Reducing the Number of Qubits in Quantum Information Set Decoding

Clémence Chevignard
Pierre-Alain Fouque
André Schrottenloher

Univ Rennes, Inria, CNRS, IRISA
Team CAPSULE

The random decoding problem

(n, k)-linear code over $\mathbb{F}_2$: dimension-k subspace of $\mathbb{F}_2^n$ specified as the kernel of a **parity-check matrix** $\boldsymbol{H} \in \mathbb{F}_2^{(n-k)\times n}$.

Random (syndrome) decoding problem

Given $\boldsymbol{H}$ sampled u.a.r., and $\boldsymbol{s} = \boldsymbol{He}$ where $\boldsymbol{e}$ is sampled u.a.r. and has weight w, find $\boldsymbol{e}$.

The level of security offered by code-based cryptosystems depends on the SDP generic complexity.

Prange's ISD algorithm

Let $S_{n,k}$ = all subsets of $\{0, \dots, n-1\}$ with $n-k$ elements. Write:

$$\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e} = (\boldsymbol{H}_0 \cdots \boldsymbol{H}_{n-1}) \begin{pmatrix} e_0 \\ \vdots \\ e_{n-1} \end{pmatrix} = e_0 \boldsymbol{H}_0 \oplus e_1 \boldsymbol{H}_1 \oplus \dots \oplus e_{n-1} \boldsymbol{H}_{n-1} \ .$$

Select a subset $I \in S_{n,k}$. Assume that all the ones in $\boldsymbol{e}$ fall in I. Then one has:

$$\boldsymbol{s} = \boldsymbol{H}\boldsymbol{e} = \bigoplus_{i \in I} e_i \boldsymbol{H}_i := \underbrace{\boldsymbol{H}_{\mathrm{I}}}_{n-k \times n-k} \boldsymbol{e}_I$$

$$\implies \boldsymbol{e}_I = \boldsymbol{H}_{\mathrm{I}}^{-1} \boldsymbol{s} \ .$$

- Pick a random $I \in S_{n,k}$
- Compute $\boldsymbol{H}_{\mathrm{I}}^{-1} \boldsymbol{s}$
- Until it has weight $w \implies$ get $\boldsymbol{e}_I$

Extend $\boldsymbol{e}_I$ by zeroes $\implies$ get $\boldsymbol{e}$

Prange's ISD (ctd.)

Probability to succeed on a random I:

$$p := \mathcal{O}\left(\frac{\binom{n-k}{w}}{\binom{n}{w}}\right)$$

Time complexity: $\mathcal{O}\left(\frac{1}{p}(n-k)^{\omega}\right)$.

Improved algorithms (Stern, Dumer, Lee-Brickell, MMT, BJMM ...)

- Less constraints on $I \implies$ less loop iterates
- More computations in the loop
- **Non-negligible memory**

Quantum ISD

[Ber10] use Grover's algorithm to search for I

$\mathcal{O}(1/\sqrt{p})$ iterations of:

- Sampling I u.a.r.
- Testing I (compute $\boldsymbol{H}_I{}^{-1}\boldsymbol{s}$, check weight) $\implies$ takes $\mathcal{O}(n^3)$ "bit operations"

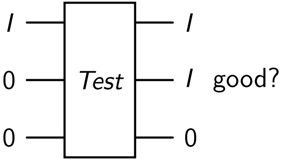
Improved algorithms [KT17], [Kir18]

- Same principles as classical algorithms
- Increase the space exponentially

Bernstein, "Grover vs. McEliece", PQCrypto 2010

Focusing on the test

- Test is a sequence of quantum operations described as a "circuit"
- Here it can actually be a **classical reversible** circuit

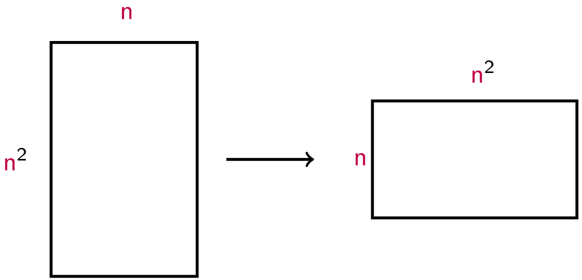


- Needs $\mathcal{O}(n^3)$ for Gaussian elimination
- Needs $\mathcal{O}(n^2)$ space to write $\boldsymbol{H}_I$

In this work

Initial: $\mathcal{O}(n^2)$ qubits and $\mathcal{O}(n^3)$ gates.

- Trade-off 1: $\mathcal{O}(n)$ qubits + $\mathcal{O}(n^3)$ gates
- Trade-off 2: $\mathcal{O}(n \log^2 n)$ qubits + $\mathcal{O}(n^3)$ gates incl. $\mathcal{O}(n^2 \log^2 n)$ nonlinear (Toffoli) gates



Quantum Prange with Wiedemann

Wiedemann's algorithm

Computing ${\boldsymbol{H}_\mathsf{I}}^{-1}\boldsymbol{s}$ only via matrix-vector products.

Assume $\boldsymbol{H}_\mathsf{I}$ invertible. Let $\boldsymbol{x} = {\boldsymbol{H}_\mathsf{I}}^{-1}\boldsymbol{s}$. Consider the space:

$$\{{\boldsymbol{H}_\mathsf{I}}^{i}\boldsymbol{s}, i \in \mathbb{N}\}$$

There is a minimal monic P such that: $P(\boldsymbol{H}_\mathsf{I})\boldsymbol{s} = 0$.

Let $Q(X) = (1 \oplus P(X))/X$. Then $\boldsymbol{x} = Q(\boldsymbol{H}_\mathsf{I})\boldsymbol{s}$. Verify that:

$$\boldsymbol{H}_\mathsf{I}\boldsymbol{x} = (\boldsymbol{H}_\mathsf{I}Q(\boldsymbol{H}_\mathsf{I}))\boldsymbol{s} = P(\boldsymbol{H}_\mathsf{I})\boldsymbol{s} \oplus \boldsymbol{s} = \boldsymbol{s} \ .$$

To reduce to a linear recurrence in $\mathbb{F}_2$, take a random $\boldsymbol{u}$ and project:

$$\{\boldsymbol{u}^T{\boldsymbol{H}_\mathsf{I}}^{i}\boldsymbol{s}, 0 \leq i \leq 2(n-k)\}$$

Wiedemann, "Solving sparse linear equations over finite fields". IEEE Trans. Inf. Theory 1986

Simplified algorithm*

Input: choice $I \in S_{n,k}$
Output: is I good?

Choose $\boldsymbol{u} = (1, 0, \dots, 0)$

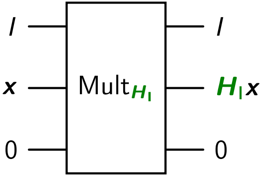
1. Compute the sequence $(\boldsymbol{u}^T \boldsymbol{H}_I{}^i \boldsymbol{s})_{0 \leq i \leq 2(n-k)}$
2. Compute the minimal polynomial $C(X)$ of the sequence
   Let $C'(X) = (C(X) \oplus 1)/X$
3. Let $\boldsymbol{y} = C'(\boldsymbol{H}_I)\boldsymbol{s}$
   If $\boldsymbol{H}_I \boldsymbol{y} = \boldsymbol{s}$, then set Success to True (False otherwise)
   Return (Success, $\boldsymbol{y}$)

Next: implement 1., 2. and 3. reversibly with $\mathcal{O}(n)$ space.

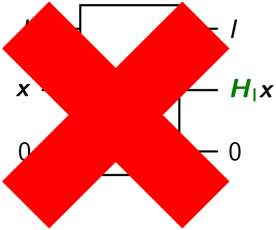
* Actually two iterates are required for constant probability of success.

Circuit Details

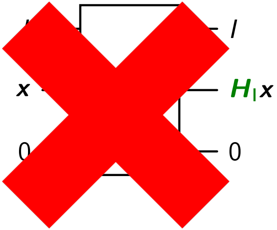
Step 0: the matrix-vector product

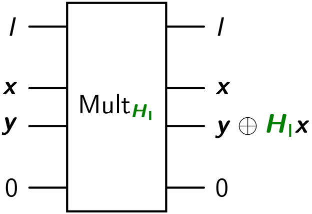


Step 0: the matrix-vector product



Step 0: the matrix-vector product





General strategy

We want to compute:

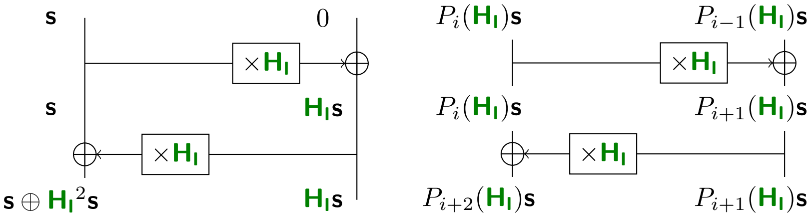
$$\boldsymbol{H}_{\mathsf{I}}\boldsymbol{x} = \boldsymbol{H}_{\mathsf{I}} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{\mathsf{n}-k} \end{pmatrix} = \boldsymbol{H} \underbrace{\begin{pmatrix} 0 \\ x_0 \\ 0 \\ 0 \\ \vdots \\ x_{\mathsf{n}-k} \\ 0 \end{pmatrix}}_{:=\boldsymbol{x}'}$$

1. construct $\boldsymbol{x}'$
2. compute $\boldsymbol{y} \leftarrow \boldsymbol{y} \oplus \boldsymbol{H}\boldsymbol{x}' \implies$ fixed linear circuit, $\boldsymbol{H}$ is built-in
3. erase $\boldsymbol{x}'$

- Our different trade-offs happen here.
- The cost depends on the representation of I.

Step 1: Evaluate the sequence

Compute the sequence $(\boldsymbol{u}^T \boldsymbol{H_I}^i \boldsymbol{s})_i$



- We can only evaluate the $P_i(\boldsymbol{H_I})\boldsymbol{s}$, not directly the powers.
- However, the $P_i(\boldsymbol{H_I})$ are a polynomial basis, so for each of the $\mathcal{O}(n)$ sequence bits:

$$\boldsymbol{u}^T \boldsymbol{H_I}^i \boldsymbol{s} = \text{linear combination of } \boldsymbol{u}^T P_j(\boldsymbol{H_I})\boldsymbol{s}$$

Step 2: Compute the minimal polynomial

With a **reversible implementation** of the Berlekamp-Massey algorithm, in $\mathcal{O}\big((n-k)^2\big)$ operations and $\mathcal{O}(n-k)$ space.

$\Longrightarrow$ non-dominating.

Step 3: Evaluate a polynomial

Given $C'(X)$, compute $C'(\boldsymbol{H}_{\mathsf{I}})\boldsymbol{s}$.

This is very similar to step 1:

$$C'(\boldsymbol{H}_{\mathsf{I}})\boldsymbol{s} = \text{ linear combination of } P_i(\boldsymbol{H}_{\mathsf{I}})\boldsymbol{s}$$

Cost of Steps 1 & 3 dominated by matrix-vector multiplications.

Conclusion

Results

Example: Classic McEliece L1: $n = 3488, k = 2720$

- **[PBP23]** 2^{22} qubits (4 millions), 2^{30} gates (2^{102} for Grover)
- **Space-optimized:** 18 098 qubits, $2^{39.3}$ gates ($\simeq 316n(n-k)^2$) ($2^{111.9}$ for Grover)
- **Toffoli-optimized:** 258 769 qubits, $2^{35.9}$ gates ($\simeq 24n(n-k)^2$), 2^{32} Toffoli (2^{104} for Grover)

Bonus: if the matrix $\boldsymbol{H}$ is structured (e.g., block-circulant), we can exploit that.

- Reduces the gate count for BIKE & HQC

Perriello, Barenghi, Pelosi, "Improving the efficiency of quantum circuits for information set decoding". ACM Transactions on Quantum Computing 2023

Conclusion

Before: qubit count in millions and gate count infeasible.

After: gate count infeasible (as expected!), but qubit count becomes closer to Shor's.

$\implies$ This circuit may be useful in other quantum cryptanalysis algorithms.

Paper: `eprint.iacr.org/2024/907`
Code: `gitlab.inria.fr/capsule/quantum-isd-less-qubits`

Thank you!