- $|U(x)\rangle$ is a random access code for x!

$$\underbrace{\log m}_{\#\text{qubits of } U(x)} \quad \geq \quad \underbrace{(1 - H(p))n}_{\text{RAC bound (Nayak 99)}}$$

Lower bound for $q > 2$ queries (sketch)

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples.

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$
- Generate a parity from each copy until you've completed a q-tuple $\in M_i$. Output its parity.

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$
- Generate a parity from each copy until you've completed a q-tuple $\in M_i$. Output its parity.
- By Birthday Paradox: $r \approx m^{1-\lfloor 2/q \rfloor}$ suffices

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$
- Generate a parity from each copy until you've completed a q-tuple $\in M_i$. Output its parity.
- By Birthday Paradox: $r \approx m^{1-\lfloor 2/q \rfloor}$ suffices
- RAC: $m^{1-\lfloor 2/q \rfloor} \cdot \log m = \Omega(n)$

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$
- Generate a parity from each copy until you've completed a q-tuple $\in M_i$. Output its parity.
- By Birthday Paradox: $r \approx m^{1-\lfloor 2/q \rfloor}$ suffices
- RAC: $m^{1-\lfloor 2/q \rfloor} \cdot \log m = \Omega(n) \Rightarrow n \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$

Lower bound for $q > 2$ queries (sketch)

- From Katz-Trevisan:
  Can assume that for each i there is a partition M_i of $[m]$ into q-tuples. For decoding x_i: pick a random element of M_i, query those q indices, and return the parity.

  Building a quantum random access code:

- Take r copies of the uniform state $|U(x)\rangle$
- Generate a parity from each copy until you've completed a q-tuple $\in M_i$. Output its parity.
- By Birthday Paradox: $r \approx m^{1-\lfloor 2/q \rfloor}$ suffices
- RAC: $m^{1-\lfloor 2/q \rfloor} \cdot \log m = \Omega(n) \Rightarrow n \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$
- Slightly improved by Woodruff

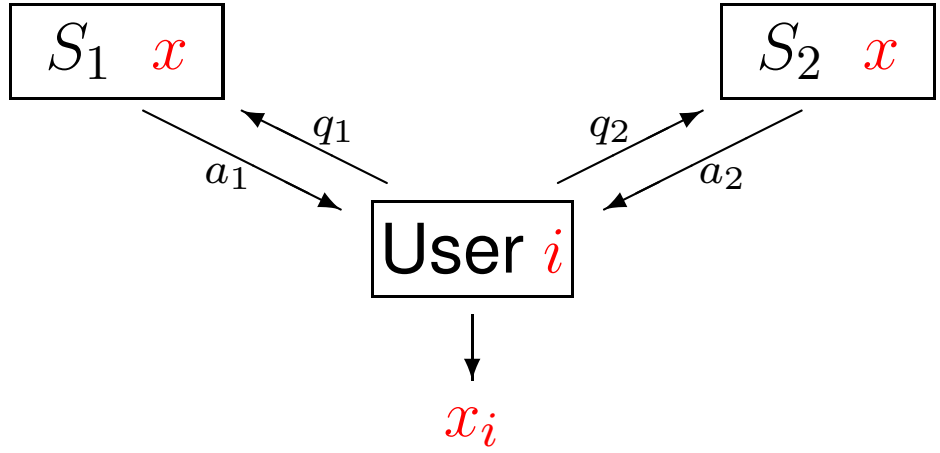
Private Information Retrieval

Private Information Retrieval

- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers

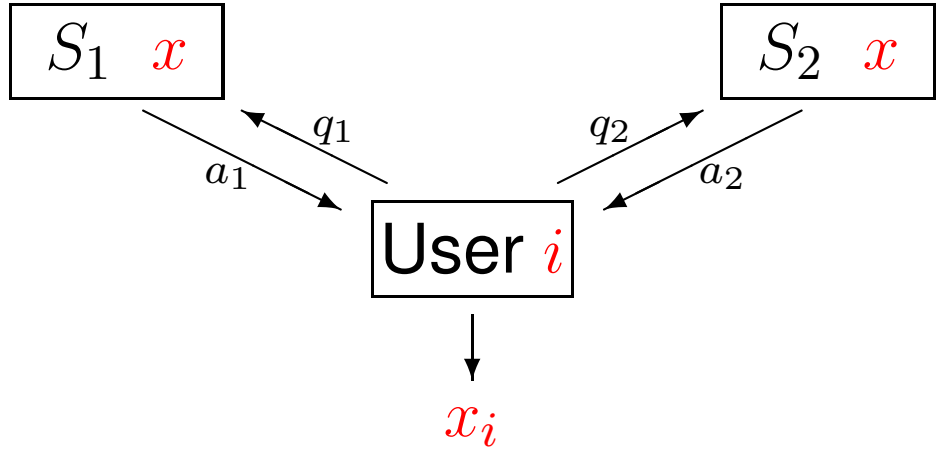
Private Information Retrieval

- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers



Private Information Retrieval

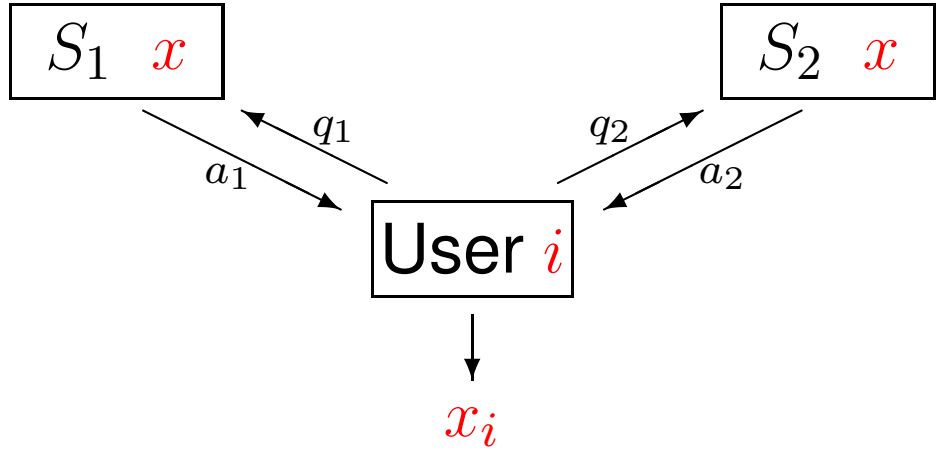
- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers



- Privacy: server learns nothing about i

Private Information Retrieval

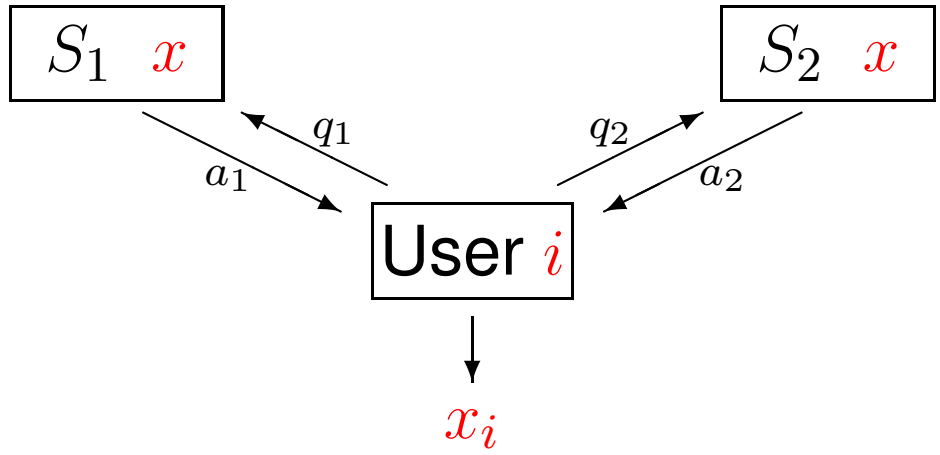
- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers



- Privacy: server learns nothing about i
- How much communication is needed?

Private Information Retrieval

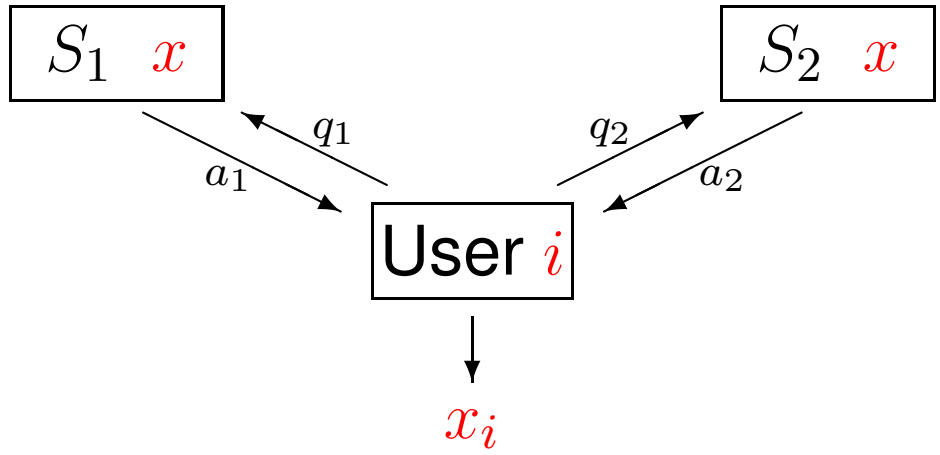
- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers



- Privacy: server learns nothing about i
- How much communication is needed?
  1-server PIR needs $\Omega(n)$ bits

Private Information Retrieval

- User retrieves x_i with probability $1/2 + \varepsilon$ from n-bit x, replicated over q non-communicating servers



- Privacy: server learns nothing about i
- How much communication is needed?
  - 1-server PIR needs $\Omega(n)$ bits
  - 2-server PIR with $O(n^{1/3})$ bits (CGKS)

Lower bound for classical binary PIR

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)
- $\Omega(n)$ lower bound for 1-server quantum PIR

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)
- $\Omega(n)$ lower bound for 1-server quantum PIR
  $\Rightarrow$
  $\Omega(n)$ lower bound for 2-server binary PIR

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)
- $\Omega(n)$ lower bound for 1-server quantum PIR
  $\Rightarrow$
  $\Omega(n)$ lower bound for 2-server binary PIR
- Previously known only for *linear* PIR (GKST)

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)
- $\Omega(n)$ lower bound for 1-server quantum PIR
  $\Rightarrow$
  $\Omega(n)$ lower bound for 2-server binary PIR
- Previously known only for *linear* PIR (GKST)
- Later, BFG found a classical proof if $\varepsilon = 1/2$

Lower bound for classical binary PIR

- Binary PIR: servers send back only 1 bit
- Can reduce 2 binary classical servers to 1 quantum server (treat servers as queries)
- $\Omega(n)$ lower bound for 1-server quantum PIR
  $\Rightarrow$
  $\Omega(n)$ lower bound for 2-server binary PIR
- Previously known only for *linear* PIR (GKST)

- Later, BFG found a classical proof if $\varepsilon = 1/2$
- Samorodnitsky replaces step 1 by linear algebra

Upper bound for quantum PIR

Upper bound for quantum PIR

- BIKR 02 give a binary 4-server PIR that uses $O(n^{3/10})$ communication and outputs XOR of the 4 bits

Upper bound for quantum PIR

- BIKR 02 give a binary 4-server PIR that uses $O(n^{3/10})$ communication and outputs XOR of the 4 bits
- Can do this with 2 quantum servers!

Upper bound for quantum PIR

- BIKR 02 give a binary 4-server PIR that uses $O(n^{3/10})$ communication and outputs XOR of the 4 bits
- Can do this with 2 quantum servers!
- This beats best known classical 2-server PIR, and beats the $\Omega(n^{1/3})$ lower bound for group-based bilinear PIR (Razborov-Yekhanin)!

Upper bound for quantum PIR

- BIKR 02 give a binary 4-server PIR that uses $O(n^{3/10})$ communication and outputs XOR of the 4 bits
- Can do this with 2 quantum servers!
- This beats best known classical 2-server PIR, and beats the $\Omega(n^{1/3})$ lower bound for group-based bilinear PIR (Razborov-Yekhanin)!

Servers	PIR complexity	QPIR complexity
$q = 1$	n	n
$q = 2$	$n^{1/3}$	$n^{3/10}$

Upper bound for quantum PIR

- BIKR 02 give a binary 4-server PIR that uses $O(n^{3/10})$ communication and outputs XOR of the 4 bits
- Can do this with 2 quantum servers!
- This beats best known classical 2-server PIR, and beats the $\Omega(n^{1/3})$ lower bound for group-based bilinear PIR (Razborov-Yekhanin)!

Servers	PIR complexity	QPIR complexity
$q = 1$	n	n
$q = 2$	$n^{1/3}$	$n^{3/10}$

- From Yekhanin's construction: $O(n^{\varepsilon})$ 2-server QPIRs

Summary

Summary

- Locally decodable codes:

Summary

- Locally decodable codes:

  1. Exponential bound for 2-query LDCs via quantum

Summary

- Locally decodable codes:

  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$

Summary

- Locally decodable codes:

  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$

- Private information retrieval:

Summary

- Locally decodable codes:
  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$
- Private information retrieval:
  1. $\Omega(n)$ lower bound for 2-server binary PIR

Summary

- Locally decodable codes:
  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$
- Private information retrieval:
  1. $\Omega(n)$ lower bound for 2-server binary PIR
  2. Upper bound $O(n^{3/10})$ for 2-server QPIR

Summary

- Locally decodable codes:
  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$
- Private information retrieval:
  1. $\Omega(n)$ lower bound for 2-server binary PIR
  2. Upper bound $O(n^{3/10})$ for 2-server QPIR
  3. Even $n^{o(1)}$ if $\exists\infty$ Mersenne primes (Yekhanin)

Summary

- Locally decodable codes:
  1. Exponential bound for 2-query LDCs via quantum
  2. q queries: polynomial bound $m \geq \left(\frac{n}{\log n}\right)^{1+\frac{1}{\lceil q/2 \rceil - 1}}$
- Private information retrieval:
  1. $\Omega(n)$ lower bound for 2-server binary PIR
  2. Upper bound $O(n^{3/10})$ for 2-server QPIR
  3. Even $n^{o(1)}$ if $\exists\infty$ Mersenne primes (Yekhanin)
- Future Work: optimal LDC bounds for $q > 2$?