CLASSICAL AND QUANTUM ALGORITHMS FOR GENERIC SYNDROME DECODING PROBLEMS

and their application to the Lee weight

André Chailloux, Thomas Debris-Alazard and Simona Etinski

PQCrypto 2021

INVENTEURS DU MONDE NUMÉRIQUE

Université de Paris

SYNDROME DECODING PROBLEM

Syndrome Decoding Problem

An NP-complete problem.[1]

[1] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. “On the inherent intractability of certain coding problems (Corresp.)”. In: (1978), pp. 384–386. DOI: 10.1109/TIT.1978.1055873.

Syndrome Decoding Problem

An NP-complete problem.[1]

For conveniently chosen parameters, exponentially hard for the best known algorithms.

[1] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)". In: (1978), pp. 384–386. DOI: 10.1109/TIT.1978.1055873.

Syndrome Decoding Problem

An NP-complete problem.[1]

For conveniently chosen parameters, exponentially hard for the best known algorithms.

Used as basis of different cryptographic protocols.[2]

---

[1] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)". In: (1978), pp. 384–386. DOI: 10.1109/TIT.1978.1055873.

[2] Jacques Stern. "A New Identification Scheme Based on Syndrome Decoding". In: 1993, pp. 13–21. DOI: 10.1007/3-540-48329-2_2.

Syndrome Decoding Problem, $\mathrm{SD}(n, k, w)$

Input – A parity check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, a syndrome $\mathbf{s} \in \mathbb{F}_q^{n-k}$, a weight function $\mathrm{wt} : \mathbb{F}_q^n \to \mathbb{N}$, and a weight $w \in \mathbb{N}$.

Goal – Find the error $\mathbf{e} \in \mathbb{F}_q^n$ such that $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$ and $\mathrm{wt}(\mathbf{e}) = w$.

Weight function, $\mathrm{wt}(\cdot)$

Let $[q]$ represent the set $\{0, 1, \ldots, q-1\}$, and $d : [q] \times [q] \to \mathbb{N}$ be a distance function.

Weight function, $\mathrm{wt}(\cdot)$

Let $[q]$ represent the set $\{0, 1, \ldots, q-1\}$, and $d : [q] \times [q] \to \mathbb{N}$ be a distance function.

A weight function over an element, $\mathrm{wt} : \mathbb{F}_q \to \mathbb{N}$, and over a vector, $\mathrm{wt} : \mathbb{F}_q^n \to \mathbb{N}$, are then defined as:

$$\forall e_i \in \mathbb{F}_q, \quad \forall \mathbf{e} = (e_0, \ldots, e_{n-1}) \in \mathbb{F}_q^n, \quad \mathrm{wt}(\mathbf{e}) = \sum_i \mathrm{wt}(e_i) = \sum_i d(e_i, 0).$$

Common weight functions

Hamming distance, $d_H(\cdot)$

$$\forall e_i, e_j \in \mathbb{F}_q,$$

$$d_H(e_i, e_j) = \begin{cases} 0, & e_i = e_j \\ 1, & e_i \neq e_j \end{cases} .$$

Common weight functions

Hamming distance, $d_H(\cdot)$

$$\forall e_i, e_j \in \mathbb{F}_q,$$

$$d_H(e_i, e_j) = \begin{cases} 0, & e_i = e_j \\ 1, & e_i \neq e_j \end{cases} .$$

Hamming weight, $wt_H(\cdot)$

$$\forall \mathbf{e} = (e_1, \dots, e_n) \in \mathbb{F}_q^n,$$

$$wt_H(\mathbf{e}) = |i \in [n] : e_i \neq 0|.$$

COMMON WEIGHT FUNCTIONS

Lee distance, $d_L(\cdot)$

$$\forall e_i, e_j \in \mathbb{F}_q, \quad q \in \mathbb{P},$$

$$d_L(e_i, e_j) = \min(|e_i - e_j|, q - |e_i - e_j|).$$

Common weight functions

Lee distance, $d_L(\cdot)$

$$\forall e_i, e_j \in \mathbb{F}_q, \quad q \in \mathbb{P},$$

$$d_L(e_i, e_j) = \min(|e_i - e_j|, q - |e_i - e_j|).$$

Lee weight, $wt_L(\cdot)$

$$\forall \mathbf{e} = (e_1, \dots, e_n) \in \mathbb{F}_q^n, \quad q \in \mathbb{P},$$

$$wt_L(\mathbf{e}) = \sum_i w_L(e_i) = \sum_i d_L(e_i, 0).$$

Goal

In this paper, we analyze the complexity of SD problems with varying alphabets' sizes and different weight functions.

Goal

In this paper, we analyze the complexity of SD problems with varying alphabets' sizes and different weight functions.

We use cryptanalytic approach where the complexity of the problem is calculated as the asymptotic running time of the algorithm that solves the problem.

INFORMATION SET DECODING

Information Set Decoding

- The best generic algorithms for solving the syndrome decoding problem.

[3] Matthieu Finiasz and Nicolas Sendrier. “Security Bounds for the Design of Code-Based Cryptosystems”. In: 2009, pp. 88–105. DOI: 10.1007/978-3-642-10366-7_6.

Information Set Decoding

- The best generic algorithms for solving the syndrome decoding problem.
- Aims to solve the SD problem by exploiting the linear structure of the code.

[3] Matthieu Finiasz and Nicolas Sendrier. “Security Bounds for the Design of Code-Based Cryptosystems”. In: 2009, pp. 88–105. DOI: 10.1007/978-3-642-10366-7_6.

Information Set Decoding

- The best generic algorithms for solving the syndrome decoding problem.
- Aims to solve the SD problem by exploiting the linear structure of the code.

In this paper, we use a framework that encompasses different ISD algorithms[3], and we generalize it to the quantum setting.

[3] Matthieu Finiasz and Nicolas Sendrier. "Security Bounds for the Design of Code-Based Cryptosystems". In: 2009, pp. 88–105. DOI: 10.1007/978-3-642-10366-7_6.

General Framework

1. Permutation step

Picks a random permutation π and permutes the columns of $\mathbf{H}$ accordingly to obtain $\mathbf{H}_\pi$.

... poly(n)

General Framework

1. Permutation step

Picks a random permutation π and permutes the columns of $\mathbf{H}$ accordingly to obtain $\mathbf{H}_\pi$.

... poly(n)

2. Partial Gaussian elimination step

Performs Gaussian elimination on the left submatrix of size $(n-k-l) \times (n-k-l)$. This operation corresponds to multiplying $\mathbf{H}_\pi$ via matrix $\mathbf{S} \in \mathbb{F}^{(n-k)\times(n-k)}$ to obtain

$$\mathbf{S}\mathbf{H}_\pi = \begin{pmatrix} \mathbf{1}_{n-k-l} & \mathbf{H}' \\ \mathbf{0} & \mathbf{H}'' \end{pmatrix}.$$

... poly(n)

After Gaussian elimination:

$$
\begin{aligned}
H_\pi e^\mathsf{T} = s^\mathsf{T} \qquad & \Leftrightarrow S H_\pi e^\mathsf{T} = S s^\mathsf{T} \\
& \Leftrightarrow \begin{pmatrix} 1_{n-k-l} & H' \\ 0 & H'' \end{pmatrix} \begin{pmatrix} e'^\mathsf{T} \\ e''^\mathsf{T} \end{pmatrix} = \begin{pmatrix} s'^\mathsf{T} \\ s''^\mathsf{T} \end{pmatrix} \\
& \Leftrightarrow \begin{cases} e'^\mathsf{T} + H' e''^\mathsf{T} = s'^\mathsf{T} \\ H'' e''^\mathsf{T} = s''^\mathsf{T}. \end{cases}
\end{aligned}
$$

After Gaussian elimination:

$$
\begin{aligned}
H_\pi e^T = s^T \qquad & \Leftrightarrow SH_\pi e^T = Ss^T \\
& \Leftrightarrow \begin{pmatrix} 1_{n-k-l} & H' \\ 0 & H'' \end{pmatrix} \begin{pmatrix} e'^T \\ e''^T \end{pmatrix} = \begin{pmatrix} s'^T \\ s''^T \end{pmatrix} \\
& \Leftrightarrow \begin{cases} e'^T + H'e''^T = s'^T \\ H''e''^T = s''^T. \end{cases}
\end{aligned}
$$

- Solve SD subproblem for $\mathbf{H}'' \in \mathbb{F}_q^{l\times(k+l)}$, $\mathbf{e}'' \in \mathbb{F}_q^{k+l}$, and $\mathbf{s}'' \in \mathbb{F}_q^{l}$ such that all the $\mathbf{e}''$ satisfy $\mathbf{H}''\mathbf{e}'' = \mathbf{s}''$ and $\mathrm{wt}(\mathbf{e}'') = p$, $p \leq w$.

After Gaussian elimination:

$$
\begin{aligned}
H_\pi e^T = s^T \qquad & \Leftrightarrow SH_\pi e^T = Ss^T \\
& \Leftrightarrow \begin{pmatrix} 1_{n-k-l} & H' \\ 0 & H'' \end{pmatrix} \begin{pmatrix} e'^T \\ e''^T \end{pmatrix} = \begin{pmatrix} s'^T \\ s''^T \end{pmatrix} \\
& \Leftrightarrow \begin{cases} e'^T + H'e''^T = s'^T \\ H''e''^T = s''^T. \end{cases}
\end{aligned}
$$

- Solve SD subproblem for $\mathbf{H}'' \in \mathbb{F}_q^{l \times (k+l)}$, $\mathbf{e}'' \in \mathbb{F}_q^{k+l}$, and $\mathbf{s}'' \in \mathbb{F}_q^{l}$ such that all the $\mathbf{e}''$ satisfy $\mathbf{H}''\mathbf{e}'' = \mathbf{s}''$ and $\mathrm{wt}(\mathbf{e}'') = p$, $p \leq w$.
- Calculate $\mathbf{e}' = \mathbf{H}'\mathbf{e}'' - \mathbf{s}'$ and check if the $\mathrm{wt}(\mathbf{e}') = w - p$.

Framework

3. SD step

Finds many the solution to SD($k+l$, l, p), the SD subproblem.

... T_{SD}

Framework

3. SD step

Finds many the solution to $\mathrm{SD}(k+l, l, p)$, the SD subproblem.

. . . T_{SD}

4. Test step

Checks if the solutions to the subproblem yield a solution to the original problem: for each $\mathbf{e}''$ (of weight p) found in the SD step, it checks if $\mathbf{e}' + \mathbf{H}'\mathbf{e}'' = \mathbf{s}'$, and if the weight of corrspoding $\mathbf{e}'$ is $w - p$.

. . . $\mathrm{poly}(n)\times$ number of solutions from SD step

One iteration of the algorithm

1. Permutation step

... $\text{poly}(n)$

2. Partial Gaussian elimination step

... $\text{poly}(n)$

3. SD step: solves the SD subproblem.

... T_{SD}

4. Test step

... $\text{poly}(n)\times$ number of solution from SD step

Lemma 1: Probability of success in the test step

Let S_r^m represent the number of vectors of weight r in the vector spaces of dimensions m over a finite field of size q.

The probability of success in the test step, P_1, is then calculated as:

$$P_1 = \min\{1, O(\frac{S_{w-p}^{n-k-l}}{\max\{1, \min\{S_w^n q^{-\ell}, q^{n-k-\ell}\}\}})\}.$$

SPHERE SURFACE AREA

Calculating Sphere Surface Area, S_w^n

- Counts the number of vectors of weight w in a vector space of dimension n.

[4] Jaakko Astola. "On the asymptotic behaviour of Lee-codes". In: (1984), pp. 13–23. DOI: 10.1016/0166-218X(84)90074-X.

Calculating Sphere Surface Area, S_w^n

- Counts the number of vectors of weight w in a vector space of dimension n.

We propose a method for calculating the sphere surface area that relies on Astola's[4] convex optimization approach and generalizes it to any alphabet size and an arbitrary weight function .

[4] Jaakko Astola. "On the asymptotic behaviour of Lee-codes". In: (1984), pp. 13–23. DOI: 10.1016/0166-218X(84)90074-X.

Sphere surface area, S_w^n

Proposition 3: Sphere Surface Area, S_w^n

Fix a parameter q, a weight function wt and a weight w, and let the set C be defined as:

$$C\{\mathbf{c} = (c_1, \cdots, c_q) : i \in [q], \quad c_i \in \mathbb{N}, \quad \sum_{i=1}^{q} c_i = n, \quad \sum_{i=1}^{q} c_i d(i, 0) = w\}.$$

Sphere surface area, S_w^n

Proposition 3: Sphere Surface Area, S_w^n

Fix a parameter q, a weight function wt and a weight w, and let the set C be defined as:

$$C\{\mathbf{c} = (c_1, \cdots, c_q) : i \in [q], \quad c_i \in \mathbb{N}, \quad \sum_{i=1}^{q} c_i = n, \quad \sum_{i=1}^{q} c_i d(i, 0) = w\}.$$

The sphere surface area, S_w^n, equals to:

$$S_w^n = \sum_{\mathbf{c} \in C} \binom{n}{\mathbf{c}}, \qquad (1)$$

where $\binom{n}{\mathbf{c}}$ denotes a multinomial coefficient.

Asymptotic sphere surface area, s_ω

Proposition 4: Asymptotic sphere surface area, s_ω

Fix a parameter q, a weight function wt and a weight w, and let the set C be defined as:

$$C\{\mathbf{c} = (c_1, \cdots, c_q) : i \in [q], \quad c_i \in \mathbb{N}, \quad \sum_{i=1}^{q} c_i = n, \quad \sum_{i=1}^{q} c_i d(i, 0) = w\}.$$

Asymptotic sphere surface area, s_ω

Proposition 4: Asymptotic sphere surface area, s_ω

Fix a parameter q, a weight function wt and a weight w, and let the set C be defined as:

$$C\{\mathbf{c} = (c_1, \cdots, c_q) : i \in [q], \quad c_i \in \mathbb{N}, \quad \sum_{i=1}^{q} c_i = n, \quad \sum_{i=1}^{q} c_i d(i, 0) = w\}.$$

Using Stirling's approximation, the asymptotic sphere surface area $s_\omega = \lim_{n\to\infty} \frac{1}{n} \log_q(S_w^n)$, is calculated as:

$$s_\omega = \lim_{n\to+\infty} \max_{\mathbf{c}\in C} \left(\sum_{i=1}^{q} -\frac{c_i}{n} \log_q \frac{c_i}{n} \right). \tag{2}$$

Asymptotic sphere surface area, s_ω

Problem 1: asymptotic sphere surface area, $s_\omega = \lim_{n\to\infty} \frac{1}{n} \log_q(S_w^n)$

Let $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_{q-1})$, and $\lambda_i \in \mathbb{R}_+$ for each $i \in [q]$.

$$\textbf{Maximize}: \quad -\sum_{i=1}^{q} \lambda_i \log_q \lambda_i,$$

$$\textbf{Subject to}: \quad \sum_{i=1}^{q} \lambda_i = 1, \quad \sum_{i=1}^{q} d(i,0)\lambda_i = \omega.$$

Let $\tilde{\boldsymbol{\lambda}} = (\tilde{\lambda}_1, ..., \tilde{\lambda}_q)$ be the solution to the problem. The asymptotic sphere surface area is then calculated as $s_\omega = -\sum_{i=1}^{q} \tilde{\lambda}_i \log_q \tilde{\lambda}_i$.

OUR APPROACH

Versions of ISD

- Differ primarily in the SD step, i.e., in the way they solve the SD subproblem.

Versions of ISD

- Differ primarily in the SD step, i.e., in the way they solve the SD subproblem.

We use a version that relies on the Schroeppel–Shamir's idea[5] and Wagner's algorithm[6] for solving a generalised k-sum problem.

[5] Richard Schroeppel and Adi Shamir. "A T=O($2^{n/2}$), S=O($2^{n/4}$) Algorithm for Certain NP-Complete Problems". In: (1981), pp. 456–464. DOI: 10.1137/0210033.

[6] David A. Wagner. "A Generalized Birthday Problem". In: ed. by Moti Yung. 2002, pp. 288–303. DOI: 10.1007/3-540-45708-9_19.

Versions of ISD

- Differ primarily in the SD step, i.e., in the way they solve the SD subproblem.

We use a version that relies on the Schroeppel–Shamir's idea and Wagner's algorithm for solving a generalised k-sum problem.
*We refer to it as ”classical Wagner's ISD algorithm”.

Versions of ISD

- Differ primarily in the SD step, i.e., in the way they solve the SD subproblem.

We use a version that relies on the Schroeppel–Shamir's idea and Wagner's algorithm for solving a generalised k-sum problem.

In the quantum setting, we combine the approach from the classical setting with Grover's algorithm[5] and the amplitude amplification[6].

---

[5]Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: 1996, pp. 212–219. DOI: 10.1145/237814.237866.

[6]Gilles Brassard and Peter Hoyer. "An Exact Quantum Polynomial-Time Algorithm for Simon's Problem". In: 1997, pp. 12–23. DOI: 10.1109/ISTCS.1997.595153.

Versions of ISD

- Differ primarily in the SD step, i.e., in the way they solve the SD subproblem.

We use a version that relies on the Schroeppel–Shamir's idea and Wagner's algorithm for solving a generalised k-sum problem.

In the quantum setting, we combine the approach from the classical setting with Grover's algorithm and the amplitude amplification.
*We refer to it as "quantum Wagner's ISD algorithm".

Checkable Multiple Syndrome Decoding Problem, $\mathrm{CMSD}(n, k, w, Y, Z)$

Input – A parity check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, a syndrome $\mathbf{s} \in \mathbb{F}_q^{n-k}$, a weight function $\mathrm{wt} : \mathbb{F}_q^n \to \mathbb{R}_+$, and a weight $w \in \mathbb{N}$.

Goal – Output the description of a function $f : [Y] \to \mathbb{F}_q^n$ such that f is efficiently computable[a] and that, if evaluated on each possible entry, yields Z soulutions to the CMSD problem, where $Z = |\{\mathbf{e} : \mathbf{e} \in \mathrm{Im}(f),\ \mathbf{s} = \mathbf{e}\mathbf{H}^\top \text{ and } \mathrm{wt}(\mathbf{e}) = w\}|$.

[a] But it need not to have an efficient description. For example, it can be stored in a large precomputed database.

Equivalence of SD and CMSD

- Z solutions to the SD problem can be found in time Y by calculating $f(1), f(2), \ldots, f(Y)$.

Equivalence of SD and CMSD

- Z solutions to the SD problem can be found in time Y by calculating $f(1), f(2), \ldots, f(Y)$.
- Given that $Y \geq Z$, if there is Z available solutions to SD, namely, $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_Z$, the function f can be constructed as:

$$f(i) = \begin{cases} \mathbf{e}_i, & \text{if } i \in [Z] \\ \mathbf{r} \in \mathbb{F}_q^n, & \text{otherwise.} \end{cases}$$

One iteration of the algorithm

1. **Permutation step**

. . . poly(n)

2. **Partial Gaussian elimination step**

. . . poly(n)

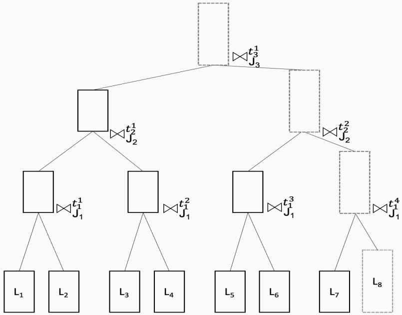
3. **CMSD step**: solves the CMSD subproblem.

. . . T_{CMSD}

4. **Test step**

. . . poly(n)$\times$ number of solution from SD step

Wagners' ISD algorithms



Merging on $a = 3$ levels.

$J_1 \dots J_a$ – partition of $[n]$

$|J_j$ – support on J_j

$\forall i \in [2^a], \forall j \in [a]$,

$t_j^i \in \mathbb{F}_q^n$: $\sum_i (t_j^i)_{|J_j} = s_{|J_j}$

$$L_1 \bowtie^t_{J_j} L_2 = \{x + y : x \in L_1, y \in L_2, x_{|J_j} + y_{|J_j} = t_{|J_j}\}.$$

Classical setting

Proposition 1: running time of the classical algorithm

Fix parameters l, p, Y, and Z of an information set decoding algorithm.

Classical setting

Proposition 1: running time of the classical algorithm

Fix parameters a, l, p, Y, and Z of an information set decoding algorithm.

The classical running time of the algorithm, T^C_{ISD}, is given as:

$$T^C_{ISD} = O\left(\max\left\{1, \frac{1}{P_1 Z}\right\} \cdot (\mathrm{poly}(n) + T_{CMSD} + \mathrm{poly}(n)Y)\right),$$

where P_1 is the probability of success in the test step, and T_{CMSD} is the running time for solving $\mathrm{CMSD}(k+l, l, p, Y, Z)$ in CMSD step.

Classical setting

Proposition 2: the running time of the CMSD step

Let then $s_\rho = \lim_{k\to\infty} \frac{1}{k}\log_q(S_p^{k+l})$, $u = \min\{\frac{s_\rho}{2^a}, (1-R)/a\}$, and $x = (1-R) - (a-1)u$.

The algorithm solves the $\mathrm{CMSD}(k+l, l, p, Y, Z)$ problem in time T_{CMSD}, where

$$T_{\mathrm{CMSD}} = q^{(k+l)(u+o(1))}, \quad Y = T_{\mathrm{CMSD}}, \quad Z = q^{(k+l)(2u-x+o(1))}.$$

Quantum setting

- Running time of a quantum algorithm: the number of gates in its corresponding circuit description.

Quantum setting

- Running time of a quantum algorithm: the number of gates in its corresponding circuit description.
- We utilize the QRAM model, for which we assume the operation:

$$U_{QRAM} : iyb_1, \ldots, b_n \to iy + x_i b_1, \ldots, b_n$$

can be done in time polylog(n) when each b_i is a single bit.

Quantum setting

Definition: Grover's algorithm[5]

Let us define a function $f : \{0,1\}^n \to \{0,1\}$ that has an efficient classical description.

[5] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: 1996, pp. 212–219. DOI: 10.1145/237814.237866.

Quantum setting

Definition: Grover's algorithm[5]

Let us define a function $f : \{0,1\}^n \to \{0,1\}$ that has an efficient classical description.

Grover's algorithm can find i such $f(i) = 1$ in time $O(\mathrm{poly}(n)2^{n/2})$ if such an i exists and output 'no solution' otherwise.

[5] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: 1996, pp. 212–219. DOI: 10.1145/237814.237866.

Quantum setting

Definition: Amplitude amplification[5]

Let us define a function $f : \{0,1\}^n \to \{0,1\}$ that has an efficient classical description.

[5] Gilles Brassard and Peter Hoyer. "An Exact Quantum Polynomial-Time Algorithm for Simon's Problem". In: 1997, pp. 12–23. DOI: 10.1109/ISTCS.1997.595153.

Quantum setting

Definition: Amplitude amplification[5]

Let us define a function $f : \{0,1\}^n \to \{0,1\}$ that has an efficient classical description.

Consider then a quantum algorithm $\mathcal{A}$ that outputs i such that $f(i) = 1$ with probability p, and $f(i) = 0$ with probability $1 - p$. The algorithm $\mathcal{A}$ does not perform intermediate quantum measurements.

[5] Gilles Brassard and Peter Hoyer. "An Exact Quantum Polynomial-Time Algorithm for Simon's Problem". In: 1997, pp. 12–23. DOI: 10.1109/ISTCS.1997.595153.

Quantum setting

Definition: Amplitude amplification[5]

Let us define a function $f : \{0,1\}^n \to \{0,1\}$ that has an efficient classical description.

Consider then a quantum algorithm $\mathcal{A}$ that outputs i such that $f(i) = 1$ with probability p, and $f(i) = 0$ with probability $1 - p$. The algorithm $\mathcal{A}$ does not perform intermediate quantum measurements.

Using amplitude amplification, one can find i such that $f(i) = 1$ by making $O(\frac{1}{\sqrt{p}})$ calls to $\mathcal{A}$.

[5] Gilles Brassard and Peter Hoyer. "An Exact Quantum Polynomial-Time Algorithm for Simon's Problem". In: 1997, pp. 12–23. DOI: 10.1109/ISTCS.1997.595153.

Quantum setting

Proposition 3: running time of the quantum algorithm

Fix parameters a, l, p, Y, and Z of an information set decoding algorithm.

The quantum running time of the algorithm, T_{ISD}^{Q}, is given as:

$$T_{ISD}^{Q} = O\left(\sqrt{\max\left\{\frac{1}{ZP_1}, 1\right\}} \cdot \left(\mathrm{poly}(n) + T_{CMSD} + \mathrm{poly}(n)\sqrt{Y}\right)\right),$$

where P_1 is the probability of success in the test step, and T_{CMSD} is the running time for solving $\mathrm{CMSD}(k + l, l, p, Y, Z)$ in CMSD step.

Quantum setting

Proposition 4: the running time of the CMSD step

Let then $s_\rho = \lim_{k\to\infty} \frac{1}{k} \log_q(S_p^{k+l})$, $u = \min\{\frac{s_\rho}{2^a+1}, (1-R)/a\}$, and $x = (1-R) - (a-1)u$.

The algorithm solves the $\mathrm{CMSD}(k+l, l, p, Y, Z)$ problem in time T_{CMSD}, where

$$T_{\mathrm{CMSD}} = q^{(k+l)(u+o(1))}, \quad Y = q^{(k+l)(2u+o(1))}, \quad Z = q^{(k+l)(3u-x+o(1))}.$$

NUMERICAL RESULTS

Asymptotic time complexity

The computational complexity is evaluated as the asymptotic running time of the algorithm when parameters l, p, and a are optimized and yield the shortest running time.

Asymptotic time complexity

The computational complexity is evaluated as the asymptotic running time of the algorithm when parameters l, p, and a are optimized and yield the shortest running time.

”binary complexity”

$$\alpha = \lim_{n\to\infty} \frac{1}{n} \log_2 T_{ISD}.$$

”q-ary complexity”

$$\hat{\alpha} = \lim_{n\to\infty} \frac{1}{n} \log_q T_{ISD}.$$

Asymptotic time complexity

Four algorithms are compared: Prange's[6] algorithm, Stern's/Dumer's algorithm[7], Wagner's ISD algorithm and it's quantum version.

[6] E. Prange. "The use of information sets in decoding cyclic codes". In: IRE Transactions on Information Theory (1962), pp. 5–9. DOI: 10.1109/TIT.1962.1057777.

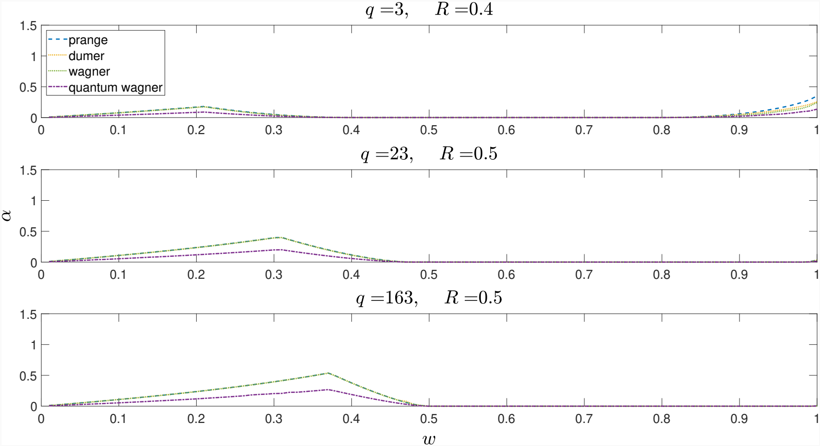
[7] Ilya Dumer. "On minimum distance decoding of linear codes". In: Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory. 1991, pp. 50–52.

Asymptotic time complexity

Four algorithms are compared: Prange's algorithm, Stern's/Dumer's algorithm , Wagner's ISD algorithm and it's quantum version.

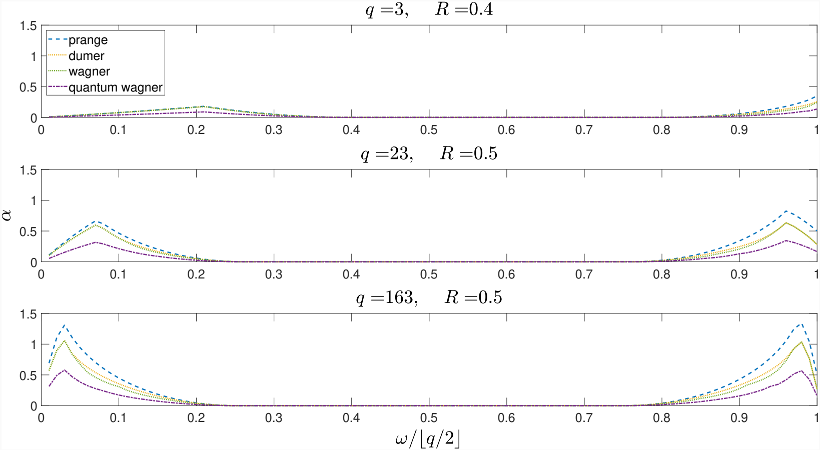
For a fixed weight function and alphabet size, the hardest instance of the problem is found as the maximum of the running time as a function of the code rate, $R = \frac{k}{n}$, and the normalized weight $\omega = \frac{w}{n}$.

Hamming weight case comparison



Asymptotic time complexity of four ISD algorithms.

Lee weight case comparison



Asymptotic time complexity of four ISD algorithms.

Hardest instances of SD problem

q	Classical Wagner's ISD algorithm			
	$\mathrm{wt}_H(\cdot)$		$\mathrm{wt}_L(\cdot)$	
	α	$\hat{\alpha}$	α	$\hat{\alpha}$
3	0.269	0.170	0.269	0.170
43	0.459	0.085	0.794	0.146
163	0.541	0.074	1.117	0.152
643	0.602	0.065	1.455	0.156

q	Quantum Wagner's ISD algorithm			
	$\mathrm{wt}_H(\cdot)$		$\mathrm{wt}_L(\cdot)$	
	α	$\hat{\alpha}$	α	$\hat{\alpha}$
3	0.148	0.093	0.148	0.093
43	0.230	0.042	0.429	0.079
163	0.271	0.037	0.607	0.083
643	0.316	0.034	0.794	0.085

Table: Hardest instances of CMSD problem in the Lee and Hamming weight.

SUMMA SUMMARUM

What did we do?

We analyzed the complexity of SD problems with varying alphabets' sizes and different weight functions.

What did we do?

We analyzed the complexity of SD problems with varying alphabets' sizes and different weight functions.

We generalized a convex optimization approach to calculating the asymptotic sphere surface area to arbitrary weight functions and alphabet sizes.

What did we do?

We analyzed the complexity of SD problems with varying alphabets' sizes and different weight functions.

We generalized a convex optimization approach to calculating the asymptotic sphere surface area to arbitrary weight functions and alphabet sizes.

We proposed a general framework that encompasses different ISD algorithms, both in the classical and quantum setting.

What did we do?

We analyzed the complexity of SD problems with varying alphabets' sizes and different weight functions.

We generalized a convex optimization approach to calculating the asymptotic sphere surface area to arbitrary weight functions and alphabet sizes.

We proposed a general framework that encompasses different ISD algorithms, both in the classical and quantum setting.

In the numerical part of the paper, we analyzed the asymptotic computational complexity of SD problem for the Hamming and Lee weight.

What did we observe?

For a fixed alphabet size $q > 3$, the complexity of the hardest instances of SD problem is higher in the Lee than in the Hamming weight.

WHAT DID WE OBSERVE?

For a fixed alphabet size $q > 3$, the complexity of the hardest instances of SD problem is higher in the Lee than in the Hamming weight.
*That is true both in the classical and quantum setting.

What did we observe?

For a fixed alphabet size $q > 3$, the complexity of the hardest instances of SD problem is higher in the Lee than in the Hamming weight.
*That is true both in the classical and quantum setting.

For the quantum setting, our algorithms have almost a quadratic improvement over the classical setting.

What did we observe?

For a fixed alphabet size $q > 3$, the complexity of the hardest instances of SD problem is higher in the Lee than in the Hamming weight.
*That is true both in the classical and quantum setting.

For the quantum setting, our algorithms have almost a quadratic improvement over the classical setting.

Nevertheless, the problem remains exponentially hard for conveniently chosen parameters both in the classical and quantum setting (for the class of the algorithms we consider).

Future work

What about other distance functions?

Future work

What about other distance functions?

What about really high values of q?

Future work

What about other distance functions?

What about really high values of q?

What about other approaches? Comparison of the results with other papers on the topic.

Future work

What about other distance functions?

What about really high values of q?

What about other approaches? Comparison of the results with other papers on the topic.

How SD problem with different underlying alphabet sizes and weight functions can be used to constructing a signature scheme?

THANK YOU FOR YOUR ATTENTION!

[a]

[a] This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 754362.