A Characterization of Optimal-Rate Linear Homomorphic Secret Sharing Schemes, and Applications

Keller Blackwell
Department of Computer Science, Stanford University, CA, USA

Mary Wootters
Department of Computer Science, Stanford University, CA, USA

Abstract

A *Homomorphic Secret Sharing* (HSS) scheme is a secret-sharing scheme that shares a secret x among s servers, and additionally allows an output client to reconstruct some function $f(x)$, using information that can be locally computed by each server. A key parameter in HSS schemes is *download rate*, which quantifies how much information the output client needs to download from each server. Recent work *(Fosli, Ishai, Kolobov, and Wootters, ITCS 2022)* established a fundamental limitation on the download rate of linear HSS schemes for computing low-degree polynomials, and gave an example of HSS schemes that meet this limit.

In this paper, we further explore optimal-rate linear HSS schemes for polynomials. Our main result is a complete characterization of such schemes, in terms of a coding-theoretic notion that we introduce, termed *optimal labelweight codes*. We use this characterization to answer open questions about the *amortization* required by HSS schemes that achieve optimal download rate. In more detail, the construction of Fosli et al. required amortization over ℓ instances of the problem, and only worked for particular values of ℓ. We show that – perhaps surprisingly – the set of ℓ's for which their construction works is in fact nearly optimal, possibly leaving out only *one* additional value of ℓ. We show this by using our coding-theoretic characterization to prove a necessary condition on the ℓ's admitting optimal-rate linear HSS schemes. We then provide a slightly improved construction of optimal-rate linear HSS schemes, where the set of allowable ℓ's is optimal in even more parameter settings. Moreover, based on a connection to the MDS conjecture, we conjecture that our construction is optimal for all parameter regimes.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives; Theory of computation → Error-correcting codes

Keywords and phrases Error Correcting Codes, Homomorphic Secret Sharing

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.16

Related Version *Full Version*: `https://arxiv.org/abs/2311.14842`

Funding *Keller Blackwell*: KB is supported by a National Science Foundation Graduate Research Fellowship and by a graduate fellowship award from Knight-Hennessy Scholars at Stanford University. KB's work is supported partially supported by NSF grant CCF-2231157.
Mary Wootters: MW's work was partially supported by NSF grants CCF-1844628, CCF-2133154, and CCF-2231157.

Acknowledgements We thank Yuval Ishai and Victor Kolobov for helpful conversations, and the anonymous referees for helpful feedback.

1 Introduction

A *Homomorphic Secret Sharing* (HSS) scheme is a secret sharing scheme that supports computation on top of the shares [5, 11, 12]. Homomorphic Secret Sharing has found many applications in recent years, from private information retrieval to secure multiparty computation (see, e.g., [8, 12]).

In more detail, a standard t-private (threshold) secret sharing scheme shares an input x as s *shares*, $\mathsf{Share}(x) = (y_1, \ldots, y_s)$, which are then distributed among s servers; the goal is that any $t+1$ of the servers can together recover the secret x, while no t of the servers can learn anything about x.[1]

A t-private *HSS* scheme has the additional feature that the servers are able to compute functions f in some function class $\mathcal{F}$, as follows. Each server j does some local computation on its share y_j to obtain an *output share* $z_j = \mathsf{Eval}(f, j, y_j)$. The output shares $z_1, \ldots, z_s$ are then sent to an *output client*, who uses these output shares to recover $f(x) = \mathsf{Rec}(z_1, \ldots, z_s)$. Formally, we say that the HSS scheme π is given by the tuple of functions $(\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ (see Definition 8).

In order for this notion to be interesting, the output shares z_j should be substantially smaller than the original shares y_j; otherwise any $t+1$ servers could just communicate their entire shares y_j to the output client, who would recover x and then compute $f(x)$. To that end, prior work [22] focused on the the *download rate* of (information-theoretically secure) HSS schemes. The download rate (see Definition 11) of an HSS scheme is the ratio of the number of bits in the output $f(x)$ (that is, the number of bits the output client wants to compute), to the number of bits in all of the output shares z_j (that is, the number of bits that the output client downloads); ideally this rate would be as close to 1 as possible.

The work [22] focused on HSS schemes for the function class $\mathcal{F} = \mathrm{POLY}_{d,m}(\mathbb{F})$, the class of all m-variate, degree-d polynomials over a finite field $\mathbb{F}$, and we will do the same here. One of the main results of that work was an infeasibility result on the download rate for *linear* HSS schemes, which are schemes where both Share and Rec are linear over some field; note that Eval (which converts the shares y_j to the output shares z_j) need not be linear.

▶ **Theorem 1** ([22] (Informal, see Theorem 15)). *Any t-private s-server linear HSS scheme for $POLY_{d,m}(\mathbb{F})$ has download rate at most $(s-dt)/s$.*

In fact, [22] proved that the bound $(s-dt)/s$ holds even if the HSS scheme is allowed to "amortize" over ℓ instances of the problem. That is, given ℓ secrets $x^{(1)}, \ldots, x^{(\ell)}$, shared *independently* as $y_j^{(i)}$ for $j \in [s], i \in [\ell]$, the j'th server may do local computation on its shares $y_j^{(1)}, \ldots, y_j^{(\ell)}$ to compute an output share z_j; the output client needs to recover $f_1(x^{(1)}), \ldots, f_\ell(x^{(\ell)})$, given $z_1, \ldots, z_s$.

Moreover, [22] complemented this infeasibility result with a construction achieving download rate $(s-dt)/s$, provided that the amortization factor ℓ is a sufficiently large multiple of $(s-dt)$:

▶ **Theorem 2** ([22]). *Suppose that $s > dt$, and suppose that $\ell = j(s-dt)$ for some integer $j \geq \log_{|\mathbb{F}|}(s)$. Then there is a t-private, s-server linear HSS scheme for $\mathrm{POLY}_{d,m}(\mathbb{F})$ (with CNF sharing, see Definition 16), download rate at least $(s-dt)/s$, and amortization parameter ℓ.*

This state of affairs leaves open two questions, which motivate our work:

(1) Are there optimal-rate linear HSS schemes for $\mathrm{POLY}_{d,m}(\mathbb{F})$ beyond the example in Theorem 2? Can we characterize them?

(2) Is some amount of amortization necessary to achieve the optimal rate? If so, which amortization parameters ℓ admit optimal-rate linear HSS schemes for $\mathrm{POLY}_{d,m}(\mathbb{F})$?

[1] In this work, we focus on *information-theoretic* security, so the above statement means that the joint distribution of any t shares does not depend on the secret x.

1.1 Main Results

We answer both questions (1) and (2) above. For all of our results, we consider *CNF sharing* [24] (see Definition 16). It is known that CNF sharing is universal for linear secret sharing schemes, in that t-CNF shares can be locally converted to shares of *any* linear t-private secret sharing scheme [17].

High-level answers to Questions (1) and (2).

(1) Our results give a complete characterization of the Rec functions for optimal-rate linear HSS schemes for $\text{POLY}_{d,m}(\mathbb{F})$. Our characterization is in terms of linear codes with optimal *labelweight*, a notion that we introduce, which generalizes distance. Similar to how optimal-distance (MDS) codes are characterized by a property of their generator matrices, we are also able to characterize the generator matrices of optimal-labelweight codes.
We hope that the notion of labelweight will find other uses, and we believe that this characterization will be useful for studying linear HSS schemes beyond our work.

(2) Using our characterization, we show that the amortization parameters ℓ given in Theorem 2 are in fact (usually) *all* of the admissible ℓ's.[2] In particular, some amortization is required, and there are parameter regimes where one cannot obtain optimal download rate with amortization parameters ℓ other than those in the construction in [22]. Moreover, we give a construction that *slightly* improves on the construction in [22], closing the gap between the lower and upper bounds in a few more parameter regimes.
We find this answer to Question (2) somewhat surprising; we expected that the particular form of ℓ in Theorem 2 was an artifact of the construction, but it turns out to be intrinsic.

We describe our results in more detail below.

(1) A characterization of optimal-rate linear HSS schemes. Our main result is a characterization of linear HSS schemes for $\text{POLY}_{d,m}(\mathbb{F})$ with optimal download rate $(s-dt)/s$. In particular, we show that the Rec algorithms for such schemes (with CNF sharing) are *equivalent* to a coding-theoretic notion that we introduce, which we refer to as codes with *optimal labelweight.*

▶ **Definition 3** (Labelweight). *Let $\mathcal{C} \subseteq \mathbb{F}^n$ be a linear code of dimension ℓ. Let $\mathcal{L} : [n] \to [s]$ be any function, which we refer to as a* labeling *function. The* labelweight *of $\mathbf{c} \in \mathcal{C}$ is the number of distinct labels that the support of $\mathbf{c}$ touches:*

$$\Delta_{\mathcal{L}}(\mathbf{c}) = |\{\mathcal{L}(i) \,:\, i \in [n], c_i \neq 0\}|.$$

The labelweight *of $\mathcal{C}$ is the minimum labelweight of any nonzero codeword:*

$$\Delta_{\mathcal{L}}(\mathcal{C}) = \min_{c \in \mathcal{C} \setminus \{0\}} \Delta_{\mathcal{L}}(\mathbf{c}).$$

In particular, if $s = n$ and $\mathcal{L}(j) = j$ for all $j \in [n]$, then $\Delta_{\mathcal{L}}(\mathcal{C})$ is just the minimum Hamming distance of $\mathcal{C}$. Thus, the labelweight of a code generalizes the standard notion of distance.

Our main characterization theorem is the following.

[2] The parenthetical "(usually)" is due to a corner case; it is possible that $\ell = j(s-dt)$ with $j = \lceil \log_{|\mathbb{F}|}(s) \rceil - 1$, which is one smaller than the bound given in Theorem 2.

▶ **Theorem 4** (Optimal linear HSS schemes are equivalent to optimal labelweight codes. (Informal, see Lemma 23 and Theorem 27))**.** *Let $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ be a t-private, s-server linear HSS for $\mathrm{POLY}_{d,m}(\mathbb{F})$, with download rate $(s-dt)/s$. Let G be the matrix that represents* Rec *(see Observation 21). Then there is some labeling function $\mathcal{L}$ so that G is the generator matrix for a code $\mathcal{C}$ with rate $(s-dt)/s$ and with $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt+1$.*

Conversely, suppose that there is a labeling function $\mathcal{L} : [n] \to [s]$ and a linear code $\mathcal{C} \subseteq \mathbb{F}^n$ with rate $(s-dt)/s$ and with $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt+1$. Then any generator matrix G of $\mathcal{C}$ describes a linear reconstruction algorithm Rec *for an s-server t-private linear HSS for $\mathrm{POLY}_{d,m}(\mathbb{F})$ that has download rate $(s-dt)/s$.*

We remark that the converse direction is constructive: given the description of such a code $\mathcal{C}$, the proof (see Theorem 27) gives an efficient construction of the Eval function as well as the Rec function.

Given Theorem 4, we now want to know when optimal labelweight codes exist. Our second main result – which may be of independent interest – characterizes these codes in terms of their generator matrices. We describe this characterization more in the Technical Overview (Section 1.2), and next we describe the implications for HSS schemes.

(2) Understanding the amortization parameter ℓ. Using our characterization above, we are able to nearly completely classify which amortization parameters ℓ admit download-optimal linear HSS schemes for $\mathrm{POLY}_{d,m}(\mathbb{F})$. Given Theorem 4, it suffices to understand when optimal labelweight codes exist. We show the following theorem.

▶ **Theorem 5** (Limitations on optimal labelweight codes. (Informal, see Theorem 31))**.** *Suppose that $G \in \mathbb{F}_q^{\ell \times n}$ is the generator matrix for a code $\mathcal{C}$ with rate $\ell/n = (s-dt)/s$, and suppose that there is a labeling function $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt+1$. Then $\ell = j(s-dt)$ for some integer j, with $j \geq \max\{\log_q(s-dt+1), \log_q(dt+1)\}$.*

This result *nearly* matches the feasibility result of [22] (Theorem 2 above). That is, Theorem 2 allows for $\ell = j(s-dt)$ for any integer $j \geq \log_q(s)$. As we always have $\max(s-dt+1, dt+1) \geq s/2$, the conclusion in Corollary 5 implies that $j \geq \log_q(s/2) = \log_q(s) - 1/\log_2(q)$. As j must be an integer, this exactly matches the amortization parameter in Theorem 2 whenever

$$\left\lceil \frac{\log(s)-1}{\log(q)} \right\rceil = \left\lceil \frac{\log(s)}{\log(q)} \right\rceil,$$

which holds for most values of s and q when q is large. Moreover, we give a construction that *very* slightly improves on the one from Theorem 2, which exactly matches Theorem 5 for even more settings of s:

▶ **Theorem 6** (Construction of optimal labelweight codes. (Informal, see Theorem 30))**.** *Let j be any integer so that*

$$j \geq \begin{cases} \log_q(s-1) & q^j \text{ odd, or } (s-dt) \notin \{3, q^j-1\} \\ \log_q(s-2) & q^j \text{ even, and } (s-dt) \in \{3, q^j-1\} \end{cases}.$$

There is an explicit construction of a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ with dimension $\ell = j(s-dt)$, block length $n = js$, (and hence rate $(s-dt)/s$), and a labeling function $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt+1$.

Given Theorem 4 (our equivalence between codes with good labelweight and linear HSS schemes), Theorem 6 immediately implies that there is a download-optimal linear HSS scheme for $\text{POLY}_{d,m}(\mathbb{F})$ with amortization parameter $\ell = j(s - dt)$ for any j as in Theorem 6. (See Corollary 33 for a formal statement).

The HSS result implied by Theorem 6 is quite close to the existing result from [22] (Theorem 2).[3] The only quantitative difference between Theorem 6 and the result of [22] is that we can take j to be either $\lceil \log_q(s-1) \rceil$ or $\lceil \log_q(s-2) \rceil$, rather than $\lceil \log_q(s) \rceil$. Meanwhile, Theorem 5 says that we must have $j \geq \lceil \log_q(s/2+1) \rceil$. So there are a few values of s where Theorem 6 is tight (matching Theorem 5) but Theorem 2 (the construction from [22]) is not. For example, if $q = 2$ and $s = 2^r + 1$, then $\lceil \log_q(s-2) \rceil = \lceil \log_q(s/2+1) \rceil = r$, but $\lceil \log_q(s) \rceil = r + 1$ is larger.

There are still a few parameter regimes where there is a gap (of size one) between the j's that work in Theorem 6 and bound of Theorem 5. We conjecture that in fact the *construction* is optimal, and Theorem 5 is loose. Our work establishes a connection between constructions of download-optimal HSS schemes and the *MDS conjecture* over extension fields, which may be of independent interest; due to space constraints, details are deferred to the full manuscript [6]. Further, we show that, assuming the MDS conjecture, our construction of Theorem 6 is optimal within a natural class of constructions, even when it does not match our infeasibility result in Theorem 5.

1.2 Technical Overview

In this section, we give a high-level overview of our techniques.

Relationship between download-optimal HSS schemes and optimal labelweight codes. Theorem 4 states that download-optimal linear HSS schemes are equivalent to optimal-labelweight codes. To give some intuition for the connection, we explain the forward direction, which is simpler. For simplicity, suppose that the function f is identity, amortized over ℓ secrets $x^{(1)}, \ldots, x^{(\ell)} \in \mathbb{F}$. This problem is called *HSS for concatenation* in [22]; the goal is to share the ℓ secrets independently, and then communicate them to the output client using significantly less information than simply transmitting $t + 1$ shares of each secret.

Consider a linear HSS scheme for this problem. By definition, the reconstruction algorithm Rec is linear, and so can be represented by a matrix $G \in \mathbb{F}^{\ell \times n}$, where n is the total number of symbols of $\mathbb{F}$ sent by all of the servers together. In more detail, we view each output share z_j as a vector over $\mathbb{F}$, and concatenate all of them to obtain a vector $\mathbf{z} \in \mathbb{F}^n$. Then we can linearly recover the ℓ concatenated secrets from $\mathbf{z}$:

$$\begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(\ell)} \end{pmatrix} = \mathsf{Rec}(z_1, \ldots, z_s) = G\mathbf{z}.$$

Now we can define a labeling function $\mathcal{L} : [n] \to [s]$ so that $\mathcal{L}(r) \in [s]$ is the identity of the server sending the r'th symbol in $\mathbf{z}$. We claim that G is the generator matrix[4] of a code $\mathcal{C}$ with $\Delta_{\mathcal{L}}(\mathcal{C}) \geq t + 1$. (Recall that in HSS for concatenation, $d = 1$, so $t + 1 = dt + 1$ is the bound on $\Delta_{\mathcal{L}}(\mathcal{C})$ that we want in Theorem 4). To see this, suppose that there were some codeword $\mathbf{m}^T G$ for some $\mathbf{m} \in \mathbb{F}^\ell$ with labelweight at most t. But consider the quantity

[3] In addition to the quantitative results being quite close, the constructions themselves are also similar. This is not surprising, given that one of our main results is that such schemes must be extremely structured. We discuss the relationship between our construction and that of [22] in the technical overview below.

[4] When we say that G is the *generator matrix* of $\mathcal{C} \subseteq \mathbb{F}^n$, we mean that $\mathcal{C}$ is the rowspan of G.

$$X := \sum_{i=1}^{\ell} m_i x^{(i)} = \mathbf{m}^T G \mathbf{z}.$$

If $\mathbf{m}^T G$ had labelweight at most t (with this labeling $\mathcal{L}$), that means that the quantity X can be computed using messages coming from only t of the servers. But this contradicts the t-privacy of the original secret sharing scheme. Indeed, suppose without loss of generality that $m_1 \neq 0$. Then if $x^{(2)} = \cdots = x^{(\ell)} = 0$, some set of t servers would be able to recover $X = m_1 x^{(1)}$ and hence $x^{(1)}$, and this should be impossible. For the full proof of this direction, we need to generalize to larger d's and more general functions, but the basic idea is the same.

The converse, showing that any optimal-labelweight code $\mathcal{C}$ implies an optimal HSS scheme, is a bit trickier. The main challenge is that the connection above tells us what Rec should be – it should be given by the generator matrix of $\mathcal{C}$ – but it does not tell us what Eval should be, or that an appropriate Eval function even exists. To find Eval, we view the output shares as vectors of polynomials in the input shares. That is, each server must return some function of the shares that it holds, and over a finite field, every function is a polynomial. In this view, we can set up an affine system to solve for Eval, where the variables are the coefficients that appear in each server's output polynomials. Then we show that this system has a solution, and that this solution indeed leads to a legitimate Eval function.

Characterization of Optimal Labelweight Codes. In order to understand when optimal labelweight codes exist, we give a characterization of them in terms of their generator matrices. We show that the generator matrices of optimal labelweight codes can be taken to be *block-totally-nonsingular*. We defer the formal definition to Definition 35, but informally, a block-totally-nonsingular matrix is made up of $j \times j$ invertible blocks $A \in GL(\mathbb{F}, j)$, with the property that any square sub-array of blocks (not necessarily contiguous) is full-rank.

▶ **Lemma 7** (Characterization of optimal label-weight codes. (Informal, see Lemma 36))**.** *Suppose that $\mathcal{C} \subseteq \mathbb{F}^n$ is a code of rate $(s - dt)/s$, and suppose that there is a labeling function $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. Then, up to a permutation of the coordinates, there is a generator matrix G for $\mathcal{C}$ that looks like $G = [I|A]$, where A is block-totally-nonsingular. Conversely, any such matrix is the generator matrix for a code $\mathcal{C}$ with $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$, where $\mathcal{L} : [n] \to [s]$ is the labeling function $\mathcal{L}(x) = \lceil x/j \rceil$.*

To give some intuition for the lemma, we first explain where the "block" structure comes from. We show (Lemma 24) that in fact any labeling function $\mathcal{L}$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) = dt + 1$ must be *balanced*, meaning that each set $\mathcal{L}^{-1}(i)$ for $i \in [s]$ has the same cardinality, $|\mathcal{L}^{-1}(i)| = j$, for some integer j. The blocks then correspond to the sets $\mathcal{L}^{-1}(i)$ for $i \in [s]$.

Next, we give some intuition for the "totally non-singular" part. The basic idea is that if there were a square sub-array of blocks that were singular, then there would be a vector $\mathbf{m} \in \mathbb{F}^{\ell}$ with support on only the relevant blocks, so that $\mathbf{m}^T A = 0$. If $G = [I|A]$ as in Lemma 7, then this implies that $\mathbf{m}^T G$ has support only on the labels that appear on the first ℓ columns of G, which it turns out is small enough to contradict the optimal labelweight property. We refer to the proof of Lemma 36 for details on both points.

Applications to Amortization. We apply the machinery described above to Question (2), about what amortization parameters ℓ are possible, as follows. As described above, we show that optimal-download-rate HSS schemes are equivalent to optimal labelweight codes, which in turn are equivalent to to block-totally-nonsingular matrices. At this point, we already

know that the amortization parameter ℓ must be equal to $j(s-dt)$ for some integer j; this follows from the $j \times j$ block structure of block-totally-nonsingular matrices. It remains only to understand for which j's such matrices exist. A limitation on such matrices follows from a standard counting argument, and this implies our infeasibility result, given formally as Theorem 31.

As noted above, there is already a near-optimal construction of HSS schemes in [22] (Theorem 2). In our language, that construction is based on systematic generator matrices of *Reed-Solomon Codes*. After an appropriate conversion to block form, such matrices are of the form $[I|A]$, where A is a block-totally-nonsingular matrix. Up to the conversion to block-form, the famous *MDS Conjecture* (see discussion in the full version [6]) implies that Reed-Solomon codes are *nearly* optimal in this context, but it is known that a slight improvement is possible: instead of requiring $j \geq \log_q(s)$ as in Theorem 2, one can get $j \geq \log_q(s-2)$ or $\log_q(s-1)$, as in Theorem 6. The change is simple: one essentially adds one or two more (block) columns to the generator matrix ((see, e.g., [28, 1]); and the proof of Theorem 30). Thus, we can slightly improve on the construction of [22] (Theorem 2) by making this slight improvement to the underlying code.

1.3 Related Work

Linear HSS schemes (for, e.g. low-degree polynomials) are implicit in classical protocols for tasks like secure multiparty computation and private information retrieval [5, 4, 14, 18, 2, 3, 15]. More recently, [12] initiated the systematic study of HSS, and in there has been a long line of work on the topic, most of which has focused on HSS schemes that are *cryptographically* secure [9, 20, 10, 11, 21, 12, 13, 7, 16, 25, 26, 19]. In contrast, we focus on *information-theoretic security*. The information-theoretic setting was explored in [12] and was further studied in [22], which is the closest to our work and also our main motivation. In particular, [22] focused the download rate of information-theoretically secure HSS schemes (both linear and non-linear), but did not focus quantitatively on the amortization parameter ℓ. In contrast, we restrict our attention to linear schemes, but focus on characterizing such schemes and on pinning down ℓ.

We note that the work [22] also obtained linear HSS schemes using coding-theoretic techniques, but the connection that they exploited is different. In particular, they show that for the case of $d = 1$ (that is, *HSS for concatenation*), the existence of linear HSS schemes for $\text{POLY}_{d=1,m}(\mathbb{F})^\ell$ is equivalent to the existence of linear codes with a particular rate and distance. However, this characterization only works when $d = 1$. In contrast, our characterization, in terms of codes with good *labelweight*, a generalization of distance, applies for general d.

Finally, we mention the *MDS conjecture*, which is related to our results. The MDS conjecture was stated by Segre in 1955 [27], and roughly says that Reed-Solomon codes have the best alphabet size possible for any *Maximum-Distance Separable* (MDS) code. After being open for over 60 years, the conjecture has been proved for *prime* order fields, and particular extension fields [1]. Our characterization of optimal-rate HSS schemes leads us to consider the related question of the best alphabet size possible for any *Totally Nonsingular* matrix, which we observe in the full version of this paper [6] is equivalent to the MDS conjecture over extension fields. This connection leads us to conjecture that our construction is in fact optimal, and we show that the MDS conjecture implies that it is, within a natural class of constructions. See the discussion in the full version of the paper [6] for more on this connection.

1.4 Open Questions and Future Directions

Before we get into the details, we take a moment to highlight some open questions.

- The most obvious question that our work leaves open is about the edge cases in the characterization of amortization parameters ℓ that are admissible for optimal-download linear HSS schemes. Given the relationship to the MDS conjecture over extension fields, resolving this may be quite a hard problem. However, there is some hope. In particular, the MDS conjecture would imply that our construction is optimal only within a natural class of constructions; it may be possible to get improved results by leaving that class.
- Another open question is to find further applications of our characterization of download optimal HSS schemes. We use this characterization to nearly resolve the question of what amortization parameters are admissible for such schemes, but we hope that it will be useful for other questions about linear information-theoretic HSS.
- Finally, it would be interesting to extend our results to linear HSS schemes that do not have optimal download rate. For example, perhaps it is possible to get a drastic improvement in the amortization parameter ℓ by backing off from the optimal download rate by only a small amount.

1.5 Organization

In Section 2, we set notation and record a few formal definitions that we will need. In Section 3, we show that download-optimal HSS schemes are equivalent to codes with good labelweight: Lemma 23 establishes that HSS schemes imply codes with good labelweight, and Theorem 27 establishes the converse. In Section 4, we state our characterization of generator matrices of codes with good labelweight (Lemma 36), and explain how this can be used to prove bounds on good labelweight codes (Theorems 30 and 31); this implies Corollary 33, which gives nearly-tight bounds on the amortization parameter ℓ in download-optimal linear HSS schemes for $\mathrm{POLY}_{d,m}(\mathbb{F})$.

Throughout this extended abstract, we have had to omit details due to space constraints; we refer the reader to the full version [6] for all the details.

2 Preliminaries

We begin by setting notation and some definitions that we will need throughout the paper.

Notation. For $n \in \mathbb{Z}^+$, we denote by $[n]$ the set $\{1, 2, \ldots, n\}$. We use bold symbols (e.g., $\mathbf{x}$) to denote vectors. For an object w in some domain $\mathcal{W}$, we use $\|w\| = \log_2(|\mathcal{W}|)$ to denote the number of bits used to represent w.

2.1 Homomorphic Secret Sharing

We consider homomorphic secret sharing (HSS) schemes with m inputs and s servers; each input is shared independently. We denote by $\mathcal{F} = \{f : \mathcal{X}^m \to \mathcal{O}\}$ the class of functions we wish to compute, where $\mathcal{X}$ and $\mathcal{O}$ are input and output domains, respectively.

▶ **Definition 8** (HSS). *Given a collection of s servers and a function class $\mathcal{F} = \{f : \mathcal{X}^m \to \mathcal{O}\}$, consider a tuple $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$, where $\mathsf{Share} : \mathcal{X} \times \mathcal{R} \to \mathcal{Y}^s$, $\mathsf{Eval} : \mathcal{F} \times [s] \times \mathcal{Y} \to \mathcal{Z}^*$, and $\mathsf{Rec} : \mathcal{Z}^* \to \mathcal{O}$ as follows[5]:*

[5] By $\mathcal{Z}^*$, we mean a vector of some number of symbols from $\mathcal{Z}$.

- $\mathsf{Share}(x_i, r_i)$*: For* $i \in [m]$, Share *takes as input a secret* $x_i \in \mathcal{X}$ *and randomness* $r_i \in \mathcal{R}$*; it outputs* s *shares* $(y_{i,j} : j \in [s]) \in \mathcal{Y}^s$*. We refer to the* $y_{i,j}$ *as* input shares*; server* j *holds shares* $(y_{i,j} : i \in [m])$*.*
- $\mathsf{Eval}\,(f, j, (y_{1,j}, y_{2,j}, \ldots, y_{m,j}))$*: Given* $f \in \mathcal{F}$*, server index* $j \in [s]$*, and server* j*'s input shares* $(y_{1,j}, y_{2,j}, \ldots, y_{m,j})$*,* Eval *outputs* $z_j \in \mathcal{Z}^{n_j}$*, for some* $n_j \in \mathbb{Z}$*. We refer to the* z_j *as* output shares*.*
- $\mathsf{Rec}(z_1, \ldots, z_s)$*: Given output shares* $z_1, \ldots, z_s$*,* Rec *computes* $f(x_1, \ldots, x_m) \in \mathcal{O}$*.*

We say that $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ *is a* s*-server HSS scheme for* $\mathcal{F}$ *if the following requirements hold:*

- ***Correctness:*** *For any* m *inputs* $x_1, \ldots, x_m \in \mathcal{X}$ *and* $f \in \mathcal{F}$*,*

$$\Pr_{\mathbf{r} \in \mathcal{R}^m}\left[\mathsf{Rec}(z_1, \ldots, z_s) = f(x_1, \ldots, x_m) : \begin{array}{l} \forall i \in [m],\ (y_{i,1}, \ldots, y_{i,s}) \leftarrow \mathsf{Share}(x_i, r_i) \\ \forall j \in [s],\ z_j \leftarrow \mathsf{Eval}\,(f, j, (y_{1,j}, \ldots, y_{m,j})) \end{array}\right] = 1$$

  Note that the random seeds $r_1, \ldots, r_m$ *are independent.*
- ***Security:*** *Fix* $i \in [m]$*; we say that* π *is* t-private *if for every* $T \subseteq [s]$ *with* $|T| \leq t$ *and* $x_i, x_i' \in \mathcal{X}$*,* $\mathsf{Share}(x_i)|_T$ *has the same distribution as* $\mathsf{Share}(x_i')|_T$*, over the randomness* $\mathbf{r} \in \mathcal{R}^m$ *used in* Share*.*

▶ Remark 9. We remark that in the definition of HSS, the reconstruction algorithm Rec does *not* need to know the identity of the function f being computed, while the Eval function does. In some contexts it makes sense to consider an HSS scheme for $\mathcal{F} = \{f\}$, in which case f is fixed and known to all. Our results in this work apply for general collections $\mathcal{F}$ of low-degree, multivariate polynomials, and in particular cover both situations.

We will focus on *linear* HSS schemes, which means that both Share and Rec are $\mathbb{F}$-linear over some finite field $\mathbb{F}$; we never require Eval to be linear. More precisely, we have the following definition.

▶ **Definition 10** (Linear HSS)**.** *Let* $\mathbb{F}$ *be a finite field.*

- *We say that an* s*-server HSS* $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ *has* linear reconstruction *if:*
  - $\mathcal{Z} = \mathbb{F}$*, so each output share* $z_i \in \mathbb{F}^{n_i}$ *is a vector over* $\mathbb{F}$*;*
  - $\mathcal{O} = \mathbb{F}^o$ *is a vector space over* $\mathbb{F}$*; and*
  - $\mathsf{Rec} : \mathbb{F}^{\sum_i n_i} \to \mathbb{F}^o$ *is* $\mathbb{F}$*-linear.*
- *We say that* π *has* linear sharing *if* $\mathcal{X}$*,* $\mathcal{R}$*, and* $\mathcal{Y}$ *are all* $\mathbb{F}$*-vector spaces, and* Share *is* $\mathbb{F}$*-linear.*
- *We say that* π *is* linear *if it has both linear reconstruction and linear sharing. Note there is no requirement for* Eval *to be* $\mathbb{F}$*-linear.*

Our main focus will be on the *download rate* of linear HSS schemes.

▶ **Definition 11** (Download cost, dowload rate)**.** *Let* s, t *be integers and let* $\mathcal{F}$ *be a class of functions with input space* $\mathcal{X}^m$ *and output space* $\mathcal{O}$*. Let* π *be a* s*-server* t*-private HSS for* $\mathcal{F}$*. Let* $z_i \in \mathcal{Z}^{n_i}$ *for* $i \in [s]$ *denote the output shares.*

- *The* download cost *of* π *is given by* $\mathsf{DownloadCost}(\pi) := \sum_{i \in [s]} \|z_i\|$*, where we recall that* $\|z_i\| = n_i \log_2 |\mathcal{Z}|$ *denotes the number of bits used to represent* z_i*.*
- *The* download rate *of* π *is given by*

$$\mathsf{DownloadRate}(\pi) := \frac{\log_2 |\mathcal{O}|}{\mathsf{DownloadCost}(\pi)}.$$

Thus, the download rate is a number between 0 and 1, and we would like it to be as close to 1 as possible.

2.2 Polynomial Function Classes

Throughout, we will be interested in classes of functions $\mathcal{F}$ comprised of low-degree polynomials.

▶ **Definition 12.** *Let $m > 0$ be an integer and $\mathbb{F}$ be a finite field. We define*

$$\mathrm{POLY}_{d,m}(\mathbb{F}) := \{f \in \mathbb{F}[X_1, \ldots, X_m] : \deg(f) \leq d\}$$

to be the class of all m-variate polynomials of degree at most d, with coefficients in $\mathbb{F}$.

We are primarily interested in *amortizing* HSS computations over ℓ instances of $\mathrm{POLY}_{d,m}(\mathbb{F})$. In this case, we will take our function class to be (a subset of) $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$ for some $\ell \in \mathbb{Z}^+$.

▶ Remark 13 (Amortization over the computation of ℓ polynomials). If $\mathcal{F}$ is (a subset of) $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$, then the definition of HSS can be interpreted as follows:

- There are $\ell \cdot m$ input secrets, $x_k^{(i)}$ for $i \in [\ell]$ and $k \in [m]$, and each are shared independently among the s servers. (That is, in Definition 8, we take $m \leftarrow \ell \cdot m$).
- There are ℓ target functions $f_1, \ldots, f_\ell \in \mathrm{POLY}_{d,m}(\mathbb{F})$, and the goal is to compute $f_i(x_1^{(i)}, \ldots, x_m^{(i)})$ for each $i \in [\ell]$. (That is, in Definition 8, $\mathbf{f} = (f_1, \ldots, f_\ell)$ is an element of $\mathcal{F} = \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$).
- Each server $j \in [s]$ sends a function z_j of all of their output shares, which importantly can combine information from across the ℓ instances; then the output client reconstructs $f_i(x_1^{(i)}, \ldots, x_m^{(i)})$ for each $i \in [\ell]$ from these shares.

In particular, we remark that this notion of amortization is interesting even for $d = m = 1$, when $f_1 = f_2 = \cdots = f_\ell$ is identity function. In [22], that problem was called *HSS for concatenation.*

As hinted at above, our infeasibility results hold not just for $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$ but also for any $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})$ that contains monomials with at least d different variables.

▶ **Definition 14.** *Let $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$. We say that $\mathcal{F}$ is* non-trivial *if there exists some $\mathbf{f} = (f_1, \ldots, f_\ell) \in \mathcal{F}$ so that for all $i \in [\ell]$, f_i contains a monomial with at least d distinct variables.*

(We note that our *feasibility* results are stated in terms of $\mathcal{F} = \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$; trivially these extend to any subset $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$).

As mentioned in the introduction (Theorem 1), the work [22] showed that any linear HSS scheme for $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$ (for any ℓ) can have download rate at most $(s - dt)/s$: We recall the following theorem from [22].

▶ **Theorem 15** ([22]). *Let t, s, d, m, ℓ be positive integers so that $m \geq d$. Let $\mathbb{F}$ be any finite field and π be a t-private s-server linear HSS scheme for $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$. Then $dt < s$, and $\mathsf{DownloadRate}(\pi) \leq (s - dt)/s$.*

2.3 CNF Sharing

The main Share function that we consider in this work is *CNF sharing* [24].

▶ **Definition 16** (t-private CNF sharing). *Let $\mathbb{F}$ be a finite field. The t-private, s-server CNF secret-sharing scheme over $\mathbb{F}$ is a function* $\mathsf{Share} : \mathbb{F} \times \mathbb{F}^{\binom{s}{t}-1} \to \left(\mathbb{F}^{\binom{s-1}{t}}\right)^s$ *that shares a secret $x \in \mathbb{F}$ as s shares $y_j \in \mathbb{F}^{\binom{s-1}{t}}$, using $\binom{s}{t} - 1$ random field elements, as follows.*

Let $x \in \mathbb{F}$, and let $\mathbf{r} \in \mathbb{F}^{\binom{s}{t}-1}$ be a uniformly random vector. Using $\mathbf{r}$, choose $y_T \in \mathbb{F}$ for each set $T \subseteq [s]$ of size t, as follows: The y_T are uniformly random subject to the equation $x = \sum_{T \subseteq [s]:|T|=t} y_T$. Then for all $j \in [s]$, define $\mathsf{Share}(x, \mathbf{r})_j = (y_T : j \notin T) \in \mathbb{F}^{\binom{s-1}{t}}$.

We observe that CNF-sharing is indeed t-private. Any $t+1$ servers between them hold all of the shares y_T, and thus can reconstruct $x = \sum_T y_T$. In contrast, any t of the servers (say given by some set $S \subseteq [s]$) are missing the share y_S, and thus cannot learn anything about x.

The main reason we focus on CNF sharing is that it is *universal* for linear secret sharing schemes:

▶ **Theorem 17** ([17]). *Suppose that $x \in \mathbb{F}$ is t-CNF-shared among s servers, so that server j holds $y_j \in \mathbb{F}^{\binom{s-1}{t}}$, and let Share' be any other linear secret-sharing scheme for s servers that is (at least) t-private. Then the shares y_j are locally convertible into shares of Share'. That, is there are functions $\phi_1, \dots, \phi_s$ so that $(\phi_1(y_1), \dots, \phi_s(y_s))$ has the same distribution as $\mathsf{Share}'(x, \mathbf{r})$ for a uniformly random vector $\mathbf{r}$.*

In particular, we prove several results of the form "no linear HSS with CNF sharing can do better than _____." Because of Theorem 17, these results imply that "no linear HSS with *any* linear sharing scheme can do better than _____."

Finally, we record a useful lemma about HSS with t-CNF sharing, which says that in order to recover a degree-d monomial, the output client must contact at least $dt+1$ servers. This lemma is implicit in Lemma 2 of [22]. For completeness, we provide a proof in the full version [6].

▶ **Lemma 18** ([22]). *Fix s, d, t so that $s \geq dt+1$, and suppose $m \geq d$. Let $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ be any linear HSS that t-CNF shares m secrets $x_1, \dots, x_m \in \mathbb{F}$ among s servers, for a nontrivial function class $\mathcal{F} \subset \mathrm{POLY}_{d,m}(\mathbb{F})$. Then Rec must depend on output shares z_j from at least $dt+1$ distinct servers $j \in [s]$.*

2.4 Linear Codes and Labelweights

Throughout, we will be working with *linear codes* $\mathcal{C} \subset \mathbb{F}^n$. Such a code is just a subspace of $\mathbb{F}^n$. For a linear code $\mathcal{C} \subseteq \mathbb{F}^n$ of dimension ℓ, we say that a matrix $G \in \mathbb{F}^{\ell \times n}$ is a *generator matrix* for $\mathcal{C}$ if $\mathcal{C} = \mathrm{rowSpan}(G)$. Note that generator matrices are not unique. If G has the form $G = [I|A]$ where $I = \mathbb{F}^{\ell \times \ell}$ is the identity matrix and $A \in \mathbb{F}^{\ell \times (n-\ell)}$, we say that G is in *systematic form*, and we refer to the first ℓ coordinates as *systematic coordinates*. The other coordinates we refer to as *non-systematic coordinates*. The *rate* of a linear code $\mathcal{C} \subset \mathbb{F}^n$ of dimension ℓ is defined as $\mathsf{Rate}(\mathcal{C}) := \frac{\ell}{n}$.

As discussed in the Introduction, our characterization of download-optimal linear HSS schemes is in terms of linear codes with good *labelweight*.

▶ **Definition 19** (Labeling Function). *Let U, V be finite domains satisfying $|U| \geq |V|$. We say that a mapping $\mathcal{L} : U \to V$ is a labeling function (or simply labeling) of U by V if $\mathcal{L}$ is a surjection.*

We will usually take $U = [n]$ and $V = [s]$ for $n, s \in \mathbb{Z}^+$ with $n \geq s$. For a code $\mathcal{C} \subseteq \mathbb{F}^n$, and a labeling function $\mathcal{L} : [n] \to [s]$, the labelweight of a codeword is the number of distinct labels that its support touches, and the labelweight of a code is the minimum labelweight of any nonzero codeword. More precisely, we have the following definition.

▶ **Definition 20** (Labelweight). *Let $\mathcal{L}$ be a labeling of $[n]$ by $[s]$ where $n, s \in \mathbb{Z}^+$ with $n \geq s$. Let $\mathbb{F}$ denote a field. Given $\mathbf{c} \in \mathbb{F}^n$, we define the labelweight of $\mathbf{c}$ by*

$$\Delta_{\mathcal{L}}(\mathbf{c}) := |\{\mathcal{L}(i) \ : \ \mathbf{c}_i \neq 0\}| .$$

For a code $\mathcal{C} \subseteq \mathbb{F}^n$, we define labelweight of $\mathcal{C}$ by

$$\Delta_{\mathcal{L}}(\mathcal{C}) := \min_{\mathbf{0} \neq \mathbf{c} \in \mathcal{C}} \Delta_{\mathcal{L}}(\mathbf{c}).$$

We note that labelweight is a generalization of Hamming weight; indeed, let $\iota : [n] \to [n]$ denote the identity function on $[n]$, which may be viewed as a labeling of $[n]$ by $[n]$. Given a linear code $\mathcal{C}$ of length n and $\mathbf{c} \in \mathcal{C}$, we see that $\Delta_\iota(\mathbf{c})$ and $\Delta_\iota(\mathcal{C})$ are equivalent to the Hamming weight of $\mathbf{c}$ and the minimum Hamming distance of $\mathcal{C}$, respectively.

3 Linear HSS Schemes and Good Labelweight Codes

In this section we show that finding optimal-download linear HSS schemes for low-degree multivariate polynomials is equivalent to finding linear codes with high labelweight. Before we show the equivalence, we make a few observations about linear reconstruction algorithms. The first is just the observation that any linear reconstruction scheme can be regarded as matrix:

▶ **Observation 21.** *Let ℓ, t, s, d, m, n be integers. Let $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ be a t-private, s-server HSS for some function class $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$ with linear reconstruction $\mathsf{Rec} : \mathbb{F}^n \to \mathbb{F}^\ell$, where $n = \sum_{j \in [s]} n_j$, and the output share z_j of server j is an element of $\mathbb{F}^{n_j}$. Let $\mathbf{z} \in \mathbb{F}^n$ be the vector of all output shares. That is, $\mathbf{z} = z_1 \circ z_2 \circ \cdots \circ z_s$, where $\circ$ denotes concatenation.*

Then there exists a matrix $G_\pi \in \mathbb{F}^{\ell \times n}$ so that, for all $f \in \mathcal{F}$ and for all secrets $\mathbf{x} \in (\mathbb{F}^m)^\ell$,

$$\mathsf{Rec}(\mathbf{z}) = G_\pi \mathbf{z} = f(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}^{(1)}) \\ f_2(\mathbf{x}^{(2)}) \\ \vdots \\ f_\ell(\mathbf{x}^{(\ell)}) \end{bmatrix}$$

For a linear HSS π, we call G_π as in the observation above the *reconstruction matrix* corresponding to Rec. We next observe that G_π has full rank.

▶ **Lemma 22.** *Let t, s, d, m, ℓ be positive integers so that $m \geq d$ and $n \geq \ell$, and let π be a t-private s-server linear HSS for some $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})$, so that $\mathcal{F}$ contains an element $(f_1, \ldots, f_\ell)$ where each f_i, $i \in [\ell]$ is non-constant. Then $G_\pi \in \mathbb{F}^{\ell \times n}$ has rank ℓ.*

Proof. Suppose that G_π does not have rank ℓ. Then there is a left kernel vector $\mathbf{v}$, so that $\mathbf{v}^T G_\pi = 0$. Let $(f_1, \ldots, f_\ell) \in \mathcal{F}$ be as in the lemma statement. By Observation 21, we then have

$$0 = \mathbf{v}^T G_\pi \mathbf{z} = \mathbf{v}^T \begin{bmatrix} f_1(\mathbf{x}^{(1)}) \\ \vdots \\ f_\ell(\mathbf{x}^{(\ell)}) \end{bmatrix}$$

for all values of secrets $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(\ell)}$. In particular, $\sum_{i=1}^{\ell} v_i f_i(\mathbf{x}^{(i)})$ is identically zero as a polynomial in the variables $x_k^{(i)}$ for $k \in [m], i \in [\ell]$. However, this is a contradiction because the variables that show up in $\mathbf{x}^{(i)}$ are different for different values of $i \in [\ell]$, and thus cannot cancel with each other. ◀

3.1 Optimal-Download HSS Implies Good Labelweight Codes

We now show the forward direction of the equivalence.

▶ **Lemma 23.** *Let ℓ, t, s, d, m be integers, with $m \geq d$. Suppose there exists a t-private, s-server HSS $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ for some non-trivial (see Definition 14) $\mathcal{F} \subseteq POLY_{d,m}(\mathbb{F})^\ell$, with download rate $\mathsf{DownloadRate}(\pi) = (s - dt)/s$. Let $n = \ell/\mathsf{DownloadRate}(\pi)$. Suppose also that π is linear over $\mathbb{F}$, and the Share is t-CNF sharing. Then there exists a linear code $\mathcal{C} \subseteq \mathbb{F}^n$ with rate $\mathsf{DownloadRate}(\pi)$ and a labeling $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$.*

Proof. Let $G_\pi \in \mathbb{F}^{\ell \times n}$ be the matrix representation of the linear reconstruction algorithm Rec, as in Observation 21. Let $\mathcal{C}$ denote the linear code spanned by the rows of G_π. By Lemma 22, G_π has rank ℓ, so the rate of $\mathcal{C}$ is $\frac{\ell}{n} = \mathsf{DownloadRate}(\pi)$. It remains only to show $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$ for some labeling function $\mathcal{L}$. Recall from Observation 21 that $n = \sum_{j \in [s]} n_j$ is the total number of symbols of $\mathbb{F}$ that appear in output shares across all of the servers, and that each column of G is associated with one such symbol. Let $\mathcal{L} : [n] \to [s]$ be the labeling function so that the j'th column of G is associated with a symbol sent by the $\mathcal{L}(j)$'th server.

Let $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_\ell) \in (\mathbb{F}^m)^\ell$ be the vector of $m\ell$ secrets, each of which is independently t-CNF shared to the s servers. Let $\mathbf{f} = (f_1, \ldots, f_\ell) \in \mathcal{F}$ be a function guaranteed by the fact that $\mathcal{F}$ is nontrivial (Definition 14).

As in Observation 21, let $\mathbf{z} \in \mathbb{F}^n$ be the vector of n output shares. Notice that with the labeling function defined above, this means that the j'th coordinate of $\mathbf{z}$ is sent by server $\mathcal{L}(j)$. Let $G_{\pi,1}, \ldots, G_{\pi,\ell}$ denote the rows of G_π and observe that

$$G_\pi \mathbf{z} = \begin{bmatrix} G_{\pi,1}^T \mathbf{z} \\ G_{\pi,2}^T \mathbf{z} \\ \vdots \\ G_{\pi,\ell}^T \mathbf{z} \end{bmatrix} = \mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}^{(1)}) \\ f_2(\mathbf{x}^{(2)}) \\ \vdots \\ f_\ell(\mathbf{x}^{(\ell)}) \end{bmatrix}$$

Let $\mathbf{v} \in \mathbb{F}^\ell \setminus \{\mathbf{0}\}$, and consider the codeword $\mathbf{c} \in \mathcal{C}$ given by $\mathbf{c} = \mathbf{v}^T G_\pi$. Then from the above,

$$\mathbf{c}^T \mathbf{z} = \sum_{i=1}^{\ell} v_i f_i(\mathbf{x}^{(i)}) =: f(\mathbf{x}).$$

In particular, for $\mathcal{F}' = \{f\}$, $\mathbf{c}$ gives a linear reconstruction algorithm Rec' for a linear HSS $\pi' = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec}')$ for $\mathcal{F}'$. Notice that $\mathcal{F}'$ is also non-trivial; indeed, each $f_i(\mathbf{x}^{(i)})$ contains distinct variables (so they cannot cancel), and each f_i has a monomial with at least d distinct variables; thus f contains a monomial with at least d distinct variables.

Then by Lemma 18, Rec' must depend on output shares from at least $dt + 1$ distinct servers $j \in [s]$. It follows from the definition of $\mathcal{L}$ that $\Delta_{\mathcal{L}}(\mathbf{c}) \geq dt + 1$. Since $\mathbf{v}$, and hence $\mathbf{c}$, was arbitrary (non-zero), this implies that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$, as desired. ◀

Lemma 23 shows that the existence of an optimal-rate linear HSS scheme for polynomials implies the existence of a linear code with with optimum labelweight. The converse is also true, but before we prove that result, we need to dive a bit deeper into the structure of codes with good labelweight, which we do next.

3.2 Structural Properties of Labeling Functions for Optimal Labelweight Codes

We establish a few structural properties of the labeling functions $\mathcal{L}$ that appear in codes with optimal labelweight. Our main structural result states that for any code $\mathcal{C}$ of rate $(s-dt)/s$, if $\mathcal{L}$ is such that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt+1$, then in fact the labeling function $\mathcal{L}$ must be "balanced," in the sense that each label shows up the same number of times. Below and throughout the paper, for $\mathcal{L}:[n]\to[s]$, we use $\mathcal{L}^{-1}(\lambda)=\{j\in[n]:\mathcal{L}(j)=\lambda\}$ to denote the preimage of $\lambda\in[s]$ under $\mathcal{L}$.

▶ **Lemma 24.** *Let $\mathcal{C}\subseteq\mathbb{F}^n$ be a rate $(s-dt)/s$ linear code and $\mathcal{L}:[n]\to[s]$ a labeling of its coordinates by $[s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C})\geq dt+1$. Then for all $\lambda\neq\lambda'\in[s]$, $\left|\mathcal{L}^{-1}(\lambda)\right|=\left|\mathcal{L}^{-1}(\lambda')\right|$.*

Proof. Since $\mathcal{C}$ has rate $(s-dt)/s$, there exists some $j\in\mathbb{Q}^+$ such that $\mathcal{C}$ has block length $n=js$ and dimension $\ell=j(s-dt)$. Let $G\in\mathbb{F}^{j(s-dt)\times js}$ be an arbitrary generator matrix of $\mathcal{C}$. The lemma holds if and only if for each $\lambda\in[s]$, we have $|\mathcal{L}^{-1}(\lambda)|=js/s=j$. Thus, we assume towards a contradiction that this is not the case and break the proof into three cases, depending on the relationship between s and $2dt$.

Case 1: $s=2dt$. In this case, consider $\lambda_1,\ldots,\lambda_{dt}$ chosen so that $\sum_{i=1}^{dt}|\mathcal{L}^{-1}(\lambda_i)|$ is minimized. As we assume towards a contradiction that there exists $\lambda\in[s]$ such that $\left|\mathcal{L}^{-1}(\lambda)\right|\neq j$, there exists a choice of $\lambda_1,\lambda_2,\ldots,\lambda_{dt}\in[s]$ so that

$$\left|\bigcup_{i=1}^{dt}\mathcal{L}^{-1}(\lambda_i)\right|=jdt-\sigma<jdt$$

for some $\sigma>0$. Without loss of generality we may assume that $\lambda_1,\ldots,\lambda_{dt}$ label the final $jdt-\sigma$ columns of G. Since $s=2dt$, G has $j(s-dt)=jdt$ rows; hence there exists some nonzero $\mathbf{m}\in\mathbb{F}^{jdt}$ such that $\mathbf{m}^TG$ has no support in its final $jdt-\sigma$ coordinates. But then $\Delta_{\mathcal{L}}(\mathbf{m}^TG)\leq dt$, contradicting $\Delta_{\mathcal{L}}(\mathcal{C})\geq dt+1$.

Case 2: $s\geq 2dt+1$. In this case, consider $\lambda_1,\ldots,\lambda_{dt}$ so that $\sum_{i=1}^{dt}|\mathcal{L}^{-1}(\lambda_i)|$ is maximized. As we assume towards a contradiction that there exists $\lambda\in[s]$ such that $\left|\mathcal{L}^{-1}(\lambda)\right|\neq j$, there exists a choice of $\lambda_1,\lambda_2,\ldots,\lambda_{dt}\in[s]$ so that

$$\left|\bigcup_{i=1}^{dt}\mathcal{L}^{-1}(\lambda_i)\right|=jdt+\sigma>jdt$$

for some $\sigma>0$. Without loss of generality we may assume that $\lambda_1,\ldots,\lambda_{dt}$ label the final $jdt+\sigma$ columns of G, leaving the first $js-(jdt+\sigma)=j(s-dt)-\sigma$ columns labeled with $[s]\setminus\{\lambda_1,\ldots,\lambda_{dt}\}$. Since G has precisely $j(s-dt)$ rows, there exists some $\mathbf{m}\in\mathbb{F}^{jdt}$ such that $\mathbf{m}^TG$ has no support in its first $j(s-dt)-\sigma$ coordinates. It follows that $\Delta_{\mathcal{L}}(\mathbf{m}^TG)\leq dt$, contradicting $\Delta_{\mathcal{L}}(\mathcal{C})\geq dt+1$.

Case 3: $s\leq 2dt-1$. Note that $s\geq dt+1$, as we assume that $\Delta_{\mathcal{L}}(\mathcal{C})\geq dt+1$. Thus, we may write $s=dt+\hat{s}$, where $\hat{s}\in[dt-1]$. In this case, consider $\lambda_1,\ldots,\lambda_{\hat{s}}$ so that $\sum_{i=1}^{\hat{s}}|\mathcal{L}^{-1}(\lambda_i)|$ is minimized. As we assume towards a contradiction that there exists $\lambda\in[s]$ such that $\left|\mathcal{L}^{-1}(\lambda)\right|\neq j$, there exists a choice of $\lambda_1,\lambda_2,\ldots,\lambda_{\hat{s}}\in[s]$ so that

$$\left|\bigcup_{i=1}^{\hat{s}}\mathcal{L}^{-1}(\lambda_i)\right|=j\hat{s}-\sigma<j\hat{s}$$

for some $\sigma > 0$. Without loss of generality we may assume that $\lambda_1, \ldots, \lambda_{\hat{s}}$ label the first $j\hat{s} - \sigma$ columns of G. Since there are $j\hat{s}$ rows of G, there exists some $\mathbf{m} \in \mathbb{F}^{j\hat{s}}$ such that $\mathbf{m}^T G$ has no support in its first $j\hat{s} - \sigma$ coordinates. It follows that $\Delta_{\mathcal{L}}(\mathbf{m}^T G) \leq dt$, contradicting $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. ◀

We next state a few corollaries; we refer to the full version [6] for the proofs.

▶ **Corollary 25.** *Let $s, d, t \in \mathbb{Z}^+$ satisfying $s - dt > 0$. Let $\mathcal{C}$ be a rate $(s - dt)/s$ linear code and $\mathcal{L}$ a labeling of its coordinates by $[s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. Then there exists some $j \in \mathbb{Z}^+$ such that:*

- **(i)** *$\mathcal{C}$ has length $n = js$ and dimension $\ell = j(s - dt)$;*
- **(ii)** *$|\mathcal{L}^{-1}(\lambda)| = j$ for all $\lambda \in [s]$;*
- **(iii)** *there is a re-ordering of the coordinates of $\mathcal{C}$ so that $\mathcal{L} : [js] \to [s]$ is given by $\mathcal{L} : x \mapsto \lceil x/j \rceil$.*

▶ **Corollary 26.** *Let j be a positive integer, and let $\mathcal{C} \subseteq \mathbb{F}_q^{js}$ be a linear code of length js and dimension $j(s - dt)$. Let $\mathcal{L} : [js] \to [s]$ be a labeling such that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. Let $G \in \mathbb{F}_q^{j(s-dt) \times js}$ be an arbitrary generator matrix for $\mathcal{C}$. Given $\Lambda \subseteq [s]$, let $G(\Lambda)$ be the submatrix of G consisting of the columns G_i of G for all $i \in [js]$ so that $\mathcal{L}(i) \in \Lambda$. Then for any $\Lambda \subseteq [s]$ with $|\Lambda| = s - dt$, $G(\Lambda) \in \mathbb{F}^{j(s-dt) \times j(s-dt)}$ and $\det(G(\Lambda)) \neq 0$.*

3.3 Good Labelweight Codes Imply Optimal-Download HSS

Now, we can prove a converse to Lemma 23.

▶ **Theorem 27.** *Let ℓ, t, s, d, m, n be integers, with $m \geq d$ and $\ell s = n(s - dt)$. Suppose that there exists a linear code $\mathcal{C} \subseteq \mathbb{F}^n$ and a labeling $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. Then there exists a t-private, s-server linear HSS $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ for $\mathrm{POLY}_{d,m}(\mathbb{F})^\ell$ with download rate $\mathsf{DownloadRate}(\pi) \geq (s - dt)/s$.*

Proof. The proof is by construction; let $\mathcal{C}$ be as in the theorem statement, and let $G \in \mathbb{F}^{\ell \times n}$ be any generator matrix for $\mathcal{C}$. In order to define π, we need to define the functions Share, Eval, and Rec. For Share, we will use t-CNF sharing (Definition 16). We define Rec using the generator matrix G. In particular, we will (soon) define Eval so that server $j \in [s]$ returns $n_j := |\mathcal{L}^{-1}(j)|$ elements of $\mathbb{F}$ as output shares. We will gather these output shares into a vector $\mathbf{z} \in \mathbb{F}^n$, where $n = \sum_{j \in [s]} n_j$. Then Rec will be given by $\mathsf{Rec}(\mathbf{z}) = G\mathbf{z}$.

Finally, it remains to define Eval. To do so, we will set up a linear system that essentially says "the Rec function we just defined is correct." Then we will show that this linear system has a solution, and that will yield our Eval function.

Below, we assume without loss of generality that the function $\mathbf{f} = (f_1, \ldots, f_\ell)$ that the HSS scheme is trying to compute has $f_j(x_1, \ldots, x_m) = \prod_{i=1}^d x_i$ for all $j \in [\ell]$, and we will define $\mathsf{Eval}(\mathbf{f}, j, \mathbf{y}_j)$ on only this $\mathbf{f}$. To obtain the general result for any polynomials $(p_1, p_2, \ldots, p_\ell) \in \mathrm{POLY}_{d,m}(\mathbb{F})^\ell$, we first observe that the argument goes through if the f_j's are *any* monomials of degree at most d, possibly with a leading coefficient; this follows by re-ordering the secrets, and possibly including some dummy secrets that are identically equal to a constant (that is, increasing the parameter m, which does not appear in any of the results) to obtain monomials of degree less than d and/or with a leading coefficient. Then to pass to general polynomials, and not just monomials, we observe that since Rec is linear, we may define Eval additively; that is, if $p_i(\mathbf{x}^{(i)}) = \sum_r f_{i,r}(\mathbf{x}^{(i)})$ for some monomials $f_{i,r}$, server j will compute Eval on each $\mathbf{f}_r = (f_{1,r}, \ldots, f_{\ell,r})$, and then return their sum.

Now we return to the task of setting up a linear system to define Eval for the particular function $\mathbf{f}$ defined above. In order to set up this linear system, we introduce some notation. Let $\mathcal{T} = \{T \subseteq [s] : |T| = t\}$ be the set of size-t subsets of $[s]$. Let $\mathbf{x} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(\ell)}) \in (\mathbb{F}^m)^\ell$ denote the secrets to be shared. For $T \in \mathcal{T}$, $r \in [\ell]$, and $j \in [m]$, let $y^{(r)}_{j,T}$ denote the CNF shares of $x^{(r)}_j$, so $x^{(i)}_j = \sum_T y^{(i)}_{j,T}$. Thus, for each $i \in [\ell]$ the function we would like to recover is

$$f_i(\mathbf{x}^{(i)}) = \sum_{\mathbf{T} \in \mathcal{T}^d} \prod_{k=1}^{d} y^{(i)}_{k,T_k}. \tag{1}$$

Let $\mathbf{y}_j$ denote the set of CNF shares that server j holds: $\mathbf{y}_j = (y^{(i)}_{k,T} : k \in [d], T \in \mathcal{T}, i \in [\ell], j \notin T)$. We will treat $\mathbf{y}_j$ as tuples of formal variables.

Next, we define the following classes of monomials, in the variables $y^{(i)}_{j,T}$. Let

$$\mathcal{M} = \left\{ y^{(i)}_{1,T_1} y^{(i)}_{2,T_2} \cdots y^{(i)}_{d,T_d} \ : \ \mathbf{T} \in \mathcal{T}^d, i \in [\ell] \right\}.$$

Given a server $j \in [s]$, let

$$\mathcal{M}_j = \left\{ y^{(i)}_{1,T_1} y^{(i)}_{2,T_2} \cdots y^{(i)}_{d,T_d} \in \mathcal{M} \ : \ \mathbf{T} \in \mathcal{T}^d, i \in [\ell], j \notin \bigcup_{k \in [d]} T_k \right\}.$$

That is, $\mathcal{M}_j$ is the subset of $\mathcal{M}$ locally computable by server j.

The function $\mathsf{Eval}(\mathbf{f}, j, \mathbf{y}_j)$ that determines server j's output shares will be defined by a sequence of $n_j = |\mathcal{L}^{-1}(j)|$ polynomials of degree d, constructed from the monomials in $\mathcal{M}_j$. To that end, we will define a vector of variables $\mathbf{e} \in \mathbb{F}^{\sum_{r \in [n]} |\mathcal{M}_{\mathcal{L}(r)}|}$, indexed by pairs (r, χ) for $\chi \in \mathcal{M}_{\mathcal{L}(r)}$. The vector $\mathbf{e}$ will encode the function Eval as follows. For each $r \in [n]$, we define $z_r = z_r(\mathbf{y}_{\mathcal{L}(r)})$ to be the polynomial in the variables $\mathbf{y}_{\mathcal{L}(r)}$ given by

$$z_r(\mathbf{y}_{\mathcal{L}(r)}) := \sum_{\chi \in \mathcal{M}_{\mathcal{L}(r)}} \mathbf{e}_{r,\chi} \cdot \chi(\mathbf{y}_{\mathcal{L}(r)}). \tag{2}$$

Then we define Eval by

$$\mathsf{Eval}(\mathbf{f}, j, \mathbf{y}_j) = (z_r(\mathbf{y}_j) : r \in \mathcal{L}^{-1}(j)) \in \mathbb{F}^{n_j}$$

for each server $j \in [s]$.

Now we will set up our system to solve for the coefficients in $\mathbf{e}$, which will define Eval as above. Define the matrix $S \in \mathbb{F}^{\ell|\mathcal{M}| \times \sum_{r \in [n]} |\mathcal{M}_{\mathcal{L}(r)}|}$ as follows.

- The rows of S are indexed by pairs $(i, \mathfrak{m}) \in [\ell] \times \mathcal{M}$.
- The columns of S are indexed by pairs (r, χ) for $r \in [s]$ and $\chi \in \mathcal{M}_r$.
- The entry of S indexed by $(i, \mathfrak{m})$ and (r, χ) is given by:

$$S[(i, \mathfrak{m}), (r, \chi)] = \begin{cases} G[i, r] & \mathfrak{m} = \chi \\ 0 & \text{else} \end{cases}.$$

Define a vector $\mathbf{g} \in \mathbb{F}^{\ell|\mathcal{M}|}$ so that the coordinates of $\mathbf{g}$ are indexed by pairs $(i, \mathfrak{m}) \in [\ell] \times \mathcal{M}$, so that

$$\mathbf{g}[(i, \mathfrak{m})] = \begin{cases} 1 & \psi_i(\mathfrak{m}) \\ 0 & \text{else} \end{cases}$$

where

$$\psi_i(\mathfrak{m}) = \begin{cases} 1 & \mathfrak{m} \text{ is of the form } \prod_{k=1}^{d} y_{k,T_k}^{(i)} \text{ for some } \mathbf{T} \in \mathcal{T}^d \\ 0 & \text{else} \end{cases}$$

Notice that (1) implies that for $i \in [\ell]$,

$$f_i(\mathbf{x}^{(i)}) = \sum_{\mathbf{T} \in \mathcal{T}^d} \prod_{k=1}^{d} y_{k,T_k}^{(i)} = \sum_{\mathfrak{m} \in \mathcal{M}} \psi_i(\mathfrak{m}) \cdot \mathfrak{m}(\mathbf{y}). \tag{3}$$

In the full version [6], we show formally that the HSS that we have just constructed is correct. That is, we prove the following claim.

▷ Claim 28. Suppose that $S \cdot \mathbf{e} = \mathbf{g}$. Then the HSS scheme $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$, where Share and Rec are as above, and Eval is defined by $\mathbf{e}$ as above, is correct. (That is, it satisfies the *Correctness* property in Definition 8).

Given Claim 28, we now only need to show that we can find a vector $\mathbf{e}$ so that $S \cdot \mathbf{e} = \mathbf{g}$. To do this, we show in the full version [6] that the matrix S has full row rank, and in particular we can solve the above affine system. Formally, we have the following claim.

▷ Claim 29. Let S be as above. Then S has full row rank.

Claim 28 says that any $\mathbf{e}$ so that $S \cdot \mathbf{e} = \mathbf{g}$ corresponds to a correct Eval function (with Share and Rec as given above), and Claim 29 implies that we can find such an $\mathbf{e}$ efficiently. Thus we can efficiently find a description of $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$. It remains to verify that $\mathsf{DownloadRate}(\pi) \geq (s - dt)/s$, which follows from construction, as the download rate of π is equal to the rate of $\mathcal{C}$, which is by definition $(s - dt)/s$. ◀

4 Linear Codes with Good Labelweight

Now that we know that download-optimal linear HSS schemes are equivalent to linear codes with good labelweight, we focus on constructions and limitations of such codes. In this section we give a construction, and a nearly-matching infeasibility result. We begin by stating the constructive result, which as noted in the Introduction is a very slight improvement over the construction in [22].

▶ **Theorem 30** (Construction of linear codes with good labelweight). *Let $\mathbb{F}$ be a finite field of size q. Let $s, d, t \in \mathbb{Z}^+$ such that $s - dt > 0$. For all integers*

$$j \geq \begin{cases} \log_q(s-1) & q^j \text{ odd, or } (s-dt) \notin \{3, q^j - 1\} \\ \log_q(s-2) & q^j \text{ even, and } (s-dt) \in \{3, q^j - 1\} \end{cases},$$

there is an explicit construction of a linear code $\mathcal{C} \subset \mathbb{F}^{js}$ of dimension $j(s-dt)$, and a labeling $\mathcal{L} : [js] \to [s]$ such that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$.

The following theorem says that Theorem 30 is basically optimal, in that we cannot take j to be substantially smaller:

▶ **Theorem 31** (Limitations on linear codes with good labelweight). *Let $\mathbb{F}$ be a finite field of size q. Let $s, d, t \in \mathbb{Z}^+$ such that $s - dt > 0$. Suppose that there is a code $\mathcal{C} \subseteq \mathbb{F}^n$ with dimension ℓ and rate $(s - dt)/s$; and a labeling function $\mathcal{L} : [n] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$. Suppose that j is such that $\ell = j(s - dt)$ and $n = js$. Then j is an integer, and*

$$j \geq \lceil \max \left\{ \log_q(s - dt + 1), \log_q(dt + 1) \right\} \rceil.$$

▶ Remark 32 (Gap between upper and lower bounds on j.). As discussed in the Introduction, Theorem 31 and Theorem 30 do not quite match. However, because j must be an integer, in fact the bounds are exactly the same for many parameter settings, especially when q is large.

We also remark that, when Theorems 31 and 30 disagree, we conjecture that the *construction* (Theorem 30) is the correct one. Indeed, our construction in Theorem 30 follows from "puffing up" a totally nonsingular (TN) matrix; the value of the parameter j has to do with the field size over which this TN matrix is defined. In fact, assuming the MDS conjecture for extension fields, the field size that we use in our construction – and hence the value of j – is the best possible; see [6] for details. Thus, if Theorem 30 is not optimal, either the MDS conjecture is false over extension fields, or else there is an alternate way to construct Block TN matrices without going through TN matrices over larger fields.

Given Lemma 23 and Theorem 27, Theorems 30 and 31 immediately imply the following corollary about linear HSS schemes.

▶ **Corollary 33** (Classification of Amortization Parameter for Linear HSS Schemes)**.** *Let s, d, t, m, ℓ be positive integers, so that $m \geq d$.*

Let $\pi = (\mathsf{Share}, \mathsf{Eval}, \mathsf{Rec})$ be a linear HSS scheme for some nontrivial (Definition 14) function class $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})$, with download rate $(s-dt)/s$. Then $\ell = j(s-dt)$ for some integer j that satisfies

$$j \geq \max\{\lceil \log_q(s-dt+1) \rceil, \lceil \log_q(dt+1) \rceil\}.$$

Conversely, let j be any integer so that j is as in Theorem 30. Then there are explicit constructions of linear HSS schemes for any $\mathcal{F} \subseteq \mathrm{POLY}_{d,m}(\mathbb{F})$ with download rate $(s-dt)/s$ and amortization parameter $\ell = j(s-dt)$.

▶ Remark 34. As in Remark 32, we observe that the bounds are quite close. Indeed, for any setting of parameters, there is at most one value of ℓ (namely, $\ell = j(s-dt)$ for one particular integer j) for which we do not know whether or not there exists a download-optimal linear HSS scheme for $\mathrm{POLY}_{d,m}(\mathbb{F})^{\ell}$. Moreover, for many parameter settings (especially when q is large), the bounds exactly match.

We prove Theorems 30 and 31 in the full version of the paper [6] However, in this extended abstract, we do state Lemma 36, which is the technical meat of the proofs. This lemma characterizes codes of high labelweight in terms of *block totally nonsingular matrices*, which we informally defined in the Introduction and which we formally define in Definition 35 below. Theorems 30 and 31 then follow from an analysis of block totally nonsingular matrices.

4.1 Block Totally-Nonsingular Matrices

We say that a matrix A is *totally nonsingular* if any square sub-matrix of A (not necessarily contiguous) is nonsingular [23]. We extend the definition of total nonsingularity to block matrices as follows.

▶ **Definition 35** (Block Totally-Nonsingular Matrices)**.** *Let $\mathbb{F}$ be a finite field, and fix integers j, r, u. Let $\mathbf{A} = [A_{i,k} : i \in [r], k \in [u]] \in (GL(\mathbb{F}, j))^{r \times u}$ be a $r \times u$ block array of invertible $j \times j$ matrices $A_{i,k}$. We say that $\mathbf{A}$ is* Block Totally Nonsingular *(Block TN) if every square sub-array of block matrices is full-rank. More precisely, for all $S \subseteq [r]$ and $S' \subseteq [u]$ of the same size, the matrix $\mathbf{A}' = [A_{i,i'} \in \mathbf{A} : i \in S, i' \in S'] \in \mathbb{F}^{j|S| \times j|S'|}$ is nonsingular.*

The main technical lemma at the heart of the proofs of Theorems 30 and 31 says that constructing codes with good labelweight is equivalent to constructing Block TN matrices:

▶ **Lemma 36.** *Let j, s, d, t be positive integers so that $s > dt$, and let q be a prime power. Then the following are equivalent.*

- **(i)** *There is a linear code $\mathcal{C} \subseteq \mathbb{F}_q^{js}$ with block length js and dimension $j(s-dt)$ and a labeling $\mathcal{L} : [js] \to [s]$ so that $\Delta_{\mathcal{L}}(\mathcal{C}) \geq dt + 1$.*
- **(ii)** *There exists a Block TN matrix $\mathbf{A} \in GL(\mathbb{F}_q, j)^{(s-dt)\times dt}$.*

Moreover, the equivalence is constructive: if $\mathbf{A}$ is a Block TN matrix as in (ii), then the code $\mathcal{C}$ in (i) is generated by the matrix $[I|\mathbf{A}]$. If $\mathcal{C}$ is a code as in (i), then it has a generator matrix of the form $[I|\mathbf{A}]$, where $\mathbf{A}$ is the Block TN matrix as in (ii).

Given Lemma 36, proving Theorems 30 and 31 amounts to giving constructions of and limitations on Block-TN matrices. Due to space limitations, we omit these analyses from this extended abstract, and give the details in the full version [6].

References

1. Simeon Ball and Jan De Beule. On sets of vectors of a finite vector space in which every subset of basis size is a basis ii. *Designs, Codes and Cryptography*, 65:5–14, 2012. URL: `https://api.semanticscholar.org/CorpusID:121889797`.
2. Donald Beaver and Joan Feigenbaum. Hiding instances in multioracle queries. In *STACS 90*, pages 37–48, 1990.
3. Donald Beaver, Joan Feigenbaum, Joe Kilian, and Phillip Rogaway. Security with low communication overhead. In *CRYPTO '90*, pages 62–76, 1990.
4. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, 1988.
5. Josh Cohen Benaloh. Secret sharing homomorphisms: Keeping shares of A secret sharing. In Andrew M. Odlyzko, editor, *CRYPTO '86*, pages 251–260, 1986.
6. Keller Blackwell and Mary Wootters. A characterization of optimal-rate linear homomorphic secret sharing schemes, and applications. *arXiv preprint*, 2023. `arXiv:2311.14842`.
7. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *CRYPTO*, pages 489–518, 2019.
8. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, and Michele Orrù. Homomorphic secret sharing: optimizations and applications. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2105–2122, 2017.
9. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *EUROCRYPT 2015, Part II*, pages 337–367, 2015.
10. Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 509–539. Springer, 2016. `doi:10.1007/978-3-662-53018-4_19`.
11. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 1292–1303. ACM, 2016. `doi:10.1145/2976749.2978429`.
12. Elette Boyle, Niv Gilboa, Yuval Ishai, Huijia Lin, and Stefano Tessaro. Foundations of homomorphic secret sharing. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 21:1–21:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. `doi:10.4230/LIPIcs.ITCS.2018.21`.

13 Elette Boyle, Lisa Kohl, and Peter Scholl. Homomorphic secret sharing from lattices without FHE. In *EUROCRYPT 2019, Part II*, pages 3–33, 2019.

14 David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, 1988.

15 Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. *J. ACM*, 1998.

16 Geoffroy Couteau and Pierre Meyer. Breaking the circuit size barrier for secure computation under quasi-polynomial LPN. In *EUROCRYPT 2021, Part II*, pages 842–870, 2021.

17 Ronald Cramer, Ivan Damgård, and Yuval Ishai. Share conversion, pseudorandom secret-sharing and applications to secure computation. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 342–362. Springer, 2005. doi:10.1007/978-3-540-30576-7_19.

18 Ronald Cramer, Ivan Damgård, and Ueli M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *EUROCRYPT*, 2000.

19 Quang Dao, Yuval Ishai, Aayush Jain, and Huijia Lin. Multi-party homomorphic secret sharing and sublinear mpc from sparse lpn. In *Annual International Cryptology Conference*, pages 315–348. Springer, 2023.

20 Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 93–122. Springer, 2016. doi:10.1007/978-3-662-53015-3_4.

21 Nelly Fazio, Rosario Gennaro, Tahereh Jafarikhah, and William E. Skeith III. Homomorphic secret sharing from Paillier encryption. In *Provable Security*, 2017.

22 Ingerid Fosli, Yuval Ishai, Victor I Kolobov, and Mary Wootters. On the download rate of homomorphic secret sharing. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

23 F.R. Gantmacher. *The Theory of Matrices*. Chelsea Publishing Company, 1980. URL: https://books.google.com/books?id=ebMuywEACAAJ.

24 Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing scheme realizing general access structure. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 72(9):56–64, 1989.

25 Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. In *EUROCRYPT 2021, Part I*, pages 678–708, 2021.

26 Lawrence Roy and Jaspal Singh. Large message homomorphic secret sharing from DCR and applications. In *CRYPTO 2021, Part III*, pages 687–717, 2021.

27 Beniamino Segre. Curve razionali normali ek-archi negli spazi finiti. *Annali di Matematica Pura ed Applicata*, 39:357–379, 1955. URL: https://api.semanticscholar.org/CorpusID:122128482.

28 Jack Keil Wolf. Adding two information symbols to certain nonbinary bch codes and some applications. *The Bell System Technical Journal*, 48(7):2405–2424, 1969. doi:10.1002/j.1538-7305.1969.tb01179.x.