# Revisiting the May–Meurer–Thomae Algorithm — Solving McEliece-1409 in One Day

Shintaro Narisada[1], Shusaku Uemura[1], Hiroki Okada[1,2], Hiroki Furue[2], Yusuke Aikawa[2], and Kazuhide Fukushima[1]

[1] KDDI Research, Inc, Japan
{sh-narisada,su-uemura,ir-okada,ka-fukushima}@kddi.com
[2] The University of Tokyo, Japan
furue-hiroki261@g.ecc.u-tokyo.ac.jp
aikawa@mist.i.u-tokyo.ac.jp

**Abstract.** As post-quantum cryptography transitions toward practical deployment, the significance of extensive cryptanalysis is on the rise. Three out of the four NIST-PQC round 4 candidates are forms of code-based cryptography. Analyses of asymptotic complexity in information set decoding (ISD) algorithms have been a central focus in the field of code-based cryptography. Recently, Esser, May and Zweydinger (Eurocrypt '22) demonstrate the practicality of the May–Meurer–Thomae (MMT) algorithm by decoding McEliece-1284. Esser and Zweydinger (Eurocrypt '23) propose the time-memory trade-off variant of Becker–Joux–May–Meurer (BJMM) decoding, which solves QC-3138. These works have paved the way for the cryptanalysis of ISD in real-world scenarios.

In this work, we further advance the progress of the abovementioned studies by performing a concrete analysis of MMT decoding. We improve the list construction in MMT so that the number of both candidates and representations in the enumeration phase is increased without the need for additional time and memory. Our new algorithm is theoretically 5.1 times faster than the BJMM algorithm for Classic McEliece I instance. We achieve the minimum time complexity across all categories of Classic McEliece among all ISD algorithms. Moreover, compared with the BJMM algorithm, our MMT algorithm reduces the bit security by 1 to 3 bits for all code based NIST-PQC round 4 candidates. Practical security estimates confirm that all the candidates have sufficiently strong bit security, except for Classic McEliece III, with a 1-bit deficiency.

In addition, we implement our new MMT algorithm in a GPU environment and provide the new record of the McEliece-1409 instance, along with implementation details and experimental analyses. Our study verifies the practical reliability of the code-based candidates against current ISD algorithms.

**Keywords:** Information Set Decoding · Representation Technique · McEliece

## 1 Introduction

Code-based cryptography is a historic public-key encryption scheme based on coding theory. More than 40 years have passed since Robert McEliece first developed the McEliece cryptography in 1978 [27]. Despite its long history, code-based cryptography is receiving renewed attention today with the advent of quantum computers, as it is considered resistant to quantum attacks.

In the NIST post-quantum cryptography standardization project (NIST-PQC), three code-based cryptographic schemes — Classic McEliece, Bit Flipping Key Encapsulation (BIKE), and Hamming Quasi-Cyclic (HQC) — are undergoing continuous evaluation in the fourth round [30]. Among the four submissions in this round, Supersingular Isogeny Key Encapsulation (SIKE), an isogeny-based cryptography, has been deprecated due to a (classical) polynomial-time attack [9]. This situation emphasizes the urgent need for an extensive security assessment of the remaining code-based candidates.

Cryptanalysis on code-based cryptography can be divided into two types: *structural attacks* and *non-structural attacks*. The former exploits certain structures in a cryptography, such as information about the code used. Various attacks and their countermeasures have been reported thus far [10,19,23].

For non-structural attacks, the most efficient approach is known as Information Set Decoding (ISD), which solves the fundamental security assumption of code-based cryptography known as the Syndrome Decoding Problem (SDP). Such algorithms are built on the framework of Prange's algorithm [32]. Thus far, several ISD algorithms have been proposed (e.g., [4,7,12,18,25,26,35]), and their asymptotic complexity has been thoroughly investigated (see Table 1). In these papers, the majority of the focus is on full/half distance decoding setting, where the weight $w = O(n)$. In the full distance setting, we fix the relative weight to be $w/n = H^{-1}(1-k/n)$, which is derived from the Gilbert–Varshamov bound, and determine the worst-case complexity while varying the relative code rate $0 \leq k/n \leq 1$, where $H(\cdot)$ represents the binary entropy function. When $w = o(n)$, all the ISD algorithms exhibit the same asymptotic behavior [8].

**Table 1.** Asymptotic time complexity $2^{\alpha n}$ for major ISD algorithms in full distance decoding setting. The exponent $\alpha$ for each algorithm is listed below. A newer algorithm is located on the right.

| Prange | Dumer | MMT | BJMM | May-Ozerov | Both-May | Sieving ISD |
|---|---|---|---|---|---|---|
| 0.121 | 0.116 | 0.112 | 0.102 | 0.0953 | 0.0951 | 0.101 |

*Accurate Bit security Assessments for Code-based Cryptography* In addition to asymptotic complexity, several contributions have been made to provide precise security estimates for actual code-based cryptography [14,20,31]. In [15,17], the

authors showed how to compute bit security estimates of code-based cryptography from the actual decoding results of medium-sized instances, employing an extrapolation technique. Recently, Esser et al. introduced a comprehensive library for cryptographic hardness estimation [16], enabling us to estimate both bit security and the optimal parameters for a specified difficulty level of an input problem.

*Real-world Cryptanalysis through Decoding Challenges* Implementation-oriented research is also crucial in this field. Decrypting higher-dimensional cryptography provides more accurate security estimates, thus aiding in the determination of the appropriate key lengths for a cryptography while balancing security and efficiency. One known benchmark for code-based cryptography is Decoding Challenge [2].

In 2022, Esser and Zweydinger successfully solved a quasi-cyclic SDP resembling BIKE and HQC with parameters $n = 3138, k = 1569, w = 56$. The above authors employed the memory-optimized MMT/BJMM algorithm [17] along with the Decoding-One-Out-of-Many (DOOM) strategy [34]. In 2023, Bernstein, Lange, and Peters obtained an initial solution to a Classic McEliece-like SDP with $n = 1347, k = 1078, w = 25$. They utilized an improved variant [6] of Stern's ISD [35]. Narisada, Fukushima and Kiyomoto found a solution to the SDP for random binary linear codes for $n = 570, k = 285, w = 70$ using a GPU implementation of the MMT algorithm [29].

*Recent Asymptotic Improvements for ISD Algorithms* Several studies have clarified the asymptotic behavior of latest ISD algorithms. For example, Esser provided a corrected analysis for the Both–May algorithm [13]. In the case of full distance decoding, the asymptotic time complexity was revised from $2^{0.0885n}$ to $2^{0.0951n}$. Additionally, Ducas et al. revealed the asymptotic complexities of the Sieving ISD [11], while Esser and Zweydinger succeeded in reducing the asymptotic space complexity of the MMT algorithm from $2^{0.053n}$ to $2^{0.0375n}$ by demonstrating its time-memory trade-offs [17].

**Contributions** In this study, we focus on the MMT algorithm, the asymptotic time complexity $2^{0.112n}$ of which is worse than that of recently proposed ISDs. However, the former is often utilized in current decoding contest [15,17,29] due to its comparatively better performance. We propose a new MMT algorithm and rigorously examine the concrete computational complexity of the proposed method with the parameters used in the NIST-PQC round 4 candidates. Our contributions provide suggestions both theoretical and practical improvements.

*New MMT Algorithm as a Generalization of BJMM Algorithm* We conduct a concrete analysis of the MMT algorithm, considering polynomial terms that have been overlooked in asymptotic complexity. Specifically, we demonstrate a reduction of 19 bits in the bit security of Category 1 Classic McEliece from the original MMT algorithm. This reduction is achieved by appropriately analyzing

the behavior of the multiple weight distributions of binary vectors occurring in the list construction phase.

Furthermore, we introduce a technical strategy to increase the number of enumerated candidates in the list construction without significantly increasing time and memory complexities, resulting in an additional speedup of approximately 40%.

This precise analysis and technical improvement lead to a new variant of the MMT algorithm, which we refer to as the *revisited MMT algorithm.* The revisited MMT algorithm is theoretically 4.0 times faster than the time-memory trade-off variant of the BJMM algorithm and 3.0 times faster than the Both–May algorithm for Classic McEliece I. We demonstrate that this new MMT algorithm is, in fact, a generalization of the depth-2 BJMM algorithm, contrary to the notion that the BJMM algorithm is a generalization of the MMT algorithm. This provides a new perspective for both algorithms and closes the gap between MMT and BJMM decoding. Additionally, we illustrate that the revisited MMT algorithm exhibits time-memory trade-offs, as observed in [17].

As asymptotic contributions, we provide detailed analysis for Dumer's algorithm and the time-memory trade-off MMT with the Shamir–Schroeppel technique. We derive new space complexities for Dumer's algorithm as $2^{0.0177n}$ and for the time-memory trade-off MMT as $2^{0.0376n}$. Additionally, we present that both depth-2 BJMM and the revisited MMT have the same asymptotic complexity, with time $2^{0.105n}$ and memory $2^{0.0659n}$.

*Rigorous Cryptanalysis and Practical Results* We implement bit-security estimation for the revisited MMT algorithm on the `CryptographicEstimators` [16], the most recent library for bit security estimation for code-based and multivariate cryptography. It is derived that the revisited MMT algorithm exhibits a relatively small cost compared to other ISDs across all categories of Classic McEliece, BIKE, and HQC.

Notably, in all categories of Classic McEliece, the revisited MMT algorithm achieved the lowest time complexity among all ISD algorithms. Additionally, we perform bit-security estimates for Classic McEliece, BIKE, and HQC under more realistic memory constraints, i.e., with a logarithmic memory cost model, where an ISD with time $T$ and memory $M$ incurs a cost of $T \log_2 M$, and a maximum of one terabyte ($M = 2^{43}$) of memory capacity. The results, nevertheless, indicate that only Category 3 of Classic McEliece falls slightly below the required security level by just one bit. On the other hand, regarding the other categories of Classic McEliece, as well as BIKE and HQC, there is a sufficient security margin even considering the dispersion of runtime.

To validate the practicality of our algorithm, we implement the revisited MMT algorithm in a GPU environment[3], building on an openly accessible MMT implementation [29]. With this implementation and 10 desktop PCs, we achieve a notable milestone by successfully solving the McEliece-1409 instance. Finally, we present the *practical* minimal/maximal time of an ISD algorithm with a

[3] Our source code will be available on `https://github.com`.

parameter $\alpha$. We conclude that our decoding results, coupled with the minimal/maximal time complexity analysis, steadfastly support and contribute to reinforcing reliable security for code-based cryptography.

**Organization** The remainder of the paper is organized as follows. Section 2 describes the notation and Information Set Decoding. In Section 3, we discuss MMT and BJMM decoding. Section 4 presents our proposed algorithm. Section 5 conducts asymptotic complexity analyses for ISD algorithms using the Shamir–Schroeppel technique. Section 6 presents cryptanalysis for code-based NIST-PQC round 4 candidates against ISD algorithms. Experimental results and analyses are provided in Section 7. Finally, Section 8 gives concluding remarks.

## 2 Preliminaries

### 2.1 Notation

Let $\mathbb{F}_2$ be the finite field with elements $\{0, 1\}$. An $n$-dimensional column vector is denoted as $\mathbf{x}^\top = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$ for a row vector $\mathbf{x} \in \mathbb{F}_2^{1\times n}$. Henceforth, we denote a column vector without the transposition symbol $\top$ for simplicity. A concatenation of two vectors $\mathbf{a} \in \mathbb{F}_2^m$ and $\mathbf{b} \in \mathbb{F}_2^n$ is written as $(\mathbf{a}, \mathbf{b}) \in \mathbb{F}_2^{m+n}$ unless otherwise specified. Thus, we regard the product of two vector spaces $\mathbb{F}_2^m \times \mathbb{F}_2^n$ as the set of concatenated vectors $\mathbb{F}_2^{m+n}$. For more than two spaces, we consider product similarly. Let the zero vector be $\mathbf{0}$. A matrix of size $m \times n$ is denoted as $\mathbf{A} \in \mathbb{F}_2^{m\times n}$. Specifically, the identity matrix is represented as $\mathbf{I}$ and the zero matrix as $\mathbf{O}$. The Hamming weight for $\mathbf{x}$ is denoted by $\mathsf{wt}(\mathbf{x}) := |\{i \mid x_i = 1\}|$. The SDP is defined as follows.

**Definition 2.1 (Syndrome Decoding Problem: SDP).** For positive integers $n, k$ and $w$ such that $k \leq n$ and $w \leq n$, we consider a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}$ and a syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$. An SDP requires finding a vector $\mathbf{e} \in \mathbb{F}_2^n$ of $\mathsf{wt}(\mathbf{e}) = w$ such that $\mathbf{He} = \mathbf{s}$.

This problem has been shown to be in the NP-complete class [5], and code-based cryptography relies on this problem for its security. In this paper, we assume that an SDP has a unique solution, which is the case in the real use of code-based cryptography. Code-based cryptography is believed to be quantum secure, as no polynomial-time algorithm for solving the SDP has been found thus far.

### 2.2 Information Set Decoding

ISD is a probabilistic algorithm that can be used to solve an SDP in exponential time, as originated in Prange [32]. In the following, we provide a brief overview of a common framework for ISD algorithms.

Algorithm 1 below provides the pseudo-code for an ISD algorithm. Until Line 5, column permutation and Gaussian elimination are applied to the parity-check

matrix and syndrome, resulting in a systematic form $\bar{\mathbf{H}}$ and a corresponding syndrome $\bar{\mathbf{s}}$. We denote a set of all permuted solutions as $\mathcal{E}_0 = \mathcal{B}_w^n$, and a set of obtainable permuted solutions as $\mathcal{E}$, which varies across ISD algorithm. A matrix $\mathbf{P}$ is referred to as a *good permutation* when $\mathbf{Pe}$ is an element of $\mathcal{E}$. For $q := \Pr[\mathbf{P} \text{ is good}] = |\mathcal{E}|/|\mathcal{E}_0|$, we utilize a specific SEARCH component for $(\bar{\mathbf{H}}, \bar{\mathbf{s}})$, producing a permuted solution $\bar{\mathbf{e}}$ with probability $q$ at Line 6. By repeating the above procedure $q^{-1}$ times, it is expected that one solution $\mathbf{P}\bar{\mathbf{e}}$ is obtained.

**Algorithm 1:** INFORMATION SET DECODING

**Input:** $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}, \mathbf{s} \in \mathbb{F}_2^{n-k}, w \in \mathbb{N}$
**Output:** $\mathbf{e} \in \mathbb{F}_2^n$ s.t. $\mathbf{He} = \mathbf{s}$

1 $q := \Pr[\mathbf{P} \text{ is good}]$
2 **repeat** `/* q^-1 times in expectation */`
3  Pick random permutation matrix $\mathbf{P}$
4  $\bar{\mathbf{H}} = [\mathbf{I}_{n-k} \mid \hat{\mathbf{H}}] = \mathbf{GHP}$
5  $\bar{\mathbf{s}} = \mathbf{Gs}$
6  $\bar{\mathbf{e}} = \text{SEARCH}(\bar{\mathbf{H}}, \bar{\mathbf{s}})$
7  **if** $\mathsf{wt}(\bar{\mathbf{e}}) = w$ and $\bar{\mathbf{H}}\bar{\mathbf{e}} = \bar{\mathbf{s}}$ **then**
8  **return** $\mathbf{P}\bar{\mathbf{e}}$

An ISD algorithm exhibits an average time complexity given by

$$q^{-1}(T_{\text{ge}} + T_{\text{search}}), \tag{1}$$

where $T_{\text{ge}}$ is the time complexity for Gaussian elimination and $T_{\text{search}}$ is the time complexity required for the SEARCH component.

For instance, in the case of Prange's algorithm, the SEARCH component checks whether $\mathsf{wt}(\bar{\mathbf{s}}) \stackrel{?}{=} w$. If this equation is true, then it returns $\bar{\mathbf{e}} = (\bar{\mathbf{s}}, \mathbf{0})$. A crucial observation regarding this algorithm is that when, fortunately, $\mathsf{wt}(\bar{\mathbf{s}}) = w$, we have that $\bar{\mathbf{H}}(\bar{\mathbf{s}}, \mathbf{0}) = \bar{\mathbf{s}}$, which satisfies both conditions for a solution. It can be stated that $\mathcal{E} = \mathcal{B}_w^{n-k} \times \mathcal{B}_0^k$, and $q$ is given by

$$q = \frac{\binom{n-k}{w}}{\binom{n}{w}}. \tag{2}$$

Eq. (1) is instantiated with substitutions from Eq. (2), $T_{\text{ge}} = (n-k)^2 n$ and $T_{\text{search}} = 1$.

To date, many efforts have been made to develop more efficient ISD algorithms that minimize Eq. (1) from an asymptotic perspective. However, relying solely on asymptotic analysis has resulted in a gap between theoretical results and actual time complexity.

## 3 May–Meurer–Thomae Algorithm

The MMT algorithm is a practical ISD algorithm that achieves a smaller time complexity than Prange's and Dumer's algorithm. The inputs to the SEARCH component in the MMT algorithm are a semi-systematic form $\bar{\mathbf{H}}$ of the parity-check matrix and the syndrome $\bar{\mathbf{s}}$:

$$\bar{\mathbf{H}} = \begin{pmatrix} \mathbf{I}_{n-k-\ell} & \mathbf{H}_1 \\ \mathbf{O} & \mathbf{H}_2 \end{pmatrix} = \mathbf{GHP}, \ \bar{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2) = \mathbf{Gs} \in \mathbb{F}_2^{n-k-\ell} \times \mathbb{F}_2^{\ell}, \tag{3}$$

where $\mathbf{H}_1 \in \mathbb{F}_2^{(n-k-\ell)\times(k+\ell)}$ and $\mathbf{H}_2 \in \mathbb{F}_2^{\ell\times(k+\ell)}$. This transformation can be achieved by applying a column permutation $\mathbf{P}$ and Gaussian elimination $\mathbf{G}$ with early abort. In the SEARCH component, it performs the merging and filtering of several lists, each consisting of a fraction of the candidates for a solution $\bar{\mathbf{e}}$. The MMT algorithm outputs a permuted solution $\bar{\mathbf{e}} \in \left(\mathcal{B}_{w-2p}^{n-k-\ell} \times \mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2}\right)$.

### 3.1 Tree-based List Construction of the Depth-2 MMT Algorithm

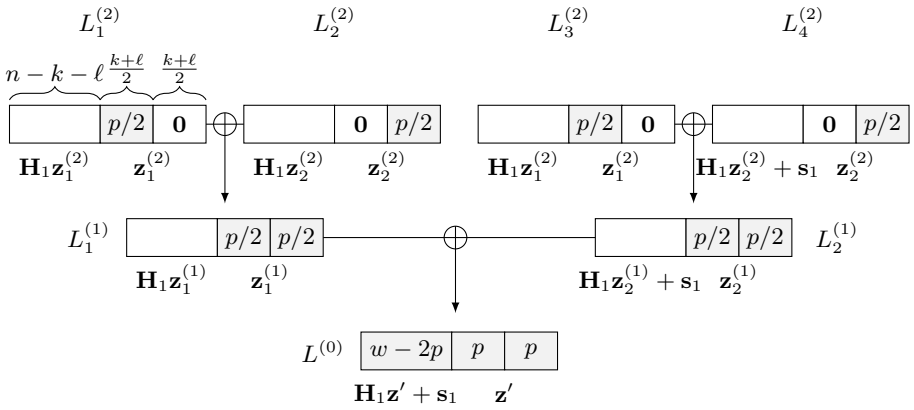


**Fig. 1.** Tree-based list construction of the depth-2 MMT algorithm.

We describe the list construction process in the standard depth-2 MMT algorithm, which has the minimal asymptotic time among all depths and has often been employed in recent decoding challenge contests [15,17,29]. The output list is $L^{(2)}$ consisting of $\mathbf{z}'$, which satisfies $\mathsf{wt}(\mathbf{z}') = 2p$ and $\mathbf{H}_2\mathbf{z}' = \mathbf{s}_2$. We traverse seven lists from the bottom (depth-2) to the top (depth 0), as depicted in Figure 1. First, four depth-2 base lists are prepared as follows:

$$L_1^{(2)} = L_3^{(2)} = \left\{ \mathbf{z}_1^{(2)} \in \mathbb{F}_2^{\frac{k+\ell}{2}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_1^{(2)}) = p/2 \right\},$$
$$L_2^{(2)} = L_4^{(2)} = \left\{ \mathbf{z}_2^{(2)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_2^{(2)}) = p/2 \right\}.$$

Then, we merge $L_1^{(2)}$ with $L_2^{(2)}$ ($L_3^{(2)}$ with $L_4^{(2)}$) to yield a depth-1 list $L_1^{(1)}$ ($L_2^{(1)}$) while filtering a pair $(\mathbf{z}_1^{(2)}, \mathbf{z}_2^{(2)})$, based on the following condition with an integer $\ell_1 \leq \ell$ and a map $\pi_{\ell_1} : \mathbb{F}_2^{\ell} \to \mathbb{F}_2^{\ell_1}, \pi_{\ell_1}(x_1, ..., x_\ell) = (x_1, ..., x_{\ell_1})$:

$$L_1^{(1)} = \left\{ \mathbf{z}_1^{(1)} \;\middle|\; \mathbf{z}_1^{(2)} \in L_1^{(2)}, \mathbf{z}_2^{(2)} \in L_2^{(2)}, \mathbf{z}_1^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)}, \pi_{\ell_1}(\mathbf{H}_2 \mathbf{z}_1^{(1)}) = \mathbf{0} \right\},$$
$$L_2^{(1)} = \left\{ \mathbf{z}_2^{(1)} \;\middle|\; \mathbf{z}_1^{(2)} \in L_3^{(2)}, \mathbf{z}_2^{(2)} \in L_4^{(2)}, \mathbf{z}_2^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)}, \pi_{\ell_1}(\mathbf{H}_2 \mathbf{z}_2^{(1)} + \mathbf{s}_2) = \mathbf{0} \right\}.$$

Note that $\mathsf{wt}(\mathbf{z}_1^{(1)}) = \mathsf{wt}(\mathbf{z}_2^{(1)}) = p$, since there is no overlap at the 1's position between $\mathbf{z}_1^{(2)}$ and $\mathbf{z}_2^{(2)}$. It may be helpful to consider that a filtering with probability $2^{-\ell_1}$ is applied to the Cartesian product $L_1^{(2)} \times L_2^{(2)}$. Then, $L_1^{(1)}$ and $L_2^{(1)}$ are merged under a specific condition to yield a list $L^{(0)}$:

$$L^{(0)} = \left\{ \mathbf{z}' \;\middle|\; \mathbf{z}_1^{(1)} \in L_1^{(1)}, \mathbf{z}_2^{(1)} \in L_2^{(1)}, \mathbf{z}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}, \mathbf{H}_2 \mathbf{z}' = \mathbf{s}_2 \right\}. \tag{4}$$

Since we already have that $\pi_{\ell_1}(\mathbf{H}_2(\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)})) = \pi_{\ell_1}(\mathbf{s}_2)$, we can consider performing filtering with probability $2^{\ell_1 - \ell}$ for $L_1^{(1)} \times L_2^{(1)}$. Now, if we set $\mathbf{z}'' = \mathbf{H}_1 \mathbf{z}' + \mathbf{s}_1$, then the first condition of a solution, $\bar{\mathbf{H}}(\mathbf{z}'', \mathbf{z}') = (\mathbf{s}_1, \mathbf{H}_2 \mathbf{z}') = \bar{\mathbf{s}}$ is satisfied. To verify the weight condition, we need to examine the distribution of 1's in $(\mathbf{z}'', \mathbf{z}')$. Namely, $(\mathbf{z}'', \mathbf{z}') \overset{?}{\in} \left( \mathcal{B}_{w-2p}^{n-k-\ell} \times \mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2} \right)$. Fortunately, when $\mathsf{wt}(\mathbf{z}'') = w - 2p$, we observe that $\mathsf{wt}(\bar{\mathbf{z}}) = w$ for $\bar{\mathbf{z}} = (\mathbf{z}'', \mathbf{z}')$. Thus, $\mathbf{P}\bar{\mathbf{z}}$ is a solution to the SDP.

### 3.2 Computational Complexity of the MMT Algorithm

We provide an overview of the time and space complexity analysis of the MMT algorithm conducted in [14,17,25]. The time complexity per iteration of the **repeat** in Algorithm 1 for the MMT algorithm is dominated by the time complexity for Gaussian elimination $T_{\mathrm{ge}} = (n-k)^2 n$ and $T_{\mathrm{search}}$ required for the SEARCH component. $T_{\mathrm{search}}$ is the sum of the time complexities for base list construction and for the merging of lists at each depth. For $|L^{(2)}| = \binom{(\ell+k)/2}{p/2}$ and $|L^{(1)}| = \max(1, 2^{-\ell_1}|L^{(2)}|^2)$, we obtain that

$$T_{\mathrm{search}} = 2|L^{(2)}| + 2\max(|L^{(2)}|, 2^{-\ell_1}|L^{(2)}|^2 + \max(|L^{(1)}|, 2^{-\ell+\ell_1}|L^{(1)}|^2). \tag{5}$$

Note that we do not concern ourselves with space complexity for $L^{(0)}$, as we can enumerate $\mathbf{z}'$ directly from $L_1^{(1)}$ and $L_2^{(1)}$ without constructing an actual list. In the MMT algorithm, the set of obtainable permuted solutions is $\mathcal{E} = \mathcal{B}_{w-2p}^{n-k-\ell} \times \mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2}$, and $q$ is given by

$$q = \frac{\binom{n-k-\ell}{w-2p}\binom{(k+\ell)/2}{p}^2}{\binom{n}{w}}. \tag{6}$$

Therefore, the average time complexity of the MMT algorithm is given by Eq. (1) instantiated with $T_{\mathrm{ge}} = (n-k)^2 n$, Eq. (5) and Eq. (6). The space complexity of the MMT algorithm is

$$(n-k)n + 2|L^{(2)}| + 2\max(1, 2^{-\ell_1}|L^{(2)}|^2). \tag{7}$$

In practice, we search for a valid integer parameter set $(p, \ell, \ell_1)$ to minimize the time complexity. To efficiently find it, several optimizers, referred to as *ISD Estimators*, have been proposed (e.g., [14,16]). In particular, the parameter $\ell_1$ must be chosen carefully as it is related to the *representations*.

A (split) representation of a weight-$\omega_1$ vector $\mathbf{z} \in \mathbb{F}_2^n$ is a pair of vectors $(\mathbf{z}_1, \mathbf{z}_2) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$, satisfying $\mathbf{z} = \mathbf{z}_1 + \mathbf{z}_2$ and $\mathsf{wt}(\mathbf{z}_1) = \mathsf{wt}(\mathbf{z}_2) = \omega_2 \geq \omega_1/2$. In the MMT algorithm, the number of representations for a weight-$2p$ $\mathbf{z}'$ as a sum of two weight-$p$ vectors $\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)}$ is

$$R = \binom{p}{p/2}^2. \tag{8}$$

In [25], valid parameters are searched under the condition that at least a single representation of a solution is expected to be contained in $L^{(0)}$, i.e., $\ell_1 \leq \log_2 R$. In [17], the authors consider the case where $\ell_1 > \log_2 R$. When $\ell_1 > \log_2 R$, the probability $\rho_{\mathrm{repr}}$ of at least one representation being included in $L^{(0)}$ is given by

$$\rho_{\mathrm{repr}} := 1 - (1 - 2^{-\ell_1})^R \approx 2^{-\ell_1} R. \tag{9}$$

They demonstrate that the decrease in the number of representations can be compensated by repeating the Search component ${\rho_{\mathrm{repr}}}^{-1}$ times. To avoid exploring the same space for each round, the constraint in $L_1^{(1)}$ and $L_2^{(1)}$ are slightly modified to $\pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{t}$ and $\pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_2^{(1)} + \mathbf{s}_2) = \mathbf{t}$, respectively. Here, $\mathbf{t} \in \mathbb{F}_2^{\ell_1}$ is a randomly chosen vector for each round. The time complexity for the MMT algorithm in the case of $\ell_1 > \log_2 R$ is given by

$$q^{-1}(T_{\mathrm{ge}} + {\rho_{\mathrm{repr}}}^{-1} T_{\mathrm{search}}).$$

The advantage of setting $\ell_1 > \log_2 R$ is to show a time-memory trade-off for cases where $\ell_1 \leq \log_2 R$, which implies that a portion of the space complexity required in the Search component can be offset by additional time complexity. Adopting a relatively large $\ell_1$ can also result in practical reductions in actual runtime, as indicated in [17,29].

### 3.3 Becker–Joux–May–Meurer Algorithm

The BJMM algorithm is a generalization of the MMT algorithm. The algorithm introduces an additional weight parameter, $p' \geq p/2 \in \mathbb{N}$, for fine-grained analysis, which is utilized in the construction of base lists:

$$L_1^{(2)} = L_3^{(2)} = \left\{ \mathbf{z}_1^{(2)} \in \mathbb{F}_2^{\frac{k+\ell}{2}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_1^{(2)}) = p' \right\},$$
$$L_2^{(2)} = L_4^{(2)} = \left\{ \mathbf{z}_2^{(2)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_2^{(2)}) = p' \right\}.$$

When merging depth-1 lists, we consider representations for a vector, $\mathbf{z}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$, such that $\mathbf{z}' \in \left(\mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2}\right)$. Now, both $\mathbf{z}_1^{(1)}$ and $\mathbf{z}_2^{(1)}$ have the same 1 distributions, i.e., $\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)} \in \left(\mathcal{B}_{p'}^{(k+\ell)/2} \times \mathcal{B}_{p'}^{(k+\ell)/2}\right)$. Instead of Eq. (8), the number of such representations is given by

$$R = \binom{p}{p/2}^2 \binom{(k+\ell)/2 - p}{p' - p/2}^2. \tag{10}$$

Choosing $p' > p/2$ increases the size of the base lists to $\binom{(k+\ell/2)}{p'}$. However, this approach significantly increases the number of representations, leading to a reduction in time complexity compared to that of the MMT algorithm. In practice, the MMT algorithm with $p/2 = 1$ has been used to solve McEliece-1223 and McEliece-1284 as reported in [15]. While the BJMM algorithm with $p' = 2, 3 > p/2$ faces the challenge of large memory consumption, often reaching several gigabytes, it is possible to reduce memory usage for the BJMM algorithm by employing the time-memory trade-off technique described in the previous subsection [17].

## 4 Revisited MMT Algorithm

In this section, we present a precise analysis of the MMT algorithm, with a particular focus on the (disjoint) weight distribution of a vector $\mathbf{z}'$ in the final list $L^{(0)}$. Then, we show how to properly modify the algorithm and estimator to increase the number of candidates enumerated in the search-tree construction while not significantly increasing computational time and memory, resulting in a reduction in overall runtime. Finally, we state that our enhanced MMT algorithm is a generalization of the depth-2 BJMM algorithm.

### 4.1 Disjoint Weight Distribution for Solution Candidates

Previously, the MMT and BJMM algorithms considered a specific 1's distribution of $\mathbf{Pe}$, i.e., $\mathcal{B}_{w-2p}^{n-k-\ell} \times \mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2}$. However, we observe that unless explicitly excluded, the final list $L^{(0)}$ may contain a fraction of a solution with a weight distribution different from the specific 1's distribution. The following proposition shows the probability that a fraction of the solution $\mathbf{e}'$ with a specific distribution is included in $L^{(0)}$.

**Proposition 4.1 (existence probability of a solution fraction in $L^{(0)}$).** *Assuming that a good permutation $\boldsymbol{P}$ permutes the solution as $\boldsymbol{Pe} = (\boldsymbol{e}'', \boldsymbol{e}')$, where $i, j$ be even integers between $0 \le i, j \le p$ and $\boldsymbol{e}' \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$. Then, for the fraction of the permuted solution $\boldsymbol{e}'$ and the final list $L^{(0)}$ defined in Eq. (4), the following equation is satisfied:*

$$\Pr\left[\boldsymbol{e}' \in L^{(0)}\right] = \rho_{i,j}, \tag{11}$$

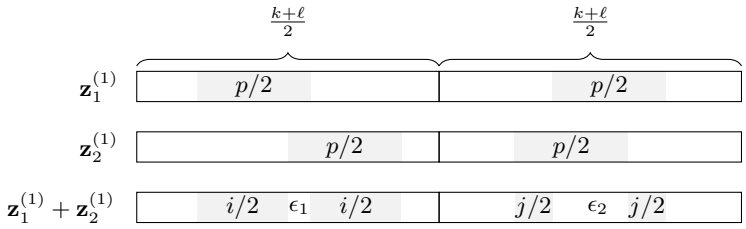


**Fig. 2.** An example of a $\mathbb{F}_2$-addition of two vectors $\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)} \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$, where $\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)} \in \left(\mathcal{B}_{p/2}^{(k+\ell)/2} \times \mathcal{B}_{p/2}^{(k+\ell)/2}\right)$. The gray region represents a one vector, while the white area represents a zero vector. There are $\epsilon_1$ (resp. $\epsilon_2$) duplicates of 1's position between $\mathbf{z}_1^{(1)}$ and $\mathbf{z}_2^{(1)}$ on the left (resp. right) interval.

*where*

$$\rho_{i,j} := \begin{cases} 1-(1-2^{-2\ell_1})^{R_0^2} & (i=j=0), \\ 1-(1-2^{-\ell_1})^{R_i R_j} & (otherwise), \end{cases}$$
$$R_i := \binom{i}{i/2}\binom{(k+\ell)/2-i}{p/2-i/2}.$$

*Proof.* The idea behind this proof is to leverage a property of $\mathbb{F}_2$-addition, i.e., $1+1=0$, as previously described in the articles that address such *extended representations* [3,4].

For $\mathbf{z}_1^{(1)} \in \left(\mathcal{B}_{p/2}^{(k+\ell)/2} \times \mathcal{B}_{p/2}^{(k+\ell)/2}\right)$ in $L_1^{(1)}$ and $\mathbf{z}_2^{(1)} \in \left(\mathcal{B}_{p/2}^{(k+\ell)/2} \times \mathcal{B}_{p/2}^{(k+\ell)/2}\right)$ in $L_2^{(1)}$, let $\epsilon_1$ (resp. $\epsilon_2$) be the number of duplicates in 1's position between $\mathbf{z}_1^{(1)}$ and $\mathbf{z}_2^{(1)}$ on the left (resp. right) interval. Now, $\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$ is in $\mathcal{B}_{p-2\epsilon_1}^{(k+\ell)/2} \times \mathcal{B}_{p-2\epsilon_2}^{(k+\ell)/2}$, as depicted in Figure 2. Given that $0 \le \epsilon_1, \epsilon_2 \le p/2$, it is shown that $\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)} \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$, through substitutions $i = p - 2\epsilon_1$ and $j = p - 2\epsilon_2$. Note that both $i$ and $j$ are even numbers satisfying $0 \le i, j \le p$ since $p, 2\epsilon_1, 2\epsilon_2$ are all even numbers.

Next, we derive the value of the following conditional probability

$$\Pr\left[\mathbf{e}' \in L^{(0)} \;\middle|\; \mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}, \mathbf{z}_1^{(1)} \in L_1^{(1)}, \mathbf{z}_2^{(1)} \in L_2^{(1)}\right] \tag{12}$$

for $\mathbf{e}' \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$ by considering two cases: (i) $i = j = 0$ and (ii) otherwise. For case (i), we have $\mathbf{z}_1^{(1)} = \mathbf{z}_2^{(1)}$ since $\mathbf{e}' = \mathbf{0}$. From the constraints in depth-1 lists, $\pi_{\ell_1}(\mathbf{s}_2) = \mathbf{0}$ is obtained. Hence, $L^{(0)}$ includes $\mathbf{e}'$ only when both $X : \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{0}$ and $Y : \pi_{\ell_1}(\mathbf{s}_2) = \mathbf{0}$ are satisfied. Given that $\Pr[X] = 2^{-\ell_1}$ and $\Pr[Y] = 2^{-\ell_1}$, as both $\mathbf{H}_2\mathbf{z}_1^{(1)}$ and $\mathbf{s}_2$ are randomly chosen, Eq. (12) is the joint probability of $X$ and $Y$, i.e., $2^{-2\ell_1}$.

For case (ii), we have $\mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$ and $\mathbf{z}_1^{(1)} \neq \mathbf{z}_2^{(1)}$ since $\mathbf{e}' \neq \mathbf{0}$. We must consider the joint probability of $X : \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{0}$ and $Z : \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_2^{(1)} + \mathbf{s}_2) =$

$\mathbf{0}$. Since $\mathbf{e}'$ is a fraction of the solution, if either $X$ or $Z$ holds true, the remaining automatically holds true. Hence, Eq. (12) is $2^{-\ell_1}$.

Recall that we have representations for $\mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$. Let $R_i$ be the number of representations of a vector $\mathbf{a} = \mathbf{b} + \mathbf{c}$, where $\mathbf{a} \in \mathcal{B}_i^{(k+\ell)/2}$ and $\mathbf{b}, \mathbf{c} \in \mathcal{B}_{p/2}^{(k+\ell)/2}$. The set of 1-coordinates in $\mathbf{a}$ can be split in $\binom{i}{i/2}$ ways as $1 = 1{+}0$ or $1 = 0{+}1$. For each split representation, the set of 0-coordinates can be split in $\binom{(k+\ell)/2-i}{\epsilon_1}$ ways by $0 = 1{+}1$. In total, $\mathbf{a}$ has $\binom{i}{i/2}\binom{(k+\ell)/2-i}{\epsilon_1}$ representations. For $\mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$, we have $\binom{i}{i/2}\binom{(k+\ell)/2-i}{\epsilon_1}\binom{j}{j/2}\binom{(k+\ell)/2-j}{\epsilon_2}$ representations. Hence,

$$\Pr\left[\mathbf{e}' \in L^{(0)} \;\middle|\; \mathbf{e}' \neq \mathbf{0}\right] = 1 - (1 - 2^{-\ell_1})^{R_i R_j}.$$

Similarly, we can show that $\Pr\left[\mathbf{e}' \in L^{(0)} \;\middle|\; \mathbf{e}' = \mathbf{0}\right] = 1 - (1 - 2^{-2\ell_1})^{R_0^2}$, which corresponds to Eq. (11). □

Assuming $2^{-\ell_1} \ll 1$, Eq. (12) is approximated by $1 - (1 - 2^{-\ell_1})^{R_i R_j} \approx \min(1, 2^{-\ell_1} R_i R_j)$ for $i = j \neq 0$ by series expansion (analogously, for $i = j = 0$). Now, we are almost ready to develop a fine-grained MMT algorithm designed to handle multiple distributions. Before delving into this chapter, we show how we allow the MMT algorithm to address odd weight distributions.

### 4.2 Multi-weight Initialization

We aim to generalize Proposition 4.1 from even distributions to encompass all distributions. To achieve this, we introduce a technique called the *multi-weight initialization*.

Now, we generalize the weight constraint for base lists by enumerating all vectors whose weights are less than or equal to $p/2$, instead of specifically enumerating weight-$p/2$ vectors, as follows:

$$\bar{L}_1^{(2)} = \bar{L}_3^{(2)} = \left\{ \mathbf{z}_1^{(2)} \in \mathbb{F}_2^{\frac{k+\ell}{2}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; 0 \leq \mathsf{wt}(\mathbf{z}_1^{(2)}) \leq p/2 \right\},$$
$$\bar{L}_2^{(2)} = \bar{L}_4^{(2)} = \left\{ \mathbf{z}_2^{(2)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{k+\ell}{2}} \;\middle|\; 0 \leq \mathsf{wt}(\mathbf{z}_2^{(2)}) \leq p/2 \right\}.$$

This approach leads to an increase in the base list size,

$$|\bar{L}^{(2)}| = \sum_{0 \leq i \leq p/2} \binom{(k+\ell)/2}{p/2 - i},$$

followed by the list $\bar{L}_1^{(1)}, \bar{L}_2^{(1)}$ with a common length of $|\bar{L}^{(1)}| = \max(1, 2^{-\ell_1}|\bar{L}^{(2)}|^2)$ and $\bar{L}^{(0)}$ with an actual length of 0. However, this approach provides practical advantages for the MMT algorithm not only by increasing the number of odd-weight candidates in the final list $\bar{L}^{(0)}$, but also by expanding the representations of even-weight candidates. By doing so, Proposition 4.1 is extended as follows.

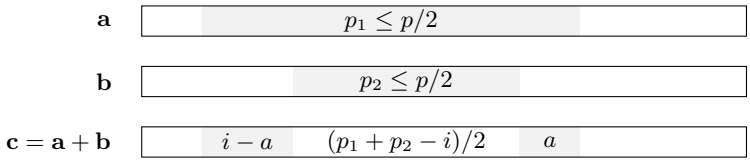


**Fig. 3.** An example of $\mathbb{F}_2$-addition that yields a weight-$i$ vector $\mathbf{c}$ from a weight-$p_1$ vector $\mathbf{a}$ and a weight-$p_2$ vector $\mathbf{b}$. We have $(p_1+p_2-i)/2$ positions of 1's duplicated between $\mathbf{a}$ and $\mathbf{b}$. In this example, we have $i-a$ ones on the left side of $\mathbf{c}$ and $a$ ones on the right side.

**Proposition 4.2 (multi-weight initialization).** *Assuming that a good permutation $\boldsymbol{P}$ permutes the solution as $\boldsymbol{P}\boldsymbol{e} = (\boldsymbol{e}'', \boldsymbol{e}')$, where $i, j$ be integers between $0 \le i, j \le p$ and $\boldsymbol{e}' \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$. Then, for the fraction of the permuted solution $\boldsymbol{e}'$ and the final list $\bar{L}^{(0)}$, the following equation is satisfied:*

$$\Pr\left[\boldsymbol{e}' \in \bar{L}^{(0)}\right] = \bar{\rho}_{i,j}, \tag{13}$$

*where*

$$\bar{\rho}_{i,j} := \begin{cases} 1-(1-2^{-2\ell_1})^{\bar{R}_0^2} & (i=j=0), \\ 1-(1-2^{-\ell_1})^{\bar{R}_i\bar{R}_j} & (otherwise), \end{cases}$$

$$\bar{R}_i := \sum_{(p_1,p_2)\in\mathcal{P}_i} \binom{i}{\lfloor i/2 \rfloor} \binom{(k+\ell)/2-i}{(p_1+p_2-i)/2}, \tag{14}$$

$$\mathcal{P}_i := \left\{ (p_1,p_2) \;\middle|\; p_1,p_2 \le \frac{p}{2}, |p_1-p_2| \le i \le p_1+p_2, p_1+p_2 \equiv i \bmod 2 \right\}. \tag{15}$$

*Proof.* Like Proposition 4.1, we enumerate the possible 1's distributions of $\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$ for $\mathbf{z}_1^{(1)} \in \left(\mathcal{B}_{p_1}^{(k+\ell)/2} \times \mathcal{B}_{p_1'}^{(k+\ell)/2}\right)$ in $\bar{L}_1^{(1)}$ and $\mathbf{z}_2^{(1)} \in \left(\mathcal{B}_{p_2}^{(k+\ell)/2} \times \mathcal{B}_{p_2'}^{(k+\ell)/2}\right)$ in $\bar{L}_2^{(1)}$, where $0 \le p_1, p_1', p_2, p_2' \le p/2$. let $\epsilon_1$ (resp. $\epsilon_2$) be the number of duplicates in 1's position between $\mathbf{z}_1^{(1)}$ and $\mathbf{z}_2^{(1)}$ on the left (resp. right) interval. Now, $\mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$ lies in $\mathcal{B}_{p_1+p_2-2\epsilon_1}^{(k+\ell)/2} \times \mathcal{B}_{p_1'+p_2'-2\epsilon_2}^{(k+\ell)/2}$. If we set $i = p_1+p_2-2\epsilon_1$, a set of valid pairs $(p_1,p_2)$ for $i$ is determined by solving the equation, $i+2\epsilon_1 = p_1+p_2$ varying $0 \le \epsilon_1 \le \min(p_1,p_2)$, which yields a set defined in Eq. (15). Similarly, we obtain $\mathcal{P}_j$ for the case of $j = p_1'+p_2'-2\epsilon_2$. Hence, $\mathbf{z}_1^{(1)}+\mathbf{z}_2^{(1)}$ is in $\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}$ with $(p_1,p_2) \in \mathcal{P}_i$ and $(p_1',p_2') \in \mathcal{P}_j$.

The following conditional probability is equivalent to Eq. (12):

$$\Pr\left[\mathbf{e}' \in \bar{L}^{(0)} \;\middle|\; \mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}, \mathbf{z}_1^{(1)} \in \bar{L}_1^{(1)}, \mathbf{z}_2^{(1)} \in \bar{L}_2^{(1)}\right].$$

Again, we need to count the number of representations for $\mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$. Let $R_{i,p_1,p_2}$ be the number of representations of a vector $\mathbf{a} = \mathbf{b} + \mathbf{c}$, where

$\mathbf{a} \in \mathcal{B}_i^{(k+\ell)/2}$, $\mathbf{b} \in \mathcal{B}_{p_1}^{(k+\ell)/2}$ and $\mathbf{c} \in \mathcal{B}_{p_2}^{(k+\ell)/2}$, as depicted in Figure 3. The set of 1-coordinates in $\mathbf{a}$ can be split in $\binom{i}{\lfloor i/2 \rfloor}$ ways as $1 = 1 + 0$ or $1 = 0 + 1$, where $\lfloor i/2 \rfloor$ is obtained by considering the case in which $i$ is an odd integer. For each split representation, the set of 0-coordinates can be split in $\binom{(k+\ell)/2-i}{(p_1+p_2-i)/2}$ ways by $0 = 1 + 1$. In total, $\mathbf{a}$ has $R_{i,p_1,p_2} = \binom{i}{\lfloor i/2 \rfloor}\binom{(k+\ell)/2-i}{(p_1+p_2-i)/2}$ representations.

For the left interval of $\mathbf{e}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$, we need to consider representations $R_{i,p_1,p_2}$ for each valid pair $(p_1, p_2) \in \mathcal{P}_i$ for a fixed $i$. Since these representations are mutually exclusive, we can sum the representations $R_{i,p_1,p_2}$ for each pair $(p_1, p_2) \in \mathcal{P}_i$, which corresponds to Eq. (14). By considering the interval on the right half, the total number of representation becomes $\bar{R}_i \bar{R}_j$. Hence, we can state that

$$\Pr\left[\mathbf{e}' \in \bar{L}^{(0)} \;\middle|\; \mathbf{e}' \neq \mathbf{0}\right] = 1 - (1 - 2^{-\ell_1})^{\bar{R}_i \bar{R}_j}.$$

Similarly, $\Pr\left[\mathbf{e}' \in \bar{L}^{(0)} \mid \mathbf{e}' = \mathbf{0}\right] = 1 - (1 - 2^{-2\ell_1})^{\bar{R}_0^2}$ is satisfied, which concludes the proof. □

Assuming $2^{-\ell_1} \ll 1$, Eq. (13) is approximated by $1 - (1 - 2^{-\ell_1})^{\bar{R}_i \bar{R}_j} \approx \min(1, 2^{-\ell_1} \bar{R}_i \bar{R}_j)$ for $i = j \neq 0$ by series expansion (analogously, for $i = j = 0$).

### 4.3 Algorithm and Computational Complexity

We incorporate Proposition 4.2 into MMT decoding and propose our revisited MMT algorithm. The revisited MMT algorithm is described in Algorithm 2 below.

First, we construct base lists using the multi-weight initialization. For these lists, depth-1 lists $\bar{L}_1^{(1)}$ and $\bar{L}_2^{(1)}$ are computed. We then check the final list $\bar{L}^{(0)} \subset \bar{L}_1^{(1)} \times \bar{L}_2^{(1)}$ to determine whether it contains a fraction of a permuted solution $\mathbf{e}' \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$ for each $i$ and $j$. To check this condition for each element in $\mathbf{z}' \in \bar{L}^{(0)}$ in a constant time, we simply need to check $\mathsf{wt}(\bar{\mathbf{z}}) \stackrel{?}{=} w$, instead of $\mathsf{wt}(\mathbf{z}'') \stackrel{?}{=} w - 2p$, where $\bar{\mathbf{z}} = (\mathbf{z}'', \mathbf{z}')$ and $\mathbf{z}'' = \mathbf{H}_1 \mathbf{z}' + \mathbf{s}_1$. If $\mathsf{wt}(\bar{\mathbf{z}}) = w$, then we can state that $\bar{\mathbf{z}} \in \left(\mathcal{B}_{w-i-j}^{n-k-\ell} \times \mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$ is a permuted solution for certain $i$ and $j$ with $0 \leq i, j \leq p$, as $\bar{\mathbf{H}}\bar{\mathbf{z}} = \bar{\mathbf{s}}$ is satisfied. The algorithm returns $\mathbf{P}\bar{\mathbf{z}}$ as the solution of the SDP.

The set of obtainable permuted solutions $\mathcal{E}$ for our proposed algorithm is given by

$$\mathcal{E} = \bigcup_{0 \leq i,j \leq p} \mathcal{C}_{i,j},$$

where $\mathcal{C}_{i,j} := \mathcal{B}_{w-i-j}^{n-k-\ell} \times \mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}$ and $|\mathcal{C}_{i,j}| = \binom{n-k-\ell}{w-i-j}\binom{(k+\ell)/2}{i}\binom{(k+\ell)/2}{j}$, instead of $|\mathcal{E}| = \binom{n-k-\ell}{w-2p}\binom{(k+\ell)/2}{p}^2$ in the MMT algorithm. From Proposition 4.2,

**Algorithm 2:** MMT-Revisited

**Input:** $\mathbf{H} \in \mathbb{F}_2^{(n-k)\times n}, \mathbf{s} \in \mathbb{F}_2^{n-k}, w \in \mathbb{N}$
**Output:** $\mathbf{e} \in \mathbb{F}_2^n$ s.t. $\mathbf{He} = \mathbf{s}$

Choose optimal $\ell, \ell_1, p$
$\mathcal{P}_i := \left\{(p_1,p_2) \;\middle|\; p_1,p_2 \le \frac{p}{2}, |p_1-p_2| \le i \le p_1+p_2, p_1+p_2 \equiv i \bmod 2\right\}$
$\bar{R}_i := \sum_{(p_1,p_2)\in\mathcal{P}_i} \binom{i}{\lfloor i/2 \rfloor}\binom{(k+\ell)/2-i}{(p_1+p_2-i)/2}$
$\bar{\rho}_{i,j} := \begin{cases} 1-(1-2^{-2\ell_1})^{\bar{R}_0^2} & (i=j=0) \\ 1-(1-2^{-\ell_1})^{\bar{R}_i\bar{R}_j} & (otherwise) \end{cases}$
$q := \binom{n}{w}^{-1} \sum_{0\le i,j\le p} \binom{n-k-\ell}{w-i-j}\binom{(k+\ell)/2}{i}\binom{(k+\ell)/2}{j}\bar{\rho}_{i,j}$
**repeat** `/* ` $q^{-1}$ `times in expectation */`
  Pick random permutation matrix $\mathbf{P}$
  $\bar{\mathbf{H}} = \begin{pmatrix} \mathbf{I}_{n-k-\ell} & \mathbf{H}_1 \\ \mathbf{O} & \mathbf{H}_2 \end{pmatrix} = \mathbf{GHP}$
  $\bar{\mathbf{s}} = (\mathbf{s}_1, \mathbf{s}_2) = \mathbf{Gs}$
  Compute
    $\bar{L}_1^{(2)} = \bar{L}_3^{(2)} = \{\mathbf{z}_1^{(2)} \in \mathbb{F}_2^{\frac{k+\ell}{2}} \times 0^{\frac{k+\ell}{2}} \mid 0 \le \mathsf{wt}(\mathbf{z}_1^{(2)}) \le p/2\}$
    $\bar{L}_2^{(2)} = \bar{L}_4^{(2)} = \{\mathbf{z}_2^{(2)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{k+\ell}{2}} \mid 0 \le \mathsf{wt}(\mathbf{z}_2^{(2)}) \le p/2\}$
  Compute
    $\bar{L}_1^{(1)} = \{\mathbf{z}_1^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)} \mid \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{0}\}$ from $\bar{L}_1^{(2)}$ and $\bar{L}_2^{(2)}$
    $\bar{L}_2^{(1)} = \{\mathbf{z}_2^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)} \mid \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)} + \mathbf{s}_2) = \mathbf{0}\}$ from $\bar{L}_3^{(2)}$ and $\bar{L}_4^{(2)}$
  Compute $\bar{L}^{(0)} = \{\mathbf{z}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)} \mid \mathbf{H}_2\mathbf{z}' = \mathbf{0}\}$ from $\bar{L}_1^{(1)}$ and $\bar{L}_2^{(1)}$
  **for** $\mathbf{z}' \in \bar{L}^{(0)}$ **do**
    $\bar{\mathbf{z}} = (\mathbf{H}_1\mathbf{z}' + \mathbf{s}_1, \mathbf{z}')$
    **if** $\mathsf{wt}(\bar{\mathbf{z}}) = w$ **then**
      **return** $\mathbf{P}\bar{\mathbf{z}}$

we expect to have

$$|\mathcal{C}_{i,j}| \cdot \Pr\left[\mathbf{e}' \in \bar{L}^{(0)} \;\middle|\; \mathbf{e}' \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)\right]$$

obtainable permuted solutions for each pair $(i,j)$. Hence, $q := \Pr[\mathbf{P} \text{ is good}]$ is given by

$$q = \frac{\sum_{0\le i,j\le p} |\mathcal{C}_{i,j}|\bar{\rho}_{i,j}}{\binom{n}{w}}. \tag{16}$$

For the time complexity, we obtain

$$T_{\text{search}} = 2|\bar{L}^{(2)}| + 2\max(|\bar{L}^{(2)}|, 2^{-\ell_1}|\bar{L}^{(2)}|^2) + \max(|\bar{L}^{(1)}|, 2^{-\ell+\ell_1}|\bar{L}^{(1)}|^2), \tag{17}$$

where, $|\bar{L}^{(2)}| = \sum_{0\le i\le p/2}\binom{(\ell+k)/2}{p/2-i}$ and $|\bar{L}^{(1)}| = \max(1, 2^{-\ell_1}|\bar{L}^{(2)}|^2)$. The average time complexity of the revisited MMT algorithm is described by Eq. (1) with

instantiations of Eq. (16), Eq. (17) and $T_{\mathrm{ge}} = (n-k)^2 n$. Space complexity of the algorithm is given by

$$(n-k)n + 2|\bar{L}^{(2)}| + 2\max(1, 2^{-\ell_1}|\bar{L}^{(2)}|^2). \tag{18}$$

From Eq. (17) and Eq. (18), in practice, the increase ratios for both time and space complexities from ogirinal MMT are dominated by $|\bar{L}^{(2)}|$, given that $|\bar{L}^{(2)}| \approx |\bar{L}^{(1)}|$ is satisfied by setting $\ell_1 \approx \log_2 |\bar{L}^{(2)}|$. The increase ratio of the base list is given by

$$\frac{|\bar{L}^{(2)}|}{|L^{(2)}|} = \frac{|L^{(2)}| + \sum_{1 \le i \le p/2} \binom{(k+\ell)/2}{p/2-i}}{|L^{(2)}|} = 1 + \frac{p}{k+\ell-p+2} + O(p^2 k^{-2}),$$

where $|L^{(2)}| = \binom{(k+\ell)/2}{p/2}$. Since $p \ll k$, the increase ratio is dominated by $pk^{-1}$. Thus, the revisited MMT algorithm shows an increase in the number of obtainable solutions from $\binom{n-k-\ell}{w-2p}\binom{(k+\ell)/2}{p}^2$ to $\sum_{0 \le i,j \le p} |\mathcal{C}_{i,j}|\bar{\rho}_{i,j}$, with an increase in time and space complexities by a factor of $pk^{-1}$.

The revisited MMT algorithm also provides time-memory trade-offs by selecting a large value for $\ell_1$. In other words, a larger $\ell_1$ reduces both the space complexity required for $|\bar{L}^{(1)}|$ and the expected number of obtainable solutions $\sum_{0 \le i,j \le p} |\mathcal{C}_{i,j}|\bar{\rho}_{i,j}$, which is directly compensated for by increasing the number of outer loops. Practical results show that choosing a larger $\ell_1$ is also effective for the revisited MMT algorithm, not only for reducing memory consumption but also for reducing total runtime, as in the original time-memory trade-off MMT algorithm described in [17].

**Relationship to BJMM Algorithm** We show that Algorithm 2 is a generalization of the depth-2 BJMM algorithm. As shown in Proposition 4.1, the revisited MMT algorithm includes extended representations in its list construction. Let $p_{\mathrm{bjmm}}$ and $p' \ge p_{\mathrm{bjmm}}/2$ denote weight parameters used in the depth-2 BJMM algorithm. Let $p, i, j$ be the weight parameters employed in the revisited MMT algorithm. Assuming $p' \leftarrow p/2$, $p_{\mathrm{bjmm}} \leftarrow i$ and $i = j$, the revisited MMT algorithm includes the set of extended representations in the BJMM algorithm. This set comprises $\binom{p_{\mathrm{bjmm}}}{p_{\mathrm{bjmm}}/2}^2 \binom{(k+\ell)/2-p_{\mathrm{bjmm}}}{p'-p_{\mathrm{bjmm}}/2}^2$ elements.

If we do not use multi-weight initialization, then both the depth-2 BJMM and revisited MMT algorithm have identical lengths for the depth-2 list, which is $\binom{(k+\ell)/2}{p'}$, and for the depth-1 list, which is $\max(1, 2^{-\ell_1}\binom{(k+\ell)/2}{p'}^2)$. Conversely, $L^{(0)}$ in the depth-2 BJMM algorithm inherently contains any vector $\mathbf{x}$ s.t. $\mathbf{x} \in \left(\mathcal{B}_i^{(k+\ell)/2} \times \mathcal{B}_j^{(k+\ell)/2}\right)$ for any even integers $0 \le i, j \le 2p'$, although this approach has not been explicitly utilized to our knowledge. Hence, as long as the depth is two, the revisited MMT algorithm can represent both MMT and BJMM decoding. The reason why we refer to Algorithm 2 as MMT rather than BJMM is that it is algorithmically MMT, as it no longer uses the BJMM weight parameter $p'$.

## 5 Asymptotic Analysis for Schroeppel–Shamir ISD

This section provides new asymptotic space complexities for Dumer's algorithm and the time-memory trade-off MMT algorithm, in conjunction with a detailed application of the Shamir–Schroeppel technique to depth-1/depth-2 ISDs. Additionally, we state that the asymptotic complexity of the revisited MMT algorithm is equivalent to that of the depth-2 BJMM algorithm.

### 5.1 The Schroeppel–Shamir Technique

The Schroeppel–Shamir technique [33] can reduce the memory complexity of a standard meet-in-the-middle (MITM) attack for the 2-list matching problem. Assuming we aim to find a pair $(\mathbf{x}_1, \mathbf{x}_2) \in L_1 \times L_2$ such that $\mathbf{x}_1 = \mathbf{x}_2$ on certain $\ell$ coordinates. This can be solved by the MITM in time $O(|D|)$ and memory $O(|D|)$, where $|D| = |L_1| = |L_2|$ and $\ell = \log_2 |D|$. When employing the Schroeppel–Shamir technique, it is known that this problem can be solved with the same time complexity of $O(|D|)$ and reduced memory complexity of $O(|D|^{1/2})$.

The algorithm decomposes $L_1 = L_{1,1} \times L_{1,2}$ and $L_2 = L_{2,1} \times L_{2,2}$, where $|L_{i,j}| = |D|^{1/2}$. We set $r = \log_2 |D|^{1/2} = \ell/2$. Then, we create a list $\tilde{L}_1$ from $L_{1,1}$ and $L_{1,2}$, which is a $2^{-r}$-fraction of $L_1$ consisting of $\mathbf{x}_1 = \mathbf{x}_{1,1} + \mathbf{x}_{1,2}$ s.t. $\pi_r(\mathbf{x}_{1,1}) + \pi_r(\mathbf{x}_{1,2}) = \mathbf{t}$ for some $\mathbf{t} \in \mathbb{F}_2^r$, where $\mathbf{x}_{1,1} \in L_{1,1}$ and $\mathbf{x}_{1,2} \in L_{1,2}$. The list $\tilde{L}_1$ is constructed in time and memory of $|D|^{1/2}$. Analogously, $\tilde{L}_2$ is constructed from $L_{2,1}$ and $L_{2,2}$ consisting of $\mathbf{x}_2 = \mathbf{x}_{2,1} + \mathbf{x}_{2,2}$, s.t. $\pi_r(\mathbf{x}_{2,1}) + \pi_r(\mathbf{x}_{2,2}) = \mathbf{t}$. We obtain a $2^{-r}$-fraction of solution pairs in time $|\tilde{L}_1||\tilde{L}_2|/2^{\ell-r} = |D|^{1/2}$ and memory $|D|^{1/2}$. Note that for all pairs $(\mathbf{x}_1, \mathbf{x}_2) \in \tilde{L}_1 \times \tilde{L}_2$, $\mathbf{x}_1 = \mathbf{x}_2$ is satisfied on $r$ coordinates. Therefore, we need to find a pair matching the remaining $\ell - r = \ell/2$ coordinates.

The above procedure is iterated $|D|^{1/2}$ times for all $\mathbf{t} \in \mathbb{F}_2^r$. In total, we obtain all solution pairs $(\mathbf{x}_1, \mathbf{x}_2) \in L_1 \times L_2$ in time $O(|D|^{1/2}|D|^{1/2}) = O(D)$ and memory $O(|D|^{1/2})$.

As for ISD algorithms, the Schroeppel–Shamir technique has previously been used in quantum ISDs (Stern/Dumer) proposed by Kachigar and Tillich [21], and later improved by Kirshanova [22], as it is also beneficial in reducing time complexity when combined with Grover search. For classical ISDs, Esser and Zweydinger apply this technique to MMT/BJMM decoding and achieve reduced asymptotic space complexity for MMT with $2^{0.0375n}$ for full distance decoding [17].

### 5.2 Dumer's Algorithm with Schroeppel–Shamir Technique

We describe how to improve the asymptotic space complexity of Dumer's algorithm. Note that Kachigar and Tillich [21] were the first to utilize the Schroeppel–Shamir technique in the quantum version of Dumer's algorithm. In this study,

we provide detailed analysis and demonstrate the reduction of the space complexity of (classical) Dumer's algorithm using the Schroeppel–Shamir technique through numerical optimization.

Assuming that we have the semi-systematic form $\bar{\mathbf{H}}$ and the syndrome $\bar{\mathbf{s}}$ as shown in Eq. (3). In Dumer's algorithm, we aim to find a permuted solution $\bar{\mathbf{e}} \in \left(\mathcal{B}_{w-2p}^{n-k-\ell} \times \mathcal{B}_p^{(k+\ell)/2} \times \mathcal{B}_p^{(k+\ell)/2}\right)$. To do so, we construct two base lists as follows:

$$L_1^{(1)} = \left\{ \mathbf{z}_1^{(1)} \in \mathbb{F}_2^{\frac{k+\ell}{2}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_1^{(1)}) = p \right\},$$
$$L_2^{(1)} = \left\{ \mathbf{z}_2^{(1)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_2^{(1)}) = p \right\}.$$

We then enumerate all pairs $(\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)}) \in L_1^{(1)} \times L_2^{(1)}$ s.t. $\mathbf{H}_2\mathbf{z}' = \mathbf{s}_2$, where $\mathbf{z}' = \mathbf{z}_1^{(1)} + \mathbf{z}_2^{(1)}$. For $\mathbf{z}'' = \mathbf{H}_1\mathbf{z}' + \mathbf{s}_1$, if $\mathsf{wt}(\mathbf{z}'') = w - 2p$, then, $\mathbf{P}(\mathbf{z}'', \mathbf{z}')$ is the solution. There is no representation in Dumer's algorithm. The asymptotic time complexity of Dumer's algorithm is

$$q^{-1} \max(|D|, 2^{-\ell}|D|^2), \tag{19}$$

where $q = \binom{k+\ell}{2p}\binom{n-k-\ell}{w-2p}\binom{n}{w}^{-1}$ and $|D| = \binom{(k+\ell)/2}{p}$. For readability, we omit Stirling's approximation from the asymptotic complexity. We employ the approximation $\binom{k+\ell}{2p} \approx \binom{(k+\ell)/2}{p}^2$, which is equivalent asymptotically. The space complexity is $|D|$ when we ignore polynomial factors.

One can apply the Schroeppel–Shamir technique in Dumer's algorithm by decomposing $L_1^{(1)} = L_1^{(2)} \times L_2^{(2)}$ and $L_2^{(1)} = L_3^{(2)} \times L_4^{(2)}$:

$$L_1^{(2)} = \left\{ \mathbf{z}_1^{(2)} \in \mathbb{F}_2^{\frac{k+\ell}{4}} \times 0^{\frac{3(k+\ell)}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_1^{(2)}) = p/2 \right\},$$
$$L_2^{(2)} = \left\{ \mathbf{z}_2^{(2)} \in 0^{\frac{k+\ell}{4}} \times \mathbb{F}_2^{\frac{(k+\ell)}{4}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_2^{(2)}) = p/2 \right\},$$
$$L_3^{(2)} = \left\{ \mathbf{z}_3^{(2)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{(k+\ell)}{4}} \times 0^{\frac{k+\ell}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_3^{(2)}) = p/2 \right\},$$
$$L_4^{(2)} = \left\{ \mathbf{z}_4^{(2)} \in 0^{\frac{3(k+\ell)}{4}} \times \mathbb{F}_2^{\frac{k+\ell}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_4^{(2)}) = p/2 \right\}.$$

We create a $2^{-r}$-fraction list $\tilde{L}_1^{(2)} \subset L_1^{(2)} \times L_2^{(2)}$, whose element is $\mathbf{z}_1^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)}$, s.t. $\pi_r(\mathbf{H}_2\mathbf{z}_1^{(2)}) + \pi_r(\mathbf{H}_2\mathbf{z}_2^{(2)}) = \mathbf{t}$, where $r = |D|^{1/2} \leq \ell$ is a parameter and $\mathbf{t} \in \mathbb{F}_2^r$ is some vector. The time and memory complexities required to construct $\tilde{L}_1^{(1)}$ are $|D|^{1/2}$. Analogously, we create a $2^{-r}$-fraction list $\tilde{L}_2^{(1)} \subset L_3^{(2)} \times L_4^{(2)}$, whose element is $\mathbf{z}_2^{(1)} = \mathbf{z}_3^{(2)} + \mathbf{z}_4^{(2)}$, s.t. $\pi_r(\mathbf{H}_2\mathbf{z}_3^{(2)}) + \pi_r(\mathbf{H}_2\mathbf{z}_4^{(2)}) = \pi_r(\mathbf{s}_2) + \mathbf{t}$.

We want to find $2^{-r}$-fraction of $\ell$-matched pairs $(\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)}) \in L_1^{(1)} \times L_2^{(1)}$ s.t. $\mathbf{H}_2\mathbf{z}_1^{(1)} + \mathbf{H}_2\mathbf{z}_2^{(1)} = \mathbf{s}_2$. Since we already have a $2^{-r}$-fraction of $r$-matched pairs $(\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)}) \in \tilde{L}_1^{(1)} \times \tilde{L}_2^{(1)}$ s.t. $\pi_r(\mathbf{H}_2\mathbf{z}_1^{(1)}) + \pi_r(\mathbf{H}_2\mathbf{z}_2^{(1)}) = \pi_r(\mathbf{s}_2)$, this can be obtained in time $\max(|D|^{1/2}, 2^{r-\ell}|D|^{1/2}|D|^{1/2})$ and memory $|D|^{1/2}$. We iterate

the above procedure for all $\mathbf{t}$. Therefore, the asymptotic time complexity of Dumer's algorithm with the Schroeppel–Shamir technique is

$$q^{-1}2^r \max(|D|^{1/2}, 2^{r-\ell}|D|). \tag{20}$$

When $\ell \geq r/2$, Eq. (20) is equivalent to Eq. (19). The asymptotic space complexity is reduced to $|D|^{1/2}$.

**Numerical Optimization** We implement Dumer's ISD with the Schroeppel–Shamir technique on the ISD optimizer developed by Esser [13][4], and conduct numerical optimizations for the full distance decoding setting. In the optimization, binomial coefficients are approximated by Stirling's approximation. For each parameters $o_i$ used in ISD algorithms, let $o_i = \tilde{o}_i \cdot n$, where $0 \leq \tilde{o}_i \leq 1$. We denote $\tilde{k} = k/n$ as the code rate. During optimization, we search for parameters that yield minimal time complexity $T_{\min}^{\tilde{k}}$ for each code rate $\tilde{k}$ and relative weight $\tilde{w} = H^{-1}(1-\tilde{k})$. Finally, we obtain the asymptotic time complexity $\max_{\tilde{k}} T_{\min}^{\tilde{k}}$ and corresponding parameters.

For full distance decoding, we obtain the asymptotic time and space complexities of

$$T = 2^{0.116n} \text{ and } S = 2^{0.0177n},$$

at $\tilde{k} = 0.43$ and $\tilde{w} = 0.1273$, with optimal parameters of

$$\tilde{p} = 0.005088,\ \tilde{r} = 0.01766,\ \tilde{\ell} = 0.03532.$$

One can confirm that the Schroeppel–Shamir technique reduces the asymptotic space complexity of Dumer's algorithm from the original value $S = 2^{0.0353n}$ to its square root $S = 2^{0.0177n}$ while maintaining the same time complexity, using parameters $r = |L^{(2)}|$ and $\ell = 2r$.

### 5.3 MMT Algorithm with Schroeppel–Shamir Technique

Let us provide a detailed analysis of the Schroeppel–Shamir technique in the depth-2 ISD, exemplified by the time-memory trade-off MMT algorithm. Following the initial analysis [17], we attempt to create depth-1 lists $L_1^{(1)}$ and $L_2^{(1)}$ via Schroeppel–Shamir.

$$L_1^{(1)} = \left\{ \mathbf{z}_1^{(1)} \;\middle|\; \mathbf{z}_1^{(2)} \in L_1^{(2)}, \mathbf{z}_2^{(2)} \in L_2^{(2)}, \mathbf{z}_1^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)}, \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{0} \right\},$$
$$L_2^{(1)} = \left\{ \mathbf{z}_2^{(1)} \;\middle|\; \mathbf{z}_3^{(2)} \in L_3^{(2)}, \mathbf{z}_4^{(2)} \in L_4^{(2)}, \mathbf{z}_2^{(1)} = \mathbf{z}_3^{(2)} + \mathbf{z}_4^{(2)}, \pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_2^{(1)} + \mathbf{s}_2) = \mathbf{0} \right\}.$$

In the MMT algorithm, $L_1^{(1)}$ comprises all vectors $\mathbf{z}_1^{(1)} \in \mathcal{B}_{p/2}^{(k+\ell)/2} \times \mathcal{B}_{p/2}^{(k+\ell)/2}$ s.t. $\pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_1^{(1)}) = \mathbf{0}$ and $L_2^{(1)}$ contains all vectors $\mathbf{z}_2^{(1)} \in \mathcal{B}_{p/2}^{(k+\ell)/2} \times \mathcal{B}_{p/2}^{(k+\ell)/2}$ s.t.

[4] Available at https://github.com/Memphisd/Revisiting-NN-ISD.

$\pi_{\ell_1}(\mathbf{H}_2\mathbf{z}_2^{(1)}) = \pi_{\ell_1}(\mathbf{s}_2)$. Since $\mathbf{z}_1^{(1)}$ is constructed from a pair $(\mathbf{z}_1^{(2)}, \mathbf{z}_2^{(2)})$, where $\mathbf{z}_1^{(1)} = \mathbf{z}_1^{(2)} + \mathbf{z}_2^{(2)}$, we can consider a $2^{-r}$ fraction of the set of pairs by imposing $\pi_r(\mathbf{H}_2\mathbf{z}_1^{(2)}) = \mathbf{t}_1$ and $\pi_r(\mathbf{H}_2\mathbf{z}_2^{(2)}) = \mathbf{t}_1$ for $r \le \ell_1$ and some $\mathbf{t}_1 \in \mathbb{F}_2^r$. Analogously, $\mathbf{z}_2^{(1)}$ is formed from a pair $(\mathbf{z}_3^{(2)}, \mathbf{z}_4^{(2)})$, where $\mathbf{z}_2^{(1)} = \mathbf{z}_3^{(2)} + \mathbf{z}_4^{(2)}$. A $2^{-r}$ fraction is obtained by imposing $\pi_r(\mathbf{H}_2\mathbf{z}_3^{(2)}) = \mathbf{t}_2$ and $\pi_r(\mathbf{H}_2\mathbf{z}_4^{(2)}) = \pi_r(\mathbf{s}_2) + \mathbf{t}_2$ for some $\mathbf{t}_2 \in \mathbb{F}_2^r$. There are $2^{2r}$ combinations for a pair $(\mathbf{t}_1, \mathbf{t}_2)$, as $\mathbf{t}_1$ is independent of $\mathbf{t}_2$. Therefore, we have $2^{-2r}$ fraction of $(\mathbf{z}_1^{(1)}, \mathbf{z}_2^{(1)}) \in L_1^{(1)} \times L_2^{(1)}$ for some pair $(\mathbf{t}_1, \mathbf{t}_2)$, i.e., for depth-2 Schroeppel–Shamir, we require $2^{-2r}$ iterations as compensation for reducing the list size to $2^{-r}$.

Algorithmically, we first create depth-3 eight lists by decomposing $L_i^{(2)} = L_{2i-1}^{(3)} \times L_{2i}^{(3)}$ for $1 \le i \le 4$.

$$
\begin{aligned}
L_1^{(3)} = L_5^{(3)} &= \left\{ \mathbf{z}_1^{(3)} \in \mathbb{F}_2^{\frac{k+\ell}{4}} \times 0^{\frac{3(k+\ell)}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_1^{(3)}) = p/4 \right\}, \\
L_2^{(3)} = L_6^{(3)} &= \left\{ \mathbf{z}_2^{(3)} \in 0^{\frac{k+\ell}{4}} \times \mathbb{F}_2^{\frac{(k+\ell)}{4}} \times 0^{\frac{k+\ell}{2}} \;\middle|\; \mathsf{wt}(\mathbf{z}_2^{(3)}) = p/4 \right\}, \\
L_3^{(3)} = L_7^{(3)} &= \left\{ \mathbf{z}_3^{(3)} \in 0^{\frac{k+\ell}{2}} \times \mathbb{F}_2^{\frac{(k+\ell)}{4}} \times 0^{\frac{k+\ell}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_3^{(3)}) = p/4 \right\}, \\
L_4^{(3)} = L_8^{(3)} &= \left\{ \mathbf{z}_4^{(3)} \in 0^{\frac{3(k+\ell)}{4}} \times \mathbb{F}_2^{\frac{k+\ell}{4}} \;\middle|\; \mathsf{wt}(\mathbf{z}_4^{(3)}) = p/4 \right\},
\end{aligned}
$$

where $|L_i^{(3)}| = \binom{(k+\ell)/4}{p/4} \approx |D|^{1/2}$ for $|D| = \binom{(k+\ell)/2}{p/2}$. For a parameter $r \le \ell_1$, we create $2^{-r}$-fraction lists $\tilde{L}_i^{(2)} \subset L_{2i-1}^{(3)} \times L_{2i}^{(3)}$ for $1 \le i \le 4$, where each element is defined as follows:

$$
\begin{aligned}
\mathbf{z}_1^{(2)} &= \mathbf{z}_1^{(3)} + \mathbf{z}_2^{(3)} \text{ s.t. } \pi_r(\mathbf{H}_2\mathbf{z}_1^{(3)}) + \pi_r(\mathbf{H}_2\mathbf{z}_2^{(3)}) = \mathbf{t}_1, \\
\mathbf{z}_2^{(2)} &= \mathbf{z}_3^{(3)} + \mathbf{z}_4^{(3)} \text{ s.t. } \pi_r(\mathbf{H}_2\mathbf{z}_3^{(3)}) + \pi_r(\mathbf{H}_2\mathbf{z}_4^{(3)}) = \mathbf{t}_1, \\
\mathbf{z}_3^{(2)} &= \mathbf{z}_1^{(3)} + \mathbf{z}_2^{(3)} \text{ s.t. } \pi_r(\mathbf{H}_2\mathbf{z}_1^{(3)}) + \pi_r(\mathbf{H}_2\mathbf{z}_2^{(3)}) = \mathbf{t}_2, \\
\mathbf{z}_4^{(2)} &= \mathbf{z}_3^{(3)} + \mathbf{z}_4^{(3)} \text{ s.t. } \pi_r(\mathbf{H}_2\mathbf{z}_3^{(3)}) + \pi_r(\mathbf{H}_2\mathbf{z}_4^{(3)}) = \pi_r(\mathbf{s}_2) + \mathbf{t}_2,
\end{aligned}
$$

where $\mathbf{t}_1, \mathbf{t}_2 \in \mathbb{F}_2^r$. The time complexity for depth-2 lists is $\max(|D|^{1/2}, 2^{-r}|D|)$. The size of a depth-2 list is $|\tilde{L}^{(2)}| = \max(1, 2^{-r}|D|)$. For depth 1-lists, we obtain $2^{-r}$-fraction lists $\tilde{L}_i^{(1)} \subset \tilde{L}_{2i-1}^{(2)} \times \tilde{L}_{2i}^{(2)}$ for $i = 1, 2$ with time $\max(|\tilde{L}^{(2)}|, 2^{r-\ell_1}|\tilde{L}^{(2)}|^2)$ and space $|\tilde{L}^{(1)}| = \max(1, 2^{r-\ell_1}|\tilde{L}^{(2)}|^2)$. Finally, we merge $\tilde{L}_1^{(1)}$ and $\tilde{L}_2^{(1)}$ in time $\max(|\tilde{L}^{(1)}|, 2^{\ell_1-\ell}|\tilde{L}^{(1)}|^2)$ and obtain a fraction of the permuted solution with probability $\rho_{\mathrm{repr}} = \min(1, 2^{-\ell_1-2r}R)$, where, $R = \binom{2p}{p} \approx \binom{p}{p/2}^2$, instead of Eq. (9). When $2^{-\ell_1-2r}R < 1$, we need to iterate the Search component $2^{\ell_1+2r}R^{-1}$ times to yield one permuted solution expectedly under a good permutation. The asymptotic time complexity of the time-memory trade-off MMT algorithm with Schroeppel–Shamir is given by

$$
q^{-1}\,{\rho_{\mathrm{repr}}}^{-1}\max(|L^{(3)}|, |\tilde{L}^{(2)}|, |\tilde{L}^{(1)}|, 2^{\ell_1-\ell}|\tilde{L}^{(1)}|^2),
$$

where $q = \binom{k+\ell}{2p}\binom{n-k-\ell}{w-2p}\binom{n}{w}^{-1}$. The asymptotic space complexity is given by $\max(|L^{(3)}|, |\tilde{L}^{(2)}|, |\tilde{L}^{(1)}|)$.

**Numerical Optimization** Similar to Dumer's algorithm, we calculated the asymptotic complexity of the time-memory trade-off MMT algorithm with the ISD optimizer. For full distance decoding, we obtain the asymptotic time and space complexities of

$$T = 2^{0.111n} \text{ and } S = 2^{0.0376n},$$

at $\tilde{k} = 0.44$ and $\tilde{w} = 0.1273$, with optimal parameters of

$$\tilde{p} = 0.01073,\ \tilde{r} = 0,\ \tilde{\ell_1} = 0.03764,\ \tilde{\ell} = 0.07527.$$

For MMT decoding, we confirm that the Schroeppel–Shamir technique does not contribute to reducing memory without sacrificing time complexity. Nevertheless, it still results in almost the same space complexity as $2^{0.0375n}$, as derived in [17], which is significantly smaller than the original value $2^{0.54n}$ [25]. This implies that $\rho_{\text{repr}}$, the time-memory trade-off term introduced in [17], plays a crucial role in reducing memory complexity in the MMT algorithm. Additionally, we observe that $T$ is minimized when $\ell_1 = \log_2 \binom{(k+\ell)/2}{p/2}$, leading to $S = |L^{(2)}| = |L^{(1)}|$. Consequently, there is no longer a need to optimize the parameter $\ell_1$.

When we set $r = \log_2 |D|^{1/2}$ and $\ell_1 = 2r$, we obtain $S = |L^{(3)}| = |\tilde{L}^{(2)}| = |\tilde{L}^{(1)}| = |D|^{1/2}$. In this setting, we observe a reduction in space complexity with an increase in time complexity:

$$T = 2^{0.119n} \text{ and } S = 2^{0.00588n},$$

at $\tilde{k} = 0.45$ and $\tilde{w} = 0.1273$, we obtain

$$\tilde{p} = 0.002639,\ \tilde{\ell} = 0.01763.$$

The time complexity falls between Prange's complexity of $2^{0.121n}$ and Dumer's complexity of $2^{0.116n}$. We defer to future work the exploration of how to leverage the Schroeppel–Shamir technique for depth-2 ISDs.

**Asymptotic Complexity for BJMM and Revisited MMT Algorithm** For BJMM decoding, it was shown that the time-memory trade-off technique can decrease asymptotic time complexity when memory capacity is limited [17], particularly for depth-3 BJMM. This holds true as well for the depth-2 time-memory trade-off BJMM with unlimited memory, which achieves the following asymptotic complexity:

$$T = 2^{0.105n} \text{ and } S = 2^{0.0659n},$$

at $\tilde{k} = 0.43$ and $\tilde{w} = 0.1273$, with optimal parameters

$$\tilde{p}' = 0.01076,\ \tilde{p} = 0.01812,\ \tilde{\ell_1} = 0.06588,\ \tilde{\ell} = 0.1318.$$

This is as the same as original BJMM algorithm [3]. We observe that $T$ is minimized when $2^{\ell_1} = S = |L^{(2)}| = |L^{(1)}| = R$, where $|L^{(2)}| = \binom{(k+\ell)/2}{p'}$ and $R \approx \binom{2p}{p}\binom{k+\ell-2p}{2p'-p}$. We state that the the revisited MMT algorithm has the same asymptotic complexity as the depth-2 time-memory trade-off BJMM, under the assumption that a specific weight distribution pair $(i, j)$, which maximizes $|\mathcal{C}_{i,j}|\bar{\rho}_{i,j}$ in Eq. (16), emerges as the dominant term among all weight distributions.

## 6 Cryptanalysis

In this section, we present security estimates for Classic McEliece, BIKE, and HQC with the revised MMT and other ISD algorithms. To calculate the bit security for each cryptosystem, we use `CryptographicEstimators`[5], which is the latest cryptanalysis library developed by Esser et al. [16]. The SDP parameter sets we target are listed in Table 2.

**Table 2.** Parameter sets for Classic McEliece, BIKE and HQC proposals.

| Scheme | Category | $n$ | $k$ | $w$ |
|---|---|---|---|---|
| Classic McEliece | 1 | 3488 | 2720 | 64 |
| | 3 | 4608 | 3360 | 96 |
| | 5 | 6688 | 5024 | 128 |
| | 5 | 6960 | 5413 | 119 |
| | 5 | 8192 | 6528 | 128 |
| BIKE (message) | 1 | 24646 | 12323 | 134 |
| | 3 | 49318 | 24659 | 199 |
| | 5 | 81946 | 40973 | 264 |
| BIKE (key) | 1 | 24646 | 12323 | 142 |
| | 3 | 49318 | 24659 | 206 |
| | 5 | 81946 | 40973 | 274 |
| HQC | 1 | 35338 | 17669 | 132 |
| | 3 | 71702 | 35851 | 200 |
| | 5 | 115274 | 57637 | 262 |

### 6.1 Cryptanalysis for Classic McEliece

First, we present the estimated bit time complexity and its corresponding space complexity for all parameter sets of Classic McEliece in Table 3.

In addition to our revised MMT algorithm and various modern ISD algorithms, we incorporate MMT-TMTO and BJMM-TMTO, time-memory trade-off

[5] https://github.com/Crypto-TII/CryptographicEstimators

**Table 3.** Estimated bit security and bit space complexity for Classic McEliece. Underlines indicate a deficiency in meeting the specified security requirements (128 bits for Category 1, 192 bits for Category 3, and 256 bits for Category 5).

| Category | 1 ($n = 3488$) | | 3 ($n = 4608$) | | 5a ($n = 6688$) | | 5b ($n = 6960$) | | 5c ($n = 8192$) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | T | M | T | M | T | M | T | M | T | M |
| MMT-REV | **140** | 98 | **<u>179</u>** | 116 | **<u>245</u>** | 146 | **<u>245</u>** | 169 | **275** | 174 |
| PRANGE | 173 | **22** | 217 | **23** | 296 | **24** | 297 | **24** | 334 | **24** |
| DUMER | 151 | 58 | 193 | 60 | 268 | 89 | 268 | 90 | 303 | 109 |
| MMT-TMTO | 148 | 59 | 190 | 70 | 261 | 90 | 261 | 91 | 294 | 102 |
| BJMM-TMTO | 142 | 98 | <u>182</u> | 122 | <u>248</u> | 162 | <u>248</u> | 160 | 277 | 189 |
| MAY-OZEROV | 141 | 87 | <u>180</u> | 115 | <u>246</u> | 165 | <u>246</u> | 160 | 276 | 194 |
| BOTH-MAY | 142 | 88 | <u>181</u> | 113 | <u>248</u> | 143 | <u>247</u> | 145 | 279 | 149 |
| SIEVING ISD | 143 | 58 | <u>184</u> | 65 | 257 | 91 | 257 | 92 | 291 | 95 |
| MMT-REV$^{\log_2 M}_{M\leq 43}$ | 147 | 43 | <u>191</u> | 43 | 267 | 43 | 268 | 43 | 304 | 43 |

variants of the MMT/BJMM decoding [17], and SIEVING ISD, the latest ISD proposed by Guo, Johansson, and Nguyen [18]. To derive the estimated complexity for SIEVING ISD, we use open-source code provided by the authors[6]. The bold font indicates the minimal bit time complexity (T) or bit space complexity (M). Among these ISD algorithms, MMT-REV achieves the smallest time complexity across all categories when assuming unlimited memory capacity and constant memory access cost. Compared to MMT and BJMM decoding, MMT-REV reduced bit security for Classic McEliece III by 11 bits from MMT-TMTO, and 3 bits from BJMM-TMTO.

In [15], the authors confirm that the assumption of the logarithmic memory access cost model aligns well with actual implementation. We also verify the validity of this assumption for our Compute Unified Device Architecture (CUDA) MMT implementation described in [29]. Previous papers [14,24,18] also enforce a maximum memory capacity of $M \leq 60$ as a low-memory setting, implying that one can utilize memory up to approximately 144 petabytes, which may appear somewhat excessive. In this paper, we also evaluate the security of NIST-PQC candidates with the revisited MMT algorithm under more realistic memory constraints, taking into account both the logarithmic access model and a maximal memory capacity of $2^{43}$ bits (1 terabyte), denoted as MMT-REV$^{\log_2 M}_{M\leq 43}$ in each table.

As a result, it is observed that the security levels of Classic McEliece for Categories 1, 5a, 5b, and 5c have sufficiently large security margins from the security requirements (128 bits for Category 1, 192 bits for Category 3, and 256 bits for Category 5) when practical memory constraints are assumed. However, for Category 3, the security level remains below the desired security level for the MMT-REV algorithm, albeit by a mere 1 bit.

[6] https://github.com/vunguyen95/Review-ISD-Sieving

### 6.2 Cryptanalysis for BIKE and HQC

Since both BIKE and HQC use a quasi cyclic code, it is known that the time complexities of several ISD algorithms can be decreased by leveraging the cyclic nature of the code. We present the results of our security estimations for BIKE and HQC in Table 4 and 5, respectively.

**Table 4.** Estimated bit security and bit space complexity for BIKE.

| | Category | 1 ($n = 24646$) | | 3 ($n = 49318$) | | 5 ($n = 81946$) | |
|---|---|---|---|---|---|---|---|
| | | T | M | T | M | T | M |
| key security | MMT-REV | 146 | 54 | 210 | 59 | 277 | 62 |
| | PRANGE | 168 | **28** | 234 | **30** | 304 | **32** |
| | DUMER | 148 | 40 | 211 | 43 | 279 | 45 |
| | MMT-TMTO | 148 | 38 | 211 | 40 | 279 | 41 |
| | BJMM-TMTO | 147 | 55 | 211 | 57 | 278 | 61 |
| | MAY-OZEROV | 147 | 55 | 210 | 57 | 278 | 61 |
| | BOTH-MAY | 147 | 55 | 210 | 57 | 278 | 61 |
| | SIEVING ISD | **141** | 46 | **204** | 50 | **271** | 53 |
| | MMT-REV$^{\log_2 M}_{M \leq 43}$ | 146 | 42 | 211 | 43 | 280 | 41 |
| message security | MMT-REV | 145 | 46 | 210 | 59 | 275 | 63 |
| | PRANGE | 174 | **28** | 242 | **30** | 309 | **32** |
| | DUMER | 146 | 41 | 211 | 44 | 276 | 46 |
| | MMT-TMTO | 146 | 38 | 211 | 40 | 276 | 41 |
| | BJMM-TMTO | 146 | 38 | 211 | 40 | 276 | 61 |
| | MAY-OZEROV | 146 | 55 | 211 | 57 | 276 | 61 |
| | BOTH-MAY | 146 | 55 | 211 | 57 | 276 | 61 |
| | SIEVING ISD | **135** | 46 | **198** | 50 | **262** | 53 |
| | MMT-REV$^{\log_2 M}_{M \leq 43}$ | 145 | 42 | 212 | 40 | 278 | 41 |

For the key security of BIKE, the time complexities of all ISD algorithms are reduced by a factor of $k$ without any additional effort. To attack the secret key of BIKE, we need to solve the quasi cyclic SDP, where the syndrome is the zero vector. This SDP contains $k$ different solutions, which decrease the expected number of loops required for any ISD by a factor of $k$. For the bit security, we present the results with $\log_2 k$ subtracted from the estimations.

In the case of message security for BIKE and HQC, several ISD algorithms can reduce time complexity by implementing the Decoding-One-Out-of-Many (DOOM) strategy, as described in [34]. For DUMER, MMT-TMTO, BJMM-TMTO, and MMT-REV, we can reduce the time complexity by $\Omega(\sqrt{k})$, where $k = n/2$, by utilizing the asymmetrical base list construction technique, as shown in [15].

**Table 5.** Estimated bit security and bit space complexity for HQC.

| Category | 1 ($n = 35338$) | | 3 ($n = 71702$) | | 5 ($n = 115274$) | |
|---|---|---|---|---|---|---|
| | T | M | T | M | T | M |
| MMT-REV | 145 | 48 | 213 | 52 | 275 | 55 |
| PRANGE | 173 | **29** | 244 | **31** | 308 | **33** |
| DUMER | 145 | 43 | 213 | 46 | 275 | 48 |
| MMT-TMTO | 145 | 38 | 213 | 40 | 275 | 42 |
| BJMM-TMTO | 145 | 38 | 213 | 40 | 275 | 42 |
| MAY-OZEROV | 146 | 39 | 214 | 42 | 276 | 44 |
| BOTH-MAY | 146 | 39 | 214 | 42 | 276 | 44 |
| SIEVING ISD | **141** | 46 | **204** | 50 | **271** | 53 |
| MMT-REV$^{\log_2 M}_{M \leq 43}$ | 145 | 43 | 214 | 40 | 276 | 42 |

We cannot achieve DOOM speedups for PRANGE, since the $i$-th rotation of the permuted syndrome $\bar{\mathbf{s}}$, denoted by $\mathrm{rot}_i(\bar{\mathbf{s}})$, satisfies $\mathsf{wt}(\bar{\mathbf{s}}) = \mathsf{wt}(\mathrm{rot}_i(\bar{\mathbf{s}}))$. This result implies that there is no way to exploit the rotations of the syndrome in the SEARCH component of the Prange algorithm, where only the weights of the (rotated) syndromes are checked.

For MAY-OZEROV and BOTH-MAY, concrete algorithms for realizing DOOM speedups have not yet been developed. In this paper, a common assumption of $\sqrt{k}$ speedup is applied to them, as in [14]. For SIEVING ISD, the authors show $k$ times speedups by leveraging the rotations of the syndrome while enlarging vectors in the search phase.

From Tables 4 and 5, we can confirm that both BIKE and HQC meet the desired level of bit security across all categories. The difference in time complexity between sieving ISD and other ISDs for quasi-cyclic codes stems mainly from discrepancies in the speedup gains of DOOM. To our knowledge, there is currently no practical evidence for the sieving ISD for quasi-cyclic SDP instances. Hence, in this paper, we also employ the revisited MMT algorithm for memory-constrained bit-security estimations for BIKE and HQC. The verification of the practicality of the sieving ISD for quasi-cyclic codes remains a future challenge.

When assuming a logarithmic memory access cost and $M \leq 43$ for the revisited MMT algorithm, BIKE and HQC of Category 3 achieve 212-bit and 214-bit security, respectively. Compared to Classic McEliece of Category 3 under the same memory constraints, the security level of Classic McEliece for this category is approximately 21 bits lower than BIKE and 23 bits lower than HQC.

## 7 Experimental Results

In this section, we provide details about our GPU implementation of the revisited MMT algorithm, its performance on a PC equipped with a consumer-grade GPU, and a practical runtime analysis based on actual decoding results. In

our experiments, we use a desktop PC with an Intel Core i9-12900 CPU and a GeForce RTX 4080 GPU unless otherwise specified.

### 7.1 GPU Implementation of the Revisited MMT Algorithm

We implement the revisited MMT algorithm by modifying the open-source CUDA MMT implementation, `cuMMT`[7]. In `cuMMT`, a depth-2 MMT with $p = 4$ is implemented in a streaming fashion. In summary, the depth 2 list $L_1^{(2)}$ and the depth 1 list $L_1^{(1)}$ are represented as one-dimensional integer lists, effectively functioning as hash maps when indexed by $\mathbf{H}_2\mathbf{e}$ for each element $\mathbf{e}$ in these lists. List merging is performed in a parallel manner on a GPU with asynchronous concurrent writing. Please refer to the original paper and reference implementation for further details [29].

We replace the MMT algorithm in `cuMMT` with the revisited MMT algorithm. Additionally, several modifications are implemented, as listed below:

1. We replaced naive Gaussian elimination in `cuMMT` with an optimized implementation of the *Method of Four Russians for Inversion* (M4RI) [1], improved by Esser, May and Zweydinger[8].
2. The outer loop in the revisited MMT algorithm is parallelized using CPU threads, while the list construction phase is parallelized using GPU threads simultaneously.
3. Using the `constexpr` feature in `C++`, we transform the non-variable values into constants, reducing memory access costs.

With these implementation modifications, `cuMMT` achieved a practical speedup of 23.4 times compared to the reference implementation for McEliece-1409 under the same experimental conditions.

Notably, replacing the original MMT algorithm with the revisited MMT algorithm alone reduces the runtime to by a factor of 3 for McEliece-1409, and this optimization can be applied to any MMT/BJMM implementation as well.

### 7.2 Decoding McEliece-1409 Challenge

We estimated the bit complexities and optimal parameters for McEliece-1409 using `CryptographicEstimators`, under constraints of $M \leq 33$ (1 gigabyte) and the logarithmic access model in Table 6. From the results, MMT-REV is theoretically shown to be 3.3 times faster than BJMM-TMTO and still 1.9 times faster than MAY-OZEROV, which yields the smallest time complexity for McEliece instances among the ISD algorithms in [14]. Note that for MAY-OZEROV and BOTH-MAY, currently there is no practical implementation due to the low efficiency of the local sensitive hashing (LSH) technique, which is the core procedure in both types of ISD. In this memory constrained setting, BJMM-TMTO is equivalent to MMT-TMTO, as it uses $p' = p/2$.

[7] The reference implementation for `cuMMT` is available at https://www.jstage.jst.go.jp/article/transfun/E106.A/3/E106.A_2022CIP0023/_pdf/

[8] Available at https://github.com/FloydZ/cryptanalysislib.

**Table 6.** Estimated complexities and optimal parameters for McEliece-1409 with $M \leq 33$ (1 gigabyte) and the logarithmic memory access cost model.

| Algorithm | T | M | $p'$ | $p$ | $\ell_1$ | $\ell$ | $w_1$ | $w_2$ | depth |
|---|---|---|---|---|---|---|---|---|---|
| MMT-REV | **70.1** | 31.5 | – | 4 | 14 | 36 | – | – | 2 |
| PRANGE | 88.6 | **18.6** | – | – | – | – | – | – | – |
| DUMER | 72.4 | 28.8 | – | 2 | – | 19 | – | – | 2 |
| MMT-TMTO | 71.8 | 32.3 | 2 | 4 | 13 | 36 | – | – | 2 |
| BJMM-TMTO | 71.8 | 32.3 | 2 | 4 | 13 | 36 | – | – | 2 |
| MAY-OZEROV | 71.0 | 32.2 | 2 | 4 | – | 13 | – | – | 2 |
| BOTH-MAY | 71.1 | 32.2 | 2 | 4 | – | 13 | 0 | 0 | 2 |

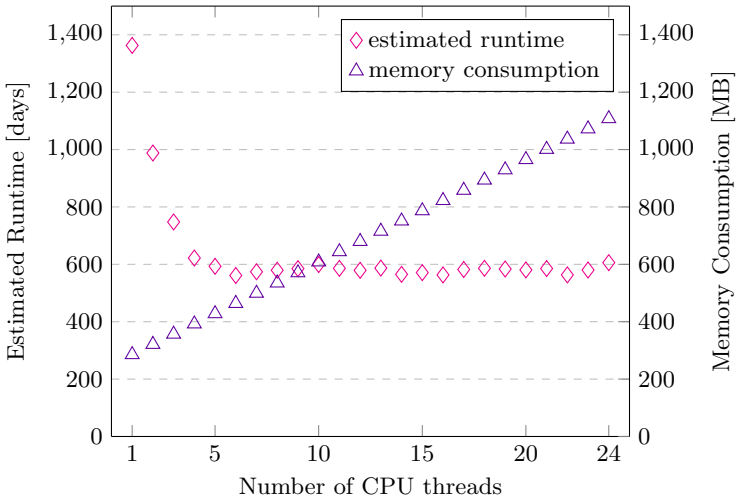


**Fig. 4.** Estimated runtime and memory consumption of our `cuMMT` implementation for McEliece-1409 with varying the number of CPU threads on the desktop PC.

In practice, we use $p = 4, \ell_1 = 14, \ell = 35$ for our `cuMMT` implementation to solve the McEliece-1409 instance, as it requires two large integer arrays of sizes $2^{\ell_1}$ and $2^{\ell-\ell_1}$. As a result, we obtain the expected runtime and the memory consumption under our experimental environment with varying CPU thread counts, as illustrated in Figure 4. The estimated runtime is given by $T_{\text{loop}}q^{-1}$, where $q$ is the probability of success of one iteration obtained by the estimator, and $T_{\text{loop}}$ is the measured runtime for one iteration with the execution of the `cuMMT`.

Based on the results, we observe that the decrease in runtime saturates as the number of threads increases. We parallelized 16 CPU threads in our experiments, requiring an expected 563 days and 822 megabytes ($2^{32.6}$ bits) on our desktop PC to solve McEliece-1409. The maximal number of GPU threads is set to

$2^{-\ell_1}|L^{(2)}|^2 = 1,684,900$ per CPU thread, resulting in a total of $26,958,400$ GPU threads per PC.

With 10 desktop PCs (5 each equipped with an RTX 4080 GPU and an Intel Core i9-12900 CPU, and 5 with an RTX 3090 GPU and the same CPU), we achieve an expected runtime of 65.3 days for McEliece-1409. As a result, we solved the McEliece-1409 instance in 29.6 hours.

### 7.3 Comparison with Latest Implementations

To compare our implementation with other recent ISD implementations, we solved several McEliece instances using the new `cuMMT`, whose results are depicted in Figure 5, as well as the estimated runtimes and bit complexities. We describe recent record computations for McEliece instances below.

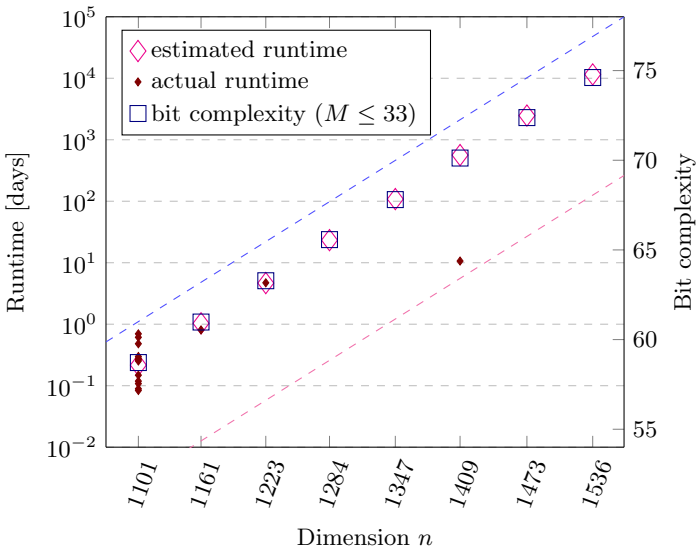


**Fig. 5.** Bit complexities and estimated running times to solve each McEliece challenge with a desktop PC equipped with an Intel Core i9-12900 CPU and an RTX 4080 GPU. Instances that were successfully solved by our implementation are marked with small squares. The red dashed line represents the minimal time $t^{\alpha}_{\min}$ and the blue dashed line represents the maximal time $t^{\alpha}_{\max}$ w.r.t. $\alpha = 0.01$, as discussed in Section 7.4. One can see that all of our records are within the range of $t^{\alpha}_{\min}$ and $t^{\alpha}_{\max}$.

McEliece-1161 was solved in 15.66 days by Narisada, Fukushima, and Kiyomoto using a GPU implementation of Dumer's algorithm on an Intel Xeon E5-2686v4 server and an NVIDIA Tesla V100 [28].

Esser, May, and Zweydinger achieved the first records for McEliece-1223 and McEliece-1284 at 2.45 days and 31.43 days, respectively, using their fast implementation of the MMT/BJMM algorithm with 4 AMD EPYC 7742 CPUs [15].

Their implementation was later improved by introducing time-memory trade-offs, achieving an expected runtime of 13.10 days for McEliece-1284 [17].

Recently, Bernstein, Lange, and Peters solved McEliece-1347 using software they developed on several clusters of computers [6]. According to the website[9], it is stated that the expected runtime of their implementation for McEliece-1284 with 4 AMD EPYC 7742 CPUs is 31.56 days.

The expected runtimes of our new `cuMMT` with one desktop PC under the memory constraints of $M \leq 33$ for McEliece-1284 and McEliece-1347 are 23.30 days and 108.39 days, respectively. For McEliece-1473 and McEliece-1536, expected runtime of our implementation with $M \leq 33$ are 2474 days and 11552 days, respectively. We refrain from making direct comparisons between individual implementations, as the objective of this paper is not to identify the best implementation among all implementations.

### 7.4 Practical Minimal/Maximal Time of ISD Algorithms

There may be concerns regarding the disparity between the expected runtime and the actual runtime, as evident in the McEliece-1409 record. Specifically, this situation raises questions about whether dimensionality $n$ influences the variance in runtime. However, we can refute this notion by extending the confidence interval analysis to the geometric distribution. Note that the best-case/worst-case time complexities of an ISD algorithm are 1 and $\infty$, respectively.

We denote the probability of success for each iteration as $q$ in any ISD algorithm. The probability density function for the $N$-th iteration at which the algorithm terminates is given by $f(N) = q(1-q)^{N-1}$, which is the geometric distribution. Since the total time complexity of an ISD algorithm up to the $N$-th iteration is $N(T_{\mathrm{ge}} + T_{\mathrm{search}})$, $f(N)$ can be extended to a map between the runtime of an ISD and the probability of success: $f(t) = q(1-q)^{t/T-1}$, where $T = T_{\mathrm{ge}} + T_{\mathrm{search}}$. The cumulative distribution function for $f(t)$ is $F(t) = 1 - (1-q)^{t/T}$. We aim to exclude the leftmost/rightmost $\alpha$ from the area formed between 0 and $f(t)$, as depicted in Figure 6.

To do so, we consider an interval, $[t^{\alpha}_{\mathrm{min}}, t^{\alpha}_{\mathrm{max}}]$, where $t^{\alpha}_{\mathrm{min}}$ satisfies $F(t^{\alpha}_{\mathrm{min}}) = \alpha$ and $t^{\alpha}_{\mathrm{max}}$ satisfies $F(t^{\alpha}_{\mathrm{max}}) = 1-\alpha$. We refer to $t^{\alpha}_{\mathrm{min}}$ as the minimal time and $t^{\alpha}_{\mathrm{max}}$ as the maximal time with respect to $\alpha$. The minimal time $t^{\alpha}_{\mathrm{min}}$ is determined by solving the following equation for $t$: $1 - (1-q)^{t/T} = \alpha$, which gives

$$t^{\alpha}_{\mathrm{min}} = \left(q^{-1}\alpha + O(q^{-1}\alpha^2)\right) T, \tag{21}$$

by series expansion. Assuming $q \ll 1$ and $\alpha \ll 1$, Eq. (21) is approximated by

$$t^{\alpha}_{\mathrm{min}} \approx q^{-1} T \alpha. \tag{22}$$

The maximal time can be determined by solving the following equation for $t$: $1 - (1-q)^{t/T} = 1 - \alpha$, which gives

$$t^{\alpha}_{\mathrm{max}} = \left(q^{-1} - \frac{1}{2} + O(q)\right) T \ln \alpha^{-1}, \tag{23}$$

[9] https://isd.mceliece.org/1347.html, published on February 26, 2023.

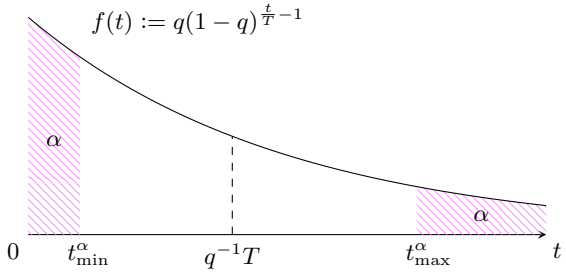


**Fig. 6.** The minimal time $t^{\alpha}_{\min}$ and maximal time $t^{\alpha}_{\max}$ of an ISD algorithm w.r.t. a parameter $0 \leq \alpha \leq 1$. $f(t) := q(1-q)^{t/T-1}$ is a map from the runtime $t$ to the success probability $f(t)$ at which an ISD algorithm terminates, where $T = T_{\mathrm{ge}} + T_{\mathrm{search}}$. We draw $f(t)$ with $q = 0.01$ in this figure for simplicity. $q^{-1}T$ is an average time complexity of an ISD. Note that the practical value of $q$ is sufficiently small, which gives approximations of $t^{\alpha}_{\min} \approx q^{-1}T\alpha$ and $t^{\alpha}_{\max} \approx q^{-1}T\ln\alpha^{-1}$. For instance, in the case of McEliece-1409, optimal parameters in the revisited MMT algorithm yield $q = 2^{-37.1}$.

by series expansion. Assuming $q \ll 1$, then Eq. (23) is approximated by

$$t^{\alpha}_{\max} \approx q^{-1}T\ln\alpha^{-1}. \tag{24}$$

It is noteworthy that Eq. (24) increases on a logarithmic scale with decreasing in $\alpha$, whereas Eq. (22) linearly decreases as $q \ll 1$ is satisfied in ISD algorithms. Therefore, the choice of $\alpha$ significantly affects the minimal runtime.

For our McEliece-1409 result, the consumed iteration count was $2^{30.8}$, whereas the average number of iterations is $q^{-1} = 2^{37.1}$, resulting in an expected runtime of 65.3 days. This result gives $\alpha \approx 2^{-6.3} = 0.013$, which falls within the runtime interval of 15.7 hours to 300.7 days with $\alpha = 0.01$.

To securely integrate code-based cryptography into real applications, a sufficiently large security margin from the average time complexity should be set by assuming that $\alpha$ is very small (e.g., $\alpha = 2^{-32}$), thereby reducing the probability of successful decoding to a negligible level.

## 8 Conclusion

In this paper, we propose the revisited MMT algorithm as a generalization of MMT and BJMM decoding. This algorithm offers the lowest bit security level for Classic McEliece among all ISD algorithms. Experimentally, we successfully solve McEliece-1409, which has 70-bit security, for the first time in 30 hours using 10 desktop PCs. These results provide both theoretical and practical evidence for the reliability of code-based NIST-PQC round 4 candidates.

Future work should include concrete analyses and practical implementations of later-proposed ISD algorithms, such as the May–Ozerov, Both–May, and sieving ISD algorithms, considering multiple weight distributions. It is important to verify the resilience of the remaining code-based NIST-PQC candidates against

structural attacks. Analyzing quantum ISD algorithms from both theoretical and practical perspectives is also crucial for ensuring the security of code-based cryptography.

**Acknowledgments**

This study was supported by JSPS KAKENHI JP22KJ0554 and the Joint Research Center for Advanced and Fundamental Mathematics for Industry, Institute of Mathematics for Industry, Kyushu University (2022a015).

## References

1. Albrecht, M., Bard, G.: The M4RI Library. The M4RI Team (2023), https://bitbucket.org/malb/m4ri
2. Aragon, N., Lavauzelle, J., Lequesne, M.: decodingchallenge.org (2019), http://decodingchallenge.org
3. Becker, A., Coron, J.S., Joux, A.: Improved Generic Algorithms for Hard Knapsacks. In: Paterson, K.G. (ed.) Advances in Cryptology – EUROCRYPT 2011. pp. 364–385. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
4. Becker, A., Joux, A., May, A., Meurer, A.: Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In: EUROCRYPT 2012. pp. 520–536 (2012)
5. Berlekamp, E., McEliece, R., Van Tilborg, H.: On the inherent intractability of certain coding problems (corresp.). IEEE Transactions on Information Theory **24**(3), 384–386 (1978)
6. Bernstein, D.J., Lange, T., Peters, C.: Attacking and Defending the McEliece Cryptosystem. In: Buchmann, J., Ding, J. (eds.) Post-Quantum Cryptography. pp. 31–46. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
7. Both, L., May, A.: Decoding linear codes with high error rate and its impact for LPN security. In: International Conference on Post-Quantum Cryptography. pp. 25–46 (2018)
8. Canto Torres, R., Sendrier, N.: Analysis of Information Set Decoding for a Sublinear Error Weight. In: Takagi, T. (ed.) Post-Quantum Cryptography. pp. 144–161. Springer International Publishing, Cham (2016)
9. Castryck, W., Decru, T.: An Efficient Key Recovery Attack on SIDH. In: Advances in Cryptology – EUROCRYPT 2023: 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V. p. 423–447. Springer-Verlag, Berlin, Heidelberg (2023)
10. Drăgoi, V., Richmond, T., Bucerzan, D., Legay, A.: Survey on cryptanalysis of code-based cryptography: From theoretical to physical attacks. In: 2018 7th International Conference on Computers Communications and Control (ICCCC). pp. 215–223 (2018). https://doi.org/10.1109/ICCCC.2018.8390461
11. Ducas, L., Esser, A., Etinski, S., Kirshanova, E.: Asymptotics and Improvements of Sieving for Codes. Cryptology ePrint Archive, Paper 2023/1577 (2023), https://eprint.iacr.org/2023/1577, https://eprint.iacr.org/2023/1577
12. Dumer, I.: On minimum distance decoding of linear codes. In: Proc. 5th Joint Soviet-Swedish Int. Workshop Inform. Theory. pp. 50–52 (1991)

13. Esser, A.: Revisiting Nearest-Neighbor-Based Information Set Decoding. Cryptology ePrint Archive, Paper 2022/1328 (2023), https://eprint.iacr.org/2022/1328, https://eprint.iacr.org/2022/1328
14. Esser, A., Bellini, E.: Syndrome Decoding Estimator. In: Public-Key Cryptography – PKC 2022. pp. 112–141 (2022)
15. Esser, A., May, A., Zweydinger, F.: McEliece Needs a Break – Solving McEliece-1284 and Quasi-Cyclic-2918 with Modern ISD. In: Advances in Cryptology – EUROCRYPT 2022. pp. 433–457 (2022)
16. Esser, A., Verbel, J., Zweydinger, F., Bellini, E.: `CryptographicEstimators`: a Software Library for Cryptographic Hardness Estimation. Cryptology ePrint Archive, Paper 2023/589 (2023), https://eprint.iacr.org/2023/589, https://eprint.iacr.org/2023/589
17. Esser, A., Zweydinger, F.: New Time-Memory Trade-Offs for Subset Sum – Improving ISD in Theory and Practice. In: Advances in Cryptology – EUROCRYPT 2023. pp. 360–390 (2023)
18. Guo, Q., Johansson, T., Nguyen, V.: A New Sieving-Style Information-Set Decoding Algorithm. Cryptology ePrint Archive, Paper 2023/247 (2023), https://eprint.iacr.org/2023/247, https://eprint.iacr.org/2023/247
19. Guo, Q., Johansson, T., Stankovski, P.: A key recovery attack on MDPC with CCA security using decoding errors. In: Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Proceedings. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10031 LNCS, pp. 789–815. Springer (2016)
20. Hamdaoui, Y., Sendrier, N.: A Non Asymptotic Analysis of Information Set Decoding. IACR Cryptol. ePrint Arch. **2013**, 162 (2013), https://api.semanticscholar.org/CorpusID:17721683
21. Kachigar, G., Tillich, J.P.: Quantum Information Set Decoding Algorithms. In: Lange, T., Takagi, T. (eds.) Post-Quantum Cryptography. pp. 69–89. Springer International Publishing, Cham (2017)
22. Kirshanova, E.: Improved Quantum Information Set Decoding. In: Lange, T., Steinwandt, R. (eds.) Post-Quantum Cryptography. pp. 507–527. Springer International Publishing, Cham (2018)
23. Kirshanova, E., May, A.: Decoding McEliece with a Hint – Secret Goppa Key Parts Reveal Everything. In: Security and Cryptography for Networks: 13th International Conference, SCN 2022, Amalfi (SA), Italy, September 12–14, 2022, Proceedings. p. 3–20. Springer-Verlag, Berlin, Heidelberg (2022)
24. Li, Y., Wang, L.P.: Security analysis of the Classic McEliece, HQC and BIKE schemes in low memory. Journal of Information Security and Applications **79**, 103651 (2023). https://doi.org/https://doi.org/10.1016/j.jisa.2023.103651
25. May, A., Meurer, A., Thomae, E.: Decoding Random Linear Codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In: ASIACRYPT 2011. pp. 107–124 (2011)
26. May, A., Ozerov, I.: On computing nearest neighbors with applications to decoding of binary linear codes. In: EUROCRYPT 2015. pp. 203–228 (2015)
27. McEliece, R.J.: A Public-Key Cryptosystem Based On Algebraic Coding Theory. Deep Space Network Progress Report **44**, 114–116 (Jan 1978)
28. Narisada, S., Fukushima, K., Kiyomoto, S.: Fast GPU Implementation of Dumer's Algorithm Solving the Syndrome Decoding Problem. In: IEEE ISPA 2021. pp. 971–977 (2021)

29. Narisada, S., Fukushima, K., Kiyomoto, S.: Multiparallel MMT: Faster ISD Algorithm Solving High-Dimensional Syndrome Decoding Problem. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **E106.A**(3), 241–252 (2023). https://doi.org/10.1587/transfun.2022CIP0023
30. National Institute of Standards and Technology: PQC Standardization Process: Announcing Four Candidates to be Standardized, Plus Fourth Round Candidates (2022), https://csrc.nist.gov/news/2022/pqc-candidates-to-be-standardized-and-round-4
31. Peters, C.: Information-set decoding for linear codes over F q. In: International Workshop on Post-Quantum Cryptography. pp. 81–94 (2010)
32. Prange, E.: The use of information sets in decoding cyclic codes. IRE Transactions on Information Theory **8**(5), 5–9 (1962)
33. Schroeppel, R., Shamir, A.: A $T = O(2^{n/2})$, $S = O(2^{n/4})$ Algorithm for Certain NP-Complete Problems. SIAM Journal on Computing **10**(3), 456–464 (1981). https://doi.org/10.1137/0210033, https://doi.org/10.1137/0210033
34. Sendrier, N.: Decoding One Out of Many. In: Post-Quantum Cryptography. pp. 51–67 (2011)
35. Stern, J.: A method for finding codewords of small weight. In: Coding Theory and Applications. pp. 106–113 (1989)