Limitations of Quantum Sieving-Based Information-Set Decoding

Lynn Engelberts[1,2], Simona Etinski[1], and Johanna Loyer[1]
lynn.engelberts(@)cwi.nl, simona.etinski(@)cwi.nl, johanna.loyer(@)cwi.nl

[1] CWI, the Netherlands
[2] QuSoft, the Netherlands

1 Introduction

Information-set decoding (ISD) algorithms are the most efficient generic attacks against the decoding problem, which is the computational problem underlying the security of code-based primitives. As these primitives are considered one of the most prominent candidates for quantum-safe cryptography, analyzing the complexity of both classical and quantum ISD algorithms attacking the decoding problem is crucial for determining their security. In this work, we are primarily interested in one of the latest advancements in cryptanalysis of the decoding problem: sieving-based ISD algorithms. Before giving more details on these algorithms, let us formally define the computational problem we are attacking and related notions. An $[n, k]$ *binary linear code* is a linear subspace $\mathcal{C} \subseteq \mathbb{F}_2^n$ of dimension k. The decoding problem is then given as follows.

Decoding problem, $DP(n, k, w)$

Given an $[n, k]$ binary linear code $\mathcal{C}$ and integer w, find a codeword $\mathbf{x}_\mathcal{C} \in \mathcal{C}$ of weight $|\mathbf{x}_\mathcal{C}| := w$.[a]

[a] The (Hamming) weight $|\mathbf{x}|$ of $\mathbf{x} \in \mathbb{F}_2^n$ is its number of non-zero coefficients in $\mathbf{x}$.

The decoding problem is NP-hard in the worst case [BMvT78]. More important for cryptographic purposes, it is known that w can be chosen to guarantee exactly one solution on average, giving us the so-called *unique decoding regime*. In this regime, the decoding problem is believed to be hard on average: for a random instance of the problem, all known algorithms (including ISD) run in time and memory exponential in n.

Sieving-Based ISD. Since the introduction of ISD by Prange [Pra62], it has been improved using various techniques. The most recent variant of ISD uses sieving as a subroutine. Sieving-based ISD was introduced in [GJN23] and further generalized and improved in [DEEK24], yielding asymptotic runtime close to that of the state-of-the-art algorithms [MMT11, BJMM12, MO15] and better time-memory trade-offs. These improvements raise natural questions on how well quantum analogs of sieving-based ISD perform, and how these compare to other quantum ISD algorithms [Ber10, KT17, Kir18].

Sieving Outside of ISD. While sieving-based algorithms are new in code-based cryptanalysis, they are abundant in lattice-based cryptanalysis and result in the state-of-the-art classical [BDGL16] and quantum [BCSS23] algorithms for the shortest vector problem. In particular, this raises yet another question whether similar quantum speed-ups can be obtained for sieving for codes (i.e., as an algorithm in itself and not necessarily inside the ISD context). Nevertheless, an important difference is that in the lattice setting, sieving is used to attack a full instance of a problem, while in the code setting, it is used as an ISD subroutine attacking a subinstance of the decoding problem.

Main Contributions. We study a natural quantum analog of the sieving-based ISD framework considered in [GJN23, DEEK24], and show it does not provide any speed-up over the first presented quantum ISD algorithm [Ber10]. Our analysis highlights that novel ideas are needed to outperform [Ber10] and, ideally, also the state-of-the-art quantum ISD algorithms [KT17, Kir18]. We suggest two natural attempts to adapt the sieving-based ISD framework, and explain why these do not allow for overcoming the found limitations. In addition, we consider quantum analogs of one component of the sieving-based ISD framework, namely the code sieving subroutine, and demonstrate that quantum speed-ups *are* possible for this part. Specifically, using the quantum-walk framework from [MNRS11] and techniques from quantum lattice sieving [CL21] (modified to the code setting), we obtain a speed-up over the best known classical algorithm from [DEEK24] and over our baseline quantum variant that uses Grover's algorithm [Gro96]. Nevertheless, our aforementioned result indicates that these quantum speed-ups for the code sieving subroutine do not lead to an improved quantum ISD algorithm.

2 Preliminaries

Information-Set Decoding (ISD). We focus on ISD algorithms following the framework from [FS09] for solving a random instance of $DP(n, k, w)$. As input, such an ISD algorithm is given an $[n, k]$ random binary linear code, an integer weight w, and a certain oracle $\mathcal{A}$. It then guesses part of the solution (corresponding to the so-called information set), thereby creating a subinstance of the decoding problem, $DP(n', k, w')$ for $k \leq n' \leq n,\ w' \leq \max(n, w)$, with many solutions. The algorithm then invokes the oracle $\mathcal{A}$ that finds many solutions to the subinstance and returns them as a list $\mathcal{L}$. Finally, the algorithm checks if any of the solutions to the subinstance from $\mathcal{L}$ yields a solution to the original problem. If so, it outputs a solution, i.e., a codeword $\mathbf{x}_{\mathcal{C}} \in \mathcal{C}$ of Hamming weight w. Otherwise, the algorithm returns to the guessing step.

Runtime of Classical ISD. The runtime of a classical ISD algorithm (within the aforementioned framework) depends on the expected runtime $T_{\mathcal{A}}$ of the oracle $\mathcal{A}$, the probability p_1 that the initial guess was correct, and the probability p_2 that a solution from $\mathcal{L}$ yields a solution to the original problem. Specifically, writing N for the size of $\mathcal{L}$, the expected runtime of this ISD algorithm is $\tilde{O}(\frac{T_{\mathcal{A}}}{p_1 p_2})$, where $p_1 := \binom{n'}{w'}\binom{n-n'}{w-w'}/\binom{n}{w}$, and $p_2 := N2^{n'-k}/\binom{n'}{w'}$. See [DEEK24, Theorem 3.1] or the original proof in [FS09].

Runtime of Quantum ISD. The best known quantum ISD algorithms [Ber10, KT17, Kir18] are based on the idea that amplitude amplification (AA) [BHMT02] allows for obtaining a quadratic speed-up when searching for a solution of the decoding problem over multiple iterations of the ISD algorithm. Furthermore, the subroutine $\mathcal{A}$ is allowed to be quantum. The runtime of the resulting quantum ISD algorithm is $\tilde{O}(\frac{T_{\mathcal{A}}}{\sqrt{p_1 p_2}})$, where p_1 and p_2 are as before, and $T_{\mathcal{A}}$ is the expected runtime of the (quantum) oracle $\mathcal{A}$.

Code Sieving. Code sieving [GJN23, DEEK24] solves an instance of $DP(n', k, w')$ where the goal is to find N codewords of weight w' in an $[n', k]$ code $\mathcal{C}'$. The algorithm starts by sampling vectors in $\mathbb{F}_2^n$ of weight w', forming an initial list. It then 'sieves' vectors from the initial list to obtain codewords in $\mathcal{C}'$. The sieving is performed iteratively, by adding one code constraint in each sieving step, to eventually obtain an output list of vectors that satisfy all code constraints of $\mathcal{C}'$. We observe here that simply adding a code constraint would shrink the list size by half in each iteration, resulting in a too-short output list. To avoid the shrinking, the algorithm adds new vectors of the desired weight in each iteration, which are obtained as a solution to the following near-neighbor search (NNS) problem.

Near-neighbor search, $NNS(n', w', N)$

Given a list $\mathcal{L}$ of N independent and uniformly random vectors in $\mathbb{F}_2^{n'}$ of weight w', find all pairs $(\mathbf{x}, \mathbf{y}) \in \mathcal{L}^2$ satisfying $|\mathbf{x} + \mathbf{y}| = w'$.[a]

[a] By '+' we denote the bitwise 'XOR' between two vectors.

Since the number of iterations corresponds to the number of code constraints, which is polynomial in n, the runtime of the sieving algorithm is determined by the runtime of an NNS algorithm.

3 Quantum Sieving-Based ISD

With *sieving-based ISD* we refer to the ISD algorithm obtained when using code sieving (as described before) as its subroutine $\mathcal{A}$. We now consider a natural quantum analog of sieving-based ISD, referred to as *quantum sieving-based ISD*, that results from using code sieving as subroutine in the quantum ISD algorithm. While the best known algorithm for sieving-based ISD has promising performance compared to the classical state-of-the-art for ISD, we show that quantum sieving-based ISD does not yield performance comparable to the quantum state-of-the-art [KT17, Kir18].

3.1 On Limitations of Quantum Sieving-Based ISD

As shown in [DEEK24], using code sieving as the oracle $\mathcal{A}$ imposes a lower bound on the size of the list $\mathcal{L}$ that $\mathcal{A}$ returns. This lower bound turns out to be a bottleneck in improving over what we refer to as *quantum Prange*, the classical Prange algorithm quantized with AA [Ber10]. More formally, we claim the following.

Claim Consider a random instance of $DP(n,k,w)$ in the unique decoding regime. For all $n', w', N \in \mathbb{N}$ satisfying $4\binom{n'}{w'}/\left(\binom{w'}{w'/2}\binom{n'-w'}{w'/2}\right) \leq N \leq \binom{n'}{w'}/2^{n'-k}$, there is no (classical or quantum) oracle $\mathcal{A}$ such that the resulting quantum ISD algorithm has a better runtime than quantum Prange.

The upper bound on N is imposed by the ISD framework itself (to guarantee the existence of N codewords in the subinstance), and the lower bound is imposed by the construction of the code sieving methods from [GJN23, DEEK24] (to guarantee that $NNS(n', w', N)$ has N solutions).

The claim is justified as follows. A trivial lower bound on the expected runtime of *any* oracle $\mathcal{A}$ is $T_{\mathcal{A}} \geq N$, since $\mathcal{A}$ is required to output N solutions. For p_1 and p_2 as defined in Section 2, the expected runtime of any such quantum ISD algorithm is thus $\Omega\left(N/\sqrt{p_1 p_2}\right) = \Omega\left(\sqrt{N\binom{n}{w}/(\binom{n-n'}{w-w'}2^{n'-k})}\right)$, which, for fixed n', w', is minimal when N equals its lower bound. Omitting constant factors, the runtime is thus lower bounded by

$$\min_{(n',w') \text{ s.t. } \binom{w'}{\frac{w'}{2}}\binom{n'-w'}{\frac{w'}{2}} \geq 2^{n'-k}} \sqrt{\frac{\binom{n'}{w'}\binom{n}{w}}{\binom{w'}{\frac{w'}{2}}\binom{n'-w'}{\frac{w'}{2}}\binom{n-n'}{w-w'}2^{n'-k}}}. \tag{1}$$

By numerically minimizing Equation 1, we obtain that, for all allowed n', w', this is never better than the runtime of quantum Prange (which is $\sqrt{\binom{n}{w}/\binom{n-k}{w}}$). In fact, the optimal values obtained through numerical optimization essentially correspond to the values characterizing quantum Prange: $n' = k$ and $w' = 0$. Our Python script is available at `https://github.com/lynnengelberts/quantum-sieving-for-codes-public.git`.

Remark 1 (Comparison with the classical setting). The same argument does not apply to *classical* sieving-based ISD, which is non-surprising since [DEEK24] have shown it outperforms classical Prange. There, a trivial lower bound on the runtime of the oracle $\mathcal{A}$ would again be N, giving a (possibly non-tight) lower bound on the time complexity of classical sieving-based ISD of $N/(p_1 p_2)$. Unlike in the quantum setting, the dependency (and, in particular, the lower bound) on N disappears as it is canceled out by p_2, namely $N/(p_1 p_2) = \binom{n}{w}/\left(\binom{n-n'}{w-w'}2^{n'-k}\right)$ since p_2 is linear in N.

3.2 On (Failed) Attempts to Overcome the Limitations

The previous subsection shows the limitations of quantum sieving-based ISD and implies that the framework should be adapted to outperform quantum Prange. Specifically, the main bottleneck in quantum sieving-based ISD appears to come from the lower bound on the output size N of the sieving subroutine, since the factors $T_{\mathcal{A}}$ and p_2 in the expected runtime $T_{\mathcal{A}}/\sqrt{p_1 p_2}$ of a quantum ISD algorithm both depend on N. Here, we present two approaches that potentially allow us to overcome these limitations and explain why neither of them works.

Approach 1: From Pairs to Tuples. Inspired by lattice-based cryptanalysis, an idea to obtain a smaller lower bound on N is to use t-tuples in the sieving subroutine for $t \geq 2$. That is, the sieving algorithm is instructed to find $(\mathbf{x}_1, \ldots, \mathbf{x}_t) \in \mathcal{L}^t$ satisfying $|\mathbf{x}_1 + \ldots + \mathbf{x}_t| = w'$. We refer to those tuples as *t-solutions* and observe that, for $t = 2$, we obtain the original sieving algorithm in which the algorithm searches for pairs.[3]

Consider a quantum ISD algorithm invoking $\mathcal{A}$ to obtain N_t t-solutions to $DP(n', k, w')$ and uses amplitude amplification. We keep the meaning of p_1, p_2 and $T_{\mathcal{A}}$ from Section 2 and, to highlight that p_2 depends on the output size N_t, we write $p_2 = p_2(N_t) = N_t q_2$. The expected runtime of quantum ISD with a t-sieving subroutine $\mathcal{A}$ is then $\tilde{O}\left(T_{\mathcal{A}}/\sqrt{p_1 q_2 N_t}\right)$. We consider that $\mathcal{A}$ is constructed similarly to the 2-tuple setting: in each iteration i, it aims to find N_t t-solutions given a list of N_t independent and uniformly random vectors from $\mathbb{F}_2^{n'}$ of weight w'. Writing $p^{(t)}$ for the probability that a t-tuple of independent and uniformly random vectors in $\mathbb{F}_2^n$ of weight w' forms a t-solution, the expected number of t-solutions in the output list is then given by $N_t^t p^{(t)}$. To ensure that, on average, there exist at least N_t t-solutions in the output list, N_t needs to satisfy $N_t^t p^{(t)} \geq N_t$. We obtain the following, and remark that the assumption seems reasonable.

Corollary 2 (A lower bound on quantum t-sieving ISD). *Assume that the runtime of any t-sieving algorithm is lower bounded by the optimal runtime of 2-sieving. Consider a quantum ISD algorithm with the aforementioned t-sieving algorithm as oracle $\mathcal{A}$. The expected runtime of this algorithm is $\Omega(1/\sqrt{p_1 q_2 p^{(2)}})$.*

[3] The intuition comes from tuple-sieving algorithms (e.g., [KMPM19]) which consider $t > 2$ to reduce the lower bound on N.

By Section 3.1, the latter is never better than quantum Prange, from which we can conclude that quantum ISD using t-sieving (with list size N_t) does no better than quantum Prange for all $t \geq 2$.

Proof. There are two cases. First, suppose that $N_t \leq 1/p^{(2)}$. By the assumption, the runtime $T_{\mathcal{A}}$ of any t-sieving algorithm is at least $1/p^{(2)}$ (since this is a lower bound on the output size of a 2-sieving algorithm), so the overall runtime is lower bounded by $T_{\mathcal{A}}/\sqrt{p_1 q_2 N_t} \geq \frac{1}{\sqrt{p_1 q_2}} \frac{1}{p^{(2)}\sqrt{N_t}} \geq \frac{1}{\sqrt{p_1 q_2 p^{(2)}}}$. On the other hand, if $N_t \geq 1/p^{(2)}$, then $T_{\mathcal{A}}/\sqrt{p_1 q_2 N_t} \geq \sqrt{N_t}/\sqrt{p_1 q_2} \geq \frac{1}{\sqrt{p_1 q_2 p^{(2)}}}$ since $T_{\mathcal{A}} \geq N_t$. □

Approach 2: Varying List Sizes. The lower bound on the output size N of the sieving subroutine comes from maintaining the same list size N throughout each iteration in the sieving algorithm. A natural question to ask is whether it would be possible to variate the list size in order to overcome this intrinsic limitation.

We propose the following approach for varying list sizes. Suppose that for given parameters n', w' there exists a (classical/quantum) sieving subroutine of the ISD framework that (for some N_i) iteratively applies a sieving step of the following form. It starts by sampling a list $\mathcal{L}_0$ of N_0 vectors of weight w'. In iteration i, the algorithm finds solution pairs $(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{L}_i^2$, satisfying $|\mathbf{x}_1 + \mathbf{x}_2| = w'$, yielding a new list $\mathcal{L}_{i+1}$ of expected size $N_{i+1} := N_i^2 p$, where $p := \binom{w'}{w'/2}\binom{n'-w'}{w'/2}/\binom{n'}{w'}$ is the probability that a random pair is a solution pair (previously denoted by $p^{(2)}$).[4]

The expected list size in the consecutive steps of the sieving algorithm can then be described via the recursive relation: $N_{i+1} = N_i^2 p$. It follows that $N_{i+1} = N_1^{2^i} p^{2^i - 1}$, and writing $r + 1 := n' - k$, the size of the output list N_{r+1} satisfies $N_{r+1} = N_1^{2^r} p^{2^r - 1}$. We now sketch the proof that there is no choice of $N_1, \ldots, N_{r+1}$ for which this algorithm yields runtime better than that of quantum Prange, denoted by T^*.

We start by observing that the overall runtime T of this algorithm is of order $T_{\max}/\sqrt{p_1 q_2 N_{r+1}}$, where $T_{\max}$ is the maximum among the runtimes of the iterations, and p_1 and $p_2 = q_2 N_{r+1}$ are the probabilities from Section 2. (Note that the output size N_{r+1} was previously denoted by N.) Since $T_{\max} \geq N_{\max} := \max_{i \geq 1} N_i$, we have $T \geq N_{\max}/\sqrt{p_1 q_2 N_{r+1}}$. By Section 3.1 we have $(1/\sqrt{p p_1 q_2}) \geq T^*$ for all allowed n', w'. To complete the proof, it thus suffices to show that there is no choice of $N_1, \ldots, N_{r+1}$ such that $N_{\max}/\sqrt{N_{r+1}} < 1/\sqrt{p}$, which can be shown by considering the two cases $N_{r+1} \geq \frac{1}{p}$ and $N_{r+1} < \frac{1}{p}$ separately.

4 Quantum Algorithms for NNS and Code Sieving

In the introduction, we also asked how well quantum analogs of code sieving perform. We present the first quantum algorithms for NNS and code sieving by combining the classical sieving algorithm from [DEEK24] with techniques from quantum lattice sieving [Laa15, CL21]. The quantum speed-ups we obtain align with the speed-ups observed in the state-of-the-art of lattice cryptanalysis. Although Section 3 shows that using these algorithms as oracle $\mathcal{A}$ does not result in quantum ISD algorithms that perform better than quantum Prange, the quantum algorithms for NNS and code sieving may be of independent interest.

Recall that code sieving repeatedly solves an NNS problem (Problem 2). The complexity of the best known algorithms for this NNS problem, and thereby the overall complexity of code sieving, depends on the cost of a subroutine that we call `FindSolutions` which, in short, searches for 'solutions' in a structured subset of the list $\mathcal{L}$. Similar to the lattice setting, our quantum algorithms aim to speed up this subroutine.

Our Quantum Algorithms. Our first quantum algorithm for `FindSolutions` is a straightforward application of Grover, and serves as a baseline for comparison. Our other quantum algorithms are a quantum-walk algorithm and a variant thereof, where we apply the sparsification technique from [CL21] to end up with a different (and better) balance of the complexity parameters. Both quantum-walk algorithms make use of locality-sensitive filtering (LSF) from [BDGL16] to obtain a speed-up over Grover in the search for 'solutions'. We illustrate the asymptotic performance of the resulting quantum algorithms for NNS (and, thus, for code sieving) using numerical results.[5] We compare the runtime and corresponding memory for the hardest instances (i.e., those weights yielding the worst asymptotic runtime) of the following algorithms: (1) the classical version from [DEEK24], (2) our version using Grover, (3) our version using a quantum walk and a second layer of LSF, and (4) the latter with sparsification. Our results are presented as follows:

- Figure 1(a) shows the asymptotic runtime of the four algorithms, which is obtained by calculating the asymptotic runtime in 100 equidistant values of $\omega := w/n$ ranging in $[0, 0.5)$, where n, w are parameters of Problem 2 and w is chosen such that there is a unique solution to the problem on average.

[4] Note here that taking N_i to be the same for all iterations, results in the original sieving algorithm.

[5] Due to space constraints, we omit the asymptotic analysis.

– Figure 1(b) specifies the asymptotic runtime for the *hardest instances*, i.e., those values of ω for which the runtime curves in Figure 1(a) reach their peak. It also contains the corresponding memory complexity.

All numerical results were obtained using Python script available in the aforementioned repository.

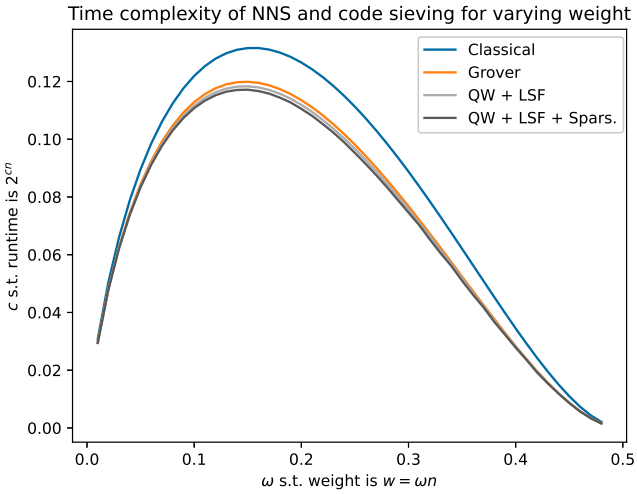

Fig. 1(a): Comparison of the asymptotic runtime of the four algorithms solving an NNS instance with $\omega = w/n$.

Algorithms	t	m_C	m_Q	m_{QRACM}	m_{QRAQM}
Classical	0.132	0.094	-	-	-
Grover*	0.120	0.094	0	0.026	-
QW+LSF*	0.118	0.094	0.024	0.031	0.024
QW+LSF+Spars.*	0.117	0.094	0.023	0.036	0.023

Fig. 1(b): The exponents of the asymptotic runtime (i.e., t s.t. runtime is $2^{tn+o(n)}$) and memory exponents for those ω where the runtime curves in Fig. 1(a) reach their peak. Our algorithms are indicated with '$*$'. Here $m_C, m_Q, m_{QRACM}, m_{QRAQM}$ correspond to classical memory, quantum memory, QRACM, and QRAQM, respectively.

Remark 3. We present the results for $NNS(n, w, N)$, but remark that when used in the ISD framework, n, w should be replaced by n', w'. Moreover, the number 0.132 for Classical in Figure 1(b) refers to the runtime of the classical sieving algorithm from [DEEK24], and is not explicitly stated in their paper.

5 Conclusion

Given the quantum speed-ups in lattice-based cryptanalysis and the recent introduction of sieving techniques for codes in the classical setting, it was essential to evaluate the impact of sieving-based techniques on the security of code-based schemes from a quantum perspective. Moreover, our new quantum algorithms for the NNS problem in the Hamming metric may be of independent interest and have the potential to be used in future algorithms for the decoding problem, for instance, within new sieving algorithms that overcome the limitations addressed in this work or within algorithms with completely different approaches. Identifying specific applications remains an open question for future work. We thus consider this work to provide valuable insights into the intrinsic limitations of the currently proposed sieving approach, which potentially suggests directions for future improvements.

References

BCSS23. X. Bonnetain, A. Chailloux, A. Schrottenloher, and Y. Shen. Finding many collisions via reusable quantum walks - application to lattice sieving. In *EUROCRYPT*, 2023.

BDGL16. A. Becker, L. Ducas, N. Gama, and T. Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, 2016.

Ber10. D. J. Bernstein. Grover vs. McEliece. In *Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010. Proceedings*, 2010.

BHMT02. G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *Quantum Computation and Information*, 2002.

BJMM12. A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in 2 n/20: How 1 + 1 = 0 improves information set decoding. In *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, 2012.

BMvT78. E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Trans. Inf. Theory*, 1978.

CL21. A. Chailloux and J. Loyer. Lattice sieving via quantum random walks. In *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, 2021.

DEEK24. L. Ducas, A. Esser, S. Etinski, and E. Kirshanova. Asymptotics and improvements of sieving for codes. In *EUROCRYPT*, 2024.

FS09. M. Finiasz and N. Sendrier. Security bounds for the design of code-based cryptosystems. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, 2009.

GJN23. Q. Guo, T. Johansson, and V. Nguyen. A new sieving-style information-set decoding algorithm. *IACR Cryptol. ePrint Arch.*, 2023.

Gro96. L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, 1996.

Kir18. E. Kirshanova. Improved quantum information set decoding. In *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, 2018.

KMPM19. E. Kirshanova, E. Mårtensson, E. W. Postlethwaite, and S. R. Moulik. Quantum algorithms for the approximate k-list problem and their application to lattice sieving. *ASIACRYPT*, 2019.

KT17. G. Kachigar and J. Tillich. Quantum information set decoding algorithms. In *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings*, 2017.

Laa15. T. Laarhoven. *Search problems in cryptography, From fingerprinting to lattice sieving.* PhD thesis, Eindhoven University of Technology, 2015.

MMT11. A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, 2011.

MNRS11. F. Magniez, A. Nayak, J. Roland, and M. Santha. Search via quantum walk. *SIAM J. Comput.*, 2011.

MO15. A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, 2015.

Pra62. E. Prange. The use of information sets in decoding cyclic codes. *IRE Trans. Inf. Theory*, 1962.