On the hardness of the Lee syndrome decoding problem

note finali coverpage

(Article begins on next page)

18 July 2024

1   **On the Hardness of the Lee Syndrome Decoding Problem**

2   Violetta Weger, Karan Khathuria, Anna-Lena Horlemann, Massimo Battaglioni, Paolo Santini,
3   and Edoardo Persichetti

Violetta Weger

Department of Electrical and Computer Engineering
Technical University of Munich
Theresienstrasse 90, 80333 München

Karan Khathuria

Institute of Computer Science
University of Tartu
51009 Tartu, Estonia

Anna-Lena Horlemann

School of Computer Science
University of St. Gallen
Dufourstrasse 50, CH-9000 St. Gallen, Switzerland

Massimo Battaglioni and Paolo Santini

Department of Information Engineering
Marche Polytechnic University
Via Brecce Bianche 12, 60131 Ancona, Italy

Edoardo Persichetti

Department of Mathematical Sciences
Florida Atlantic University
777 Glades Road, Boca Raton, FL 33431, U.S.A.

(Communicated by the associate editor name)

Abstract. In this paper we study the hardness of the syndrome decoding
problem over finite rings endowed with the Lee metric. We first prove that the
decisional version of the problem is NP-complete, by a reduction from the 3-
dimensional matching problem. Then, we study the complexity of solving the
problem, by translating the best known solvers in the Hamming metric over
finite fields to the Lee metric over finite rings, as well as proposing some novel
solutions. For the analyzed algorithms, we assess the computational complexity
in the asymptotic regime and compare it to the corresponding algorithms in
the Hamming metric.

1   1. INTRODUCTION

NP-complete problems play a fundamental role in cryptography, since systems based on these problems offer strong security arguments. In this work, we focus on the *Syndrome Decoding Problem (SDP)*, that is, the problem of decoding an arbitrary linear code, which is the basis of code-based cryptography. This branch of public-key cryptography originated from the the seminal work of McEliece in 1978 [30] and Niederreiter in 1986 [33], and is currently regarded as one of the most studied and consolidated areas in post-quantum cryptography [16]. SDP in the Hamming metric was proven to be NP-complete over the binary finite field in [9] and over arbitrary finite fields in [5].

In order to solve this problem, an adversary is forced to apply a generic decoding algorithm. Currently, the best known algorithms of this type, for the Hamming metric, are those in the *Information Set Decoding (ISD)* family. These exponential-time algorithms are very well studied and understood, and constitute an important tool to determine the size of the parameters that are needed to achieve a given security level (for an overview see [31, 4]).

In the last few years, in order to reduce the size of the public keys associated to code-based cryptosystems, there has been a growing interest in exploring metrics other than Hamming's; for example, in [19] the hardness of the rank SDP and in [36] the hardness of the sum-rank SDP were studied. By exploring these new settings one might find a metric or an ambient space that is better suited for certain scenarios, in the sense that ISD algorithms are more costly than in the classical case, for a given size of input. In this paper we study the hardness of decoding a random linear code over $\mathbb{Z}/m\mathbb{Z}$ equipped with the Lee metric. This metric, first introduced in [26, 41], has recently received attention within the cryptographic community, for example in [23] where codes defined over $\mathbb{Z}/4\mathbb{Z}$ in the Lee metric were considered. After the online version of this manuscript was posted, the authors of [15] published a related article, where they propose classical and quantum algorithms based on Wagner's approach to solve the Hamming and the Lee metric SDP over finite fields, thus restricting to $\mathbb{Z}/p\mathbb{Z}$ for the Lee metric. In the common scenarios, the results in [15] match with ours: ISD algorithms in the Lee metric have a larger cost than their classical counterparts and thus we encourage further research of the Lee metric and its applications in cryptography. In particular, the Lee metric has the potential of providing shorter sizes in a zero-knowledge identification scheme such as Stern's [39], and thus, via the Fiat-Shamir transform, [17] also in signature schemes. Apart from cryptographic purposes, devising a generic decoding algorithm for this metric is interesting per se, from a theoretical point of view.

In this paper we show how the reduction from [5] also proves the NP-completeness of SDP over any finite ring endowed with an additive weight, i.e., where the weight of a vector is given by the sum of the weights of its entries. Thus, this proof is also valid for codes defined over $\mathbb{Z}/m\mathbb{Z}$, equipped with the Lee metric. In addition, we provide algorithms to solve SDP in the Lee metric over $\mathbb{Z}/p^s\mathbb{Z}$, where p is a prime and $s \in \mathbb{N}$ and perform a complexity analysis. In particular, we translate the ideas of algorithms such as Stern's ISD algorithm and the BJMM ISD algorithm to our setting but also propose new algorithms that are tailored to the particular structure of a parity-check matrix over $\mathbb{Z}/p^s\mathbb{Z}$. For the asymptotic complexity analysis, we also require additional computations with respect to the results in [20], where only the case where p is odd was treated. We observe that the ISD algorithms in the Lee metric have a larger cost than in the Hamming metric, which makes the Lee

metric a promising alternative for the Hamming metric regarding cryptographic applications.

The paper is organized as follows: in Section 2 we introduce the notation used throughout the paper and formulate some general properties of the Lee metric. The proof of the NP-completeness of the Lee metric version of the SDP and the Given Weight Codeword Problem (GWCP) is given in Section 3. In Section 4 we present information set decoding algorithms over $\mathbb{Z}/p^s\mathbb{Z}$ with respect to the Lee metric and assess their complexities. In Section 5 we compare the asymptotic complexity of all the Lee ISD algorithms and the Hamming ISD algorithms. Finally, we draw some concluding remarks in Section 6.

2. Notation and Preliminaries

In this section we provide the notation and the preliminaries used throughout the paper.

2.1. Notation. Let p be a prime, q a prime power and m a positive integer. We denote with $\mathbb{Z}/m\mathbb{Z}$ the ring of integers modulo m, and with $\mathbb{F}_q$ the finite field with q elements, as usual. Given an integer x, we denote its absolute value as $|x|$. The cardinality of a set V is denoted as $|V|$ and its complement by V^C. We use bold lower case (respectively, upper case) letters to denote vectors (respectively, matrices). By abuse of notation, a tuple in a module over a ring, will still be denoted by vector. The identity matrix of size k is denoted by Id_k; the $k \times n$ zero matrix is denoted as $\mathbf{0}_{k \times n}$, while $\mathbf{0}_n$ simply denotes the zero vector of length n. Given a vector $\mathbf{x}$ of length n and a set $S \subseteq \{1, \ldots, n\}$, we denote by $\mathbf{x}_S$ the projection of $\mathbf{x}$ to the coordinates indexed by S. In the same way, $\mathbf{M}_S$ denotes the projection of the $k \times n$ matrix $\mathbf{M}$ to the columns indexed by S. The support of a vector $\mathbf{a}$ is defined as $\mathrm{S}(\mathbf{a}) := \{j \mid a_j \neq 0\}$. For $S \subseteq \{1, \ldots, n\}$, we denote by $(\mathbb{Z}/m\mathbb{Z})^n(S)$ the vectors in $(\mathbb{Z}/m\mathbb{Z})^n$ having support in S.

2.2. Coding Theory in the Lee Metric. In this section we recall the definitions and main properties of linear codes over finite rings endowed with the Lee metric, see also [8].

Definition 2.1. For $x \in \mathbb{Z}/m\mathbb{Z}$ we define the *Lee weight* to be
$$\mathrm{wt}_L(x) := \min\{x, \mid m - x \mid\},$$
Then, for $\mathbf{x} \in (\mathbb{Z}/m\mathbb{Z})^n$, we define the *Lee weight* to be the sum of the Lee weights of its coordinates,
$$\mathrm{wt}_L(\mathbf{x}) := \sum_{i=1}^n \mathrm{wt}_L(x_i).$$
We define the *Lee distance* of $\mathbf{x}, \mathbf{y} \in (\mathbb{Z}/m\mathbb{Z})^n$ as
$$\mathrm{d}_L(\mathbf{x}, \mathbf{y}) := \mathrm{wt}_L(\mathbf{x} - \mathbf{y}).$$
The Lee sphere and the Lee ball of radius w are denoted by
$$S_L(n, w, m) := \{\mathbf{v} \in (\mathbb{Z}/m\mathbb{Z})^n \mid \mathrm{wt}_L(\mathbf{v}) = w\},$$
$$B_L(n, w, m) := \{\mathbf{v} \in (\mathbb{Z}/m\mathbb{Z})^n \mid \mathrm{wt}_L(\mathbf{v}) \leq w\},$$
respectively. We denote their sizes by $F_L(n, w, m) := |S_L(n, w, m)|$ and $V_L(n, w, m) := |B_L(n, w, m)|$, respectively, and study their asymptotics in Subsection 2.3.

Definition 2.2. Let m be a positive integer. A *ring-linear code* over $\mathbb{Z}/m\mathbb{Z}$ of length n is a $\mathbb{Z}/m\mathbb{Z}$-submodule of $(\mathbb{Z}/m\mathbb{Z})^n$.

The *type* of $\mathcal{C} \subseteq (\mathbb{Z}/m\mathbb{Z})^n$ is defined as

$$k = \log_m(|\,\mathcal{C}\,|)$$

and the *rate* of $\mathcal{C}$ is then given by $R = k/n$.

The *minimum Lee distance* $d_L(\mathcal{C})$ of a code $\mathcal{C} \subseteq (\mathbb{Z}/m\mathbb{Z})^n$ is the minimum of all Lee distances of distinct codewords of $\mathcal{C}$:

$$d_L(\mathcal{C}) = \min\{d_L(x,y) \mid x \neq y \in \mathcal{C}\}.$$

Linear codes can be completely represented through a generator or a parity-check matrix.

Definition 2.3. A matrix $\mathbf{G}$ is called a *generator matrix* for a (ring) linear code $\mathcal{C}$ if its row space corresponds to $\mathcal{C}$. In addition, we call a matrix $\mathbf{H}$ a *parity-check matrix* for $\mathcal{C}$ if its kernel corresponds to $\mathcal{C}$.

Note that such generator and parity-check matrices are not unique. If m is not prime, even the number of rows of such matrices is not unique.

From Section 4 onward we will restrict ourselves to $\mathbb{Z}/p^s\mathbb{Z}$, for some prime p and positive integer s, since $\mathbb{Z}/p^s\mathbb{Z}$ provides the most studied case for ring-linear codes.

Using the fundamental theorem of finite abelian groups, we note that any code $\mathcal{C} \subseteq (\mathbb{Z}/p^s\mathbb{Z})^n$ is isomorphic to

$$(\mathbb{Z}/p^s\mathbb{Z})^{k_1} \times (\mathbb{Z}/p^{s-1}\mathbb{Z})^{k_2} \times \cdots \times (\mathbb{Z}/p\mathbb{Z})^{k_s},$$

for some $k_1, \ldots, k_s \in \mathbb{N}$. Thus, we can give the code additional parameters: $(k_1, \ldots, k_s)$ is called the *subtype* of $\mathcal{C}$ and $K = \sum_{i=1}^{n} k_i$ is called the *rank*. Moreover, the parameter k_1 is called the *free rank* of $\mathcal{C}$, and if $k_1 = k = K$, then the code is said to be *free*.

Proposition 1 (Systematic Form). *Let $\mathcal{C}$ be a linear code in $(\mathbb{Z}/p^s\mathbb{Z})^n$ of subtype $(k_1, \ldots, k_s)$. Then $\mathcal{C}$ is permutation equivalent to a code having the following systematic parity-check matrix $\mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times n}$,*

$$(1) \qquad \mathbf{H} = \begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \cdots & \mathbf{A}_{1,s-1} & \mathbf{A}_{1,s} & \mathrm{Id}_{n-K} \\ p\mathbf{A}_{2,1} & p\mathbf{A}_{2,2} & \cdots & p\mathbf{A}_{2,s-1} & p\mathrm{Id}_{k_s} & \mathbf{0}_{k_s\times(n-K)} \\ p^2\mathbf{A}_{3,1} & p^2\mathbf{A}_{3,2} & \cdots & p^2\mathrm{Id}_{k_{s-1}} & \mathbf{0}_{k_{s-1}\times k_s} & \mathbf{0}_{k_{s-1}\times(n-K)} \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ p^{s-1}\mathbf{A}_{s,1} & p^{s-1}\mathrm{Id}_{k_2} & \cdots & \mathbf{0}_{k_2\times k_{s-1}} & \mathbf{0}_{k_2\times k_s} & \mathbf{0}_{k_2\times(n-K)} \end{pmatrix},$$

where $\mathbf{A}_{1,j} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K)\times k_j}, \mathbf{A}_{i,j} \in (\mathbb{Z}/p^{s+1-i}\mathbb{Z})^{k_{s-i+2}\times k_j}$ for $i > 1$.

In order to define information set decoders for the codes we consider, we need a suitable notion of information set, which is given next.

Definition 2.4. Consider a code $\mathcal{C}$ over $\mathbb{Z}/p^s\mathbb{Z}$ of length n and type k. An *information set* of $\mathcal{C}$ is a set $I \subseteq \{1, \ldots, n\}$ of minimal size such that

$$|\,\{\mathbf{c}_I \mid \mathbf{c} \in \mathcal{C}\}\,| = |\,\mathcal{C}\,|.$$

Using parity-check matrices of the form (1), we can see that an information set for the respective code has cardinality K, corresponding to the first K columns of $\mathbf{H}$.

1   Taking a random code over $\mathbb{Z}/p^s\mathbb{Z}$, it was shown in [11] that we have to consider
2 the parity-check matrix in such a form, since the probability of having a non-free
3 code is non-negligible.

2.3. ASYMPTOTICS. In this section we provide the asymptotic analysis of the volume of the Lee ball (and Lee sphere) and the Gilbert-Varshamov bound. In the whole section, we assume that $q = p^s$ and define $M := \lfloor \frac{q}{2} \rfloor$. Furthermore, we assume that all the code parameters $k = k(n)$, $K = K(n)$, $k_1 = k_1(n)$ are functions of n, with the usual restrictions

$$0 \leq k_1(n) \leq k(n) \leq K(n) \leq n.$$

4 Further, let $t = t(n)$ be another function of n, which corresponds to the Lee weight
5 of a vector of length n, hence it satisfies $0 \leq t(n) \leq Mn$.

Since these quantities are bounded, we may define their relative limits in n

$$\lim_{n\to\infty} \frac{k(n)}{n} =: R, \qquad \lim_{n\to\infty} \frac{t(n)}{n} =: T,$$
$$\lim_{n\to\infty} \frac{k_1(n)}{n} =: R_1, \qquad \lim_{n\to\infty} \frac{K(n)}{n} =: R_I.$$

The volume of the Lee balls (and spheres) can be approximated with the following results. We will use generating functions of the sizes of Lee spheres and Lee balls, which are known (see e.g. [2]) to be $f(x)^n$ and $\frac{f(x)^n}{1-x}$, respectively, where

$$f(x) := \begin{cases} 1 + 2\sum_{i=1}^{M} x^i & \text{if } q \text{ is odd,} \\ 1 + 2\sum_{i=1}^{M-1} x^i + x^M & \text{if } q \text{ is even.} \end{cases}$$

6 We denote the coefficient of x^t in a function $\Phi(x)$ by $[x^t]\Phi(x)$.
7   We will make use of the following reformulation of a result from [20] for estimating
8 the size of the Lee balls and Lee spheres, for large n.

Lemma 2.5. *[20, Corollary 1] Let $\Phi(x) = f(x)^n g(x)$ with $f(0) \neq 0$, and $t(n)$ be a function in n. Set $T := \lim_{n\to\infty} t(n)/n$ and ρ as the solution to*

$$\underbrace{\frac{xf'(x)}{f(x)}}_{=:\Delta(x)} = T.$$

If $\Delta'(\rho) > 0$, and the modulus of any singularity of $g(x)$ is larger than ρ, then for large n

$$\frac{1}{n}\log_q([x^{t(n)}]\Phi(x)) \approx \log_q(f(\rho)) - T\log_q(\rho) + o(1).$$

9   Using this, we get the following asymptotic behavior of Lee spheres and balls.

Lemma 2.6.   *1. If q is odd and $0 \leq T < M$, then*

$$\lim_{n\to\infty} \frac{1}{n}\log_q(F_L(n, t(n), q)) = \log_q(f(\rho)) - T\log_q(\rho),$$

where ρ is the unique real positive solution of $2\sum_{i=1}^{M}(i-T)x^i = T$ and

$$f(\rho) = 1 + 2\sum_{i=1}^{M}\rho^i = \frac{M(\rho+1)+1}{(1-\rho)(M-T)+1}.$$

If moreover $T < \frac{M(M+1)}{2M+1}$, then also

$$\lim_{n \to \infty} \frac{1}{n} \log_q(V_L(n, t(n), q)) = \log_q(f(\rho)) - T \log_q(\rho),$$

2. *If q is even and $0 \leq T < M$, then*

$$\lim_{n \to \infty} \frac{1}{n} \log_q(F_L(n, t(n), q)) = \log_q(g(\rho')) - T \log_q(\rho'),$$

where ρ' is the unique real positive solution of

$$2 \sum_{i=1}^{M-1} (i - T)x^i + (M - T)x^M = T$$

and

$$
\begin{aligned}
g(\rho') &= 1 + 2 \sum_{i=1}^{M-1} \rho'^i + \rho'^M \\
&= \frac{\rho'^{M+1}(T - M) + \rho'^M(T - M + 1) + \rho'(T - M) + T + M + 1}{\rho'(T - M) + M + 1 - T}.
\end{aligned}
$$

If moreover $T < \frac{M}{2}$, then also

$$\lim_{n \to \infty} \frac{1}{n} \log_q(V_L(n, t(n), q)) = \log_q(g(\rho')) - T \log_q(\rho').$$

Proof. 1. If q is odd, with $f(z) := 1 + 2\sum_{i=1}^{M} x^i$, we have that the generating functions for $F_L(n, t, q)$ and $V_L(n, t, q)$ are given by $f(x)^n$ and $\frac{f(x)^n}{1-x}$, respectively. The statement now follows from Lemma 2.5, where the derivation of $f(\rho)$ can be found in the proof of [20, Theorem 4]. Notice that we do not need any restriction on T in the first case, since here $g(x) = 1$ does not have any singularities. In the second case the only singularity of $g(x) = \frac{1}{1-x}$ is 1, hence we need that $\rho < 1$. This is provided with the condition on T, since $\Delta(x) = \frac{2\sum_{i=1}^{M} ix^i}{1 + 2\sum_{i=1}^{M} x^i}$ is strictly increasing for $x \geq 1$ and

$$\Delta(1) = \frac{M(M+1)}{2M+1} > T = \Delta(\rho).$$

2. If q is even, with $g(z) := 1 + 2\sum_{i=1}^{M-1} x^i + x^M$, we have again that the generating functions for $F_L(n, t, q)$ and $V_L(n, t, q)$ are given by $g(x)^n$ and $\frac{g(x)^n}{1-x}$, respectively, and that the statement follows from Lemma 2.5, where the derivation of $g(\rho)$ can be found in the proof of [20, Theorem 4]. We again need that $\rho < 1$ in the second case, because $\Delta(x) = \frac{2\sum_{i=1}^{M-1} ix^i + Mx^M}{1 + 2\sum_{i=1}^{M-1} x^i + x^M}$ is strictly increasing for $x \geq 1$ and

$$\Delta(1) = \frac{M}{2} > T = \Delta(\rho).$$

$\square$

Remark 1. Notice that asymptotically the size of the ball equals the size of the largest sphere inside the ball, as long as T fulfills the prescribed conditions, which is approximately $T < M/2$ for both cases.

Note that if q is odd and $T \geq M(M+1)/(2M+1)$ or if q is even and $T \geq M/2$, we get that $\lim_{n\to\infty} \frac{1}{n} \log_q(V_L(n, t(n), q)) = 1$. This can be easily observed as

$$
\begin{aligned}
1 &\geq \lim_{n\to\infty} \frac{1}{n} \log_q(V_L(n, t(n), q)) \\
&\geq \begin{cases} \lim_{n\to\infty} \frac{1}{n} \log_q(V_L(n, M(M+1)/(2M+1), q)) & \text{if } q \text{ is odd} \\ \lim_{n\to\infty} \frac{1}{n} \log_q(V_L(n, M/2, q)) & \text{if } q \text{ is even} \end{cases} \\
&= 1.
\end{aligned}
$$

The last equality follows from Lemma 2.6, where we get $\rho = 1$ and $\rho' = 1$, respectively.

Let $AL(n, d, q)$ denote the maximal cardinality of a code $\mathcal{C} \subseteq (\mathbb{Z}/q\mathbb{Z})^n$ of minimum Lee distance d and let us consider the maximal information rate

$$
R(n, d, q) := \frac{1}{n} \log_q(AL(n, d, q)),
$$

for $0 \leq d \leq nM$. We define the relative minimum distance to be $\delta := \frac{d}{nM}$.

Theorem 2.7 (Asymptotic Gilbert-Varshamov Bound [3])**.** *It holds that*

$$
\liminf_{n\to\infty} R(n, \delta M n, q) \geq \lim_{n\to\infty} \left(1 - \frac{1}{n} \log_q(V_L(n, \delta M n, q)) \right).
$$

In [11, Theorem 22] it is shown that a code generated by a random matrix achieves the Gilbert-Varshamov bound with high probability.

In ISD complexity analyses, we are interested in computing $\lim_{n\to\infty} \frac{1}{n} \log_q(f(n))$, where $f(n)$ denotes the actual complexity function, all polynomial terms will become negligible. Note that a finite sum results in the asymptotics in a maximum of the limits of the summands, i.e., for a function $g(i)$ with $\lim_{n\to\infty} 1/n \log_q(g(i)) = G(i)$, we have that

$$
\lim_{n\to\infty} 1/n \log_q \left(\sum_{i=1}^{\ell} g(i) \right) = \max\{G(i) \mid i \in \{1, \ldots, \ell\}\}.
$$

2.4. UNIFORM DISTRIBUTION OVER $\mathbb{Z}/p^s\mathbb{Z}$. In the complexity analysis we will assume that a random vector-matrix product is uniformly distributed. In fact, this is a key part for ISD algorithms, as we study the average-case complexity. The following lemma gives the mathematical base for this assumption.

Lemma 2.8. *Consider a random vector $\mathbf{x} \in (\mathbb{Z}/p^s\mathbb{Z})^K$.*

    1. If $\mathbf{x} \in p(\mathbb{Z}/p^s\mathbb{Z})^K$, then $p | \mathrm{wt}_L(\mathbf{x})$.

    2. If $p | \mathrm{wt}_L(\mathbf{x})$, then $\mathbf{x} \in p(\mathbb{Z}/p^s\mathbb{Z})^K$ with probability $\frac{F_L(K, \frac{v}{p}, p^{s-1})}{F_L(K, v, p^s)}$.

    3. Let $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{K \times n}$ be chosen uniformly at random. Moreover, let $\mathbf{x} \in (\mathbb{Z}/p^s\mathbb{Z})^K$ be chosen uniformly at random among the vectors that contain a unit, i.e., that do not live in $p(\mathbb{Z}/p^s\mathbb{Z})^K$, possibly with a weight restriction. Then $\mathbf{x}\mathbf{A}$ is uniformly distributed in $(\mathbb{Z}/p^s\mathbb{Z})^n$.

Proof. 1. It follows from the definition of the Lee weight, that if $p | x_i$ for $i \in \{1, \ldots, K\}$, then $p | \mathrm{wt}_L(x_i)$, which implies the statement.

2. The number of $\mathbf{x} \in p(\mathbb{Z}/p^s\mathbb{Z})^K$ with $\mathrm{wt}_L(\mathbf{x}) = v$ is equal to the number of $\bar{\mathbf{x}} \in (\mathbb{Z}/p^{s-1}\mathbb{Z})^K$ with $\mathrm{wt}_L(\bar{\mathbf{x}}) = v/p$, which implies the statement.

3. Let x_i be a unit entry of $\mathbf{x}$. Then $\mathbf{x}\mathbf{A} = \left(\sum_{j \neq i} x_j \mathbf{A}_j \right) + x_i \mathbf{A}_i$, where $\mathbf{A}_i$ denotes

the i-th row of $\mathbf{A}$, and since $x_i \mathbf{A}_i$ is uniformly distributed in $(\mathbb{Z}/p^s\mathbb{Z})^n$, so is $\mathbf{x}\mathbf{A}$.

□

Note that in the SDP it is assumed that $\mathbf{s}$ is any element of $(\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$. For ISD algorithms, however, it is assumed that $\mathbf{s}$ is indeed a syndrome, that is, that there exists a solution $\mathbf{e}$ of weight t such that $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$.

In addition, ISD algorithms assume that the input $\mathbf{H}$ is chosen uniformly at random, which by Lemma 2.8 implies that $\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$ is uniformly distributed.

3. The NP-Completeness of the General Syndrome Decoding Problem

In this section we consider a more general definition of weight and prove that the Syndrome Decoding Problem (SDP) is NP-complete in this general setting, which includes the Lee weight setting as a special case. Note that all problems in this section are easily seen to be in NP, since testing a solution clearly requires polynomial time. Hence showing NP-hardness automatically implies NP-completeness.

Let R be a finite ring with identity, denoted by 1 and $\mathrm{wt} : R \to \mathbb{R}_{\geq 0}$ be a function on R that satisfies the following properties:

1. $\mathrm{wt}(0) = 0$,
2. $\mathrm{wt}(1) = 1$ and $\mathrm{wt}(x) \geq 1$ for all $x \neq 0$.

We call such a function a *weight* (function). We extend this weight additively to R^n by writing $\mathrm{wt}(\mathbf{x}) = \sum_{i=1}^{n} \mathrm{wt}(x_i)$ for all $\mathbf{x} = (x_1, \ldots, x_n) \in R^n$. In this section, we restrict ourselves to such *additive weights*. Clearly, the Hamming weight and the Lee weight are examples of additive weights.

Note that all results of this section hold also for any weight fulfilling condition 1. and a more general version of condition 2., i.e., $\mathrm{wt}(x) \geq \lambda$ for all $x \neq 0$ and $\mathrm{wt}(1) = \lambda$ for some $\lambda \in \mathbb{R}_{>0}$. The argument is then similar, due to scaling.

We then define the syndrome decoding problem as:

Problem 1. (R, wt)-**Syndrome Decoding Problem** $((R, \mathrm{wt})$-**SDP**)
Given $\mathbf{H} \in R^{(n-k) \times n}$, $\mathbf{s} \in R^{n-k}$ and $t \in \mathbb{N}$, is there a vector $\mathbf{e} \in R^n$ such that $\mathrm{wt}(\mathbf{e}) \leq t$ and $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$?

Berlekamp, McEliece and van Tilborg proved in [9] the NP-completeness of the syndrome decoding problem for the case of binary linear codes equipped with the Hamming metric (i.e., employing the Hamming weight in the problem definition). In particular, their proof was based on a reduction from the 3-dimensional matching (3DM) problem to SDP. In [5], Barg generalized this proof to an arbitrary alphabet size.

Proposition 2. *The (R, wt)-SDP is NP-complete for any finite ring R with identity and for any additive weight.*

As a corollary we obtain that also SDP in the Lee metric is NP-complete.

Since the proof of Proposition 2 is very similar to the original proof of [9] and [5], we include it in the appendix for the sake of completeness.

Let R be a finite ring with identity and $k \leq n$ be positive integers. Let wt be an additive weight function on R^n and $k \leq n$ be positive integers. We now define the given weight codeword problem as follows:

Problem 2. (R, wt)-***Given Weight Codeword Problem (***(R, wt)***-GWCP)***

Given $\mathbf{H} \in R^{(n-k) \times n}$ *and* $w \in \mathbb{N}$*, is there a vector* $\mathbf{c} \in R^n$ *such that* $\mathrm{wt}(\mathbf{c}) = w$ *and* $\mathbf{c}\mathbf{H}^\top = \mathbf{0}_{n-k}$*?*

In order to prove the NP-completeness of of Problem 2, we give a polynomial time reduction from the 3-dimensional matching problem.

Problem 3. *3-Dimensional Matching (3DM) Problem*

Let T *be a finite set and* $U \subseteq T \times T \times T$*. Given* U, T*, decide if there exists a set* $W \subseteq U$ *such that* $|W| = |T|$ *and no two elements of* W *agree in any coordinate.*

Proposition 3. *The* (R, wt)*-GWCP is NP-complete for any finite ring* R *with identity and for any additive weight.*

The proof is similar to the original reduction of [9, 5]; however, the adaptation is not trivial and therefore we report it next.

Proof. We prove the NP-completeness by a reduction from the 3DM problem. To this end, we start with a random instance of 3DM, i.e., T of size t, and $U \subseteq T \times T \times T$ of size u. Let us denote the elements in $T = \{b_1, \ldots, b_t\}$ and in $U = \{\mathbf{a}_1, \ldots, \mathbf{a}_u\}$. From this we build the matrix $\overline{\mathbf{H}}^\top \in R^{u \times 3t}$, like in [9]:

- for $j \in \{1, \ldots, t\}$, we set $h_{i,j} = 1$ if $\mathbf{a}_i[1] = b_j$ and $h_{i,j} = 0$ else,
- for $j \in \{t+1, \ldots, 2t\}$, we set $h_{i,j} = 1$ if $\mathbf{a}_i[2] = b_j$ and $h_{i,j} = 0$ else,
- for $j \in \{2t+1, \ldots, 3t\}$, we set $h_{i,j} = 1$ if $\mathbf{a}_i[3] = b_j$ and $h_{i,j} = 0$ else.

Let M be the maximal weight of an element in R. Then we construct $\mathbf{H}^\top \in R^{(3tMu+3t+u) \times (3tMu+3t)}$ in the following way.

$$\mathbf{H}^\top = \begin{pmatrix} \overline{\mathbf{H}}^\top & \mathrm{Id}_u & \cdots & \mathrm{Id}_u \\ -\mathrm{Id}_{3t} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & -\mathrm{Id}_u & & \mathbf{0} \\ \vdots & & \ddots & \\ \mathbf{0} & \mathbf{0} & & -\mathrm{Id}_u \end{pmatrix},$$

where we have repeated the size-u identity matrix $3tM$ times in the first row. Let us set $w = 3t^2 M^2 + 4tM$ and assume that we can solve the (R, wt)-GWCP on the instance given by $\mathbf{H}, w$ in polynomial time. Let us consider two cases.

Case 1: In the first case the (R, wt)-GWCP solver returns as answer 'yes', since there exists a $\mathbf{c} \in R^{3tMu+3t+u}$, of weight equal to w, such that $\mathbf{c}\mathbf{H}^\top = \mathbf{0}_{3tMu+3t}$. Let us write this $\mathbf{c}$ as

$$\mathbf{c} = (\overline{\mathbf{c}}, \mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{3tM}),$$

where $\overline{\mathbf{c}} \in R^u, \mathbf{c}_0 \in R^{3t}$ and $\mathbf{c}_i \in R^u$ for all $i \in \{1, \ldots, 3tM\}$. Then, $\mathbf{c}\mathbf{H}^\top = \mathbf{0}_{3tMu+3t}$ gives the equations

$$\overline{\mathbf{c}}\,\overline{\mathbf{H}}^\top - \mathbf{c}_0 = \mathbf{0},$$

$$\overline{\mathbf{c}} - \mathbf{c}_1 = \mathbf{0},$$

$$\vdots$$

$$\overline{\mathbf{c}} - \mathbf{c}_{3tM} = \mathbf{0}.$$

Hence, we have that $\mathrm{wt}(\overline{\mathbf{c}}\,\overline{\mathbf{H}}^\top) = \mathrm{wt}(\mathbf{c}_0)$ and

$$\mathrm{wt}(\overline{\mathbf{c}}) = \mathrm{wt}(\mathbf{c}_1) = \cdots = \mathrm{wt}(\mathbf{c}_{3tM}).$$

Due to the coordinate-wise additivity of the weight, we have that

$$\mathrm{wt}(\mathbf{c}) = \mathrm{wt}(\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top}) + (3tM + 1)\mathrm{wt}(\overline{\mathbf{c}}).$$

Since $\mathrm{wt}(\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top}) \leq 3tM$, we have that $\mathrm{wt}(\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top})$ and $\mathrm{wt}(\overline{\mathbf{c}})$ are uniquely determined as the remainder and the quotient, respectively, of the division of $\mathrm{wt}(\mathbf{c})$ by $3tM + 1$. In particular, if $\mathrm{wt}(\mathbf{c}) = 3t^2M^2 + 4tM$, then we must have $\mathrm{wt}(\overline{\mathbf{c}}) = tM$ and $\mathrm{wt}(\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top}) = 3tM$. Hence, the first u parts of the found solution $\mathbf{c}$, i.e., $\overline{\mathbf{c}}$, give a matching for the 3DM in a similar way as in the proof of Proposition 2. For this we first observe that $\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top}$ is a full support vector and it plays the role of the syndrome, i.e., $\overline{\mathbf{c}}\overline{\overline{\mathbf{H}}}^{\top} = (x_1, \ldots, x_{3t})$, where $x_i \in R \setminus \{0\}$ having $\mathrm{wt}(x_i) = M$ for all $i \in \{1, \ldots, 3t\}$. We note that $\overline{\mathbf{c}}$ has exactly t non-zero entries, which corresponds to a solution of 3DM (for details see the proof of Proposition 2 in the appendix).

Case 2: If the solver returns as answer 'no', this is also the correct answer for the 3DM problem. In fact, it is easy to see that the above construction also associates any solution W of the 3DM to a solution $\mathbf{c}$ of the corresponding (R, wt)-GWCP.

Thus, if such a polynomial time solver for the (R, wt)-GWCP exists, we can also solve the 3DM problem in polynomial time. $\qquad\square$

Remark 2. We remark that the bounded version of this problem, i.e., deciding if a codeword $\mathbf{c}$ with $\mathrm{wt}(\mathbf{c}) \leq w$ exists, can be solved by applying the solver of Problem 2 at most w many times.

Remark 3. The computational versions of Problems 2 and 4 are at least as hard as their decisional counterparts. Trivially, any operative procedure that returns a vector with the desired properties (when it exists) can be used as a direct solver for the above problems.

As mentioned in the introduction, the previous results imply that the corresponding problems in the Lee metric are also NP-complete.

Problem 4. *Lee - Syndrome Decoding Problem (L-SDP) Let m and $k \leq n$ be positive integers. Given $\mathbf{H} \in (\mathbb{Z}/m\mathbb{Z})^{(n-k)\times n}$, $\mathbf{s} \in (\mathbb{Z}/m\mathbb{Z})^{n-k}$ and $t \in \mathbb{N}$, is there a vector $\mathbf{e} \in (\mathbb{Z}/m\mathbb{Z})^n$ such that $\mathrm{wt}_L(\mathbf{e}) \leq t$ and $\mathbf{e}\mathbf{H}^{\top} = \mathbf{s}$?*

Problem 5. *Given Lee Weight Codeword Problem (GLWCP) Let m and $k \leq n$ be positive integers. Given $\mathbf{H} \in (\mathbb{Z}/m\mathbb{Z})^{(n-k)\times n}$ and $w \in \mathbb{N}$, is there a vector $\mathbf{c} \in (\mathbb{Z}/m\mathbb{Z})^n$ such that $\mathrm{wt}_L(\mathbf{e}) = w$ and $\mathbf{c}\mathbf{H}^{\top} = \mathbf{0}_{n-k}$?*

Corollary 1. *The L-SDP and the GLWCP are NP-complete for any fixed $m \in \mathbb{N}$.*

4. INFORMATION SET DECODING IN THE LEE METRIC

After showing the NP-completeness of the syndrome decoding problem in the Lee metric, we now aim at assessing the complexity of solving it. For this, we adapt some well known ISD algorithms from the Hamming metric to the Lee metric, as well as derive new ISD algorithms that emerge from the special form of the parity-check matrix over a finite ring $\mathbb{Z}/p^s\mathbb{Z}$. For each algorithm we provide the asymptotic analysis of the average cost, where the input parity-check matrix and syndrome are uniformly distributed over $\mathbb{Z}/p^s\mathbb{Z}$.

The first ISD algorithm was proposed by Prange in 1962 [35] and assumes that no errors happen in a randomly chosen information set. Although one iteration requires a low computational cost, the whole algorithm has a large overall cost,

1 as many iterations have to be performed. There have been many improvements
2 upon the original algorithm by Prange, focusing on a more likely error pattern.
3 Indeed, these approaches increase the cost of one iteration but, on average, require
4 a smaller number of iterations (see [27, 28, 38, 12, 13, 14, 29, 18, 10, 6]). For an
5 overview of the binary case see [31, 4]. With new cryptographic schemes proposed
6 over general finite fields, most of these algorithms have been generalized to $\mathbb{F}_q$ (see
7 [34, 32, 21, 22, 25]).

8   In this paper we provide five different ISD algorithms for the ring $\mathbb{Z}/p^s\mathbb{Z}$ equipped
9 with the Lee metric. We start with the two-blocks algorithm, that may be seen as
10 a generalization of Stern's algorithm in the Hamming metric [38]. Due to the spe-
11 cial form of the parity-check matrix, dividing it in s-blocks, we also propose the
12 s-blocks algorithm, which at the best of our knowledge has no known counterpart
13 in the Hamming metric. In addition, we use the idea of Partial Gaussian Elimina-
14 tion (PGE) [18], where one reduces the initial SDP to a smaller SDP instance. For
15 this scenario, we provide both Wagner's approach [42] of partitioning and the rep-
16 resentation technique approach [7], where we allow subvectors to overlap. Finally,
17 following the idea of [7] we also introduce a combination of their approaches. We
18 notice that ISD algorithms in the Lee metric, for the special case of $\mathbb{Z}/4\mathbb{Z}$, have
19 already been studied in [23].

20   We have also considered generalized birthday decoding algorithms (GBA); how-
21 ever, we have found that they do not improve PGE algorithms. GBA is exploiting,
22 as the name suggests, the birthday paradox; this allows in the Hamming metric to
23 choose smaller list sizes, thus decreasing the cost and, at the same time, still finding
24 a solution. However, the situation in the Lee metric is different: even if we consider
25 the maximal list sizes, we run into the following problem: one has to ensure that
26 at least one vector leading to a solution lives in the lists. This can be enforced by
27 either repeating this step and thus employing this success probability or choosing
28 a small enough number of positions on which one merges. In the Lee metric both
29 solutions are not satisfactory, since we either have to choose the number of positions
30 for merging to be 0, or the lowest cost is achieved by using the algorithm on just
31 one level. Therefore, using even smaller lists would only worsen the situation and
32 the cost. However, if one is interested in such an approach, it can be found for
33 Wagner's algorithm over $\mathbb{Z}/p\mathbb{Z}$ in [15].

34 4.1. TWO-BLOCKS ALGORITHM. In this section, we adapt Stern's algorithm from
35 the Hamming metric to the Lee metric, which also encompasses the basic Lee-
36 Brickell's and Prange's algorithms as special cases.

37   The idea of Stern's algorithm in the Hamming metric is to partition the chosen
38 information set I into two sets X and Y containing v_1 and v_2 errors, respectively.
39 Moreover, it is assumed that there exists a zero-window Z of size z outside of the
40 information set where no errors happen. Our adaptation is as follows.

Let us assume that the information set is $I = \{1, \ldots, K\}$, and that the zero-
window is $Z = \{K+1, \ldots, K+z\}$; furthermore, let us define $J = \{1, \ldots, n\} \setminus (I \cup Z) = \{K+z+1, \ldots, n\}$. To bring the parity-check matrix $\mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1) \times n}$ into
systematic form, we multiply by an invertible matrix $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1) \times (n-k_1)}$.
We can write the error vector partitioned into the information set part I, the zero-
window part Z and the remaining part J as $\mathbf{e} = (\mathbf{e}_I, \mathbf{0}_z, \mathbf{e}_J)$, with $\mathrm{wt}_L(\mathbf{e}_I) = v_1 + v_2$

and $\text{wt}_L(\mathbf{e}_J) = t - v_1 - v_2$. We are in the following situation

$$\mathbf{e}\mathbf{H}^\top\mathbf{U}^\top = \begin{pmatrix} \mathbf{e}_I & \mathbf{0}_z & \mathbf{e}_J \end{pmatrix} \begin{pmatrix} \mathbf{A}^\top & \mathbf{B}^\top & p\mathbf{C}^\top \\ \text{Id}_z & \mathbf{0}_{z\times(n-K-z)} & \mathbf{0}_{z\times(K-k_1)} \\ \mathbf{0}_{(n-K-z)\times z} & \text{Id}_{n-K-z} & \mathbf{0}_{(n-K-z)\times(K-k_1)} \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{s}_1 & \mathbf{s}_2 & p\mathbf{s}_3 \end{pmatrix} = \mathbf{s}\mathbf{U}^\top,$$

where $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{z\times K}, \mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K-z)\times K}$, $\mathbf{C} \in (\mathbb{Z}/p^{s-1}\mathbb{Z})^{(K-k_1)\times K}$ and $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^z, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-z}, \mathbf{s}_3 \in (\mathbb{Z}/p^{s-1}\mathbb{Z})^{K-k_1}$. From this, we get the following three conditions

$$\tag{2} \mathbf{e}_I\mathbf{A}^\top = \mathbf{s}_1,$$

$$\tag{3} \mathbf{e}_I\mathbf{B}^\top + \mathbf{e}_J = \mathbf{s}_2,$$

$$\tag{4} p\mathbf{e}_I\mathbf{C}^\top = p\mathbf{s}_3.$$

In the algorithm, we will define a set P that contains all vectors of the form $\mathbf{e}_1\mathbf{A}^\top$ and $\mathbf{e}_1\mathbf{C}^\top$, and a set Q that contains all vectors of the form $\mathbf{s}_1 - \mathbf{e}_2\mathbf{A}^\top$ and $\mathbf{s}_3 - \mathbf{e}_2\mathbf{C}^\top$. Whenever a vector in P and a vector in Q coincide, we call such a pair a *collision*. For each collision, we define $\mathbf{e}_J$ such that Condition (3) is satisfied, *i.e.*,

$$\mathbf{e}_J = \mathbf{s}_2 - \mathbf{e}_I\mathbf{B}^\top.$$

If in addition $\text{wt}_L(\mathbf{e}_J) = t - v_1 - v_2$, we have found the wanted error vector. The two-blocks algorithm in the Lee metric is depicted in Algorithm 1.

For the following complexity analysis, we first recall the assumptions made in Section 2.3. Let all the code parameters $k(n), k_1(n), K(n), t(n)$ be functions of n, and define:

$$R := \lim_{n\to\infty}\frac{k(n)}{n}, R_1 := \lim_{n\to\infty}\frac{k_1(n)}{n}, R_I := \lim_{n\to\infty}\frac{K(n)}{n}, T := \lim_{n\to\infty}\frac{t(n)}{n}.$$

We fix the real numbers V, L with $0 \le V \le T/2$ and $0 \le L \le 1 - R$, such that $0 \le T - 2V \le M(1 - R - L)$. Then we fix the internal algorithm parameters $m_1 = m_2 = R_I/2$ and v_1, v_2, ℓ which we see as functions depending on n, such that $\lim_{n\to\infty}\frac{v_i}{n} = V$ and $\lim_{n\to\infty}\frac{\ell}{n} = L$.

In order to ease the asymptotic formulas, we introduce the following notation

$$S(R, V) := \lim_{n\to\infty}\frac{1}{n}\log_q(F_L(k, v, q)) = R\lim_{k\to\infty}\frac{1}{k}\log_q(F_L(k, v, q)),$$

$$L(R, V) := \lim_{n\to\infty}\frac{1}{n}\log_q(V_L(k, v, q)) = R\lim_{k\to\infty}\frac{1}{k}\log_q(V_L(k, v, q)),$$

for which we will use the results of Lemma 2.6.

Theorem 4.1. *Algorithm 1 over* $\mathbb{Z}/p^s\mathbb{Z}$ *equipped with the Lee metric has the following asymptotic average time complexity:*

$$-2S(R_I/2, V) - S(1 - R_I - L, T - 2V) + S(1, T) +$$
$$\max\{S(R_I/2, V), 2S(R_I/2, V) - L - R_I + R_1\}.$$

Proof. The only steps that contribute to the asymptotic complexity, are the construction of P and Q, finding the collisions between the two sets and the number of iterations required.

The construction of the sets P and Q costs asymptotically $S(R_I/2, V)$, which is the asymptotic size of the Lee sphere of weight v_i and length m_i.

Algorithm 1 Two-Blocks Algorithm over $\mathbb{Z}/p^s\mathbb{Z}$ in the Lee metric

---

Input: $\mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times n}$, $\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$, $t \in \mathbb{N}$, $K = m_1 + m_2, z < n - K$
and $v_i < \min\{\lfloor \frac{p^s}{2}\rfloor m_i, \lfloor \frac{t}{2}\rfloor\}$ for $i \in \{1,2\}$.
Output: $\mathbf{e} \in (\mathbb{Z}/p^s\mathbb{Z})^n$ with $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$ and $\mathrm{wt}_L(\mathbf{e}) = t$.

1: Choose an information set $I \subset \{1,...,n\}$ of size K and choose a zero-window
$\quad Z \subset I^C$ of size z, and define $J = (I \cup Z)^C$.
2: Partition I into X of size m_1 and Y of size $m_2 = K - m_1$.
3: Compute $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times(n-k_1)}$, such that

$$(\mathbf{UH})_I = \begin{pmatrix} \mathbf{A} \\ \mathbf{B} \\ p\mathbf{C} \end{pmatrix}, \ (\mathbf{UH})_Z = \begin{pmatrix} \mathrm{Id}_z \\ \mathbf{0}_{(n-K-z)\times z} \\ \mathbf{0}_{(K-k_1)\times z} \end{pmatrix}, \ (\mathbf{UH})_J = \begin{pmatrix} \mathbf{0}_{z\times(n-K-z)} \\ \mathrm{Id}_{n-K-z} \\ \mathbf{0}_{(K-k_1)\times(n-K-z)} \end{pmatrix},$$

$\quad$ where $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{z\times K}, \mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K-z)\times K}$ and $\mathbf{C} \in (\mathbb{Z}/p^{s-1}\mathbb{Z})^{(K-k_1)\times K}$.
4: Compute $\mathbf{s}\mathbf{U}^\top = \begin{pmatrix} \mathbf{s}_1 & \mathbf{s}_2 & p\mathbf{s}_3 \end{pmatrix}$, where $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^z, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-z}$ and
$\quad \mathbf{s}_3 \in (\mathbb{Z}/p^{s-1}\mathbb{Z})^{K-k_1}$.
5: Compute the set

$$P = \{(\mathbf{e}_1\mathbf{A}^\top, \mathbf{e}_1\mathbf{C}^\top, \mathbf{e}_1) \mid \mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^K(X), \ \mathrm{wt}_L(\mathbf{e}_1) = v_1\}.$$

6: Compute the set

$$Q = \{(\mathbf{s}_1 - \mathbf{e}_2\mathbf{A}^\top, \mathbf{s}_3 - \mathbf{e}_2\mathbf{C}^\top, \mathbf{e}_2) \mid \mathbf{e}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^K(Y), \ \mathrm{wt}_L(\mathbf{e}_2) = v_2\}.$$

7: **for** $(\mathbf{a},\mathbf{b},\mathbf{e}_1) \in P$ **do**
8: $\quad$ **for** $(\mathbf{a},\mathbf{b},\mathbf{e}_2) \in Q$ **do**
9: $\quad\quad$ **if** $\mathrm{wt}_L(\mathbf{s}_2 - (\mathbf{e}_1+\mathbf{e}_2)\mathbf{B}^\top) = t - v_1 - v_2$ **then**
10: $\quad\quad\quad$ Return $\mathbf{e}$, such that $\mathbf{e}_I = \mathbf{e}_1 + \mathbf{e}_2$, $\mathbf{e}_Z = \mathbf{0}_z$ and $\mathbf{e}_J = \mathbf{s}_2 - (\mathbf{e}_1+\mathbf{e}_2)\mathbf{B}^\top$.
11: Start over with Step 1 and a new selection of I.

In the next step, we check for collisions between P and Q. The resulting vectors $\mathbf{e}_1\mathbf{A}^\top$, respectively $\mathbf{s}_1 - \mathbf{e}_2\mathbf{A}^\top$, live in $(\mathbb{Z}/p^s\mathbb{Z})^z$, whereas the second part of the resulting vectors $\mathbf{e}_1\mathbf{C}^\top$ and $\mathbf{s}_3 - \mathbf{e}_2\mathbf{C}^\top$ lives in $p(\mathbb{Z}/p^s\mathbb{Z})^{(K-k_1)}$, and from Lemma 2.8 we can assume that they are uniformly distributed. If v_1, v_2 are chosen coprime to p, then this is certainly true. Therefore, we have to check asymptotically $2S(R_I/2, V) - L - R_I + R_1$ many times.

Finally, since the success probability of one iteration is given by

$$F_L(m_1, v_1, p^s)F_L(m_2, v_2, p^s)F_L(n - K - z, t - v_1 - v_2, p^s)F_L(n, t, p^s)^{-1},$$

the asymptotic number of iterations is given by

$$-2S(R_I/2, V) - S(1 - R_I - L, T - 2V) + S(1, T).$$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \square$

We remark that, if we set $m_1 = K$ and thus $m_2 = 0$, and ask for Lee weight $v_1 = v$ and $v_2 = 0$ in the above algorithm, we get Lee-Brickell's approach, and if we further choose $\ell = 0$ and $v = t$ we get Prange's approach, costing asymptotically

$$S(1, T) - S(1 - R_I, T).$$

4.2. *s*-BLOCKS ALGORITHM. In this section, we present an algorithm that takes advantage of the structure of the parity-check matrix over the ring $\mathbb{Z}/p^s\mathbb{Z}$. For a

code of subtype $(k_1, \ldots, k_s)$ and rank K we set $k_{s+1} = n - K$ and bring the parity-check matrix into the systematic form in (1). The idea, in this algorithm, is to split the error vector into $s+1$ parts, where the first s parts belong to the information set and the last part is outside the information set. Then, we go through all the error vectors having weight v in the information set and $t - v$ outside the information set. In order to go through all such error vectors, we fix a weak compositions of v into s parts, which represent the weight distribution of the first s parts of the error vector. Let us denote by $W_s(v)$ the set of all weak compositions of v into s parts.

For simplicity, we assume that the information set is $I = \{1, \ldots, K\}$. Let $(v_1, \ldots, v_s)$ be a weak composition of v into s positive integers, and let $v_{s+1} = t - v$. Therefore, the error vector is of the form $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_{s+1})$, where $\mathbf{e}_i \in (\mathbb{Z}/p^s\mathbb{Z})^{k_i}$ with $\mathrm{wt}_L(\mathbf{e}_i) = v_i$ for each $i \in \{1, \ldots, s+1\}$. We first bring the parity-check matrix $\mathbf{H}$ into the systematic form (1), by multiplying it with an invertible matrix $\mathbf{U}$. Thus, if we also partition the syndrome $\mathbf{s}$ into parts of the same size as the (row-)parts of $\mathbf{UH}$, we obtain the following situation:

$$
\mathbf{UHe}^\top = \begin{pmatrix}
\mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \cdots & \mathbf{A}_{1,s-1} & \mathbf{A}_{1,s} & \mathrm{Id}_{k_{s+1}} \\
p\mathbf{A}_{2,1} & p\mathbf{A}_{2,2} & \cdots & p\mathbf{A}_{2,s-1} & p\mathrm{Id}_{k_s} & \mathbf{0}_{k_s \times k_{s+1}} \\
p^2\mathbf{A}_{3,1} & p^2\mathbf{A}_{3,2} & \cdots & p^2\mathrm{Id}_{k_{s-1}} & \mathbf{0}_{k_{s-1} \times k_s} & \mathbf{0}_{k_{s-1} \times k_{s+1}} \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
p^{s-1}\mathbf{A}_{s,1} & p^{s-1}\mathrm{Id}_{k_2} & \cdots & \mathbf{0}_{k_2 \times k_{s-1}} & \mathbf{0}_{k_2 \times k_s} & \mathbf{0}_{k_2 \times k_{s+1}}
\end{pmatrix}
\begin{pmatrix}
\mathbf{e}_1^\top \\
\mathbf{e}_2^\top \\
\mathbf{e}_3^\top \\
\vdots \\
\mathbf{e}_{s+1}^\top
\end{pmatrix}
$$

$$
= \begin{pmatrix}
\mathbf{s}_1^\top \\
p\mathbf{s}_2^\top \\
\vdots \\
p^{s-1}\mathbf{s}_s^\top
\end{pmatrix} = \mathbf{Us}^\top.
$$

From this we obtain s conditions

$$
(5) \qquad \mathbf{A}_{1,1}\mathbf{e}_1^\top + \mathbf{A}_{1,2}\mathbf{e}_2^\top + \cdots + \mathbf{A}_{1,s}\mathbf{e}_s^\top + \mathbf{e}_{s+1}^\top = \mathbf{s}_1^\top
$$

$$
(6) \qquad p\left(\mathbf{A}_{2,1}\mathbf{e}_1^\top + \mathbf{A}_{2,2}\mathbf{e}_2^\top + \cdots + \mathbf{A}_{2,s-1}\mathbf{e}_{s-1}^\top + \mathbf{e}_s^\top\right) = p\mathbf{s}_2^\top
$$

$$
\vdots
$$

$$
(7) \qquad p^{s-2}\left(\mathbf{A}_{s-1,1}\mathbf{e}_1^\top + \mathbf{A}_{s-1,2}\mathbf{e}_2^\top + \mathbf{e}_3^\top\right) = p^{s-2}\mathbf{s}_{s-1}^\top
$$

$$
(8) \qquad p^{s-1}\left(\mathbf{A}_{s,1}\mathbf{e}_1^\top + \mathbf{e}_2^\top\right) = p^{s-1}\mathbf{s}_s^\top.
$$

Next, we go through all error vectors of selected weight distribution and check the above conditions using backward recursion. For this, we use the following notation to expand vectors over $\mathbb{Z}/p\mathbb{Z}$: for a vector $\mathbf{x} \in (\mathbb{Z}/p^s\mathbb{Z})^\ell$, we write $\mathbf{x}^{(i)} = \mathbf{x} \pmod{p^i}$ for all $i \in \{1, \ldots, s\}$, and similarly for a matrix $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{m \times \ell}$, we write $\mathbf{B}^{(i)} = \mathbf{B} \pmod{p^i}$ for all $i \in \{1, \ldots, s\}$. Moreover, we define $\mathbf{x}^{(i)} + p^i (\mathbb{Z}/p^s\mathbb{Z})^\ell = \{\mathbf{x}^{(i)} + p^i\mathbf{y} \mid \mathbf{y} \in (\mathbb{Z}/p^s\mathbb{Z})^\ell\}$ for any $i \in \{1, \ldots, s\}$.

In the next step, we iterate over all the vectors $\mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{k_1}$ having Lee weight $\mathrm{wt}_L(\mathbf{e}_1) = v_1$. Now, for a given $\mathbf{e}_1$, Condition (8) defines

$$
\mathbf{e}_2^{(1)^\top} := \mathbf{s}_s^{(1)^\top} - \mathbf{A}_{s,1}^{(1)}\mathbf{e}_1^{(1)^\top}.
$$

Thence, we iterate over all the vectors $\mathbf{e}_2 \in \mathbf{e}_2^{(1)} + p(\mathbb{Z}/p^s\mathbb{Z})^{k_2}$. If $\mathbf{e}_2$ has Lee weight $\mathrm{wt}_L(\mathbf{e}_2) = v_2$, we proceed; otherwise, we choose another $\mathbf{e}_2$. Now, as a consequence

of Condition (7), we obtain

$$\mathbf{e}_3^{(2)\top} := \mathbf{s}_{s-1}^{(2)\ \top} - \mathbf{A}_{s-1,1}^{(2)}\mathbf{e}_1^{(2)\top} - \mathbf{A}_{s-1,2}^{(2)}\mathbf{e}_2^{(2)\top}.$$

Then, we iterate over all the vectors $\mathbf{e}_3 \in \mathbf{e}_3^{(2)} + p^2 (\mathbb{Z}/p^s\mathbb{Z})^{k_3}$ and check whether any has the correct Lee weight v_3.

We proceed in this fashion until we obtain the $\mathbf{e}_{s+1}$ from Condition (5) in the following way

$$\mathbf{e}_{s+1}^\top := \mathbf{s}_1^\top - \sum_{i=1}^{s} \mathbf{A}_{1,i}\mathbf{e}_i^\top.$$

And finally we check whether $\mathbf{e}_{s+1}$ has the remaining Lee weight v_{s+1}.

The choice of $(v_1, \ldots, v_s) \in W_s(v)$ are inputs of the algorithm that can be optimized, i.e., chosen in such a way that the cost is the lowest.

The details are provided in Algorithm 2.

Algorithm 2 s-Blocks Algorithm over $\mathbb{Z}/p^s\mathbb{Z}$ in the Lee metric

---

Input: $\mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times n}$, $\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$, $t \in \mathbb{N}$, $v < \min\{K\lfloor\frac{p^s-1}{2}\rfloor, t\}$ and $(v_1, \ldots, v_s) \in W_s(v)$

Output: $\mathbf{e} \in (\mathbb{Z}/p^s\mathbb{Z})^n$ with $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$ and $\mathrm{wt}_L(\mathbf{e}) = t$.

1: Choose an information set $I \subset \{1, ..., n\}$ of size K.
2: Compute $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times(n-k_1)}$ and an $n \times n$ permutation matrix $\mathbf{P}$, such that $\mathbf{UHP}$ is in the systematic form as in (1).
3: Compute $\mathbf{sU}^\top = \begin{pmatrix} \mathbf{s}_1 & p\mathbf{s}_2 & \cdots & p^{s-1}\mathbf{s}_s \end{pmatrix}$, where $\mathbf{s}_i \in (\mathbb{Z}/p^{s-i+1}\mathbb{Z})^{k_{s-i+2}}$.
4: **for** $\mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{k_1}$ with $\mathrm{wt}_L(\mathbf{e}_1) = v_1$ **do**
5: $\quad \mathbf{e}_2^{(1)\top} := \mathbf{s}_s^{(1)\ \top} - \mathbf{A}_{s,1}^{(1)}\mathbf{e}_1^{(1)\top}.$
6: $\quad$ **for** $\mathbf{e}_2 \in \mathbf{e}_2^{(1)} + p(\mathbb{Z}/p^s\mathbb{Z})^{k_2}$ **do**
7: $\quad\quad$ **if** $\mathrm{wt}_L(\mathbf{e}_2) = v_2$ **then**
8: $\quad\quad\quad \mathbf{e}_3^{(2)\top} := \mathbf{s}_{s-1}^{(2)\ \top} - \mathbf{A}_{s-1,1}^{(2)}\mathbf{e}_1^{(2)\top} - \mathbf{A}_{s-1,2}^{(2)}\mathbf{e}_2^{(2)\top}$
9: $\quad\quad\quad$ **for** $\mathbf{e}_3 \in \mathbf{e}_3^{(2)} + p^2 (\mathbb{Z}/p^s\mathbb{Z})^{k_3}$ **do**
10: $\quad\quad\quad\quad$ **if** $\mathrm{wt}_L(\mathbf{e}_3) = v_3$ **then**
11: $\quad\quad\quad\quad\quad \ddots$
12: $\quad\quad\quad\quad\quad \mathbf{e}_s^{(s-1)\top} = \mathbf{s}_2^{(s-1)\ \top} - \sum_{i=1}^{s-1} \mathbf{A}_{2,i}^{(s-1)}\mathbf{e}_i^{(s-1)\top}$
13: $\quad\quad\quad\quad\quad$ **for** $\mathbf{e}_s \in \mathbf{e}_s^{(s-1)} + p^{s-1}(\mathbb{Z}/p^s\mathbb{Z})^{k_s}$ **do**
14: $\quad\quad\quad\quad\quad\quad$ **if** $\mathrm{wt}_L(\mathbf{e}_s) = v_s$ **then**
15: $\quad\quad\quad\quad\quad\quad\quad \mathbf{e}_{s+1}^\top := \mathbf{s}_1^\top - \sum_{i=1}^{s} \mathbf{A}_{1,i}\mathbf{e}_i^\top.$
16: $\quad\quad\quad\quad\quad\quad\quad$ **if** $\mathrm{wt}_L(\mathbf{e}_{s+1}) = t - v$ **then**
17: $\quad\quad\quad\quad\quad\quad\quad\quad$ Return $\mathbf{e} = (\mathbf{e}_1, \ldots, \mathbf{e}_{s+1})\mathbf{P}$
18: Start over with Step 1 and a new selection of I.

For the complexity analysis, we fix the real numbers $V_1, \ldots, V_s$ with $V_i \leq MR_i, 0 \leq \sum_{i=1}^{s} V_i \leq T$ and $T - \sum_{i=1}^{s} V_i \leq (1 - R_I)M$. Then we fix the internal algorithm parameters v_i such that $\lim_{n\to\infty} \frac{v_i}{n} = V_i$, and the code parameters k_i such that $\lim_{n\to\infty} \frac{k_i}{n} = R_i$.

Theorem 4.2. *The asymptotic average time complexity of Algorithm 2 is*

$$S(1,T) - \sum_{i=2}^{s} S(R_i, V_i) - S\left(1 - R_I, T - \sum_{i=1}^{s} V_i\right)$$

$$+ \max\left\{\sum_{i=2}^{j-1} \frac{-R_i(i-1)}{s} + S(R_i, V_i) \mid 2 \le j \le s+1\right\}.$$

Proof. In one iteration we go through all $\mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{k_1}$ of Lee weight v_1, which are $F_L(k_1, v_1, p^s)$ many. To compute the cost of the subsequent steps, we divide them in groups of three steps, i.e., at the j-th stage we first compute $\mathbf{e}_j^{(j-1)}$ according to the syndrome conditions, and then, we go through all $\mathbf{e}_j \in \mathbf{e}_j^{(j-1)} + p^{j-1}(\mathbb{Z}/p^s\mathbb{Z})^{k_j}$ such that they have Lee weight v_j. The cost of computing $\mathbf{e}_j^{(j-1)}$ is given by

$$f(\mathbf{e}_j) := \sum_{\ell=1}^{j-1} k_j k_\ell,$$

and the average number of $\mathbf{e}_j$'s we go through in the set $\mathbf{e}_j^{(j-1)} + p^{j-1}(\mathbb{Z}/p^s\mathbb{Z})^{k_j}$ having Lee weight v_j is given by

$$g(\mathbf{e}_j) := \frac{p^{k_j(s-j+1)}}{p^{k_j s}} F_L(k_j, v_j, p^s).$$

Thus, the cost of one iteration is given by

$$C(v_1, \ldots, v_s) := F_L(k_1, v_1, p^s)\left(\sum_{j=2}^{s+1} f(\mathbf{e}_j) \prod_{\ell=2}^{j-1} g(\mathbf{e}_\ell)\right).$$

This corresponds to the following asymptotic cost

$$\lim_{n\to\infty} \frac{1}{n}\log_q\left(C(v_1, \ldots, v_s)\right)$$

$$= \lim_{n\to\infty} \frac{1}{n}\log_q\left(F_L(k_1, v_1, p^s)\sum_{j=2}^{s+1}\prod_{\ell=2}^{j-1} p^{-k_\ell(\ell-1)} F_L(k_\ell, v_\ell, p^s)\right)$$

$$= S(R_1, V_1) + \lim_{n\to\infty} \frac{1}{n}\log_q\left(\sum_{j=2}^{s+1} p^{\sum_{\ell=2}^{j-1} -k_\ell(\ell-1)} \prod_{\ell=2}^{j-1} F_L(k_\ell, v_\ell, p^s)\right)$$

$$= S(R_1, V_1) + \max\left\{\sum_{\ell=2}^{j-1} \frac{-R_\ell(\ell-1)}{s} + S(R_\ell, V_\ell) \mid 2 \le j \le s+1\right\}.$$

The success probability of one iteration is given by

$$\prod_{i=1}^{s+1} F_L(k_i, v_i, p^s) F_L(n, t, p^s)^{-1},$$

where, again, we set $k_{s+1} = n - K$ and $v_{s+1} = t - v$. This corresponds to the asymptotic number of iterations

$$S(1,T) - \sum_{i=1}^{s} S(R_i, V_i) - S\left(1 - R_I, T - \sum_{i=1}^{s} V_i\right).$$

$\square$

Remark 4. Another variant of this s-blocks algorithm can be obtained by looping over all possible $(v_1, \ldots, v_s) \in W_s(v)$.

4.3. Partial Gaussian Elimination Algorithms.

In this section we adapt the Partial Gaussian Elimination (PGE) algorithms from the Hamming metric to the Lee metric, namely [18, 29, 7]. They all follow the same strategy: by only applying partial Gaussian elimination, depending on a parameter ℓ, we are able to solve the original syndrome decoding instance by solving a smaller one.

More in detail, we find a matrix $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times(n-k_1)}$, such that

$$\mathbf{UH} = \begin{pmatrix} \mathbf{A} & \mathrm{Id}_{n-K-\ell} \\ \mathbf{B} & \mathbf{0} \end{pmatrix},$$

where $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K-\ell)\times(K+\ell)}$ and $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1)\times(K+\ell)}$. Thus, partitioning the error vector $\mathbf{e}$ accordingly, the syndrome decoding problem becomes

$$\mathbf{UHe}^\top = \begin{pmatrix} \mathbf{A} & \mathrm{Id}_{n-K-\ell} \\ \mathbf{B} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{e}_1^\top \\ \mathbf{e}_2^\top \end{pmatrix} = \begin{pmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \end{pmatrix},$$

where $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-\ell}, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$, and $\mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell}$ has Lee weight v and $\mathbf{e}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-\ell}$ has Lee weight $t-v$. Thus, the initial decoding problem splits into two equations:

$$(9) \qquad\qquad\qquad \mathbf{A}\mathbf{e}_1^\top + \mathbf{e}_2^\top = \mathbf{s}_1^\top,$$

$$(10) \qquad\qquad\qquad \mathbf{B}\mathbf{e}_1^\top = \mathbf{s}_2^\top.$$

Note that Condition (10) is again a syndrome decoding instance but with smaller parameters. It is enough to solve this smaller instance and then to compute $\mathbf{e}_2 = \mathbf{s}_1 - \mathbf{e}_1\mathbf{A}^\top$, i.e., such that Condition (9) is satisfied, and check whether it has the remaining weight $t-v$. The technique to solve the smaller instance will also impact the whole algorithm, as it contributes to the success probability of one iteration.

In order to solve the smaller syndrome decoding instance, several techniques have been introduced in the Hamming metric, such as Wagner's approach and the representation technique.

4.3.1. *Wagner's Approach.*

The main idea of Wagner's approach [42], which has been applied to the syndrome decoding problem in [18], on a levels, is to partition the searched error vector into 2^a subvectors and to store them in a list together with their corresponding partial syndromes. After this, the lists are merged in a special way. We will now provide a description of Wagner's approach in more detail.

The smaller syndrome decoding instance is given by $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1)\times(K+\ell)}$, $\mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$ and we are searching for an error vector $\mathbf{e}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell}$ of Lee weight v, such that $\mathbf{B}\mathbf{e}_1^\top = \mathbf{s}_2^\top$.

Let a be a positive integer. To ease the notation in Wagner's algorithm let us assume that $K + \ell$ and v are divisible by 2^a (if this is not the case, a small tweak using for example $\lfloor \frac{K+\ell}{2^a} \rfloor$ can be applied); then, one defines 2^a index sets of size $\frac{K+\ell}{2^a}$:

$$I_j := \left\{ (j-1)\frac{K+\ell}{2^a} + 1, \ldots, j\frac{K+\ell}{2^a} \right\},$$

for all $j \in \{1, \ldots, 2^a\}$. We then split the input matrix $\mathbf{B}$ into 2^a submatrices $\mathbf{B}_j^{(0)}$, which have columns indexed by I_j, i.e.,

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_1^{(0)} & \cdots & \mathbf{B}_{2^a}^{(0)} \end{pmatrix}$$

and partition the error vector into the corresponding subsets I_j as

$$\mathbf{e}_1 = (\mathbf{e}_1^{(0)}, \ldots, \mathbf{e}_{2^a}^{(0)}).$$

Let us denote by

$$I_b^{(i-1)} = I_{2b-1}^{(i-2)} \cup I_{2b}^{(i-2)}$$

1   for $b \in \{1, \ldots, 2^{a-i+1}\}, i \in \{2, \ldots, a+1\}$ and $I_b^{(0)} = I_b$. Then, we can define
2   $\mathbf{B}_{2j-1}^{(i-1)} = \mathbf{B}_{I_{2j-1}^{(i-1)}}$.

We build the initial lists

$$\mathcal{L}_j^{(0)} := \{\mathbf{e}_j^{(0)} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell)/(2^a)} \mid \mathrm{wt}_L\left(\mathbf{e}_j^{(0)}\right) = v/2^a\},$$

for $j \in \{1, \ldots, 2^a\}$. Let

$$0 = u_0 \leq u_1 \leq \cdots \leq u_{a-1} \leq u_a = K + \ell - k_1,$$

where u_i is a positive integer indicating the number of entries, on which the merging procedure on level i is based. We will say that $\mathbf{x} + \mathbf{y} =_u \mathbf{z}$, if $\mathbf{x} + \mathbf{y}$ are equal to $\mathbf{z}$ on the last u positions. On the i-th level one has 2^{a-i+1} input lists $\mathcal{L}^{(i-1)}$ and wants to construct 2^{a-i} output lists $\mathcal{L}^{(i)}$. These are constructed in the following way

$$\mathcal{L}_j^{(i)} = \mathcal{L}_{2j-1}^{(i-1)} +\!\!+_{\mathbf{0}} \mathcal{L}_{2j}^{(i-1)} :=$$
$$\{(\mathbf{e}_{2j-1}^{(i-1)}, \mathbf{e}_{2j})^{(i-1)} \mid \mathbf{e}_b^{(i-1)} \in \mathcal{L}_b^{(i-1)}, \mathbf{B}_{2j-1}^{(i-1)}(\mathbf{e}_{2j-1}^{(i-1)})^\top =_{u_i} -\mathbf{B}_{2j}^{(i-1)}(\mathbf{e}_{2j}^{(i-1)})^\top\},$$

for $j \in \{1, \ldots, 2^{a-i} - 1\}$, whereas for $j = 2^{a-i}$ we have

$$\mathcal{L}_j^{(i)} = \mathcal{L}_{2j-1}^{(i-1)} +\!\!+_{\mathbf{s}_2} \mathcal{L}_{2j}^{(i-1)} :=$$
$$\{(\mathbf{e}_{2j-1}^{(i-1)}, \mathbf{e}_{2j}^{(i-1)}) \mid \mathbf{e}_b^{(i-1)} \in \mathcal{L}_b^{(i-1)}, \mathbf{B}_{2j-1}^{(i-1)}(\mathbf{e}_{2j-1}^{(i-1)})^\top =_{u_i} \mathbf{s}_2^\top - \mathbf{B}_{2j}^{(i-1)}(\mathbf{e}_{2j}^{(i-1)})^\top\}.$$

Algorithm 3 Merge-concatenate

Input: The input lists $\mathcal{L}_1, \mathcal{L}_2$, the positive integers $0 < u < k$,
$\mathbf{B}_1, \mathbf{B}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{k \times n}$ and $\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^k$.
Output: $\mathcal{L} = \mathcal{L}_1 +\!\!+_{\mathbf{s}} \mathcal{L}_2$.
  1: Lexicographically sort $\mathcal{L}_1$ according to the last u positions of $\mathbf{B}_1\mathbf{e}_1^\top$ for $\mathbf{e}_1 \in \mathcal{L}_1$.
     We also store the last u positions of $\mathbf{B}_1\mathbf{e}_1^\top$ in the sorted list.
  2: **for** $\mathbf{e}_2 \in \mathcal{L}_2$ **do**
  3:      **for** $\mathbf{e}_1 \in \mathcal{L}_1$ with $\mathbf{B}_1\mathbf{e}_1^\top = \mathbf{s}^\top - \mathbf{B}_2\mathbf{e}_2^\top$ on the last u positions **do**
  4:          $\mathcal{L} = \mathcal{L} \cup \{(\mathbf{e}_1, \mathbf{e}_2)\}$.
  5: Return $\mathcal{L}$.

Lemma 4.3. *The asymptotic of the average cost of Algorithm 3 is*

$$\lim_{n \to \infty} \frac{1}{n} \max\left\{\log_{p^s}(\mid \mathcal{L}_1 \mid), \log_{p^s}(\mid \mathcal{L}_2 \mid)\right\}.$$

Proof. As a first step, we sort the list $\mathcal{L}_1$ according to the last u positions of $\mathbf{B}_1\mathbf{e}_1^\top$. Then, for each $\mathbf{e}_2 \in \mathcal{L}_2$, we compute the last u positions of $\mathbf{s}^\top - \mathbf{B}_2\mathbf{e}_2^\top$ and check for a collision in the sorted list $\mathcal{L}_1$. The asymptotic cost to compute $\mathbf{B}_1\mathbf{e}_1^\top$ on the last u positions for each $\mathbf{e}_1 \in \mathcal{L}_1$, and to sort $\mathcal{L}_1$ is the same:

$$\lim_{n\to\infty} \frac{1}{n} \log_{p^s} (|\mathcal{L}_1|).$$

Similarly, to compute $\mathbf{s}^\top - \mathbf{B}_2\mathbf{e}_2^\top$ on the last u positions for each $\mathbf{e}_2 \in \mathcal{L}_2$, asymptotically costs

$$\lim_{n\to\infty} \frac{1}{n} \log_{p^s} (|\mathcal{L}_2|).$$

$\square$

After the merge on level a, one is left with a final list, which contains now error vectors $\mathbf{e}_1^{(a)} = (\mathbf{e}_1^{(0)}, \ldots, \mathbf{e}_{2^a}^{(0)})$ of Lee weight v such that

$$\mathbf{B}_1^{(0)}(\mathbf{e}_1^{(0)})^\top + \cdots + \mathbf{B}_{2^a}^{(0)}(\mathbf{e}_{2^a}^{(0)})^\top = \mathbf{s}_2^\top$$

and can hence solve the smaller syndrome decoding instance.

This algorithm based on Wagner's approach succeeds if the target vector $\mathbf{e}_1$ can be split into subvectors $\mathbf{e}_1^{(0)}, \ldots, \mathbf{e}_{2^a}^{(0)}$ of length $(K + l)/2^a$ and weight $v/2^a$, and at each level i the syndrome equations are satisfied on some fixed u_i positions. In particular, the success probability is then given by the probability that at least one representation of the target vector lives in the lists $\mathcal{L}^{(i)}$. Let us assume that $v \neq 0$.

For the base lists, we have the success probability that we can partition the target vector $\mathbf{e}_1$ into vectors of length $(K + \ell)/2^a$ of Lee weight $v/2^a$ is given by

$$(11) \qquad \frac{F_L(n - K - \ell, t - v, p^s)F_L\left(\frac{K+\ell}{2^a}, \frac{v}{2^a}, p^s\right)^{2^a}}{F_L(n, t, p^s)}.$$

For the lists after one merge, i.e., $\mathcal{L}_i^{(1)}$, we now want that the vectors $\mathbf{e}_i^{(1)} \in \mathcal{L}_i^{(1)}$ are such that $\mathbf{B}_i^{(1)}(\mathbf{e}_i^{(1)})^\top =_{u_1} \mathbf{0}^\top$. Since $\mathbf{B}_i^{(1)}, \mathbf{e}_i^{(1)}$ are assumed to be uniformly distributed, we can use Lemma 2.8, to get that

1. If $p \nmid v/2^{a-1}$, then $\mathbf{e}_i^{(1)} \notin p\left(\mathbb{Z}/p^s\mathbb{Z}\right)^{(K+\ell)/2^{a-1}}$ with probability 1.
2. If $p \mid v/2^{a-1}$, then $\mathbf{e}_i^{(1)} \notin p\left(\mathbb{Z}/p^s\mathbb{Z}\right)^{(K+\ell)/2^{a-1}}$ with probability

$$1 - \frac{F_L((K + \ell)/2^{a-1}, v/(p2^{a-1}), p^{s-1})}{F_L((K + \ell)/2^{a-1}, v/2^{a-1}, p^s)}.$$

Thus, $\mathbf{e}_i^{(1)} \notin p\left(\mathbb{Z}/p^s\mathbb{Z}\right)^{(K+\ell)/2^{a-1}}$ with high probability and again due to Lemma 2.8, we get that $\mathbf{B}_i^{(1)}(\mathbf{e}_i^{(1)})^\top$ is uniformly distributed. Thus, $\mathbf{B}_i^{(1)}(\mathbf{e}_i^{(1)})^\top =_{u_1} \mathbf{0}^\top$ has probability p^{-su_1}. Note that the condition $\mathbf{B}_i^{(1)}(\mathbf{e}_i^{(1)})^\top =_{u_1} \mathbf{0}^\top$ is imposed on all 2^{a-1} vectors (and for the last one a similar condition holds but summing to $\mathbf{s}_2$). However, ensuring only half of them satisfy the condition is sufficient, because with the condition on the next merges we can ensure that the other half satisfies this, as well. For example, since $\mathbf{B}_j^{(2)}(\mathbf{e}_j^{(2)})^\top = \mathbf{B}_{2j-1}^{(1)}(\mathbf{e}_{2j-1}^{(1)})^\top + \mathbf{B}_{2j}^{(1)}(\mathbf{e}_{2j}^{(1)})^\top =_{u_2} \mathbf{0}^\top$ and $\mathbf{B}_{2j-1}^{(1)}(\mathbf{e}_{2j-1}^{(1)})^\top =_{u_1} \mathbf{0}^\top$, it directly follows that $\mathbf{B}_{2j}^{(1)}(\mathbf{e}_{2j}^{(1)})^\top =_{u_1} \mathbf{0}^\top$.

For the next merge, we impose that $\mathbf{B}_i^{(2)}(\mathbf{e}_i^{(2)})^\top =_{u_2} \mathbf{0}^\top$, knowing that

$$\mathbf{B}_i^{(2)}(\mathbf{e}_i^{(2)})^\top =_{u_1} \mathbf{0}^\top$$

from the condition on the lists $\mathcal{L}_i^{(1)}$, thus this results in a probability that the above condition is verified of $p^{-s(u_2-u_1)}$ and is imposed on 2^{a-3} vectors.

We can continue in the same fashion for the subsequent merges, resulting for the algorithm on a levels in a total probability of

$$\prod_{i=1}^{a-1} \left(p^{-s(u_i-u_{i-1})} \right)^{2^{a-i-1}}.$$

We expect by Lemma 2.8 all lists $\mathcal{L}_b^{(i)}$ to be of size

$$L_i = \frac{L_{i-1}^2}{(p^s)^{u_i-u_{i-1}}}.$$

Hence, for $i \in \{1,\ldots,a\}$ we have that

$$L_i = \frac{L_0^{2^i}}{(p^s)^{\beta(i)}},$$

where $\beta_i = u_i + \sum_{j=1}^{i-1} 2^{i-j-1} u_j$.

Observe that the final merge on $K + \ell - k_1$ positions, that is

$$\mathbf{B}_1^{(a-1)}(\mathbf{e}_1^{(a-1)})^\top = \mathbf{s}_2^\top - \mathbf{B}_2^{(a-1)}(\mathbf{e}_2^{(a-1)})^\top$$

is not considered in the computation of the probability, as here we are not losing any representation of the target vector, since we are not assuming that some parts sum to 0.

In the following, we compute the asymptotic average cost of Wagner's algorithm for level one and level two. This is enough, as it turns out that level one gives the optimal complexity, and increasing the levels does not improve the cost, as one can see in the comparison in Section 5.

Theorem 4.4.   *1. For Wagner's approach on one level, let us fix the real numbers V, L with $0 \leq V \leq T$ and $0 \leq L \leq 1 - R_I$ such that $V \leq M(R_I + L)$ and $T - V \leq M(1 - R_I - L)$. We fix the internal algorithm parameters v, ℓ, such that $\lim_{n\to\infty} \frac{v}{n} = V, \lim_{n\to\infty} \frac{\ell}{n} = L$. The asymptotic average time complexity of Wagner's approach on one level is then given by*

$$S(1,T) - S(1 - R_I - L, T - V) - 2S((R_I + L)/2, V/2) +$$
$$\max\{S((R_I + L)/2, V/2), 2S((R_I + L)/2, V/2) - (R_I + L - R_1)\}.$$

2. For Wagner's approach on two levels, let us fix additionally the real number U, such that $0 \leq U \leq R_I + L - R_1$ and we fix the internal algorithm parameters u_1, such that $\lim_{n\to\infty} \frac{u_1}{n} = U$. The asymptotic average time complexity of Wagner's approach on two levels is then given by

$$S(1,T) - S(1 - R_I - L, T - V) - 4S((R_I + L)/4, V/4) + U +$$
$$\max\{S((R_I + L)/4, V/4), 2S((R_I + L)/4, V/4) - U,$$
$$4S((R_I + L)/4, V/4) - (R_I + L - R_1) - U\}.$$

Proof.   1. The initial lists $\mathcal{L}_1^{(0)}, \mathcal{L}_2^{(0)}$ are of size $L_0 = F_L\left(\frac{K+\ell}{2}, \frac{v}{2}, p^s\right)$. We now have to compute $\mathcal{L}_1^{(1)} = \mathcal{L}_1^{(0)} +_{\mathbf{s}_2} \mathcal{L}_2^{(0)}$. Which due to Algorithm 3 costs

$$S((R_I + L)/2, V/2).$$

Algorithm 4 Wagner on a levels

Input: $0 = u_0 \leq u_1 \leq \cdots \leq u_a = K + \ell - k_1, 0 \leq v \leq t, \mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times n}$ and
$\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$.
Output: $\mathbf{e} \in (\mathbb{Z}/p^s\mathbb{Z})^n$ with $\mathrm{wt}_L(\mathbf{e}) = t$ and $\mathbf{He}^\top = \mathbf{s}^\top$.

1: Choose an $n \times n$ permutation matrix $\mathbf{P}$.
2: Find $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times(n-k_1)}$, such that

$$\mathbf{UHP} = \begin{pmatrix} \mathbf{A} & \mathrm{Id}_{n-K-\ell} \\ \mathbf{B} & \mathbf{0} \end{pmatrix},$$

  where $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K-\ell)\times(K+\ell)}$ and $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1)\times(K+\ell)}$.
3: Compute $\mathbf{Us}^\top = \begin{pmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \end{pmatrix}$, where $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-\ell}, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$.
4: Partition $I = \{1, \ldots, K + \ell\}$ into 2^a subsets of size $\frac{K+\ell}{2^a}$:

$$I_j = \left\{ (j-1)\frac{K+\ell}{2^a} + 1, \ldots, j\frac{K+\ell}{2^a} \right\},$$

  for $j \in \{1, \ldots, 2^a\}$.
5: Partition $\mathbf{B}$ into 2^a submatrices $\mathbf{B}_j$, containing the columns of $\mathbf{B}$ indexed by I_j, for all $j \in \{1, \ldots, 2^a\}$.
6: Set

$$\mathcal{L}_j^{(0)} = \{\mathbf{e}_j^{(0)} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell)/(2^a)}, \mathrm{wt}_L\left(\mathbf{e}_j^{(0)}\right) = v/(2^a)\}$$

  for all $j \in \{1, \ldots, 2^a\}$.
7: **for** $i \in \{1, \ldots a\}$ **do**
8: **for** $j \in \{1, \ldots, 2^{a-i}\}$ **do**
9: Compute $\mathcal{L}_j^{(i)} = \begin{cases} \mathcal{L}_{2j-1}^{(i-1)} +_{\mathbf{0}} \mathcal{L}_{2j}^{(i-1)} & \text{for } 1 \leq j < 2^{a-i}, \\ \mathcal{L}_{2j-1}^{(i-1)} +_{\mathbf{s}_2} \mathcal{L}_{2j}^{(i-1)} & \text{for } j = 2^{a-i}. \end{cases}$
10: **for** $\mathbf{e}_1 \in \mathcal{L}_1^{(a)}$ **do**
11: **if** $\mathrm{wt}_L\left(\mathbf{s}_1 - \mathbf{e}_1\mathbf{A}^\top\right) = t - v$ **then**
12: Return $\mathbf{e} = \mathbf{P}(\mathbf{e}_1, \mathbf{s}_1 - \mathbf{e}_1\mathbf{A}^\top)$
13: Else start over at step 1.

Finally, we check for all $\mathbf{e}_1 \in \mathcal{L}_1^{(1)}$ if $\mathrm{wt}_L\left(\mathbf{s}_1 - \mathbf{e}_1\mathbf{A}^\top\right) = t - v$, which asymptotically costs

$$2S((R_I + L)/2, V/2) - (R_I + L - R_1),$$

which is the asymptotic size of $\mathcal{L}_1^{(1)}$. The success probability is given by

$$\frac{F_L(n - K - \ell, t - v, p^s)F_L((K + L)/2, v/2, p^s)^2}{F_L(n, t, p^s)},$$

and hence the asymptotic number of iterations is

$$S(1, T) - S(1 - R_I - L, T - V) - 2S((R_I + L)/2, V/2).$$

2. In the level 2 case, we have 4 base lists, $\mathcal{L}_b^{(0)}$ of size L_0, for $b \in \{1, \ldots, 4\}$. We first merge $\mathcal{L}_1^{(1)} = \mathcal{L}_1^{(0)} +_{\mathbf{0}} \mathcal{L}_2^{(0)}$, and $\mathcal{L}_2^{(1)} = \mathcal{L}_3^{(0)} +_{\mathbf{s}_2} \mathcal{L}_4^{(0)}$, which both cost $S((R_I+L)/4, V/4)$. The final merge $\mathcal{L}_1^{(2)} = \mathcal{L}_1^{(1)} +_{\mathbf{s}_2} \mathcal{L}_2^{(2)}$ costs $2S((R_I+L)/4, V/4$, which is the asymptotic size of $\mathcal{L}_b^{(1)}$ for $b \in \{1, 2\}$. To check for all $\mathbf{e}_1 \in \mathcal{L}_1^{(2)}$

if $\mathrm{wt}_L\left(\mathbf{s}_1 - \mathbf{e}_1 \mathbf{A}^\top\right) = t - v$, asymptotically costs

$$4S((R_I + L)/4, V/4) - (R_I + L - R_1) - U,$$

which is the asymptotic size of $\mathcal{L}_1^{(2)}$.

The success probability is composed of the probability that $\mathbf{e}$ has the assumed weight distribution and that a merging on u_1 positions results in the sought-after $\mathbf{e}$, which is given by

$$\frac{F_L(n - K - \ell, t - v, p^s) F_L((K + L)/4, v/4, p^s)^4}{F_L(n, t, p^s) p^{u_1}},$$

and hence the asymptotic number of iterations is

$$S(1, T) - S(1 - R_I - L, T - V) - 4S((R_I + L)/4, V/4) + U.$$

$\square$

4.3.2. *Representation Technique.* In contrast to the above description of Wagner's approach, in [7] the authors allow the subvectors of the error vector to overlap; this is called the subset sum representation technique. Note that the approach in [29] is similar to [7], but in the former the vectors have no overlap. In this section we will adapt the representation technique approach of [7] to solve the smaller syndrome decoding instance to the Lee metric.

In this algorithm, we create the list $\mathcal{L}$ of vectors having Lee weight v by merging two lists $\mathcal{L}_1, \mathcal{L}_2$ containing vectors of Lee weight $v/2 + \varepsilon$, where ε represents the overlapping part that cancels out after adding the vectors. Thus, if we want to reach a $\mathbf{x} \in (\mathbb{Z}/p^s\mathbb{Z})^n$ of Lee weight v, we can write it as $\mathbf{x} = \mathbf{y} + \mathbf{z}$, for $\mathbf{y}$ and $\mathbf{z}$ having Lee weight $v/2 + \varepsilon$. For $M = \left\lfloor \frac{p^s}{2} \right\rfloor$, this ε can be between 0 and $Mn - v$, as we can reach weight v be using two times $v/2$ and the room we are left with for overlappings has weight $Mn - v$. Algorithm 5 describes the process of merging with overlapping. We apply this merging process at each level, similar to Wagner's approach.

The weight in the overlapping parts is described by the choice of the positive integers $\varepsilon_0, \ldots, \varepsilon_{a-1}$. Let $v_0 = \frac{v}{2^a} + \sum_{b=0}^{a-1} \frac{\varepsilon_b}{2^b}$. We start by computing the base lists

$$\mathcal{L}_j^{(0)} = \left\{ \mathbf{e}_j^{(0)} \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell} \mid \mathrm{wt}_L\left(\mathbf{e}_j^{(0)}\right) = v_0 \right\},$$

for $j \in \{1, \ldots, 2^a\}$. The algorithm is divided again into a levels, and at level $i \in \{1, \ldots, a\}$ the vectors have Lee weight ε_{i-1} in the overlapping part, which cancels out after adding two vectors, and the rest of the merging works as in Wagner's approach. In other words, we merge the lists $\mathcal{L}_{2j-1}^{(i-1)}$ and $\mathcal{L}_{2j}^{(i-1)}$ to obtain a list $\mathcal{L}_j^{(i)}$ such that the overlapping part cancels out, where the lists are given by

$$\mathcal{L}_j^{(i)} = \left\{ \mathbf{e}_j^{(i)} \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell} \mid \mathrm{wt}_L\left(\mathbf{e}_j^{(i)}\right) = v_i, \mathbf{B}\mathbf{e}_j^{(i)\top} =_{u_i} \mathbf{s}_2^\top \right\}.$$

Here $v_i = \frac{v}{2^{a-i}} + \sum_{b=i}^{a-1} \frac{\varepsilon_b}{2^{b-i}}$, for each $j \in \{1, \ldots, 2^{a-i}\}$, and

$$1 \le u_1 \le \cdots \le u_a = K + \ell - k_1$$

is an non-decreasing sequence of integers.

The resulting list $\mathcal{L}_j^{(i)}$ is expected to have size

$$L_i = F_L(K + \ell, v_i, p^s)/(p^s)^{u_i}.$$

In order to ensure that at least one representation $(\mathbf{e}_1^{(i)}, \mathbf{e}_2^{(i)})$ of a vector $\mathbf{e}_1^{(i+1)} = \mathbf{e}_1^{(i)} + \mathbf{e}_2^{(i)}$ is in $\mathcal{L}_1^{(i)} \times \mathcal{L}_2^{(i)}$, we have to choose the u_i's in a more restrictive way. For example any value smaller than the logarithm of the number of representations will suffice.

For a vector $\mathbf{x}$ of weight v of length $K + \ell$ we observe that the number of representations $\mathbf{y} + \mathbf{z}$ of weight $v/2 + \varepsilon$ depends on the choice of $\mathbf{x}$ and thus does not have a generic formula. Thus, we can either work with a lower bound or with an expected number of representations. Using a lower bound ensures that for any error vector the algorithm works; however, on average, the expected number of representations should be enough to ensure this as well.

Let us denote by $C(w, \sigma, \eta)$ the number of compositions of w into σ parts such that each part is at most η, which can be computed as

$$C(w, \sigma, \eta) = \sum_{j=0}^{\min\{\sigma, \lfloor \frac{w-\sigma}{\eta} \rfloor\}} (-1)^j \binom{\sigma}{j} \binom{w - j\eta - 1}{\sigma - 1},$$

as shown in [1].

Lemma 4.5. *Let $\mathbf{x} \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell}$ with $\mathrm{wt}_L(\mathbf{x}) = v$. The number of representations $\mathbf{x} = \mathbf{y} + \mathbf{z}$ with $\mathrm{wt}_L(\mathbf{y}) = \mathrm{wt}_L(\mathbf{z}) = v/2 + \epsilon$ is greater than or equal to*

$$2 \sum_{\sigma=1}^{\min\{\varepsilon, K+\ell\}} \binom{K+\ell}{\sigma} C(\varepsilon, \sigma, \lfloor p^s/4 \rfloor).$$

Proof. Let us denote by $M = \lfloor \frac{p^s}{2} \rfloor$. To get a lower bound, one construction of a representation is enough. One can write each representation $\mathbf{y} + \mathbf{z}$ as

$$(\mathbf{y}_1 + \mathbf{y}_2) + (\mathbf{z}_1 + \mathbf{z}_2),$$

where $\mathrm{wt}_L(\mathbf{y}_1) = \mathrm{wt}_L(\mathbf{z}_1) = v/2$ and $\mathrm{wt}_L(\mathbf{y}_2) = \mathrm{wt}_L(\mathbf{z}_2) = \varepsilon$. We will focus on a particular representation, where we half the weight of each entry of $\mathbf{x}$. Thus, it is enough to consider one entry of $\mathbf{x}$, e.g. the first $\mathbf{x}_1$. Clearly, the choice of $\mathbf{y}_1$ then completely defines the choice of $\mathbf{z}_1$ and also the choice of $\mathbf{y}_2$ determines the choice of $\mathbf{z}_2 = -\mathbf{y}_2$. For each non-zero entry (and thus also for the first entry) of $\mathbf{y}_1$ there exist always two choices, namely for the first entry these are $\lfloor \mathbf{x}_1/2 \rfloor$ and $\lceil (\mathbf{x}_1 - p^s)/2 \rceil$. The choice of $\mathbf{y}_2$ is then restricted, since we want to add weight by computing $\mathbf{y}_1 + \mathbf{y}_2$, but for certain choices of $\mathbf{y}_2$ this reduces the weight. In fact, for any choice of first entry of $\mathbf{y}_1$ (assuming it is non-zero), we have at least $\lfloor M/2 \rfloor$ choices for the first entry of $\mathbf{y}_2$. To see this, imagine the first entry of $\mathbf{y}_1$ called α to be in $[-M, M]$. We denote by β the first entry of $\mathbf{y}_2$. If $\alpha > 0$, then in order for $\beta + \alpha \in [0, M]$ we must choose $\beta \in [0, \lfloor M/2 \rfloor]$, and if $\alpha < 0$, then in order for $\beta + \alpha \in [-M, 0]$ we must choose $\beta \in [-\lfloor M/2 \rfloor, 0]$. Thus, with this particular construction of $\mathbf{y}$ and $\mathbf{z}$ we are allowed to compose the weight ε into σ parts, denoting the support size, of size up to $\lfloor M/2 \rfloor$. $\square$

Thus, using this lower bound we choose

$$u_i = \left\lfloor \log_{p^s} \left(2 \sum_{\sigma=1}^{\min\{\varepsilon, K+\ell\}} \binom{K+\ell}{\sigma} C(\varepsilon, \sigma, \lfloor p^s/4 \rfloor) \right) \right\rfloor.$$

As mentioned earlier, another approach for the choice of u_i can be based on the expected number of representations. However, due to the difficulty of calculating the expected number of representations, we leave this approach for future works.
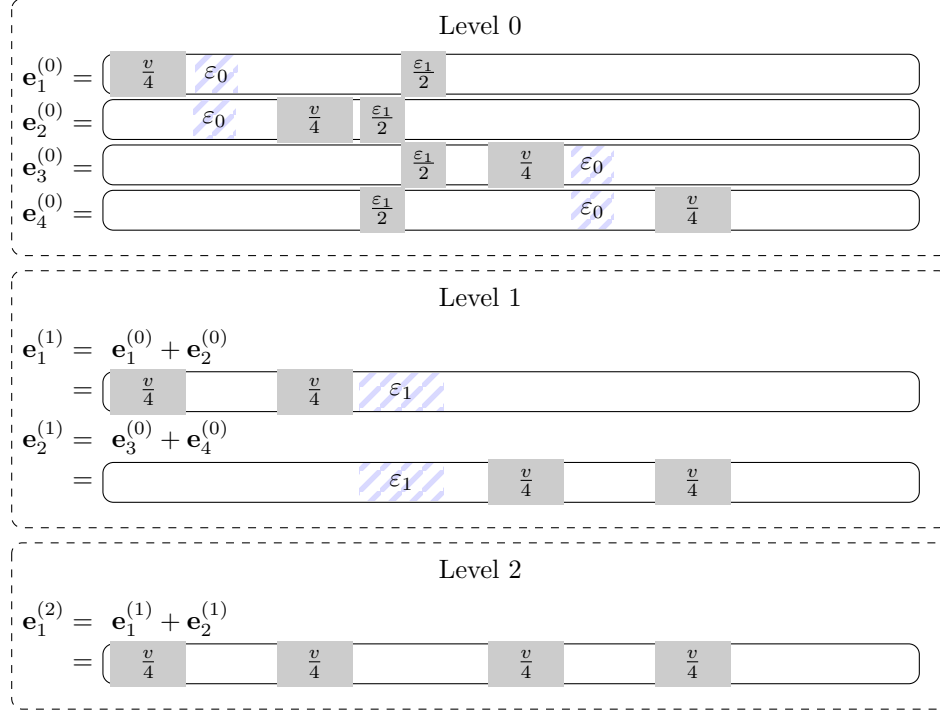

FIGURE 1. Illustration of the error vectors at each level of the representation technique algorithm for two levels. At each level, the striped region represents the overlapping part.

Algorithm 5 Merge

Input: The input lists $\mathcal{L}_1, \mathcal{L}_2$, the positive integers $0 < u < k$ and $0 \leq v \leq \left\lfloor \frac{p^s}{2} \right\rfloor n$, the matrix $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{k \times n}$ and the syndrome $\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^k$.

Output: $\mathcal{L} = \mathcal{L}_1 \bowtie \mathcal{L}_2$.

1: Lexicographically sort $\mathcal{L}_1$ according to the last u positions of $\mathbf{Be}_1^\top$ for $\mathbf{e}_1 \in \mathcal{L}_1$. We also store the last u positions of $\mathbf{Be}_1^\top$ in the sorted list.

2: **for** $\mathbf{e}_2 \in \mathcal{L}_2$ **do**

3:     **for** $\mathbf{e}_1 \in \mathcal{L}_1$ with $\mathbf{Be}_1^\top = \mathbf{s}^\top - \mathbf{Be}_2^\top$ on the last u positions **do**

4:         **if** $\mathrm{wt}_L(\mathbf{e}_1 + \mathbf{e}_2) = v$ **then**

5:             $\mathcal{L} = \mathcal{L} \cup \{\mathbf{e}_1 + \mathbf{e}_2\}$.

6: Return $\mathcal{L}$.

Lemma 4.6. *The asymptotic average time complexity of the merge algorithm (Algorithm 5) is given by*

$$\lim_{n \to \infty} \frac{1}{n} \left(\log_{p^s}(L_1) + \log_{p^s}(L_2) \right) - U,$$

where $U = \lim_{n\to\infty} \frac{u}{n}$ and $L_i = |\mathcal{L}_i|$ for $i = 1, 2$.

Proof. We sort the list $\mathcal{L}_1$ while computing the vectors $\mathbf{Be}_1^\top$ for each $\mathbf{e}_1 \in \mathcal{L}_1$. Once we know the u positions of this vector, we append $\mathbf{e}_1^\top$ to the sorted list containing previously computed vectors. Let L_1 be the size of $\mathcal{L}_1$. Then, the average cost of sorting $\mathcal{L}_1$ according to the last u positions of $\mathbf{Be}_1^\top$ is asymptotically

$$\lim_{n\to\infty} \frac{1}{n} \left(\log_{p^s}(L_1) \right).$$

Then for each $\mathbf{e}_2 \in \mathcal{L}_2$, we compute the last u positions of $\mathbf{s}^\top - \mathbf{Be}_2^\top$ and find $\mathbf{e}_1$ in the sorted list $\mathcal{L}_1$ satisfying $\mathbf{Be}_1^\top =_u \mathbf{s}^\top - \mathbf{Be}_2^\top$. This step will cost on average

$$\lim_{n\to\infty} \frac{1}{n} \left(\log_{p^s}(L_2) \right).$$

Next, for each collision we compute the vector $\mathbf{e}_1 + \mathbf{e}_2$, which is asymptotically the most dominant step. Using the estimation for the expected number of collisions, we get that the overall average cost of the algorithm is asymptotically given by

$$\lim_{n\to\infty} \frac{1}{n} \left(\log_{p^s}(L_1) + \log_{p^s}(L_2) \right) - U,$$

where $U = \lim_{n\to\infty} \frac{u}{n}$. $\square$

Algorithm 5 is applied for all levels $i < a$ whereas, on the last level a, we do a similar merging algorithm, with a small twist: instead of Step 5 of Algorithm 5, we check whether

$$\mathrm{wt}_L(\mathbf{s}_1 - (\mathbf{e}_1 + \mathbf{e}_2)\mathbf{A}^\top) = t - v,$$

which is necessary to guarantee that the considered solution $(\mathbf{e}_1 + \mathbf{e}_2)$ of the smaller SDP instance can be extended to a solution of the whole SDP instance, being $(\mathbf{e}_1 + \mathbf{e}_2, \mathbf{s}_1 - (\mathbf{e}_1 + \mathbf{e}_2)\mathbf{A}^\top)$. Taking this condition into account inside the merge algorithm prevents us from storing the final list, thus decreasing the cost of the algorithm. Thus, in the last level, as soon as this condition is satisfied, one can abort and return the found error vector; we denote this merging process as

$$\mathcal{L}_1 \bowtie_{\mathbf{s}_2, \mathbf{s}_1, \mathbf{A}, w}^{(a)} \mathcal{L}_2.$$

Algorithm 6 Last Merge

Input: The input lists $\mathcal{L}_1, \mathcal{L}_2$, the positive integers $w, v, 0 < u < k$,
$\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{k \times n}, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^k$ and $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^r, \mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{r \times 2n}$.
Output: $\mathbf{e} \in \mathcal{L}_1 \bowtie_{\mathbf{s}_2, \mathbf{s}_1, \mathbf{A}, w}^{(a)} \mathcal{L}_2$.

1: Lexicographically sort $\mathcal{L}_1$ according to the last u positions of $\mathbf{Be}_1^\top$ for $\mathbf{e}_1 \in \mathcal{L}_1$.
   We also store the last u positions of $\mathbf{Be}_1^\top$ in the sorted list.
2: **for** $\mathbf{e}_2 \in \mathcal{L}_2$ **do**
3:     **for** $\mathbf{e}_1 \in \mathcal{L}_1$ with $\mathbf{Be}_1^\top = \mathbf{s}_2^\top - \mathbf{Be}_2^\top$ on the last u positions **do**
4:         **if** $\mathrm{wt}_L(\mathbf{e}_1 + \mathbf{e}_2) = v$ and $\mathrm{wt}_L(\mathbf{s}_1 - (\mathbf{e}_1 + \mathbf{e}_2)\mathbf{A}^\top) = w$ **then**
5:             Return $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{s}_1 - (\mathbf{e}_1 + \mathbf{e}_2)\mathbf{A}^\top)$.

Note that the cost of this last merge is a bit smaller on the last step, but asymptotically the difference is negligible.

Corollary 2. *The asymptotic average time complexity of the last merge (Algorithm 6) is given by*

$$\lim_{n\to\infty} \frac{1}{n} \left(\log_{p^s}(L_1) + \log_{p^s}(L_2) \right) - U,$$

where $U = \lim_{n\to\infty} \frac{u}{n}$ and $L_i = |\mathcal{L}_i|$ for $i = 1, 2$.

Algorithm 7 Representation Technique on a levels

Input: $0 \le \ell \le n - K$, $0 \le \varepsilon_i \le M(K + \ell) - v_{i+1}$, for all $i \in \{0, \ldots, a-1\}$ where $v_i = v/(2^{a-i}) + \sum_{b=i}^{a-1} \frac{\varepsilon_b}{2^{b-i}}$,
$0 = u_0 \le u_1 \le \cdots \le u_a = K + \ell - k_1, t, v, \mathbf{H} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times n}$ and
$\mathbf{s} \in (\mathbb{Z}/p^s\mathbb{Z})^{n-k_1}$.
Output: $\mathbf{e} \in (\mathbb{Z}/p^s\mathbb{Z})^n$ with $\mathrm{wt}_L(\mathbf{e}) = t$ and $\mathbf{He}^\top = \mathbf{s}^\top$.

1: Choose an $n \times n$ permutation matrix $\mathbf{P}$.
2: Find $\mathbf{U} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-k_1)\times(n-k_1)}$, such that

$$\mathbf{UHP} = \begin{pmatrix} \mathbf{A} & \mathrm{Id}_{n-K-\ell} \\ \mathbf{B} & \mathbf{0} \end{pmatrix},$$

 where $\mathbf{A} \in (\mathbb{Z}/p^s\mathbb{Z})^{(n-K-\ell)\times(K+\ell)}$ and $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1)\times(K+\ell)}$.
3: Compute $\mathbf{Us}^\top = \begin{pmatrix} \mathbf{s}_1^\top \\ \mathbf{s}_2^\top \end{pmatrix}$, where $\mathbf{s}_1 \in (\mathbb{Z}/p^s\mathbb{Z})^{n-K-\ell}$, $\mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$.
4: Set

$$\mathcal{L}_j^{(0)} = \left\{ \mathbf{e}_j^{(0)} \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell} \mid \mathrm{wt}_L\left(\mathbf{e}_j^{(0)}\right) = \frac{v}{2^a} + \sum_{b=0}^{a-1} \frac{\varepsilon_b}{2^b} \right\}$$

 for all $j \in \{1, \ldots, 2^a\}$.
5: **for** $i \in \{1, \ldots a-1\}$ **do**
6:     **for** $j \in \{1, \ldots, 2^{a-i}\}$ **do**
7:         Compute $\mathcal{L}_j^{(i)} = \mathcal{L}_{2j-1}^{(i-1)} \bowtie \mathcal{L}_{2j}^{(i-1)}$ using Algorithm 5.
8: Compute $\mathbf{e} \in \mathcal{L}_1^{(a-1)} \bowtie_{\mathbf{s},2\mathbf{s}_1,\mathbf{A},t-v}^{(a)} \mathcal{L}_2^{(a-1)}$.
9: Return $\mathbf{Pe}$.
10: Else start over at step 1.

In the following, we compute the asymptotic average cost of the representation technique for one level and two levels. In fact, in the case of free codes using two levels gives only a small improvement on using one level and for non-free codes using one level is the optimal choice, as one can see in the comparison in Section 5.

Theorem 4.7.   *1. Let us consider the representation technique on one level, here we need to fix the real numbers V, L, E, such that $0 \le E \le (R_I + L)M - V$. We fix the internal algorithm parameters v, ℓ, ε_0, such that $\lim_{n\to\infty} \frac{v}{n} = V$, $\lim_{n\to\infty} \frac{\ell}{n} = L$, $\lim_{n\to\infty} \frac{\varepsilon_0}{n} = E$. If $V = 0$, then the asymptotic average time complexity is given by $S(1, T) - S(R_I + L, V) - S(1 - R_I - L, T - V)$. If $V > 0$, then the asymptotic average time complexity is given by*

$$S(1, T) - S(R_I + L, V) - S(1 - R_I - L, T - V)$$
$$+ \max\{S(R_I + L, V/2 + E), 2S(R_I + L, V/2 + E) - (R_I + L - R_1)\}.$$

2. If we consider the representation technique on two levels, we fix the real numbers V, L, E_0, E_1 and U, such that $0 \le E_1 \le (R_I + L)M - V$, $0 \le E_0 \le$

$(R_I + L)M - V/2 - E_1$ *and* $0 \leq U \leq R_I + L - R_1$. *We fix the internal algorithm parameters* $\varepsilon_0, \varepsilon_1$ *and* u_1, *such that* $\lim\limits_{n\to\infty} \frac{\varepsilon_0}{n} = E_0, \lim\limits_{n\to\infty} \frac{\varepsilon_1}{n} = E_1$ *and* $\lim\limits_{n\to\infty} \frac{u_1}{n} = U$, *which we recall is by Lemma 4.5 fixed. Then the asymptotic average time complexity is given by*

$$S(1,T) - S(R_I + L, V) - S(1 - R_I - L, T - V) +$$
$$\max\{S(R_I + L, V/4 + E_0 + E_1/2), S(R_I + L, V/2 + E_1) - U,$$
$$2S(R_I + L, V/4 + E_0 + E_1/2) - U,$$
$$2S(R_I + L, V/2 + E_1) - (R_I + L - R_1) - U\}.$$

Proof.   1. The initial lists $\mathcal{L}_1^{(0)}$ and $\mathcal{L}_2^{(0)}$ are both of size $L_0 = F_L(K + \ell, v_0, p^s)$, where $v_0 = \frac{v}{2} + \varepsilon$. We compute $\mathcal{L}_1^{(1)} = \mathcal{L}_1^{(0)} \bowtie \mathcal{L}_2^{(0)}$ using Algorithm 6 on the following inputs: the positive integers $v, t - v, u = K + \ell - k_1$ and $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1) \times (K+\ell)}, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$. This has the asymptotic cost of

$$2S(R_I + L, V/2 + E) - (R_I + L - R_1).$$

Finally, we note that the probability that the error vector has the assumed weight distribution of the representation technique is

$$F_L(K + \ell, v, p^s) F_L(n - K - \ell, t - v, p^s) F_L(n, t, p^s)^{-1}.$$

Hence, the asymptotic number of iterations is given by

$$S(1, T) - S(R_I + L, V) - S(1 - R_I - L, T - V).$$

2. The initial lists $\mathcal{L}_j^{(0)}$ are all of size $L_0 = F_L(K + \ell, v_0, p^s)$, where $v_0 = \frac{v}{4} + \varepsilon_0 + \varepsilon_1/2$. On the first level, the two merges $\mathcal{L}_1^{(1)} = \mathcal{L}_1^{(0)} \bowtie \mathcal{L}_2^{(0)}$ and $\mathcal{L}_2^{(1)} = \mathcal{L}_3^{(0)} \bowtie \mathcal{L}_4^{(0)}$ on u_1 positions cost asymptotically

$$2S(R_I + L, V/4 + E_0 + E_1/2) - U.$$

The lists $\mathcal{L}_b^{(1)}$ have asymptotic sizes $S(R_I + L, V/2 + E_1) - U$.

On the second level we compute $\mathcal{L}_1^{(2)} = \mathcal{L}_1^{(1)} \bowtie \mathcal{L}_2^{(1)}$ using Algorithm 6 on the following inputs: the positive integers $v, t - v, u = K + \ell - k_1$ and $\mathbf{B} \in (\mathbb{Z}/p^s\mathbb{Z})^{(K+\ell-k_1) \times (K+\ell)}, \mathbf{s}_2 \in (\mathbb{Z}/p^s\mathbb{Z})^{K+\ell-k_1}$. This has the asymptotic cost of

$$2(S(R_I + L, V/2 + E_1) - U) - (R_I + L - R_1) + U.$$

Note that the addition of U results from the fact, that we already did merge on u_1 positions, thus we only need to merge on $K + \ell - k_1 - u_1$ positions. The success probability stays the same as in the first level, since it only considers the splitting into the weights v and $t - v$.   □

4.4. LEE-BJMM ALGORITHM. The ISD algorithm BJMM [7] in the Hamming metric is a mixture of the two techniques considered above. We consider the case of two levels and three levels. In both the cases, merging at the first level is done using Algorithm 3 (similar to Wagner's algorithm), and merging at later levels is done using Algorithm 5 (similar to the representation technique). Note also, that Lee BJMM at one level is same as Wagner's approach on one level. Note that the optimal choice for the Hamming metric is three levels. For the Lee metric, we observe in Section 5, that two levels is the optimal choice.

Corollary 3. *1. Let us consider Lee-BJMM on two levels, and fix the real numbers $V, L, E,$ and U such that $0 \leq E \leq (R_I + L)M - V,$ and $0 \leq U \leq R_I + L - R_1$. We fix the internal algorithm parameters v, ℓ, ε_0 and u_1, such that $\lim_{n \to \infty} \frac{v}{n} = V, \lim_{n \to \infty} \frac{\ell}{n} = L, \lim_{n \to \infty} \frac{\varepsilon_0}{n} = E$ and $\lim_{n \to \infty} \frac{u_1}{n} = U$, which we recall is by Lemma 4.5 fixed. The asymptotic average time complexity of BJMM on two levels is given by*

$$S(1, T) - S(R_I + L, V) - S(1 - R_I - L, T - V)$$
$$+ \max\{S((R_I + L)/2, V/4 + E/2), S(R_I + L, V/2 + E) - U,$$
$$2(S(R_I + L, V/2 + E) - U) - (R_I + L - R_1) + U\}.$$

2. Let us consider now Lee-BJMM on three levels, and fix the real numbers $V, L,$ $E_0, E_1,$ and U_0, U_1 such that $0 \leq E_1 \leq (R_I + L)M - V, 0 \leq E_0 \leq (R_I + L)M - V/2 - E_1$ and $0 \leq U_0 \leq U_1 \leq R_I + L - R_1$. We fix the internal algorithm parameters $v, \ell, \varepsilon_0, \varepsilon_1, u_0$ and u_1, such that $\lim_{n \to \infty} \frac{v}{n} = V, \lim_{n \to \infty} \frac{\ell}{n} = L, \lim_{n \to \infty} \frac{\varepsilon_i}{n} = E_i$ and $\lim_{n \to \infty} \frac{u_i}{n} = U_i$ for $i \in \{0, 1\}$, which we recall are by Lemma 4.5 fixed. The asymptotic average time complexity of BJMM at three level is given by

$$S(1, T) - S(R_I + L, V) - S(1 - R_I - L, T - V)$$
$$+ \max\{S((R_I + L)/2, V/8 + E_1/4 + E_0/2), S(R_I + L, V/4 + E_1/2 + E_0) - U_0,$$
$$2S(R_I + L, V/4 + E_1/2 + E_0) - U_1 - U_0, S(R_I + L, V/2 + E_1) - U_1,$$
$$2S(R_I + L, V/2 + E_1) - (R_I + L - R_1) - U_1\}.$$

The proof works similar to Theorem 4.4 and Theorem 4.7, where in the case of two levels, we start with 4 base lists of size $F_L((K + \ell)/2, v/4 + \varepsilon_0/2)$ and in the case of three levels we start with 8 base lists of size $F_L((K+\ell)/2, v/8+\varepsilon_1/4+\varepsilon_0/2)$.

5. COMPARISON

In this section, we numerically compute the asymptotic complexity of the Lee ISD algorithms proposed in Section 4. For our analysis we will consider codes that achieve the asymptotic Gilbert-Varshamov bound in Theorem 2.7. We will exemplary consider $q = 7^2$ and assume that $R_1 = \lambda R$, for $\lambda \in \{0.5, 0.75, 1\}$ and that in general it holds that

$$R \leq \min\left\{1, \lambda R + \frac{(1 - \lambda)Rs}{s - 1}\right\} \leq R_I \leq \min\left\{(1 - \lambda)Rs + \lambda R, 1\right\}.$$

In [11, Theorem 22] it is shown that a code generated by a random matrix is free and achieves the Gilbert-Varshamov bound with high probability. We will hence assume that $R_I = \min\left\{1, \lambda R + \frac{(1-\lambda)Rs}{s-1}\right\}$ in our asymptotic analysis. Note that for such choices, not all rates R may be achieved.

In Figure 2, we compare the asymptotic complexity of all the ISD algorithms at different rates R by optimizing the internal parameters of each algorithm. In Table 1, we find the complexity in the worst case of each algorithm, i.e., we find $e(R^*, q)$ for $R^* = \text{argmax}_{0 \leq R \leq 1}(e(R, q))$. Finally, in Table 2, we compare the worst-case complexity of Lee ISD algorithms with Hamming ISD algorithms, while fixing the base $q = 4$ and $\lambda = 1$.

For free codes, we can see from Figure 2 (A) that BJMM on two levels is the fastest algorithm. In addition, we have observed that Wagner's approach on one
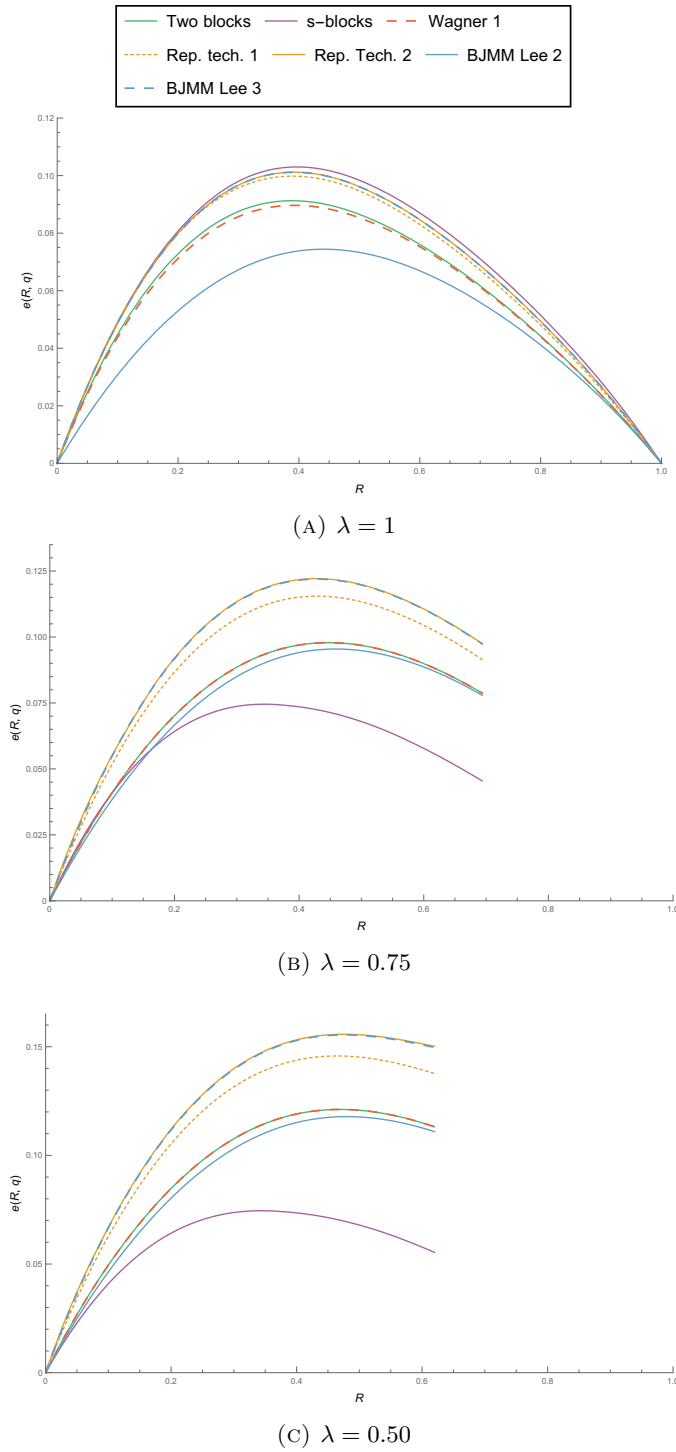
(A) $\lambda = 1$

(B) $\lambda = 0.75$

(C) $\lambda = 0.50$

FIGURE 2. Comparing asymptotic complexity of different algorithms for different values of λ. The values are calculated for $q = 7^2$.

	$\lambda = 1$		$\lambda = 0.75$		$\lambda = 0.5$	
	R^*	$e(R^*, q)$	R^*	$e(R^*, q)$	R^*	$e(R^*, q)$
Two-Blocks	0.3886	0.0913	0.4473	0.0978	0.4694	0.1211
s-Blocks	0.3969	0.1030	0.3441	0.0745	0.3441	0.07453
Wagner $a = 1$	0.3925	0.0897	0.4473	0.0978	0.4694	0.1211
Wagner $a = 2$	0.3925	0.0897	0.4473	0.0978	0.4694	0.1211
Rep. tech. $a = 1$	0.3896	0.0998	0.4288	0.1155	0.4648	0.1457
Rep. tech. $a = 2$	0.3922	0.1012	0.4275	0.1221	0.4757	0.1557
BJMM level 2	0.4414	0.07440	0.4587	0.0954	0.4801	0.1178
BJMM level 3	0.3921	0.1012	0.4282	0.1220	0.4754	0.1554

TABLE 1. Comparison of the asymptotic complexity of all the algorithms at rate $R^* = \mathrm{argmax}_{0 \le R \le 1} (e(R, q))$. The values are calculated for $q = 7^2$.

	$e(R^*, q)$
Lee Metric	
Prange	0.0575
s-Blocks	0.0575
Two-Blocks	0.0556
Wagner $a = 1$	0.0556
Rep. tech. $a = 1$	0.0569
Rep. tech. $a = 2$	0.0571
BJMM level 2	0.05265
BJMM level 3	0.0557
Hamming Metric	
BJMM-MO	0.04294
Stern	0.04987
Prange	0.05095

TABLE 2. Comparison with Hamming metric for $q = 4$ and $\lambda = 1$. The values for Hamming metric ISD algorithms BJMM-MO and Stern are from [21, Table 3] and [22, Table 1], respectively.

level and the two-blocks algorithms have a similar cost, outperforming the representation technique algorithms.

The situation changes drastically when we consider non-free codes. In this case, the s-blocks algorithm outperforms all other algorithms, even BJMM.

Remark 5. Asymptotically, the algorithm based on Wagner's approach on two levels results in the same cost as the level one variant. Indeed it is easy to see that the term U in the asymptotic complexity either gets canceled out or the cost minimizes at $U = 0$. This is also observed in the values in Table 1.

If we compare the asymptotic cost of the Lee-metric ISD algorithms to the corresponding algorithms in the Hamming metric, namely BJMM-MO, Stern and Prange we observe that decoding a random code in the Lee metric has a larger cost than in the Hamming metric, which makes it a promising alternative for cryptographic applications.

We provide all the SAGE [40] and Mathematica [24] programs which can be used to compute the finite and asymptotic workfactors, respectively, at https://git.math.uzh.ch/isd/lee-isd/lee-isd-algorithm-complexities.git

6. Conclusion

In this paper we analyzed the hardness of the syndrome decoding problem in the Lee metric.

We extended the reduction of Barg [5] to the syndrome decoding and the given weight codeword problem of additive weights, thus including also the NP-completeness of the syndrome decoding problem in the Lee metric. In order to analyze more in depth the hardness of this problem, we provided several decoding algorithms for random linear Lee-metric codes and provided their asymptotic cost.

The NP-completeness and the observation that, for a fixed set of code parameters, the cost of decoding a random code in the Lee metric is larger than in the Hamming metric, make the Lee metric a promising candidate for code-based cryptography. We thus emphasize the need of further research on Lee-metric codes with suitable parameters for applications in cryptography. For example, Low-Lee-Density Parity-Check codes were recently introduced [37].

References

[1] M. Abramson. Restricted combinations and compositions. *Fibonacci Quart.*, pages 439–452, 1976.

[2] H. Astola and I. Tabus. Bounds on the size of Lee-codes. In *2013 8th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 471–476. IEEE, 2013.

[3] J. Astola. On the asymptotic behaviour of Lee-codes. *Discrete applied mathematics*, 8(1):13–23, 1984.

[4] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini. A finite regime analysis of information set decoding algorithms. *Algorithms*, 12(10), 2019.

[5] S. Barg. Some new NP-complete coding problems. *Problemy Peredachi Informatsii*, 30(3):23–28, 1994.

[6] A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EURO-CRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 520–536. Springer Verlag, 2012.

[7] A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 520–536. Springer, 2012.

[8] E. Berlekamp. *Algebraic coding theory*. World Scientific, 1968.

[9] E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. on Inf. Theory*, 24(3):384–386, May 1978.

[10] D. J. Bernstein, T. Lange, and C. Peters. Smaller decoding exponents: ball-collision decoding. In *Annual Cryptology Conference*, pages 743–760. Springer, 2011.

[11] E. Byrne, A.-L. Horlemann, K. Khathuria, and V. Weger. Density of free modules over finite chain rings. *arXiv preprint arXiv:2106.09403*, 2021.

[12] A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Trans. on Inf. Theory*, 44(1):367–378, 1998.

[13] A. Canteaut and N. Sendrier. Cryptanalysis of the original McEliece cryptosystem. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 187–199. Springer, 1998.

[14] F. Chabaud. Asymptotic analysis of probabilistic algorithms for finding short codewords. In *Eurocode'92*, pages 175–183. Springer, 1993.

[15] A. Chailloux, T. Debris-Alazard, and S. Etinski. Classical and quantum algorithms for generic syndrome decoding problems and applications to the Lee metric. *arXiv preprint arXiv:2104.12810*, 2021.

[16] L. Chen, Y.-K. Liu, S. Jordan, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone. Report on post-quantum cryptography. Technical Report NISTIR 8105, National Institute of Standards and Technology, 2016.

[17] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the theory and application of cryptographic techniques*, pages 186–194. Springer, 1986.

[18] M. Finiasz and N. Sendrier. Security bounds for the design of code-based cryptosystems. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 88–105. Springer, 2009.

[19] P. Gaborit and G. Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Trans. on Inf. Theory*, 62(12):7245–7252, 2016.

[20] D. Gardy and P. Solé. Saddle point techniques in asymptotic coding theory. In *Workshop on Algebraic Coding*, pages 75–81. Springer, 1991.

[21] C. T. Gueye, J. B. Klamti, and S. Hirose. Generalization of BJMM-ISD using May-Ozerov nearest neighbor algorithm over an arbitrary finite field $\mathbb{F}_q$. In *Codes, Cryptology and Information Security*, pages 96–109. Springer International Publishing, 2017.

[22] S. Hirose. May-Ozerov algorithm for nearest-neighbor problem over $\mathbb{F}_q$ and its application to information set decoding. In *International Conference for Information Technology and Communications*, pages 115–126. Springer, 2016.

[23] A.-L. Horlemann-Trautmann and V. Weger. Information set decoding in the Lee metric with applications to cryptography. *Advances in Mathematics of Communications*, online, 2019.

[24] Wolfram Research, Inc. Mathematica, Version 12.3.1. Champaign, IL, 2021.

[25] C. Interlando, K. Khathuria, N. Rohrer, J. Rosenthal, and V. Weger. Generalization of the ball-collision algorithm. *Journal of Algebra Combinatorics Discrete Structures and Applications*, 7(2):195–207, 2018.

[26] C. Lee. Some properties of nonbinary error-correcting codes. *IRE Trans. Inf. Theory*, 4(2):77–82, Jun. 1958.

[27] P. Lee and E. Brickell. An observation on the security of McEliece's public-key cryptosystem. In *Advances in Cryptology - EUROCRYPT 88*, pages 275–280. Springer Verlag, 1988.

[28] J. S. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Trans. on Inf. Theory*, 34(5):1354–1359, September 1988.

[29] A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 107–124. Springer, 2011.

[30] R. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Progress Report*, pages 114–116, 1978.

[31] A. Meurer. *A coding-theoretic approach to cryptanalysis*. PhD thesis, Ruhr Universität Bochum, 2013.

[32] R. Niebuhr, E. Persichetti, P.-L. Cayrel, S. Bulygin, and J. Buchmann. On lower bounds for information set decoding over $\mathbb{F}_q$ and on the effect of partial knowledge. *Int. J. Inf. Coding Theory*, 4(1):47–78, 2017.

[33] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Probl. Contr. and Inform. Theory*, 15:159–166, 1986.

[34] C. Peters. Information-set decoding for linear codes over $\mathbb{F}_q$. In *International Workshop on Post-Quantum Cryptography*, pages 81–94. Springer, 2010.

[35] E. Prange. The use of information sets in decoding cyclic codes. *IRE Trans. Inf. Theory*, 8(5):5–9, Sep. 1962.

[36] S. Puchinger, J. Renner, and J. Rosenkilde. Generic decoding in the sum-rank metric. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 54–59. IEEE, 2020.

[37] P. Santini, M. Battaglioni, F. Chiaraluce, M. Baldi, and E. Persichetti. Low-Lee-density parity-check codes. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[38] J. Stern. A method for finding codewords of small weight. In *International Colloquium on Coding Theory and Applications*, pages 106–113. Springer, 1988.

[39] J. Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO' 93*, pages 13–21, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.

[40] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.4)*, 2018. https://www.sagemath.org.

[41] W. Ulrich. Non-binary error correction codes. *The Bell System Technical Journal*, 36(6):1341–1388, Nov. 1957.

[42] D. Wagner. A generalized birthday problem. In *Annual International Cryptology Conference*, pages 288–304. Springer, 2002.

Appendix

For completeness we include here the proof of Proposition 2.

Proof. We prove the NP-completeness by a reduction from the 3DM problem. For this, we start with a random instance of 3DM with T of size t, and $U \subseteq T \times T \times T$ of size u. Let us denote the elements in $T = \{b_1, \ldots, b_t\}$ and in $U = \{\mathbf{a}_1, \ldots, \mathbf{a}_u\}$. From this we build the matrix $\mathbf{H}^\top \in R^{u \times 3t}$ like $\overline{\mathbf{H}}^\top$ in the proof of Proposition 3.

With this construction, we have that each column of $\mathbf{H}$ corresponds to an element in U, and has weight 3. Let us set the syndrome $\mathbf{s}$ as the all-one vector of length $3t$. Assume that we can solve the (R, wt)-SDP on the instances $\mathbf{H}, \mathbf{s}$ and t in polynomial time. Let us consider two cases.

<u>Case 1:</u> First, assume that the (R, wt)-SDP solver returns as answer 'yes', i.e., there exists an $\mathbf{e} \in R^u$, of weight less than or equal to t and such that $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$.

- We first observe that we must have $\mathrm{wt}(\mathbf{e}) = |\mathrm{S}(\mathbf{e})| = t$. For this note that each column of $\mathbf{H}$ adds at most 3 non-zero entries to $\mathbf{s}$. Therefore, we need to add at least t columns to get $\mathbf{s}$, i.e., $|\mathrm{S}(\mathbf{e})| \geq t$ and hence $\mathrm{wt}(\mathbf{e}) \geq t$. As we also have $\mathrm{wt}(\mathbf{e}) \leq t$ by hypothesis, this implies that $\mathrm{wt}(\mathbf{e}) = |\mathrm{S}(\mathbf{e})| = t$. Moreover, we have that $\mathrm{wt}(e_i) = 1$ for all $i \in \mathrm{S}(\mathbf{e})$.

- Secondly, we observe that the weight t solution must be a vector with entries in $\{0, 1\}$. For this we note that the matrix $\mathbf{H}$ has entries in $\{0, 1\}$ and has constant column weight three, and since $|\mathrm{S}(\mathbf{e})| = t$, the supports of the t columns of $\mathbf{H}$ that sum up to the all-one vector have to be disjoint. Therefore, we get that the j-th equation from the system of equations $\mathbf{e}\mathbf{H}^\top = \mathbf{s}$ is of the form $e_i h_{i,j} = 1$ for some $i \in \mathrm{S}(\mathbf{e})$. Since $h_{i,j} = 1$, we have $e_i = 1$.

Recall from above that the columns of $\mathbf{H}$ correspond to the elements of U. The t columns corresponding to the support of $\mathbf{e}$ are now a solution W to the 3DM problem. This follows from the fact that the t columns have disjoint supports and add up to the all-one vector, which implies that each element of T appears exactly once in each coordinate of the elements of W.

<u>Case 2:</u> If the solver returns as answer 'no', this is also the correct answer for the 3DM problem. In fact, it is easy to see that the above construction also associates any solution W of the 3DM to a solution $\mathbf{e}$ of the corresponding (R, wt)-SDP.

Thus, if such a polynomial time solver exists, we can also solve the 3DM problem in polynomial time. $\qquad \square$

In Figure 3 we provide a toy example to clarify how the construction in our reduction works. Notice that the very same construction is used also in Proposition 3, where we generalize the canonical problem of finding codewords with given weight, and prove it to be NP-complete.

$\mathbf{a}_1 = \{D, A, B\}$

$\mathbf{a}_2 = \{C, B, A\}$

$\boxed{\mathbf{a}_3 = \{D, A, B\}}$

$\boxed{\mathbf{a}_4 = \{B, C, D\}}$

$\boxed{\mathbf{a}_5 = \{C, D, A\}}$

$\mathbf{a}_6 = \{A, D, A\}$

$\boxed{\mathbf{a}_7 = \{A, B, C\}}$

(A)

$$\mathbf{H}^\top = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(B)

FIGURE 3. Example on how to build an (R, wt)-SDP instance from a 3DM instance, as in Proposition 2. In this example, we have $t = 4$, $u = 7$ and $T = \{A, B, C, D\}$. The triples in the set $U = \{\mathbf{a}_1, \cdots, \mathbf{a}_7\}$ are shown in (a), while (b) reports the construction of $\mathbf{H}^\top$ from U. In (a) we highlighted the subset $W = \{\mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5, \mathbf{a}_7\}$, which constitutes a solution to the 3DM instance. In (b) we highlighted the rows corresponding to the triples in W; it is easily seen that summing the rows of $\mathbf{H}^\top$ corresponding to the triples in W results in the all-one vector, i.e., the target syndrome.