Computationally Relaxed Locally Decodable Codes, Revisited

Alexander R. Block[†] and Jeremiah Blocki[‡]
[†]Georgetown University and University of Maryland, College Park. Email: alexander.block@georgetown.edu
[‡]Purdue University. Email: jblocki@purdue.edu

***Abstract*—We revisit computationally relaxed locally decodable codes (crLDCs) (Blocki et al., Trans. Inf. Theory '21) and give two new constructions. Our first construction is a Hamming crLDC that is conceptually simpler than prior constructions, leveraging digital signature schemes and an appropriately chosen Hamming code. Our second construction is an extension of our Hamming crLDC to handle insertion-deletion (InsDel) errors, yielding an InsDel crLDC. This extension crucially relies on the noisy binary search techniques of Block et al. (FSTTCS '20) to handle InsDel errors. Both crLDC constructions have binary codeword alphabets, are resilient to a constant fraction of Hamming and InsDel errors, respectively, and under suitable parameter choices have poly-logarithmic locality and encoding length linear in the message length and polynomial in the security parameter. These parameters compare favorably to prior constructions in the poly-logarithmic locality regime.**

I. Introduction

Locally decodable codes (LDCs) are error-correcting codes that admit super-efficient (i.e., poly-logarithmic time) recovery of individual symbols of an encoded message by querying only a few locations into a received word. For an alphabet Σ and a (normalized) metric dist, a pair of algorithms $\mathsf{Enc}\colon \Sigma^k \to \Sigma^K$ and $\mathsf{Dec}\colon [k] \to \Sigma$ (for $[k] := \{1, \ldots, k\}$) is a (ℓ, ρ, p)-LDC if Dec is a randomized oracle algorithm such that for any message x and any received word y', if $\mathrm{dist}(\mathsf{Enc}(x), y') \leqslant \rho$ then for every i, $\mathsf{Dec}^{y'}(i)$ makes at most ℓ queries to y' and outputs x_i with probability at least p. Here, k and K are the *message* and *block lengths*, respectively, k/K is the *rate*, ℓ is the *locality*, ρ is the *error-rate*, and p is the *success probability*.

Studied extensively in the context of worst-case *Hamming errors* [1]–[9] where dist is the normalized Hamming distance (HAM), *Hamming* LDC*s* seem to have irreconcilable trade-offs between the rate, error-rate, and locality. For constant error-rate (the target of most applications), the best known constructions with constant $\ell \geqslant 3$ locality have super-polynomial rate [4]–[6], for $\ell = 2$ it is known that $K = \Theta(\exp(k))$ [3], and the best known constructions with constant rate have super-logarithmic (sub-polynomial) locality [9]. Furthermore, the best known lower bounds for general Hamming LDCs with constant error-rate and locality $\ell \geqslant 3$ are $K = \Omega(k^{\frac{\ell+1}{\ell-1}})$ [10], and any locality $\ell = 3$ linear Hamming LDC has $K = \Omega(k^2/\log(k))$ [11]. See surveys [7], [8] for more details.

To remedy these dramatic tradeoffs, Ben-Sasson et al. [12] introduced *relaxed* LDC*s* (rLDCs). Relaxed LDCs are LDCs that additionally allow the decoder to output a symbol $\perp \notin \Sigma$, which signifies that the decoder does not know the correct value, under the following restrictions: the decoder (a) does not output $\perp$ "too often"; and (b) never outputs $\perp$ when the queried codeword is uncorrupted. This relaxation yields LDCs with constant locality and block length $K = k^{1+\varepsilon}$ for (small) constant $\varepsilon > 0$. Blocki et al. consider a further relaxation of rLDCs known as *computationally relaxed* LDC*s* (crLDCs) [13]: rLDCs that are only resilient against adversarial channels that are computationally bounded (i.e., probabilistic polynomial time (PPT) channels). This relaxation, inspired by the work of Lipton [14], yields crLDCs with constant rate, constant error-rate, and polylog locality.

Recently, advances in coding theory have turned their focus to understanding and constructing codes which are resilient to *insertion-deletion errors* (InsDel errors) [15]–[33], where an adversarial channel inserts and deletes a bounded number of symbols into and from the encoded message. Known as *InsDel codes*, the metric dist considered is the (normalized) edit distance ED, defined as the minimum number of symbol insertions and deletions to transform a string u into a string v, normalized by $2\max\{|u|, |v|\}$. Only recently have efficient InsDel codes with asymptotically optimal rate and error-rate been well-understood [20], [24]–[26], [30].

The study of LDCs resilient to InsDel errors (InsDel LDCs) has been scarce, with only a handful of results to date. Introduced by Ostrovsky and Paskin-Cherniavsky [34], to the best of our knowledge, *all* InsDel LDC constructions follow [34] and utilize a so-called "Hamming-to-InsDel compiler" [35]–[37], which transforms any Hamming LDC into an InsDel LDC, increasing both the rate and error-rate by a constant factor and increasing the locality by a polylog factor. For example, any locality-3 Hamming LDC with block length K and error-rate ρ can be compiled into a locality-$3\cdot\mathrm{polylog}(K)$ InsDel LDC with block length $\Theta(K)$ and error-rate $\Theta(\rho)$.

A. Overview of Results

In this work, we revisit crLDCs with respect to both Hamming and InsDel errors. We begin by defining crLDCs.

Definition 1 (Computationally Relaxed Locally Decodable Codes)**.** *Let $\mathcal{C} = \{C_\lambda[K, k, q_1, q_2]\}_{\lambda\in\mathbb{N}}$ be a code family with encoding algorithms $\{\mathsf{Enc}_\lambda\colon \Sigma_1 \to \Sigma_2\}_{\lambda\in\mathbb{N}}$ where $|\Sigma_i| = q_i$. We say $\mathcal{C}$ is a $(\ell, \rho, p, \delta, \mathrm{dist})$-*computationally relaxed locally decodable code *(*crLDC*) if there exists a family of randomized oracle decoding algorithms $\{\mathsf{Dec}_\lambda\colon [k] \to \Sigma_1\}_{\lambda\in\mathbb{N}}$ such that:*

1) *For all $\lambda \in \mathbb{N}$ and any $\tilde{y} \in \Sigma_2^*$, $\mathsf{Dec}_\lambda^{\tilde{y}}(i)$ makes at most ℓ queries to $\tilde{y}$ for any $i \in [k]$;*

2) *For all* $\lambda \in \mathbb{N}$ *and any* $x \in \Sigma_1^k$*, we have* $\Pr[\mathsf{Dec}_\lambda^{\mathsf{Enc}_\lambda(x)}(i) = x_i] = 1$ *for all* $i \in [k]$*;*

3) *Define binary predicate* $\mathsf{Fool}(\tilde{y}, \rho, p, x, y, \lambda) = 1$ *iff (a)* $\mathrm{dist}(y, \tilde{y}) \leqslant \rho$*; and (b)* $\exists i \in [k]$ *such that* $\Pr[\mathsf{Dec}_\lambda^{\tilde{y}}(i) \in \{x_i, \perp\}] < p$*, where the probability is taken over* Dec_λ*; otherwise* $\mathsf{Fool}(\tilde{y}, \rho, p, x, y, \lambda) = 0$*. We require that for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\varepsilon_{\mathsf{F}}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *and all* $x \in \Sigma_1^k$*, we have* $\Pr[\mathsf{Fool}(\mathcal{A}(y), \rho, p, x, y, \lambda) = 1] \leqslant \varepsilon_{\mathsf{F}}(\lambda)$*, where the probability is taken over* $\mathcal{A}$ *and* $y = \mathsf{Enc}_\lambda(x)$*.*

4) *Define binary predicate* $\mathsf{Limit}(\tilde{y}, \rho, \delta, x, y, \lambda) = 1$ *iff the following hold: (a)* $\mathrm{dist}(y, \tilde{y}) \leqslant \rho$*; and (b)* $|\mathsf{Good}(\tilde{y})| < \delta k$*, where* $\mathsf{Good}(\tilde{y}) := \{i \colon \Pr[\mathsf{Dec}_\lambda^{\tilde{y}}(i) = x_i] > 2/3\}$*, where the probability is taken over* Dec_λ*; otherwise* $\mathsf{Limit}(\tilde{y}, \rho, \delta, x, y, \lambda) = 0$*. We require that for all adversaries PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\varepsilon_{\mathsf{L}}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *and all* $x \in \Sigma_1^k$*, we have* $\Pr[\mathsf{Limit}(\mathcal{A}(y), \rho, \delta, x, y, \lambda) = 1] \leqslant \varepsilon_{\mathsf{L}}(\lambda)$*, where the probability is taken over* $\mathcal{A}$ *and* $y = \mathsf{Enc}_\lambda(x)$*.*

If dist *is the normalized Hamming distance* HAM*, we say the code is a* Hamming crLDC*; if* dist *is the normalized edit distance* ED*, we say the code is a* InsDel crLDC*. Here,* ℓ *is the* locality*,* ρ *is the* error-rate*,* p *is the* success probability*, and a function is* negligible *if it is* $o(x^{-c})$ *for all constants* $c > 0$*. If* $q_2 = 2$*, we say that* $\mathcal{C}$ *is a family of* binary crLDCs*, and if* $q_1 = q_2$ *we simply write* $C_\lambda[K, k, q_1]$*.*

Definition 1 closely follows the crLDC definition of Blocki et al. [13] with a few modifications. First, the constructions of [13] utilize a public random seed for a collision-resistant hash function, so their crLDC definition is quantified over the randomness of the seed generation algorithm. Our constructions do not require a public random seed so we omit this algorithm from our definition and instead quantify the security of our crLDC over a code family $\{C_\lambda\}_{\lambda \in \mathbb{N}}$. This quantification also captures the notion of asymptotic security when interacting with PPT adversaries, which differs from standard (r)LDC definitions that consider information-theoretic adversaries. Moreover, [13] requires the public random seed to be generated in an honest (i.e., trusted) way, and our definition and constructions circumvent this requirement. Second, we slightly strengthen the security definition by tweaking the predicate Fool: in Definition 1, the adversary wins if there *exists* an index i (not necessarily known by the adversary) such that the probability the decoder outputs correctly on input i is less than p. In contrast, [13] requires the adversary to output corrupt codeword y' and a target index i such that the probability the decoder outputs correctly on index i is less than p. Note that requiring Definition 1 to hold for $p = 2/3$, $\varepsilon_{\mathsf{F}}(\lambda) = \varepsilon_{\mathsf{L}}(\lambda) = 0$, and for all computationally unbounded adversaries $\mathcal{A}$ results in the original rLDC definition [12].

Our first contribution is constructing a family of binary Hamming crLDCs satisfying Definition 1. Our construction borrows from code concatenation techniques [38], which utilize an outer code $C_{out} = (\mathsf{Enc}_{out}, \mathsf{Dec}_{out})$ and an inner code $C_{in} = (\mathsf{Enc}_{in}, \mathsf{Dec}_{in})$ and encodes a message x as follows: (1) compute $y = \mathsf{Enc}_{out}(x)$; (2) partition y into some number d of blocks $y^{(1)} \| \ldots \| y^{(d)}$; (3) compute $Y^{(i)} = \mathsf{Enc}_{in}(y^{(i)})$ for all i; and (4) output $Y = Y^{(1)} \| \ldots \| Y^{(d)}$; here, $\|$ denotes string concatenation. In our construction, we use the identity function as C_{out}, utilize a suitable *digital signature scheme* to sign each block $y^{(i)}$, and use a classical Hamming code as C_{in}. Briefly, a digital signature scheme with signatures of length $r(\cdot)$ is a tuple of PPT algorithms $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ that satisfy the following properties: (1) Gen takes as input security parameter $\lambda \in \mathbb{N}$ (in unary) and outputs a key pair $(\mathsf{pk}, \mathsf{sk})$, where pk is the *public/verification key* and sk is the *private/signing key*; (2) Sign takes as input a message m of arbitrary length and the signing key sk and outputs a signature $\sigma \in \{0,1\}^{r(\lambda)}$ of message m. (3) Ver is deterministic and takes as input a message m, some signature σ, and a verification key pk outputs 1 iff σ is a valid signature of message m and 0 otherwise. (4) For all PPT adversaries $\mathcal{A}$, for $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, if $\mathcal{A}$ is given pk as input and given oracle access to $\mathsf{Sign}_{\mathsf{sk}}(\cdot)$, then Π is *secure* if, except with negligible probability in λ, $\mathcal{A}$ cannot output a pair $(\tilde{m}, \tilde{\sigma})$ such that $\mathsf{Ver}_{\mathsf{pk}}(\tilde{m}, \tilde{\sigma}) = 1$ and $\mathcal{A}$ never queried $\mathsf{Sign}_{\mathsf{sk}}(\tilde{m})$. Given a secure digital signature scheme and any binary Hamming code, we obtain our first main result.

Theorem 1. *Let* Π *be a* $r := r(\lambda)$ *length signature scheme. Let* C_{in} *be a binary Hamming code with rate* β_{in} *and error-rate* ρ_{in}*. Then for every positive polynomial* $k(\cdot)$ *and constant* $c \in (0, 1/2)$*, there exists a code family* $\mathcal{C}_{\mathsf{H}} := \{C_{\mathsf{H},\lambda}[K, k(\lambda), 2]\}_{\lambda \in \mathbb{N}}$ *and function* $\mu := \mu(\lambda)$ *such that* $\mathcal{C}_{\mathsf{H}}$ *is a* (ℓ, ρ, p, δ)*-Hamming* crLDC *with* $K = O((1/\beta_{in}) \max\{k(1 + \log(k)/r), r\})$*,* $\ell = O((\mu/\beta_{in}) \cdot (r + \log(k)))$*,* $\rho = c\rho_{in}$*,* $p = 1 - \exp(-\mu(1/2 - c)^2/2(1 - c)) > 2/3$*, and* $\delta = 1/2$*, where* $k := k(\lambda)$*.*

Our code family $\mathcal{C}_{\mathsf{H}}$ is constant rate whenever $\beta_{in} = \Theta(1)$ and $\Omega(\log(k(\lambda))) = r(\lambda) \leqslant k(\lambda)$. Our construction allows for $r(\lambda) > k(\lambda)$, but this results in locality $\ell \geqslant K$, so it is more efficient to use a Hamming code with comparable rate and error-rate. Any choice of μ satisfying $p > 2/3$ ensures that $\delta = 1/2$; e.g., $\mu(\lambda) := O(\log^{1+\epsilon}(\lambda))$ for constant $\epsilon > 0$ gives us polylog locality and success probability $1 - \mathsf{negl}(\lambda)$, where negl denotes some unspecified negligible function.

We can instantiate Theorem 1 with a constant rate and error-rate binary Hamming code C_{in} (e.g., [39]) and an appropriate signature scheme to achieve a constant rate and error-rate Hamming crLDC with polylog locality. Our construction shines when $r(\lambda) = \mathrm{polylog}(\lambda)$ and under standard idealized models there exist signature schemes with $r(\lambda)$ as small as $\Theta(\log^{1+\epsilon}(\lambda))$ for small constant $\epsilon > 0$ [40], [41], assuming these schemes satisfy the following notion of concrete security: for security parameter λ, any adversary running in time $2^{\lambda/2}$ can violate the security of the scheme with probability at most $2^{-\lambda/2}$ for signatures of length $r(\lambda) = \lambda$. Plugging in $\lambda' = \Theta(\log^{1+\epsilon}(\lambda))$, said schemes are secure against super-polynomial time adversaries with negligible security in λ, which implies they satisfy our definition of security for signature schemes. Using such a scheme with a constant rate

and error-rate Hamming code C_{in} and $\mu(\lambda) := O(\log^{1+\epsilon}(\lambda))$, we obtain the following corollary.

Corollary 1. *Let Π be a $r(\lambda) = \Theta(\log^{1+\epsilon}(\lambda))$ length signature scheme for constant $\epsilon > 0$. Then for all sufficiently large positive polynomials $k(\cdot)$, there exists code family $\{C_{\mathsf{H},\lambda}[K, k(\lambda), 2]\}_{\lambda\in\mathbb{N}}$ that is a (ℓ, ρ, p, δ)-Hamming* crLDC *with $K = O(k)$, $\ell = O(\log^{2(1+\epsilon)}(\lambda))$, $\rho = \Theta(1)$, $p = 1 - \mathsf{negl}(\lambda)$, and $\delta = 1/2$, where $k := k(\lambda)$.*

The parameters of Corollary 1 are comparable to the Hamming crLDC construction of [13], which achieves $K = O(k)$, $\ell = \mathrm{polylog}(n)$, $\rho = \Theta(1)$, $p = 1 - \mathsf{negl}(\lambda)$, and $\delta = \Theta(1)$. Our construction is arguably conceptually simpler than that of [13], which utilizes local expander graphs and collision-resistant hash functions (with a trusted setup), whereas our construction simply partitions, signs, and encodes. Moreover, our use of signatures does not require public key infrastructure as such schemes exist from one-way functions [42].

1) Extension to InsDel Errors: Our second contribution is extending the construction of Theorem 1 to handle InsDel errors. Prior constructions of InsDel LDCs utilized a so-called "Hamming-to-InsDel" compiler [34], [35]. Key to this compiler is a *noisy binary search* algorithm, which intuitively allows one to search an almost sorted list and find most entries with high probability. We use this algorithm to find blocks of codewords that are not "too corrupt", allowing us to handle more general InsDel errors. We use the noisy binary search tools of Block et al. [35] and the well-known Schulman-Zuckerman InsDel code [43] for C_{in} to extend Theorem 1 to the InsDel setting. Together with a secure digital signature scheme, we obtain our second main result.

Theorem 2. *Let Π be a $r := r(\lambda)$ length signature scheme. There exists a constant $c \in (0, 1/2)$ such that for every positive polynomial $k(\cdot)$ and constant $\rho^* \in (0, 1/3)$, there exists a code family $\mathcal{C}_{\mathsf{Ins}} := \{C_\lambda[K, k(\lambda), 2]\}_{\lambda\in\mathbb{N}}$ and a function $\mu := \mu(\lambda)$ such that $\mathcal{C}_{\mathsf{Ins}}$ is a (ℓ, ρ, p, δ)-InsDel* crLDC *for parameters $K = O(\max\{k(1+\log(k)/r), r\})$, $\ell = O((\log^3(n)+\mu)\cdot(r+\log(k)))$, $\rho = \Theta(1)$, $p = 1 - \rho^* - \exp(-\mu(1/2 - c)^2/2(1 - c)) > 2/3$, and $\delta = 1 - \Theta(\rho)$, where $k := k(\lambda)$.*

As with Theorem 1, our family $\mathcal{C}_{\mathsf{Ins}}$ is constant rate whenever $\Omega(\log(k(\lambda))) = r(\lambda) \leqslant k(\lambda)$, and additionally has the same downside whenever $r(\lambda) > k(\lambda)$, in which case it is more efficient to directly encode with an (asymptotically) optimal InsDel code (e.g., [43]). We again choose μ such that $p = 1 - \mathsf{negl}(\lambda) > 2/3$; moreover, under the same set of assumptions on the underlying signature scheme as with our Hamming crLDC (e.g., [40], [41]), for $\mu(\lambda) = \Theta(\log^{1+\epsilon}(\lambda))$ for small constant $\epsilon > 0$, we obtain the following corollary.

Corollary 2. *Let Π be a $r(\lambda) = \Theta(\log^{1+\epsilon}(\lambda))$ length signature scheme for constant $\epsilon > 0$. Then for all sufficiently large positive polynomials $k(\cdot)$, there exists code family $\{C_{\mathsf{I},\lambda}[K, k(\lambda), 2]\}_{\lambda\in\mathbb{N}}$ that is a (ℓ, ρ, p, δ)-InsDel* crLDC *with $K = O(k)$, $\ell = O(\log^{3(1+\epsilon)}(\lambda))$, $\rho = \Theta(1)$, $p = 1 - \mathsf{negl}(\lambda)$, and $\delta = 1 - \Theta(\rho)$, where $k := k(\lambda)$.*

To the best of our knowledge, our InsDel crLDCs are the first of their kind and compare favorably to the prior InsDel LDCs of Block et al. [35] and are comparable to the private and resource-bounded LDCs of Block and Blocki [37].

B. Related Work

Classical InsDel codes were initially studied in [15], inspiring a rich line of research into these codes; see surveys [44]–[46] for more information. Recently, k-deletion correcting codes with optimal rate were constructed, answering a long standing open question [32]. Randomized codes with positive rate that are correct a large fraction of deletions are studied in [16], [17]. Another line of work extends list decoding to InsDel codes [19], [26], [30]. Finally, [20] constructs explicit synchronization strings which can be "locally decoded" in the following sense: each index of the string is computable using symbols located at a small number of other locations in the string. These synchronization strings are used to construct near linear time interactive coding schemes for InsDel errors.

[14] initiated the study of codes resilient to errors introduced by computationally bounded channels. Several follow-up works adopt this channel model, yielding Hamming codes with better parameters than their classical counterparts [47]–[49]. It has been argued that any real-world communication channel can be reasonably modeled as a computationally bounded channel [14], [50], so one can reasonably expect error patterns encountered in nature to be modeled by some (possibly unknown) PPT algorithm. This channel model has also been extended to the LDC setting for both Hamming [13], [50]–[53] and, more recently, InsDel errors [37].

[12] introduced the notion of relaxed locally decodable codes. In a follow-up work, [54] introduced and construct *relaxed locally correctable codes* (rLCC) for Hamming errors: codes with local correction algorithms which can correct corrupt codeword symbols via querying a few locations into the received word. Their construction has significantly better parameters than classical Hamming LCCs, achieving constant locality, constant error-rate, and polynomial block length. Furthermore, their rLCC is also a rLDC since their code is systematic. [13] studies Hamming rLDCs/rLCCs in the context of computationally bounded channels (crLDC/crLCC). Our work directly adapts this model but for InsDel errors.

[34] initiated the study of InsDel LDCs. They give a compiler which transforms any Hamming LDC into an InsDel LDC, asymptotically preserving the rate and error-rate of the underlying Hamming LDC at the cost of a poly-logarithmic increase in the locality. [35] reproves this result with a conceptually simpler analysis using techniques borrowed from the study of a cryptographic object known as memory-hard functions [55]–[58]. [36] proposes the notion of Hamming/InsDel LDCs with randomized encodings in various settings, including when the encoder and decoder share randomness or when the channel adds error patterns non-adaptively. In the InsDel case, [36] invokes the compiler of [34] and obtain a code with block length $O(k)$ or $O(k\log(k))$ and $\mathrm{polylog}(k)$ locality. Recently, [37] extends the compiler of [35] to the private-

key setting of [51], where the encoder and decoder share a secret key unknown to the channel, and to the resource-bounded setting of [50], where the channel is assumed to be resource constrained in some way. While it is likely that applying the "Hamming-to-InsDel" compiler to the crLDC of [13] or our crLDCs would yield an InsDel crLDC, this has not been formally claimed or proven in prior work. Finally, there has been recent progress in obtaining lower bounds for InsDel LDCs. [59] proved that InsDel LDCs with constant locality, even in the private-key setting, require exponential block length, and also show that linear 2-query InsDel LDCs do not exist. This makes it all the more surprising that a constant rate InsDel crLDC in the polylog locality regime exist.

II. Technical Overview

The main technical ingredients for both our Hamming and InsDel crLDC constructions are the use of a digital signature scheme Π with r-length signatures along with a suitable inner code C_{in}. The encoding algorithms for both codes are nearly identical, with the main difference being the choice of C_{in}. The decoding algorithms are also similar: the InsDel decoder is a (non-trivial) modification of the Hamming decoder to handle InsDel errors using noisy binary search techniques.

1) Hamming crLDC *Construction:* Let C_{in} be an appropriate Hamming code (i.e., non-local), and let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be an r-length signature scheme.

The Hamming Encoder: We define a family of encoding algorithms $\{\mathsf{Enc}_{\mathsf{H},\lambda}\}_\lambda$. Let $\lambda \in \mathbb{N}$ be the security parameter. For any message $x \in \{0,1\}^k$, encoder $\mathsf{Enc}_{\mathsf{H},\lambda}$ partitions x into $d = \lceil k/r(\lambda) \rceil$ blocks $x = x^{(1)} \| \cdots \| x^{(d)}$, where $x^{(i)} \in \{0,1\}^{r(\lambda)}$ for all i (padding with 0 as necessary). Each $x^{(i)}$ is now signed using Π: $\mathsf{Enc}_{\mathsf{H},\lambda}$ generates key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ and computes signature $\sigma^{(i)} \leftarrow \mathsf{Sign}_{\mathsf{sk}}(x^{(i)} \| i)$. Next, the block $x^{(i)} \| \sigma^{(i)} \| \mathsf{pk} \| i$ is encoded using C_{in} to obtain codeword $c^{(i)}$, where pk is the public key generated previously. Finally, $\mathsf{Enc}_{\mathsf{H},\lambda}$ outputs $C = c^{(1)} \| \cdots \| c^{(d)} \in \{0,1\}^K$. If $r(\lambda) \geqslant k$, only a single block is signed and encoded at the cost of locality $\geqslant K$, so it is more efficient to use a Hamming code with similar rate and error-rate rather than $\mathsf{Enc}_{\mathsf{H},\lambda}$.

The Hamming Decoder: We define a family of decoding algorithms $\{\mathsf{Dec}_{\mathsf{H},\lambda}\}_\lambda$. Let $\lambda \in \mathbb{N}$, $\mu \in \mathbb{N}$ be a parameter of our choice, $x \in \{0,1\}^k$, $C = \mathsf{Enc}_{\mathsf{H},\lambda}(x)$, and $\tilde{C} \leftarrow \mathcal{A}(C)$ such that $\mathsf{HAM}(C, \tilde{C}) \leqslant \rho$, where $\mathcal{A}$ is a PPT adversary. On input $i \in [k]$ and given oracle access to $\tilde{C}$, the decoder $\mathsf{Dec}_{\mathsf{H},\lambda}$ tries to recover x_i via a two-step process. First, $\mathsf{Dec}_{\mathsf{H},\lambda}$ tries to recover the true public key pk. It begins by uniformly sampling $j_1, \ldots, j_\mu \xleftarrow{\$} [d]$. Parsing $\tilde{C}$ as $\tilde{C}^{(1)} \| \cdots \| \tilde{C}^{(d)}$, for each $\kappa \in [\mu]$ $\mathsf{Dec}_{\mathsf{H},\lambda}$ (1) recovers some $\tilde{m}^{(j_\kappa)} \leftarrow \mathsf{Dec}_{in}(\tilde{C}^{(j_\kappa)})$; (2) parses $\tilde{m}^{(j_\kappa)}$ as $\tilde{x}^{(j_\kappa)} \| \tilde{\sigma}^{(j_\kappa)} \| \tilde{\mathsf{pk}}^{(j_\kappa)} \| \tilde{j}$; and (3) recovers $\tilde{\mathsf{pk}}^{(j_\kappa)}$. The decoder then sets $\mathsf{pk}^* = \mathsf{majority}(\tilde{\mathsf{pk}}^{(j_1)}, \ldots, \tilde{\mathsf{pk}}^{(j_\mu)})$. Second, $\mathsf{Dec}_{\mathsf{H},\lambda}$ computes j such that x_i lies in $x^{(j)}$, computes $\tilde{m}^{(j)} \leftarrow \mathsf{Dec}_{in}(\tilde{C}^{(j)})$, parses it as $\tilde{x}^{(j)} \| \tilde{\sigma}^{(j)} \| \tilde{\mathsf{pk}}^{(j)} \| \tilde{j}$, then checks if $\mathsf{Ver}_{\mathsf{pk}^*}(\tilde{x}^{(j)} \| j, \tilde{\sigma}^{(j)}) = 1$, outputting $\tilde{x}^{(j)}_{i^*}$, where i^* is the index of $x^{(j)}$ that corresponds to x_i if true; else $\mathsf{Dec}_{\mathsf{H},\lambda}$ outputs $\perp$. Here it is crucial for $\mathsf{Dec}_{\mathsf{H},\lambda}$ to use its computed value j, otherwise it is possible for an adversary to swap two blocks $C^{(j_1)}$ and $C^{(j_2)}$ where $x^{(j_1)} \neq x^{(j_2)}$, violating Item 3.

Theorem 1 Proof Overview: The main technical challenge of proving Theorem 1 is showing that $\{(\mathsf{Enc}_{\mathsf{H},\lambda}, \mathsf{Dec}_{\mathsf{H},\lambda})\}_\lambda$ satisfies Items 3 and 4 of Definition 1. Towards Item 3, for any $x \in \{0,1\}^k$, PPT adversary $\mathcal{A}$, and $i \in [k]$, we analyze the probability that $\mathsf{Dec}^{\tilde{C}}_{\mathsf{H},\lambda}(i) \in \{x_i, \perp\}$ for $\tilde{C} \leftarrow \mathcal{A}(\mathsf{Enc}_{\mathsf{H},\lambda}(x))$ such that $\mathsf{HAM}(\tilde{C}, \mathsf{Enc}_{\mathsf{H},\lambda}(x)) \leqslant \rho$. If $\mathsf{Dec}^{\tilde{C}}_{\mathsf{H},\lambda}(i) = x_i$, then $\tilde{x}^{(j)}_{i^*} = x_i$ and $\mathsf{Ver}_{\mathsf{pk}^*}(\tilde{x}^{(j)} \| j, \tilde{\sigma}^{(j)}) = 1$. Conditioning on $(\tilde{x}^{(j)} \| j, \tilde{\sigma}^{(j)})$ not breaking the security of Π, successful verification implies $\tilde{x}^{(j)} = x^{(j)}$ and $\tilde{\sigma}^{(j)} = \sigma^{(j)}$, and verification succeeds whenever $\mathsf{pk}^* = \mathsf{pk}$. By Chernoff, we can ensure that $\mathsf{pk}^* = \mathsf{pk}$ with high probability (depending on μ) as long as more than half of the blocks $\tilde{C}^{(j)}$ have less than ρ_{in}-fraction of Hamming errors, which is achieved by setting $\rho = c\rho_{in}$ for any $c \in (0, 1/2)$. Now by definition, $\tilde{\sigma}^{(i)}$ does not break security of Π with probability at least $1 - \varepsilon_\Pi(\lambda)$ (i.e., either it is a correct signature or verification fails), where $\varepsilon_\Pi(\lambda)$ is a negligible function for the security of Π. As index i was arbitrary here, we establish Item 3 via a union bound for $p = 1 - \exp(-\mu(1/2 - c)^2/2(1 - c))$ and $\varepsilon_{\mathsf{F}}(\lambda) = k \cdot \varepsilon_\Pi(\lambda)$, where $\varepsilon_{\mathsf{F}}(\lambda)$ is negligible since $k = \mathrm{poly}(\lambda)$.

Towards Item 4, define $\mathcal{J} := \{j : \mathsf{HAM}(\tilde{C}^{(j)}, C^{(j)}) \leqslant \rho_{in}\}$. Then $\rho = c\rho_{in}$ for $c \in (0, 1/2)$ implies $|\mathcal{J}| \geqslant d/2$. Moreover, $\mathsf{Dec}_{in}(\tilde{C}^{(j)}) = x^{(j)} \| \sigma^{(j)} \| \mathsf{pk} \| j$ for any $j \in \mathcal{J}$. Now for any $i \in [k]$, if x_i lies in $x^{(j)}$ for $j \in \mathcal{J}$, the probability $\mathsf{Dec}_{\mathsf{H},\lambda}$ outputs x_i equals the probability that $\mathsf{Dec}_{\mathsf{H},\lambda}$ correctly recovers $\mathsf{pk}^* = \mathsf{pk}$. We choose μ to ensure $\Pr[\mathsf{pk}^* = \mathsf{pk}] > 2/3$. This along with $|\mathcal{J}| \geqslant d/2$ implies that $|\mathsf{Good}(\tilde{C})| \geqslant k/2$ and $\delta = 1/2$. By our choice of ρ, $|\mathcal{J}| \geqslant d/2$ and $|\mathsf{Good}(\tilde{C})| \geqslant k/2$ holds *for any* $\tilde{C}$ such that $\mathsf{HAM}(C, \tilde{C}) \leqslant \rho$; Thus Item 4 holds with $\delta = 1/2$ and $\varepsilon_{\mathsf{L}}(\lambda) := 0$. We refer the reader to the full version of our work [60] for the complete proof.

2) InsDel crLDC *Construction:* Let C_{in} be the Schulman-Zuckerman InsDel code (SZ code) [43] and let $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be an r-length signature scheme.

Challenges to Decoding InsDel Errors: InsDel errors allow an adversary to insert symbols into and delete symbols from codewords, which introduces challenges that do not arise with Hamming errors. One may hope to simply use our family $\{(\mathsf{Enc}_{\mathsf{H},\lambda}, \mathsf{Dec}_{\mathsf{H},\lambda})\}_\lambda$ with $C_{in} = \mathsf{SZ}$ to achieve Theorem 2; however this yields a trivial InsDel crLDC. Let $\mathsf{Enc}'_{\mathsf{H},\lambda}$ be identical to $\mathsf{Enc}_{\mathsf{H},\lambda}$ except we use the SZ code as the code C_{in}. For any x and $C = \mathsf{Enc}'_{\mathsf{H},\lambda}(x)$, there is a simple attack to ensure that $\mathsf{Dec}_{\mathsf{H},\lambda}$ *always* outputs $\perp$: the adversary simply transforms $C = C^{(1)} \| \ldots \| C^{(d)}$ into $\tilde{C} = C^{(d)}_1 \| C^{(1)} \| \cdots \| C^{(d-1)} \| C^{(d)}_0$, where $C^{(d)}_0, C^{(d)}_1$ are the first and second halves of $C^{(d)}$, respectively. This implies that $\{(\mathsf{Enc}'_{\mathsf{H},\lambda}, \mathsf{Dec}_{\mathsf{H},\lambda})\}_\lambda$ is an InsDel crLDC with $\delta = 0$; i.e., it always outputs $\perp$ given a corrupt codeword. However, we can handle this and more general attacks by leveraging the noisy binary search techniques of Block et al. [35].

Noisy Binary Search Overview: To understand the noisy binary search algorithm NBS and its guarantees, we require the notion of γ-*goodness*. For $x, y \in \{0,1\}^*$, we say that y is

γ-good with respect to x if $\mathsf{ED}(x, y) \leqslant \gamma$. The notion of γ-goodness (albeit under different formal definitions) has been useful in the design and analysis of depth-robust graphs, a combinatorial object used extensively in the study of memory-hard functions [55]–[57], and it is essential to the success of NBS. Intuitively, for a fixed "correct" ordered list of strings $A = (a_1, \ldots, a_n)$, each of length κ, and some other list of strings $B = (b_1, \ldots, b_{n'})$, the algorithm NBS finds any string b_j that is γ-good with respect to the string a_j for $j \in [n]$, except with negligible probability. In our context, each b_j corresponds to blocks in the (possibly corrupt) codeword. Given a tolerance parameter $\rho^* \in (0, 1/2)$, the NBS algorithm on input $j \in [n]$ outputs b_j for at least $(1-\rho^*)$-fraction of the γ-good indices j, except with negligible probability. Moreover, NBS runs in time $\kappa \cdot \mathrm{polylog}(n')$, which is only possible by allowing NBS to fail on a small fraction of γ-good indices, else the algorithm requires $\Omega(\kappa n')$ time.

Suppose that $\tilde{C} \in \{0,1\}^{n'}$ for some n' is a corrupt codeword (from an appropriate encoding algorithm) and let $i \in [k]$. We use the NBS algorithm to search $\tilde{C}$ for some (possibly corrupt) block $\tilde{m}^{(j)}$ which contains the desired symbol x_i. So long as $\tilde{C}$ and $\tilde{m}^{(j)}$ are "not too corrupt", then NBS outputs $\tilde{m}^{(j)}$ with high probability. For searching, NBS utilizes a block decoding algorithm BlockDec to find $\tilde{m}^{(j)}$ within $\tilde{C}$ with the following guarantee: for input i, if i is within a (small) ball around a γ-good block $\tilde{m}^{(j)}$, then BlockDec outputs $\tilde{m}^{(j)}$ with probability at least $1 - \gamma$. Assuming $\tilde{m}^{(j)}$ is not too corrupt, we can parse it as $\tilde{x}^{(j)} \| \tilde{\sigma}^{(j)} \| \tilde{\mathsf{pk}} \| \tilde{j}$ and use Ver to ensure that $\tilde{x}^{(j)}$ is correct. Note that both NBS and BlockDec can fail and output $\perp$, which we take into consideration for our decoder.

The InsDel Encoder: We define a family of encoding algorithms $\{\mathsf{Enc}_{\mathsf{I},\lambda}\}_\lambda$. Let $\lambda \in \mathbb{N}$ be the security parameter and let α be a constant specified by the NBS algorithm [35]. For any message $x \in \{0,1\}^k$, encoder $\mathsf{Enc}_{\mathsf{I},\lambda}$ behaves identically to $\mathsf{Enc}_{\mathsf{H},\lambda}$ by partitioning x into $d = \lceil x/r(\lambda) \rceil$ blocks, generating $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$, computing $\sigma^{(j)} \leftarrow \mathsf{Sign}_{\mathsf{sk}}(x^{(j)} \| j)$, and computing $c^{(j)} = \mathsf{SZ.Enc}(x^{(j)} \| \sigma^{(j)} \| \mathsf{pk} \| j)$ for every j. Next, the encoder computes buffered codewords $C^{(j)} = 0^{\alpha r(\lambda)} \| c^{(j)} \| 0^{\alpha r(\lambda)}$ for every j, where $0^{\alpha r(\lambda)}$ is a all-zero vector of length $\alpha r(\lambda)$ and ensure the success of the NBS and BlockDec algorithms. Finally, $\mathsf{Enc}_{\mathsf{I},\lambda}$ outputs $C = C^{(1)} \| \cdots \| C^{(d)}$. Again, if $r(\lambda) \geqslant k$, it is more efficient to simply to encode x using the SZ code.

The InsDel Decoder: We define a family of decoding algorithms $\{\mathsf{Dec}_{\mathsf{I},\lambda}\}_\lambda$. Let $\lambda \in \mathbb{N}$, $\mu \in \mathbb{N}$ be a parameter of our choice, $x \in \{0,1\}^k$, $C = \mathsf{Enc}_{\mathsf{I},\lambda}(x)$, and $\tilde{C} \leftarrow \mathcal{A}(C)$ such that $\mathsf{ED}(C, \tilde{C}) \leqslant \rho$, where $\mathcal{A}$ is a PPT adversary and $\tilde{C} \in \{0,1\}^{K'}$ for some K'. Then on input $i \in [k]$ and given oracle access to $\tilde{C}$, the decoder $\mathsf{Dec}_{\mathsf{I},\lambda}$ tries to recover x_i via the same two-step process as $\mathsf{Dec}_{\mathsf{H},\lambda}$: first, recover the public key pk; and second, find block j that is supposed to contain x_i and use the recovered pk to verify its integrity. Recovery of pk is done similarly to $\mathsf{Dec}_{\mathsf{H},\lambda}$, except we leverage BlockDec to find blocks with potential public keys. $\mathsf{Dec}_{\mathsf{I},\lambda}$ first samples $i_1, \ldots, i_\mu \xleftarrow{\$} [n]$ uniformly at random, then obtains $\tilde{m}^{(j_\kappa)} \leftarrow \mathsf{BlockDec}(i_\kappa)$ for each $\kappa \in [\mu]$, where $j_\kappa \in [d]$. Intuitively, in the InsDel setting we need to search for each block j_κ whereas in the Hamming setting we knew exactly where each block was located. If $\tilde{m}^{(j_\kappa)} = \perp$, then we set $\mathsf{pk}_\kappa = \perp$; else, we parse $\tilde{m}^{(j_\kappa)}$ as $\tilde{x}^{(j_\kappa)} \| \tilde{\sigma}^{(j_\kappa)} \| \tilde{\mathsf{pk}}^{(j_\kappa)} \| \tilde{j}$ and set $\mathsf{pk}_\kappa = \tilde{\mathsf{pk}}^{(j_\kappa)}$. Finally, we let $\mathsf{pk}^* = \mathsf{majority}(\mathsf{pk}_1, \ldots, \mathsf{pk}_\kappa)$. Next, $\mathsf{Dec}_{\mathsf{I},\lambda}$ computes j such that x_i lies in $x^{(j)}$. Then $\mathsf{Dec}_{\mathsf{I},\lambda}$ obtains $\tilde{m}^{(j)} \leftarrow \mathsf{NBS}(j)$. If either $\tilde{m}^{(j)}$ or pk^* are $\perp$, then $\mathsf{Dec}_{\mathsf{I},\lambda}$ aborts and outputs $\perp$. Otherwise, $\tilde{m}^{(j)}$ is parsed as $\tilde{x}^{(j)} \| \tilde{\sigma}^{(j)} \| \tilde{\mathsf{pk}}^{(j)} \| \tilde{j}$ and then $\mathsf{Dec}_{\mathsf{I},\lambda}$ checks if $\mathsf{Ver}_{\mathsf{pk}^*}(\tilde{x}^{(j)} \| j, \tilde{\sigma}^{(j)}) = 1$. If not, $\mathsf{Dec}_{\mathsf{I},\lambda}$ outputs $\perp$; otherwise, $\mathsf{Dec}_{\mathsf{I},\lambda}$ outputs $\tilde{x}^{(j)}_{i^*}$, where i^* is the index of $x^{(j)}$ corresponding to x_i.

Theorem 2 Proof Overview: The main technical challenge of proving Theorem 2 is showing that $\{(\mathsf{Enc}_{\mathsf{I},\lambda}, \mathsf{Dec}_{\mathsf{I},\lambda})\}_\lambda$ satisfies Items 3 and 4 of Definition 1. Towards Item 3, for any $x \in \{0,1\}^k$, PPT adversary $\mathcal{A}$, and $i \in [k]$, we analyze the probability that $\mathsf{Dec}^{\tilde{C}}_{\mathsf{I},\lambda}(i) \in \{x_i, \perp\}$ for $\tilde{C} \leftarrow \mathcal{A}(\mathsf{Enc}_{\mathsf{I},\lambda}(x))$ such that $\mathsf{ED}(\tilde{C}, \mathsf{Enc}_{\mathsf{I},\lambda}(x)) \leqslant \rho$. The proof proceeds identically to the Hamming crLDC with the following key changes. First, when recovering the public key, we must consider the success probability of BlockDec in our Chernoff bound to ensure $\mathsf{pk}^* = \mathsf{pk}$ with high probability. Second, we must consider the success probability of NBS when recovering block j that contains x_i. Careful selection of parameters and the guarantees of BlockDec and NBS ensures Item 3 holds.

Towards Item 4, the proof again proceeds nearly identically to the Hamming crLDC case, except again we must take into consideration the recovery of public key $\mathsf{pk}^* = \mathsf{pk}$ via BlockDec and the recovery of block j with NBS. The noisy binary search algorithm recovers any block that is γ-good with probability greater than $2/3$ (under suitable parameter choices), except with negligible probability. This directly translates to the fraction $\delta = 1 - \Theta(\rho)$ of indices we are able to decode from for Item 4. We refer the reader to the full version of our work [60] for the complete proof.

ACKNOWLEDGMENTS

Alexander R. Block was supported in part by the National Science Foundation under NSF CCF #1910659 and in part by DARPA under agreement No. HR00112020022 and No. HR00112020025. Jeremiah Blocki was supported in part by the National Science Foundation under awards CNS #2047272, CNS #1931443 and CCF #1910659. The views, opinions, findings, conclusions and/or recommendations expressed in this material are those of the author and should not be interpreted as reflecting the position or policy of the Department of Defense or the U.S. Government, and no official endorsement should be inferred.

REFERENCES

[1] J. Katz and L. Trevisan, "On the efficiency of local decoding procedures for error-correcting codes," in *STOC*. ACM, 2000.

[2] M. Sudan, L. Trevisan, and S. P. Vadhan, "Pseudorandom generators without the XOR lemma," *J. Comput. Syst. Sci.*, vol. 62, no. 2, 2001.

[3] I. Kerenidis and R. de Wolf, "Exponential lower bound for 2-query locally decodable codes via a quantum argument," *J. Comput. Syst. Sci.*, vol. 69, no. 3, 2004.

[4] S. Yekhanin, "Towards 3-query locally decodable codes of subexponential length," *J. ACM*, vol. 55, no. 1, 2008.

[5] Z. Dvir, P. Gopalan, and S. Yekhanin, "Matching vector codes," *SIAM J. Comput.*, vol. 40, no. 4, 2011.

[6] K. Efremenko, "3-query locally decodable codes of subexponential length," *SIAM J. Comput.*, vol. 41, no. 6, 2012.

[7] S. Yekhanin, "Locally decodable codes," *Found. Trends Theor. Comput. Sci.*, vol. 6, no. 3, 2012.

[8] S. Kopparty and S. Saraf, "Guest column: Local testing and decoding of high-rate error-correcting codes," *SIGACT News*, vol. 47, no. 3, 2016.

[9] S. Kopparty, O. Meir, N. Ron-Zewi, and S. Saraf, "High-rate locally correctable and locally testable codes with sub-polynomial query complexity," *J. ACM*, vol. 64, no. 2, 2017.

[10] D. P. Woodruff, "New lower bounds for general locally decodable codes," *Electron. Colloquium Comput. Complex.*, vol. TR07-006, 2007.

[11] ——, "A quadratic lower bound for three-query linear locally decodable codes over any field," *J. Comput. Sci. Technol.*, vol. 27, no. 4, 2012.

[12] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. P. Vadhan, "Robust pcps of proximity, shorter pcps, and applications to coding," *SIAM J. Comput.*, vol. 36, no. 4, 2006.

[13] J. Blocki, V. Gandikota, E. Grigorescu, and S. Zhou, "Relaxed locally correctable codes in computationally bounded channels," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, 2021.

[14] R. J. Lipton, "A new approach to information theory," in *STACS*, vol. 775. Springer, 1994.

[15] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals." *Soviet Physics Doklady*, vol. 10, no. 8, 1966, doklady Akademii Nauk SSSR, V163 No4 845-848 1965.

[16] M. A. Kiwi, M. Loebl, and J. Matousek, "Expected length of the longest common subsequence for large alphabets," in *LATIN*, vol. 2976. Springer, 2004.

[17] V. Guruswami and C. Wang, "Deletion codes in the high-noise and high-rate regimes," *IEEE Trans. Inf. Theory*, vol. 63, no. 4, 2017.

[18] J. Brakensiek, V. Guruswami, and S. Zbarsky, "Efficient low-redundancy codes for correcting multiple deletions," *IEEE Trans. Inf. Theory*, vol. 64, no. 5, 2018.

[19] B. Haeupler, A. Shahrasbi, and M. Sudan, "Synchronization strings: List decoding for insertions and deletions," in *ICALP*, vol. 107. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[20] B. Haeupler and A. Shahrasbi, "Synchronization strings: explicit constructions, local decoding, and applications," in *STOC*. ACM, 2018.

[21] K. Cheng, B. Haeupler, X. Li, A. Shahrasbi, and K. Wu, "Synchronization strings: Highly efficient deterministic constructions over small alphabets," in *SODA*. SIAM, 2019.

[22] K. Cheng, Z. Jin, X. Li, and K. Wu, "Block edit errors with transpositions: Deterministic document exchange protocols and almost optimal binary codes," in *ICALP*, vol. 132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[23] V. Guruswami and R. Li, "Polynomial time decodable codes for the binary deletion channel," *IEEE Trans. Inf. Theory*, vol. 65, no. 4, 2019.

[24] B. Haeupler, A. Rubinstein, and A. Shahrasbi, "Near-linear time insertion-deletion codes and $(1+\epsilon)$-approximating edit distance via indexing," in *STOC*. ACM, 2019.

[25] B. Haeupler, "Optimal document exchange and new codes for insertions and deletions," in *FOCS*. IEEE Computer Society, 2019.

[26] S. Liu, I. Tjuawinata, and C. Xing, "On list decoding of insertion and deletion errors," *CoRR*, vol. abs/1906.09705, 2019.

[27] K. Cheng, V. Guruswami, B. Haeupler, and X. Li, "Efficient linear and affine codes for correcting insertions/deletions," in *SODA*. SIAM, 2021.

[28] V. Guruswami and R. Li, "Coding against deletions in oblivious and online models," *IEEE Trans. Inf. Theory*, vol. 66, no. 4, 2020.

[29] K. Cheng and X. Li, "Efficient document exchange and error correcting codes with asymmetric information," in *SODA*. SIAM, 2021.

[30] V. Guruswami, B. Haeupler, and A. Shahrasbi, "Optimally resilient codes for list-decoding from insertions and deletions," *IEEE Trans. Inf. Theory*, vol. 67, no. 12, 2021.

[31] B. Haeupler and A. Shahrasbi, "Synchronization strings: Codes for insertions and deletions approaching the singleton bound," *J. ACM*, vol. 68, no. 5, 2021.

[32] J. Sima and J. Bruck, "On optimal k-deletion correcting codes," *IEEE Trans. Inf. Theory*, vol. 67, no. 6, 2021.

[33] K. Cheng, Z. Jin, X. Li, and K. Wu, "Deterministic document exchange protocols and almost optimal binary codes for edit errors," *J. ACM*, vol. 69, no. 6, 2022.

[34] R. Ostrovsky and A. Paskin-Cherniavsky, "Locally decodable codes for edit distance," in *ICITS*, vol. 9063. Springer, 2015.

[35] A. R. Block, J. Blocki, E. Grigorescu, S. Kulkarni, and M. Zhu, "Locally decodable/correctable codes for insertions and deletions," in *FSTTCS*, vol. 182. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[36] K. Cheng, X. Li, and Y. Zheng, "Locally decodable codes with randomized encoding," *IACR Cryptol. ePrint Arch.*, 2020.

[37] A. R. Block and Jeremiah, "Private and resource-bounded locally decodable codes for insertions and deletions," in *ISIT*. IEEE, 2021.

[38] G. D. F. Jr., "Generalized minimum distance decoding," *IEEE Trans. Inf. Theory*, vol. 12, no. 2, 1966.

[39] J. Justesen, "Class of constructive asymptotically good algebraic codes," *IEEE Trans. Inf. Theory*, vol. 18, no. 5, pp. 652–656, 1972.

[40] C. Schnorr, "Efficient identification and signatures for smart cards," in *CRYPTO*, vol. 435. Springer, 1989.

[41] J. Blocki and S. Lee, "On the multi-user security of short schnorr signatures with preprocessing," in *EUROCRYPT*. Springer, 2022.

[42] L. Lamport, "Constructing digital signatures from a one way function," Tech. Rep., 1979.

[43] L. J. Schulman and D. Zuckerman, "Asymptotically good codes correcting insertions, deletions, and transpositions," *IEEE Trans. Inf. Theory*, vol. 45, no. 7, 1999.

[44] N. Sloane, "On single-deletion-correcting codes," *arxiv:Combinatorics*, 2002.

[45] M. Mitzenmacher, "A survey of results for deletion channels and related synchronization channels," in *SWAT*, vol. 5124. Springer, 2008.

[46] H. Mercier, V. K. Bhargava, and V. Tarokh, "A survey of error-correcting codes for channels with symbol synchronization errors," *IEEE Commun. Surv. Tutorials*, vol. 12, no. 1, 2010.

[47] S. Micali, C. Peikert, M. Sudan, and D. A. Wilson, "Optimal error correction against computationally bounded noise," in *TCC*, vol. 3378. Springer, 2005, pp. 1–16.

[48] V. Guruswami and A. D. Smith, "Optimal rate code constructions for computationally simple channels," *J. ACM*, vol. 63, no. 4, 2016.

[49] R. Shaltiel and J. Silbak, "Explicit list-decodable codes with optimal rate for computationally bounded channels," *Comput. Complex.*, vol. 30, no. 1, p. 3, 2021.

[50] J. Blocki, S. Kulkarni, and S. Zhou, "On locally decodable codes in resource bounded channels," in *ITC*, vol. 163. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, pp. 16:1–16:23.

[51] R. Ostrovsky, O. Pandey, and A. Sahai, "Private locally decodable codes," in *ICALP*, vol. 4596. Springer, 2007, pp. 387–398.

[52] B. Hemenway and R. Ostrovsky, "Public-key locally-decodable codes," in *CRYPTO*, vol. 5157. Springer, 2008, pp. 126–143.

[53] B. Hemenway, R. Ostrovsky, M. J. Strauss, and M. Wootters, "Public key locally decodable codes with short keys," in *APPROX-RANDOM*, vol. 6845. Springer, 2011, pp. 605–615.

[54] T. Gur, G. Ramnarayan, and R. Rothblum, "Relaxed locally correctable codes," *Theory Comput.*, vol. 16, pp. 1–68, 2020.

[55] P. Erdös, R. L. Graham, and E. Szemeredi, "On sparse graphs with dense long paths." Stanford University, Stanford, CA, USA, Tech. Rep., 1975.

[56] J. Alwen, J. Blocki, and B. Harsha, "Practical graphs for optimal side-channel resistant memory-hard functions," in *CCS*. ACM, 2017.

[57] J. Alwen, J. Blocki, and K. Pietrzak, "Sustained space complexity," in *EUROCRYPT (2)*, vol. 10821. Springer, 2018.

[58] J. Blocki and B. Holman, "Sustained space and cumulative complexity trade-offs for data-dependent memory-hard functions," in *CRYPTO (3)*, vol. 13509. Springer, 2022.

[59] J. Blocki, K. Cheng, E. Grigorescu, X. Li, Y. Zheng, and M. Zhu, "Exponential lower bounds for locally decodable and correctable codes for insertions and deletions," in *FOCS*. IEEE, 2021.

[60] A. R. Block and J. Blocki, "Computationally relaxed locally decodable codes, revisited," *CoRR*, vol. abs/2305.01083, 2023.