DTU Library

Generic Decoding in the Sum-Rank Metric

Puchinger, Sven; Renner, Julian; Rosenkilde, Johan

Published in:
Proceedings of 2020 IEEE International Symposium on Information Theory

Link to article, DOI:
10.1109/ISIT44484.2020.9174497

Publication date:
2020

Document Version
Peer reviewed version

Citation (APA):
Puchinger, S., Renner, J., & Rosenkilde, J. (2020). Generic Decoding in the Sum-Rank Metric. In *Proceedings of 2020 IEEE International Symposium on Information Theory* (pp. 54-59). Article 9174497 IEEE. https://doi.org/10.1109/ISIT44484.2020.9174497

Generic Decoding in the Sum-Rank Metric

Sven Puchinger[1], Julian Renner[2], Johan Rosenkilde[1]
[1]Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), Denmark
[2]Institute for Communications Engineering, Technical University of Munich (TUM), Germany
Email: svepu@dtu.dk, julian.renner@tum.de, jsrn@dtu.dk

***Abstract*—We propose the first non-trivial generic decoding algorithm for codes in the sum-rank metric. The new method combines ideas of well-known generic decoders in the Hamming and rank metric. For the same code parameters and number of errors, the new generic decoder has a larger expected complexity than the known generic decoders for the Hamming metric and smaller than the known rank-metric decoders.**

***Index Terms*—Sum-Rank Metric, Generic Decoding**

I. Introduction

The sum-rank metric is really a family of metrics, which contains both Hamming and rank metric as special cases, and in general can be seen as a mix of the two. It was introduced under the name "extended rank metric" as a suitable distance measure for multi-shot network coding in 2010 [1]. Since then, several code constructions and efficient decoders have been proposed for the metric [2]–[9]. The codes have also been studied in the context of distributed storage [10] and further aspects of network coding [9].

A generic decoder is an algorithm that takes a code and a received word as input and outputs a codeword that is close to the received word, without any restriction on or knowledge about the structure of the code. Designing such algorithms has a long tradition in coding theory, both for theoretical and practical reasons: studying the complexity of generic decoding is essential to evaluate the practial security level of code-based cryptosystems such as the McEliece [11], Niederreiter [12] and Gabidulin–Paramonov–Tretjakov [13] cryptosystems, or the numerous variants thereof. A trivial generic decoding algorithm is to simply tabulate the input code and compare each codeword with the received word, but there are much more efficient approaches. For the Hamming metric, the related decision problem is NP-hard [14], and there is also a hardness reduction for the analog problem in the rank metric [15], and so it is not surprising that all known generic decoding algorithms still have exponential running time in the code parameters.

Prange [16] presented in 1962 a generic decoder whose type is now know as information-set decoding. The basic idea is to repeatedly choose $n-d$ random positions, where n is the length and d the minimum distance, until the chosen positions contain all the errors. This event can be detected by re-encoding on the remaining d positions, obtaining a codeword, and seeing that this is close to the received word. There have been more than 25 papers improving Prange's algorithm, which have significantly reduced the exponent of the exponential in the complexity expression. A full list of publications up to 2016 can be found in [17, Section 4.1].

In the rank metric, the first generic decoder was proposed in 1996 [18] and since then, there have also been several improvements: [19], [20], [21], and [22]. The complexity of generic decoding in the rank metric remains significantly higher than in the Hamming metric, which results in a substantial advantage of rank-metric-based cryptosystems over their Hamming-metric analogs. The idea here is to repeatedly choose a sub row-space (or column space) of the received word until this contains the error row-space (resp. column space), and when it does use rank-erasure decoding techniques to decode using linear algebra.

In this paper, we propose the first non-trivial generic decoding algorithm for the sum-rank metric. The algorithm combines the sketched ideas for the Hamming and rank metric: we simultaneously randomly choose both the distribution of rank errors across blocks, as well as row spaces inside each block, and the process succeeds when the error row space in each block is covered. The decoding is then performed using sum-rank erasure decoding using linear algebra.

We propose two ways of making this random choice and analyze the resulting expected complexities of the generic decoder. Roughly, the expected complexity smoothly interpolates between the basic generic decoders in the two "extremal" cases of the sum-rank metric: Hamming and rank metric. As a related result, we also propose an efficient algorithm to compute the number of vectors of a given sum-rank weight.

Our work can be seen as a proof-of-concept that known methods of generic decoding can be adapted to the sum-rank metric. Though out of scope of this paper, it seems reasonable that many improvements for generic decoding in Hamming and rank metric can also be applied, which might further reduce the complexity. Moreover, the results open up the possibility to study code-based cryptosystems in the sum-rank metric.

II. Preliminaries

Let q be a prime power and m be a positive integer. Throughout the paper n is the length of the studied codes, and ℓ is a blocking parameter satisfying $\ell \mid n$. We also let $\eta := n/\ell$ and $\mu := \min\{\eta, m\}$. We denote by $\mathbb{F}_q$ the finite field of size q and by $\mathbb{F}_{q^m}$ its extension field of extension degree m, which we will often consider as an $\mathbb{F}_q$-vector space of dimension m. For a vector $\boldsymbol{x} \in \mathbb{F}_{q^m}^\eta$, we define $\mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{x}) := \dim_{\mathbb{F}_q} \langle x_1, \dots, x_\eta \rangle_{\mathbb{F}_q}$. The sum-rank metric is defined as follows.

Definition 1. *The (ℓ-)sum-rank weight is defined as*

$$\mathrm{wt}_{\mathrm{SR},\ell} : \mathbb{F}_{q^m}^n \to \mathbb{Z}_{\geq 0}, \quad \boldsymbol{x} \mapsto \textstyle\sum_{i=1}^{\ell} \mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{x}_i),$$

where we write $\boldsymbol{x} = [\boldsymbol{x}_1 | \boldsymbol{x}_2 | \dots | \boldsymbol{x}_\ell]$ with $\boldsymbol{x}_i \in \mathbb{F}_{q^m}^\eta$. We call

$$[\mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{x}_1), \dots, \mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{x}_\ell)] \in \{0, \dots, \mu\}^\ell$$

the weight decomposition *of $\boldsymbol{x}$. Furthermore, the (ℓ-)sum-rank distance is defined as*

$$\mathrm{wt}_{\mathrm{SR},\ell} : \mathbb{F}_{q^m}^n \times \mathbb{F}_{q^m}^n \to \mathbb{Z}_{\geq 0}, \quad [\boldsymbol{x}_1, \boldsymbol{x}_2] \mapsto \mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{x}_1 - \boldsymbol{x}_2).$$

The family of sum-rank metrics includes two well-known metrics as extremal cases: For $\ell = 1$, it coincides with the rank metric and for $\ell = n$, it is the Hamming metric. In between, we have $\mathrm{wt}_{\mathrm{R}}(\boldsymbol{x}) \leq \mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{x}) \leq \mathrm{wt}_{\mathrm{H}}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$, where $\mathrm{wt}_{\mathrm{R}}(\boldsymbol{x})$ and $\mathrm{wt}_{\mathrm{H}}(\boldsymbol{x})$ are the rank and Hamming weight of $\boldsymbol{x}$.

In the paper, we aim at solving the following problem, independently of the code $\mathcal{C}$ (i.e., generically).

Problem 1 (Generic Sum-Rank-Metric Decoding)**.** *Let $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ be a code, t a positive integer with $0 \leq t \leq \mu\ell$, and $\boldsymbol{r} \in \mathbb{F}_{q^m}^n$ be a vector (received word). If it exists, find a vector (error) $\boldsymbol{e}$ with $\mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{e}) = t$ such that $\boldsymbol{r} - \boldsymbol{e} \in \mathcal{C}$.*

This work has been supported by the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement no. 713683 (COFUNDfellowsDTU) and under the European Research Council (ERC) grant agreement no. 801434.

For non-negative integers a and b, the Gaussian binomial $\begin{bmatrix} a \\ b \end{bmatrix}_q$ is defined by the number of b-dimensional subspaces of $\mathbb{F}_q^a$. We have the bounds [23]:

$$q^{(a-b)b} \leq \begin{bmatrix} a \\ b \end{bmatrix}_q \leq 4q^{(a-b)b}. \quad (1)$$

We let $\mathrm{NM}_q(a, b, i)$ denote the number of $a \times b$ matrices over $\mathbb{F}_q$ of rank exactly i, for $0 \leq i \leq \min\{a, b\}$. We have [24]:

$$\mathrm{NM}_q(a, b, i) = \prod_{j=0}^{i-1} \frac{(q^a - q^j)(q^b - q^j)}{q^i - q^j} \leq 4q^{i(a+b)-i^2}. \quad (2)$$

III. Counting Error Vectors

As the generic decoding problem can be solved by brute-forcing through all vectors of a given sum-rank weight, we are interested in finding the number of such vectors. The question of counting is also related to explicitly writing down a list of such vectors (hence, how to realize this naive generic decoder) and provides a comparative line for the complexity of our new generic decoder that we present in the next section. In the extremal cases of the Hamming and rank metric, simple closed-form expressions are easy to obtain. The question seems more involved for the general sum-rank metric.

We denote by $\mathcal{N}_{q,\eta,m}(t, \ell)$ the number of vectors in $\mathbb{F}_{q^m}^{\eta\ell}$ of ℓ-sum rank weight $t \leq \mu\ell$. The recursion in the following lemma implies an efficient dynamic programming routine to compute the number, which we outline in Algorithm 1. We prove its complexity in Theorem 3 below.

Lemma 2. *$\mathcal{N}_{q,\eta,m}(t, \ell) = 0$ for $t > \mu\ell$. Otherwise:*

$$\mathcal{N}_{q,\eta,m}(t, \ell) = \begin{cases} \mathrm{NM}_q(m, \eta, t), & \text{if } \ell = 1, \\ \displaystyle\sum_{t'=0}^{\min\{\eta,m,t\}} \mathrm{NM}_q(m, \eta, t') \cdot \mathcal{N}_{q,\eta,m}(t - t', \ell - 1), & \text{if } \ell > 1, \end{cases},$$

where $\eta = \frac{n}{\ell}$ and m are constants throughout the recursion.

Proof. The first claim is obvious since each of the ℓ blocks can have at most rank weight μ. For $\ell = 1$, the formula is simply the number of $m \times \eta$ matrices of rank t. For larger ℓ, we sum up over the number of possibilities to choose the rank weight t' of the first block multiplied with the number of sum-rank weight words in the remaining $\ell - 1$ blocks. □

Algorithm 1: Compute $\mathcal{N}_{q,\eta,m}(t, \ell)$

Input : Integers q, η, m, ℓ, t: $t \leq \mu\ell$, $\mu = \min\{\eta, m\}$
Output : Number $\mathcal{N}_{q,\eta,m}(t, \ell)$ of vectors in $\mathbb{F}_{q^m}^{\eta\ell}$ of ℓ-sum-rank weight t

1 Initialize table of integers $\{\mathsf{N}(t', \ell')\}_{t'=0,\ldots,t}^{\ell'=1,\ldots,\ell}$
2 **for** $t' = 0, \ldots, t$ **do**
3     $\mathsf{N}(t', 1) \leftarrow \mathrm{NM}_q(m, \eta, t')$
4 **for** $\ell' = 2, \ldots, \ell$ **do**
5   **for** $t' = 0, \ldots, t$ **do**
6         $\mathsf{N}(t', \ell') \leftarrow \displaystyle\sum_{t''=t'-\mu(\ell'-1)}^{\min\{\mu,t'\}} \mathrm{NM}_q(m, \eta, t'')\mathsf{N}(t' - t'', \ell' - 1)$
7 **return** $\mathsf{N}(t, \ell)$

Theorem 3. *Algorithm 1 is correct and has bit complexity*

$$O^{\sim}\left(\ell t^3(n + m - t)\log_2(q)\}\right).$$

Proof. The algorithm computes a table that fulfills $\mathsf{N}(t', \ell') = \mathcal{N}_{q,\eta,m}(t', \ell')$ for all $t' = 0, \ldots, t$ and $\ell' = 1, \ldots, \ell$ using the recursive formula in Lemma 2. The limits of the sum in Line 6 are chosen such that $0 \leq t'' \leq \min\{\mu, t'\}$ and $0 \leq t' - t'' \leq \mu(\ell' - 1)$ (we have $\mathsf{N}(t' - t'', \ell' - 1) = 0$ for greater $t' - t''$). This implies the correctness.

Complexity-wise, the algorithm performs ℓt^2 integer multiplications, where the (non-negative) integers can be rather large. A very rough upper bound is given by $\mathcal{N}_{q,\eta,m}(t, \ell) \leq \sum_{i=0}^{t} \mathrm{NM}_q(m, n, i) \leq 4(t+1)q^{t(n+m-t)}$, which follows from (2). Since integer multiplication can be implemented with quasi-linear bit operations in the bit size of the involved integers [25], each multiplication costs at most $O^{\sim}\big(t(n+m-t)\log_2(q)\big)$. □

Algorithm 1 can be adapted to create a list of all errors of sum-rank weight t: instead of storing the number of vectors in the table $\mathsf{N}(\cdot, \cdot)$, we store lists of the respective vectors. By brute-forcing the overall list and checking whether the received word minus each error is a codeword (this costs at most $Cn(n-k)m^2$ operations over $\mathbb{F}_q$ for a small constant $C > 0$), we obtain a generic sum-rank-metric decoder with complexity at most

$$W_{\text{errors}} := Cn(n - k)m^2\mathcal{N}_{q,\eta,m}(t, \ell). \quad (3)$$

Remark 4. *The recursion in Lemma 2 can also be turned into an efficient algorithm to draw uniformly at random from the set of sum-rank errors of weight t. For the first block, draw uniformly at random an integer D from the set $\{1, 2, \ldots, \mathcal{N}_{q,\eta,m}(t, \ell)\}$. Then, map D to a sum-rank error by the following method for $i = 1, \ldots, \ell$: choose the entry t_i of the weight decomposition as the largest integer with*

$$D_i := \sum_{t'=t-\mu\ell}^{t_i} \mathrm{NM}_q(m, \eta, t') \cdot \mathcal{N}_{q,\eta,m}(t - t', \ell - 1) < D.$$

Use well-known methods from the rank metric to choose an error $\boldsymbol{e}_i \in \mathbb{F}_{q^m}^{\eta}$ of rank weight t_i. Proceed with the next i, $D \leftarrow D - D_i$, $\ell \leftarrow \ell - 1$.

IV. Generic Decoding in the Sum-Rank Metric

In this section, we present a new generic decoding algorithm for the sum-rank metric. The idea is similar to the generic decoders in the Hamming and rank metric: first we find the "support" of an error (e.g., the error positions in the Hamming metric) in a randomized fashion and second we compute the full error by erasure decoding (e.g., computing the error values after having found the error positions).

Since it is not immediately clear what the analog of a support in the sum-rank metric is compared to the notions of support in Hamming and rank metric, we first introduce this notion in Section IV-A. We also explain under which conditions we can uniquely recover an error given its support (erasure decoding). In Section IV-B, we explain how we determine the support of the error in a randomized way and derive a simple upper bound on the expected complexity of the resulting generic decoder. Section IV-C presents an improved generic decoder. Its expected complexity cannot be determined as easily as the first one, but we present an efficient method to determine an upper bound it.

A. Support and Erasure Decoding in the Sum-Rank Metric

The following lemma gives rise to a notion of "support" in the sum-rank metric, which we state in Definition 2 below.

Lemma 5. *Let $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ have ℓ-sum-rank weight t and let $\boldsymbol{t}$ be its weight decomposition. Then there are vectors*

$$\boldsymbol{a}_i \in \mathbb{F}_{q^m}^{t_i}, \mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{a}_i) = t_i, \quad \text{for } i = 1, \ldots, \ell,$$

as well as matrices over the sub-field $\mathbb{F}_q$:

$$\boldsymbol{B}_i \in \mathbb{F}_q^{t_i \times \eta}, \mathrm{rk}_{\mathbb{F}_q}(\boldsymbol{B}_i) = t_i, \quad \text{for } i = 1, \ldots, \ell,$$

such that

$$\boldsymbol{e} = \overbrace{\begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \boldsymbol{a}_3 & \ldots & \boldsymbol{a}_\ell \end{bmatrix}}^{=:\,\boldsymbol{a}\,\in\,\mathbb{F}_{q^m}^t} \cdot \overbrace{\begin{bmatrix} \boldsymbol{B}_1 & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_2 & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_3 & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{B}_\ell \end{bmatrix}}^{=:\,\boldsymbol{B}\,\in\,\mathbb{F}_q^{t\times n}}.$$

Furthermore, the decomposition is unique up to elementary row operations on the matrices $\boldsymbol{B}_i$. In particular, the row spaces of the matrices $\boldsymbol{B}_i$ are uniquely determined by $\boldsymbol{e}$.

Proof. By basic linear algebra, see e.g. [26], there is an $\boldsymbol{a}_i \in \mathbb{F}_{q^m}^{t_i}$ and $\boldsymbol{B}_i \in \mathbb{F}_q^{t_i \times \eta}$ such that $\boldsymbol{e}_i = \boldsymbol{a}_i \boldsymbol{B}_i$. Also the uniqueness up to row operations follows directly from the analogous results in the rank metric. This implies the result. $\square$

Definition 2. *For $\boldsymbol{e} \in \mathbb{F}_{q^m}^n$ of sum-rank weight t, we define its* (ℓ-sum-rank) support *by*

$$\mathrm{supp}_{\mathrm{SR},\ell}(\boldsymbol{e}) := \mathcal{E}_1 \times \mathcal{E}_2 \times \cdots \times \mathcal{E}_\ell,$$

where $\mathcal{E}_i$ (with $\dim \mathcal{E}_i = t_i$) is the $\mathbb{F}_q$-row space of $\boldsymbol{B}_i$ as in Lemma 5. A super-support of $\boldsymbol{e}$ is a product

$$\mathcal{F} := \mathcal{F}_1 \times \mathcal{F}_2 \times \cdots \times \mathcal{F}_\ell$$

of subspaces $\mathcal{F}_i \subseteq \mathbb{F}_q^\eta$ such that $\mathcal{E}_i \subseteq \mathcal{F}_i$ for all i. We write $\mathrm{supp}_{\mathrm{SR},\ell}(\boldsymbol{e}) \subseteq \mathcal{F}$ and say that $\mathcal{F}$ has weight $\sum_{i=1}^{\ell} \dim(\mathcal{F}_i)$.

Remark 6. *Definition 2 specializes the usual notions of support for Hamming resp. rank metric when $\ell = n$ resp. $\ell = 1$.*

Remark 7. *As in the rank metric (see, e.g., [21]), there are at least two possible definitions of support in the sum-rank metric: we could also use the product of the $\mathbb{F}_q$-spans of the entries of the error's i-th block $\boldsymbol{e}_i$ (i.e., the column space of a matrix representation of $\boldsymbol{e}_i$). This would be advantageous for codes with $m < \eta$. Due to space restrictions, however, we only use the row space definition in this paper since most known codes in the sum-rank metric require $m \geq \eta$.*

The following theorem generalizes the classical Hamming metric statement that $d-1$ is the maximal number of linearly independent columns, as well as the analogous statement in rank metric [26, Theorem 1]:

Lemma 8. *Let $\boldsymbol{H} \in \mathbb{F}_{q^m}^{k \times n}$ be a parity-check matrix of a code $\mathcal{C}[n,k]_{\mathbb{F}_{q^m}}$. Define for any integer $0 \leq t \leq n$ the set*

$$\mathcal{B}_{\ell,t} := \Bigg\{ \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}_1 & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_2 & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B}_3 & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{B}_\ell \end{bmatrix} \in \mathbb{F}_q^{t \times n} : \boldsymbol{B}_i \in \mathbb{F}_q^{t_i \times (n/\ell)},\ \mathrm{rk}(\boldsymbol{B}_i) = t_i,\ \textstyle\sum_{i=1}^{\ell} t_i = t \Bigg\}$$

Then, $\mathcal{C}$ has minimum ℓ-sum-rank distance d if and only if

- *we have $\mathrm{rk}_{\mathbb{F}_{q^m}}(\boldsymbol{H}\boldsymbol{B}^\top) = d-1$ for any $\boldsymbol{B} \in \mathcal{B}_{\ell,d-1}$ and*
- *we have $\mathrm{rk}_{\mathbb{F}_{q^m}}(\boldsymbol{H}\boldsymbol{B}^\top) < d$ for at least one $\boldsymbol{B} \in \mathcal{B}_{\ell,d}$.*

Proof. The proof follows by the decomposition of words of a given ℓ-sum-rank weight in Lemma 5, together with the definition of the minimum sum-rank distance, i.e., that $\boldsymbol{H}\boldsymbol{x}^\top \neq \boldsymbol{0}$ for any word of $\mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{x}) = d-1$ and there is at least one $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ with $\mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{x}) = d$ and $\boldsymbol{H}\boldsymbol{x}^\top = \boldsymbol{0}$. $\square$

Theorem 8 implies the following statement about erasure decoding in the sum-rank metric.

Theorem 9. *Let $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e} \in \mathbb{F}_{q^m}^n$ be a received word, where $\boldsymbol{c}$ is an unknown codeword of a code with minimum sum-rank distance d and $\boldsymbol{e}$ is an unknown error of sum-rank weight at most $d-1$. If we know a super-support $\mathcal{F}$ of $\boldsymbol{e}$ of weight at most $d-1$, then we can uniquely recover $\boldsymbol{c}$ from $\boldsymbol{r}$ with complexity $C(n-k)^3 m^2$ in operations over $\mathbb{F}_q$, where $C > 0$ is a constant.*

Proof. It follows from Lemma 5 that $\boldsymbol{e}$ can be written as $\boldsymbol{a}\boldsymbol{B}$, where $\boldsymbol{B}$ is a block-diagonal matrix containing bases of the super-support entries $\mathcal{F}_i$. Let $\boldsymbol{H}$ be a parity-check matrix of the given code of minimum sum-rank distance $\mathcal{C}$. Since $\mathcal{F}$ has weight $t \leq d-1$, by Lemma 8 (note that we can add suitable rows to $\boldsymbol{B}$ such that it has $\mathbb{F}_q$-rank $d-1$, and then remove the corresponding columns of $\boldsymbol{H}\boldsymbol{B}^\top$), the matrix $\boldsymbol{H}\boldsymbol{B}^\top \in \mathbb{F}_{q^m}^{(n-k)\times t}$ has $\mathbb{F}_{q^m}$-rank t. Hence, the linear system

$$\boldsymbol{H}\boldsymbol{r}^\top = \boldsymbol{H}\boldsymbol{e}^\top = (\boldsymbol{H}\boldsymbol{B}^\top)\boldsymbol{a}^\top,$$

where $\boldsymbol{a}$ is unknown, and $\boldsymbol{r}$, $\boldsymbol{H}$, and $\boldsymbol{B}$ are known, has a unique solution $\boldsymbol{a}$ and we can uniquely determine $\boldsymbol{a}$, $\boldsymbol{e}$, and thus $\boldsymbol{c}$ using linear-algebraic operations. Using elementary matrix multiplication, Gaussian elimination, and polynomial multiplication algorithms, the involved operations have the following complexities, where the constants C_1, C_2, C_3 are close to 1. Multiplying $\boldsymbol{H}\boldsymbol{B}^\top$ costs at most $C_1(n-k)s\eta m$ operations in $\mathbb{F}_q$ since each row of $\boldsymbol{B}$ has at most η non-zero entries. The only remaining step is solving the linear system $\big(\boldsymbol{H}\boldsymbol{B}^\top\big)\boldsymbol{a}^\top = \boldsymbol{s}^\top$, where $\boldsymbol{s}$ is the syndrome of the received word. This costs at most $C_2 s^2(n-k)$ operations over $\mathbb{F}_{q^m}$, and any operation in $\mathbb{F}_{q^m}$ costs again $C_3 m^2$ operations in $\mathbb{F}_q$. $\square$

B. Randomized Support Finding Algorithm

By the analysis in Section IV-A, it suffices to find the support of an error in the rank weight, or a super-support of weight at most $d-1$ thereof, in order to successfully recover an error of sum-rank weight at most $d-1$. Our strategy is to find such a super-support by a randomized algorithm. Therefore we investigate the probability that a random choice is a super-support of the error. We assume first that we know the weight decomposition $\boldsymbol{t}$ of the error and have chosen a weight decomposition $\boldsymbol{s}$ which is element-wise greater than $\boldsymbol{t}$.

Lemma 10. *Let $\boldsymbol{e}$ be an error of support $\mathcal{E} := \mathcal{E}_1 \times \mathcal{E}_2 \times \cdots \times \mathcal{E}_\ell$ with $t_i := \dim \mathcal{E}_i$. Let $\boldsymbol{e}$ be an error of support $\mathcal{E} := \mathcal{E}_1 \times \mathcal{E}_2 \times \cdots \times \mathcal{E}_\ell$ with $t_i := \dim \mathcal{E}_i$. For $s_1, \dots, s_\ell$ with $t_i \leq s_i \leq \eta$, let*

$$\mathcal{F} := \mathcal{F}_1 \times \mathcal{F}_2 \times \cdots \times \mathcal{F}_\ell$$

be a product of uniformly at randomly drawn vector spaces $\mathcal{F}_i \subseteq \mathbb{F}_q^\eta$ of dimension $\dim \mathcal{F}_i = s_i$. Then, the probability that $\mathcal{F}$ is a super-support of $\boldsymbol{e}$ is

$$\Pr(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t} \text{ known}) = \prod_{i=1}^{\ell} \frac{\begin{bmatrix} s_i \\ t_i \end{bmatrix}_q}{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q} \geq 4^{-\ell} q^{-\sum_{i=1}^{\ell} t_i(\eta - s_i)}. \quad (4)$$

Proof. Since the subspaces $\mathcal{F}_i$ are drawn independently, the probability equals the product of the probabilities that the i-th subspace $\mathcal{F}_i$ is a superspace of $\mathcal{E}_i$. This probability is given by $\begin{bmatrix} \eta - t_i \\ s_i - t_i \end{bmatrix}_q \begin{bmatrix} \eta \\ s_i \end{bmatrix}_q^{-1}$, where the numerator counts the number of possibilities to expand the t_i-dimensional subspace $\mathcal{E}_i$ into an s_i-dimensional space and the denominator gives the total number of s_i-dimensional subspaces of $\mathbb{F}_q^\eta$. By properties of the Gaussian binomial coefficient, we get $\begin{bmatrix} \eta - t_i \\ s_i - t_i \end{bmatrix}_q \begin{bmatrix} \eta \\ s_i \end{bmatrix}_q^{-1} = \begin{bmatrix} s_i \\ t_i \end{bmatrix}_q \begin{bmatrix} \eta \\ t_i \end{bmatrix}_q^{-1}$. We obtain the upper bound by (1). $\square$

Algorithm 2 presents a strategy for randomly choosing a super-support for an error $\boldsymbol{e}$ of sum-rank weight t, support $\mathcal{E}$, and weight decomposition $\boldsymbol{t}$. The idea is to first choose a random decomposition $\boldsymbol{t}_{\mathrm{drawn}}$. If $\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}$, then we can lower-bound the success probability $\Pr(\mathcal{E} \subseteq \mathcal{F})$ using Lemma 10, i.e.,

$$\begin{aligned} \Pr(\mathcal{E} \subseteq \mathcal{F}) &\geq \Pr\big(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\big) \cdot \Pr\big(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\big) \\ &\geq \Pr\big(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\big) \cdot 4^{-\ell} q^{-\sum_{i=1}^{\ell} t_i(\eta - s_i)}, \end{aligned} \quad (5)$$

where $\boldsymbol{s}$ is the weight decomposition of $\mathcal{F}$ which element-wise is greater than $\boldsymbol{t}$. The strategy of computing $\boldsymbol{s}$ from $\boldsymbol{t}$ and the integer s (Lines 2–5) is chosen to maximize (for a given $\boldsymbol{t}$) the exponent of the expression $q^{-\sum_{i=1}^{\ell} t_i(\eta - s_i)}$: The $s-t$ "extra" dimensions (where s is usually chosen as $d-1$ to ensure that erasure decoding works) available in $\boldsymbol{s}$ are assigned to those positions i where t_i is largest. In the following, we will refer to this $\boldsymbol{s}$ as $\mathsf{scomp}(\boldsymbol{t}, s)$. We present a lower bound on the probability (5) in Lemma 11, which is independent of the error's weight decomposition $\boldsymbol{t}$ and in this sense a worst-case bound. To state the it, we need the following auxiliary definition.

Definition 3. *Let $t \leq \mu\ell$. Then, we define the* (ordered) quasi-equal decomposition *as*

$$\boldsymbol{t}^{\mathrm{eq}}(t) := \big[t^*, \dots, t^*, t_*, \dots, t_*\big] \in \{0, \dots, \mu\}^\ell,$$

where $t_* := \lfloor \frac{t}{\ell} \rfloor$, $t^* := t_* + 1$, and the number of entries with t^* and t_* is chosen such that $\sum_{i=1}^{\ell} \boldsymbol{t}_i^{\mathrm{eq}} = t$.

Algorithm 2: Random Choice of Super-Support

Input : Integers t and s with $0 \leq t \leq s \leq n-k$

Output : $\mathcal{F} := \mathcal{F}_1 \times \mathcal{F}_2 \times \cdots \times \mathcal{F}_\ell$ of weight s

$\boldsymbol{t}_{\text{drawn}} \leftarrow$ uniformly at random from

$$\mathcal{T}_{t,\ell,\mu} := \left\{[t_1, \ldots, t_\ell] \in \{0, 1, \ldots, \mu\}^\ell : \textstyle\sum_{i=1}^{\ell} t_i = t\right\}$$

$[s_1, \ldots, s_\ell] \leftarrow \boldsymbol{t}_{\text{drawn}}$; $\quad \delta \leftarrow s - t$

while $\delta > 0$ **do**

  $j \leftarrow$ smallest j such that $s_j = \max\limits_{i\,:\,s_i \neq \eta} \{s_i\}$

  $\Delta \leftarrow \min\{\delta, \eta - s_j\}$; $\quad s_j \leftarrow s_j + \Delta$; $\quad \delta \leftarrow \delta - \Delta$

for $\forall i \in [1, \ldots, \ell]$ **do**

  $\mathcal{F}_i \leftarrow$ uniformly at random from $\mathrm{Gr}(\mathbb{F}_q^\eta, s_i)$

return $\mathcal{F} := \mathcal{F}_1 \times \mathcal{F}_2 \times \cdots \times \mathcal{F}_\ell$

Lemma 11. *Let $\boldsymbol{e}$ be an error of ℓ-sum-rank weight t. Let s be an integer with $t \leq s \leq n-k$. Then, Algorithm 2 returns a super-support of $\boldsymbol{e}$ (of support $\mathcal{E}$) with probability at least*

$$\Pr(\mathcal{E} \subseteq \mathcal{F}) \geq \binom{\ell+t-1}{\ell-1}^{-1} 4^{-\ell} \prod_{i=1}^{\ell} q^{-t_i'(\eta - s_i')}$$
$$\geq \binom{\ell+t-1}{\ell-1}^{-1} 4^{-\ell} q^{-\frac{(n-s)t}{\ell} + s - n + \frac{t^2}{\ell^2} - \eta\frac{t}{\ell} - \eta - t - \ell - 1},$$

where $\boldsymbol{t}' := \boldsymbol{t}^{\text{eq}}(t)$ and $\boldsymbol{s}' = \mathsf{scomp}(\boldsymbol{t}', s)$.

Proof. Since $\boldsymbol{t}$ is chosen uniformly at random from $\mathcal{T}_{t,\ell,\mu}$, we have $\Pr\left(\boldsymbol{t}_{\text{drawn}} = \boldsymbol{t}\right) = |\mathcal{T}_{t,\ell,\mu}|^{-1}$, where $|\mathcal{T}_{t,\ell,\mu}|$ can be upper-bounded by $\binom{\ell+t-1}{\ell-1}$ using a stars-&-bars argument.

If $\boldsymbol{t}$ is guessed correctly, the probability of choosing a suitable super-support is lower-bounded by $4^{-\ell} q^{-\sum_{i=1}^{\ell} t_i(\eta - s_i)}$, cf. (4). We show that the quasi-equal distribution minimizes the exponent $-\sum_{i=1}^{\ell} t_i(\eta - s_i)$ among all $\boldsymbol{t}$. Since $-\sum_{i=1}^{\ell} t_i(\eta - s_i)$ is invariant when permuting $\boldsymbol{t}$, w.l.o.g., assume that the t_i are sorted in non-increasing order, i.e., $t_1 \geq t_2 \geq \cdots \geq t_\ell$. Then, Algorithm 2 chooses $\boldsymbol{s}$ by "filling up" the first blocks up to η until the $s-t$ extra dimensions are used. Hence, there is a unique index h (which depends on $\boldsymbol{t}$ and s) with

$$\boldsymbol{s} = \left[\eta, \eta, \ldots, \eta, t_h + \delta, t_{h+1}, \ldots, t_\ell\right],$$

where $0 \leq \delta < \eta - t_h$. Note that we can reach any sorted vector $\boldsymbol{t}$ by starting with a sorted quasi-equal decomposition vector $\boldsymbol{t}' = \boldsymbol{t}^{\text{eq}}$. and iteratively decreasing (by one) the rightmost entry t_j' of $\boldsymbol{t}'$ that is larger than t_j, while simultaneously increasing (by one) the left-most entry t_i' of $\boldsymbol{t}'$ that is smaller than t_i, until $\boldsymbol{t}' = \boldsymbol{t}$. We analyze how $-\sum_{i=1}^{\ell} t_i'(\eta - s_i')$ changes under such an operation. First observe that

$$-\sum_{i=1}^{\ell} t_i'(\eta - s_i') = \underbrace{-t_h'(\eta - t_h' - \delta)}_{=:A} \underbrace{- \sum_{i=h+1}^{\ell} t_i'(\eta - t_i')}_{=:B}$$

where h and δ are defined as above, depending on $\boldsymbol{t}'$ and s. We distinguish the following five cases:

(1) If $i, j < h$, then h, δ, as well as all the t_κ' for $\kappa \geq h$ are unchanged. Hence, the sum does not change. (2) If $i < h$ and $j = h$, then h is unchanged, δ is increased by one, and t_h' is decreased by 1. Hence, A is increased by $\eta - t_h' - \delta > 0$ and B is unchanged, so overall the sum is increased. (3) If $i < h$ and $j > h$, then h is unchanged and δ is increased by one. Thus, A is increased by t_h' and B is changed by $+\eta - 2t_j' + 1$. Due to $t_h' \geq t_j'$ and $\eta \geq t_j'$, the sum is overall increased. Hence, the overall sum is increased by at least 1 (since $t_h' \geq t_j'$). (4) If $i = h$ and $j > h$, then h and δ are unchanged. Thus, A changes by $-(\eta - \delta) + 2t_h' + 1$ and$B$ is changed by $+\eta - 2t_j' + 1$. Due to $t_h' \geq t_j'$, the sum is overall increased. (5) If $i, j > h$, then h, δ, and t_h' are unchanged. Thus, A is unchanged and B is changed by $-\eta + 2t_i' + 1 + \eta - 2t_j' + 1 = 2(t_i' - t_j' + 1)$. Due to $t_i' \geq t_j'$, the sum is overall increased.

Summarized, the sum is never decreased by such an operation, which means that it is minimized for the sorted quasi-equal decomposition $\boldsymbol{t}^{\text{eq}}$. We now bound the sum for this case. With $\frac{t}{\ell} - 1 < t_* \leq t_i^{\text{eq}} \leq t^* < \frac{t}{\ell} + 1$ and $(\eta - t_*)h \geq s - t$ (thus, $h \geq \frac{s-t}{\eta - t_*}$), we obtain

$$-\sum_{i=1}^{\ell} t_i'(\eta - s_i') \geq -\sum_{i=h}^{\ell} t_i'(\eta - t_i') \geq -\sum_{i=h}^{\ell} t^*(\eta - t_*)$$
$$= -(\ell - h + 1)t^*(\eta - t_*) \geq -\left(\ell - \tfrac{s-t}{\eta - t_*} + 1\right) t^*(\eta - t_*)$$
$$\geq -\left[(\ell + 1)\left(\eta - \tfrac{t}{\ell} + 1\right) - s + t\right]\left(\tfrac{t}{\ell} + 1\right)$$
$$= -\tfrac{(n-s)t}{\ell} + s - n + \tfrac{t^2}{\ell^2} - \eta\tfrac{t}{\ell} - \eta - t - \ell - 1$$

This proves the claim. □

Lemma 11 implies an upper bound on the expected complexity of a Las-Vegas type generic decoder that randomly draws a super-support and then applies erasure decoding. We outline the algorithm in Algorithm 3 and state the resulting upper bound on the expected complexity in Theorem 12.

Algorithm 3: Generic Sum-Rank Decoder

Input : Parity-check matrix $\boldsymbol{H}$ of an $[n, k]$ code $\mathcal{C}$ over $\mathbb{F}_{q^m}$ with minimum sum-rank distance d. Let $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$, where $\boldsymbol{c} \in \mathcal{C}$ and $\boldsymbol{e}$ has sum-rank weight t, integer s with $t \leq s \leq n-k$.

Output : $\boldsymbol{e}'$ of sum-rank weight t such that $\boldsymbol{r} - \boldsymbol{e}' \in \mathcal{C}$.

$\boldsymbol{e}' \leftarrow \boldsymbol{0}$

while $\boldsymbol{H}(\boldsymbol{r} - \boldsymbol{e}')^\top \neq \boldsymbol{0}$ *or* $\mathrm{wt}_{\mathrm{SR},\ell}(\boldsymbol{e}') \neq t$ **do**

  $\mathcal{F} := \mathcal{F}_1 \times \mathcal{F}_2 \times \cdots \times \mathcal{F}_\ell \leftarrow$ random support using Algorithm 2 with parameters t and s

  $\boldsymbol{e}' \leftarrow$ erasure decoding w.r.t. $\mathcal{F}$, $\boldsymbol{H}$, $\boldsymbol{r}$ (cf. Thm. 9)

return $\boldsymbol{e}'$

Theorem 12. *Let $\boldsymbol{c}$ be a codeword of a sum-rank metric code $\mathcal{C}$ of minimum sum-rank distance d. Further, let $\boldsymbol{e}$ be an error of sum-rank weight $t < d$. Then, Algorithm 3 with input $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$ and parameter s with $t \leq s < d$ returns an error $\boldsymbol{e}'$ of sum-rank weight t such that $\boldsymbol{r} - \boldsymbol{e}'$ is a codeword. Its expected runtime (in operations over $\mathbb{F}_q$) is at most*

$$W_{\text{new}}^{(\text{simple})} \leq C(n-k)^3 m^2 \binom{\ell+t-1}{\ell-1} 4^\ell \prod_{i=1}^{\ell} q^{t_i'(\eta - s_i')} \tag{6}$$

$$\leq C(n-k)^3 m^2 q^{\frac{(n-s)t}{\ell} - s + n - \frac{t^2}{\ell^2} + \eta\frac{t}{\ell} + \eta + t + \ell + 1 + (\ell-1)\log_q(t+\ell-1)} \tag{7}$$

for a constant $C > 0$, $\boldsymbol{t}' := \boldsymbol{t}^{\text{eq}}(t)$, and $\boldsymbol{s}' = \mathsf{scomp}(\boldsymbol{t}', s)$.

Proof. Correctness follows since if a suitable $\boldsymbol{e}$ exists, there is a non-zero probability that a super-support of $\boldsymbol{e}$ is drawn, and erasure decoding has a unique result for a super-support of weight $s < d$. The complexity is given by the product of the cost of erasure decoding (cf. Theorem 9) and the expected number iterations until we have found a valid super-support $\Pr(\mathcal{E} \subseteq \mathcal{F})^{-1}$ (cf. Lemma 11). The bound follows from

$$\binom{\ell+t-1}{\ell-1} 4^\ell \leq \frac{(t+\ell-1)^{(\ell-1)}}{(\ell-1)!} 4^\ell \leq \tilde{C} q^{(\ell-1)\log_q(t+\ell-1)}$$

for some constant $\tilde{C}$. □

Remark 13. *We can guarantee uniqueness of erasure decoding in Algorithm 3 only for $s < d$, but it might work up to $s = n-k$, depending on the chosen super-support. Note that some generic Hamming- and rank-metric decoding papers use $s = n - k$ without analyzing the erasure decoding success probability.*

C. Improved Support Finding

In the previous subsection, we obtained a simple-to-evaluate upper bound on the expected complexity of our new generic decoder. In this section we improve the strategy of Algorithm 2 for drawing a random super-support. This drawing is somewhat more involved and comes at the cost of a more complicated complexity expression. We do not have a good closed-form bound on this expression, but we show that it is efficiently computable for given input parameters.

Our proof strategy so far was to independently bound $\Pr\left(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right)$ and $\Pr\left(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right)$. We designed the randomized support finding algorithm in a way such that $\Pr(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t})$ is as large as possible in the worst case, i.e., uniformly distributed: $\Pr(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}) = |\mathcal{T}_{t,\ell,\mu}|^{-1}$.

Then we independently bounded $\Pr(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t})$ for the worst case (i.e., the worst $\boldsymbol{t}$). However, this second term depends heavily on $\boldsymbol{t}$ and we can expect to increase both the actual success probability as well as our lower bound by changing the distribution of $\boldsymbol{t}_{\mathrm{drawn}}$ accordingly such that the *entire* term on the right-hand side of (5) is maximized in the worst case. We achieve this by setting

$$\Pr\left(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right) := \frac{\Pr\left(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right)^{-1}}{\sum_{\boldsymbol{t}' \in \mathcal{T}_{t,\ell,\mu}} \Pr\left(\mathcal{E}(\boldsymbol{t}') \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}'\right)^{-1}} \tag{8}$$

since in this case we have

$$\begin{aligned} &\Pr\left(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right) \cdot \Pr\left(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right) \\ &= \left(\sum_{\boldsymbol{t}' \in \mathcal{T}_{t,\ell,\mu}} \Pr\left(\mathcal{E}(\boldsymbol{t}') \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}'\right)^{-1}\right)^{-1}, \end{aligned}$$

which is independent of $\boldsymbol{t}$. By Lemma 10, we have

$$\Pr\left(\mathcal{E} \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right) = \prod_{i=1}^{\ell} \begin{bmatrix} s_i \\ t_i \end{bmatrix}_q \begin{bmatrix} \eta \\ t_i \end{bmatrix}_q^{-1},$$

where $\boldsymbol{s} = \mathsf{scomp}(\boldsymbol{t}, s)$. To stress that each entry s_i of $\boldsymbol{s}$ is a function of $\boldsymbol{t}$, we write $s_i(\boldsymbol{t})$ in the following.

The discussion above directly implies the following result.

Theorem 14. *Suppose that we change Line 1 of Algorithm 2 such that $\boldsymbol{t}_{\mathrm{drawn}}$ is drawn from $\mathcal{T}_{t,\ell,\mu}$ according to the distribution $\Pr\left(\boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right)$ as in (8) instead of uniformly.*

Then, the expected complexity of iterations in Algorithm 3 is upper-bounded by

$$W_{\mathrm{new}}^{(\mathrm{impr})} = C\Big((n-k)^3 m^2 + \mathrm{C}_{\mathrm{draw}}(t,\ell,\eta)\Big) \cdot M, \tag{9}$$

where $\mathrm{C}_{\mathrm{draw}}(t,\ell,\eta)$ is the complexity of drawing according to the distribution in (8), $C > 0$ is a constant, and

$$\begin{aligned} M &:= \sum_{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu}} \Pr\left(\mathcal{E}(\boldsymbol{t}) \subseteq \mathcal{F} \mid \boldsymbol{t}_{\mathrm{drawn}} = \boldsymbol{t}\right)^{-1} \\ &= \sum_{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu}} \prod_{i=1}^{\ell} \begin{bmatrix} \eta \\ t_i \end{bmatrix}_q \begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q^{-1}, \end{aligned} \tag{10}$$

Since the cardinality of the set $\mathcal{T}_{t,\ell,\mu}$, and thus the number of summands in (10), is large, the complexity bound given in Theorem 14 appears to be difficult to compute. However, we can again apply a dynamic programming approach to compute M efficiently (i.e., in small-degree polynomial time in the code parameters). Such an algorithm can also be used to implement the drawing procedure according to the distribution in (8). Due to space restrictions, however, we only briefly discuss the idea of computing M.

Due to the complicated choice of $\boldsymbol{s}$ (in particular the fact that s_i depends on $\boldsymbol{t}$ and not only on t_i), we first rearrange the sum in (10) such that we only sum over ordered weight decompositions:

$$\sum_{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu}} \prod_{i=1}^{\ell} \frac{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q}{\begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q} = \sum_{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu}^{\mathrm{ord}}} \pi(\boldsymbol{t}) \prod_{i=1}^{\ell} \frac{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q}{\begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q}, \tag{11}$$

where $\mathcal{T}_{t,\ell,\mu}^{\mathrm{ord}} := \{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu} : t_1 \geq t_2 \geq \dots \geq t_\ell\}$ and $\pi(\boldsymbol{t})$ denotes the number of possibilities to permute the entries of $\boldsymbol{t}$. It is well-known that $\pi(\boldsymbol{t}) = \frac{\ell!}{\prod_{i=0}^{\eta} j_i!}$, where j_i is the number of times the integer i appears in $\boldsymbol{t}$. Hence, we can split the product in the sum of (11) as follows:

$$\begin{aligned} &\sum_{\boldsymbol{t} \in \mathcal{T}_{t,\ell,\mu}^{\mathrm{ord}}} \pi(\boldsymbol{t}) \prod_{i=1}^{\ell} \frac{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q}{\begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q} = \ell! \sum_{t_1, j \in \mathcal{J}(t,\ell,\eta)} \frac{1}{j!} \prod_{i=1}^{j} \frac{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q}{\begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q} \cdot \\ &\underbrace{\left(\sum_{\boldsymbol{t} \in \mathcal{T}_{t-j\cdot t_1, \ell-j, t_1-1}^{\mathrm{ord}}} \frac{\pi(\boldsymbol{t})}{(\ell-j)!} \prod_{i=1}^{\ell-j} \frac{\begin{bmatrix} \eta \\ t_i \end{bmatrix}_q}{\begin{bmatrix} s_i(\boldsymbol{t}) \\ t_i \end{bmatrix}_q}\right)}_{=: M(t_1, j_1)}, \end{aligned}$$

where $\mathcal{J}(t,\ell,\eta) := \big\{[t_1, j] \,:\, 0 \leq t_1 \leq \min\{\eta, t\},\, 1 \leq j \leq \ell,\, -t_1 j \geq 0,\, (\ell-j)(t_1-1) \geq t - t_1 j\big\}$ and $s_1, \dots, s_j$ only depend on t_1, t, and s (i.e., fill up the first j positions of the $\boldsymbol{t}$ vector (all values $= t_1$) until all but $s - (t - t_1 j)$ of s is used - the remaining $t - t_1 j$ are needed to ensure $s_i \geq t_i$ for $i > j$).

Since the values $M(t_1, j_1)$ are again of the same type as M, we can turn the computation of M into a dynamic programming routine. The table that needs to be computed is indexed by 4 parameters, but we only need to compute a sparse subtable. Hence, it can be implemented efficiently using memoization.

Remark 15. *Algorithm 3 sometimes succeeds if $\boldsymbol{t}_{\mathrm{drawn}} \neq \boldsymbol{t}$, though with much smaller probability. By analyzing this effect, we could get a better upper (and a lower) complexity bound.*

V. Comparison to Other Generic Decoders

Two naive strategies for generic decoding are given by brute-forcing the codewords (complexity $W_{\mathcal{C}} = q^{mk} m^2 kn$, where $m^2 kn$ is cost of encoding) and brute-forcing the errors (complexity W_{errors} as in (3)). Also, for $m \gg \eta$ we can use a generic rank-metric decoder since an error of sum-rank weight t has rank weight t with high probability. Using the basic algorithm in [21], we get a work factor of

$$W_{\mathrm{rank}}^{(\mathrm{basic})} \approx (n-k)^3 m^2 q^{t(n-s)}, \tag{12}$$

where $s < d$ is the weight of the randomly chosen rank supports.

In Table I, we compare the average complexities of these generic decoding algorithms with the algorithms that we propose. We show the $\log_2$ of the number of operations required, where we use $C = 1$ and assume that $\mathrm{C}_{\mathrm{draw}}(t,\ell,\eta)$ is negligible compared to $(n-k)^3 m^2$. Further, we use (6) to compute $W_{\mathrm{new}}^{(\mathrm{simple})}$. For $\ell = 1$, the new algorithm is equal to the basic rank-metric generic decoder of [21]. For $\ell > 1$, we observe that the new algorithms have a significantly reduced complexity compared to all the other possibilities.

Table I

Base-2 logarithms of Complexities of generic decoders for $\ell \in [2, 30]$ and $q = 2$, $m = 40$, $n = 60$, $k = 30$, $t = 10$, $s = 20$.

ℓ	η	$W_{\mathcal{C}}$	W_{errors}	$W_{\mathrm{rank}}^{(\mathrm{basic})}$	$W_{\mathrm{new}}^{(\mathrm{simple})}$	$W_{\mathrm{new}}^{(\mathrm{impr})}$
1	60	1222	913	426	428	426
2	30	1222	676	426	233	233
3	20	1222	595	426	164	157
4	15	1222	555	426	136	127
5	12	1222	532	426	126	112
6	10	1222	517	426	111	97
10	6	1222	488	426	102	74
12	5	1222	481	426	98	69
15	4	1222	475	426	97	64
20	3	1222	469	426	100	58
30	2	1222	463	426	115	54

We also roughly compare the exponential terms in the complexities of the basic algorithms for solving the Hamming- [16], sum-rank- (this work), and rank-metric generic decoding problems (basic algorithm in [21]) for the same length n, error weight t and weight s of the randomly chosen super-supports. In the Hamming metric, we have an exponent of roughly a constant times t, in the sum-rank metric $\approx t\frac{n-s}{\ell}$ (using the upper bound of Theorem 12 with $t \gg \ell$), and in the rank metric $\approx t(n-s)$ (cf. (12)). Hence, the new decoder's complexity smoothly interpolates between the two extreme cases of ℓ.

References

[1] R. W. Nóbrega and B. F. Uchoa-Filho, "Multishot codes for network coding using rank-metric codes," in *2010 Third IEEE International Workshop on Wireless Network Coding*. IEEE, 2010, pp. 1–6.

[2] A. Wachter, V. R. Sidorenko, M. Bossert, and V. V. Zyablov, "On (partial) unit memory codes based on gabidulin codes," *Problems of Information Transmission*, vol. 47, no. 2, pp. 117–129, 2011.

[3] A. Wachter-Zeh and V. Sidorenko, "Rank metric convolutional codes for random linear network coding," in *2012 International Symposium on Network Coding (NetCod)*. IEEE, 2012, pp. 1–6.

[4] A. Wachter-Zeh, M. Stinner, and V. Sidorenko, "Convolutional codes in rank metric with application to random network coding," *IEEE Transactions on Information Theory*, vol. 61, no. 6, pp. 3199–3213, 2015.

[5] D. Napp, R. Pinto, J. Rosenthal, and P. Vettori, "Mrd rank metric convolutional codes," in *2017 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2017, pp. 2766–2770.

[6] ——, "Faster decoding of rank metric convolutional codes," in *23rd International Symposium on Mathematical Theory of Networks and Systems*, 2018.

[7] U. Martínez-Peñas, "Skew and linearized reed–solomon codes and maximum sum rank distance codes over any division ring," *Journal of Algebra*, vol. 504, pp. 587–612, 2018.

[8] D. Boucher, "An algorithm for decoding skew reed-solomon codes with respect to the skew metric," in *Workshop on Coding and Cryptography*, 2019.

[9] U. Martínez-Peñas and F. R. Kschischang, "Reliable and secure multishot network coding using linearized reed-solomon codes," *IEEE Transactions on Information Theory*, 2019.

[10] ——, "Universal and dynamic locally repairable codes with maximal recoverability via sum-rank codes," *IEEE Transactions on Information Theory*, 2019.

[11] R. J. McEliece, "A Public-Key Cryptosystem Based On Algebraic Coding Theory," *Deep Space Network Progress Report*, vol. 42, no. 44, pp. 114–116, 1978.

[12] H. Niederreiter, "Knapsack-Type Cryptosystems and Algebraic Coding Theory," *Problems of Control and Information Theory*, vol. 15, no. 2, pp. 159–166, 1986.

[13] E. M. Gabidulin, A. Paramonov, and O. Tretjakov, "Ideals Over a Non-Commutative Ring and Their Application in Cryptology," in *Advances in Cryptology—EUROCRYPT'91*. Springer, 1991, pp. 482–489.

[14] E. Berlekamp, R. McEliece, and H. Van Tilborg, "On the Inherent Intractability of Certain Coding Problems (Corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978.

[15] P. Gaborit and G. Zémor, "On the hardness of the decoding and the minimum distance problems for rank codes," *IEEE Transactions on Information Theory*, vol. 62(12), pp. 7245–7252, 2016.

[16] E. Prange, "The Use of Information Sets in Decoding Cyclic Codes," *IRE Transactions on Information Theory*, vol. 8, no. 5, pp. 5–9, 1962.

[17] Bernstein et al., "Supporting Documentation of Submission: "Classic McEliece" to the NIST Post-Quantum Standardization," https://classic.mceliece.org/nist.html, version as of March 30, 2019, 2019.

[18] F. Chabaud and J. Stern, "The cryptographic security of the syndrome decoding problem for rank distance codes," in *Advances in Cryptology — ASIACRYPT '96*, K. Kim and T. Matsumoto, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 368–381.

[19] A. V. Ourivski and T. Johansson, "New technique for decoding codes in the rank metric and its cryptography applications," *Problems of Information Transmission*, vol. 38, no. 3, pp. 237–246, Jul 2002. [Online]. Available: https://doi.org/10.1023/A:1020369320078

[20] P. Gaborit, O. Ruatta, and J. Schrek, "On the complexity of the rank syndrome decoding problem," *IEEE Trans. Inform. Theory*, vol. 62, no. 2, pp. 1006–1019, Feb 2016.

[21] N. Aragon, P. Gaborit, A. Hauteville, and J.-P. Tillich, "A new algorithm for solving the rank syndrome decoding problem," in *2018 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2018, pp. 2421–2425.

[22] M. Bardet, P. Briaud, M. Bros, P. Gaborit, V. Neiger, O. Ruatta, and J.-P. Tillich, "An algebraic attack on rank metric code-based cryptosystems," Tech. Rep., 2019, arXiv:1910.00810v1.

[23] R. Koetter and F. R. Kschischang, "Coding for Errors and Erasures in Random Network Coding," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3579–3591, 2008.

[24] T. Migler, K. E. Morrison, and M. Ogle, "Weight and rank of matrices over finite fields," 2004.

[25] D. Harvey and J. van der Hoeven, "Faster integer multiplication using short lattice vectors," *The Open Book Series*, vol. 2, no. 1, pp. 293–310, 2019.

[26] E. M. Gabidulin, "Theory of Codes with Maximum Rank Distance," *Problems of Information Transmission*, vol. 21, no. 1, pp. 3–16, 1985.