

Silver: Silent VOLE and Oblivious Transfer from Hardness of Decoding Structured LDPC Codes

Geoffroy Couteau, Peter Rindal, Srinivasan Raghuraman

Abstract. We put forth new protocols for oblivious transfer extension and vector OLE, called *Silver*, for SILent Vole and oblivious transfER. Silver offers extremely high performances: generating 10 million random OTs on one core of a standard laptop requires only 300ms of computation and 122KB of communication. This represents 37% less computation and $\sim 1300\times$ less communication than the standard IKNP protocol, as well as $\sim 4\times$ less computation and $\sim 14\times$ less communication than the recent protocol of Yang et al. (CCS 2020). Silver is *silent*: after a one-time cheap interaction, two parties can store small seeds, from which they can later *locally* generate a large number of OTs *while remaining offline*. Neither IKNP nor Yang et al. enjoys this feature; compared to the best known silent OT extension protocol of Boyle et al. (CCS 2019), upon which we build up, Silver has $19\times$ less computation, and the same communication. Due to its attractive efficiency features, Silver yields major efficiency improvements in numerous MPC protocols.

Our approach is a radical departure from the standard paradigm for building MPC protocols, in that we do *not* attempt to base our constructions on a well-studied assumption. Rather, we follow an approach closer in spirit to the standard paradigm in the design of symmetric primitives: we identify a set of fundamental structural properties that allow us to withstand all known attacks, and put forth a candidate design, guided by our analysis. We also rely on extensive experimentations to analyze our candidate and experimentally validate their properties. In essence, our approach boils down to constructing new families of linear codes with (plausibly) high minimum distance and extremely low encoding time. While further analysis is of course welcomed in order to gain total confidence in the security of Silver, we hope and believe that initiating this approach to the design of MPC primitives will pave the way to new secure primitives with extremely attractive efficiency features.

1 Introduction

Secure multiparty computation (MPC) allows n parties to jointly evaluate a function f , while leaking no information on their own input beyond the output of the function. It is a fundamental problem in cryptography, which has received considerable attention since its introduction in the seminal works of Yao [Yao86], and Goldreich, Micali, and Wigderson [GMW87b, GMW87a]. While early feasibility results for MPC were mainly of theoretical interest, MPC protocols have enjoyed tremendous improvements in the past decade.

Oblivious transfers (OT) are perhaps the most fundamental building block for MPC protocols. In a random OT, two parties receive respectively (s_0, s_1) and

(s_b, b) , where (s_0, s_1) are two random *strings*, and b is a random *selection bit*. Random OT is a complete primitive for secure computation [Kil88], and modern MPC protocols rely on it. Efficiency improvements in protocols for generating OTs directly translate into improvements for a plethora of MPC protocols.

OT Extension. A long line of work, initiated with the breakthrough work of [IKNP03], has therefore sought to develop increasingly efficient protocols for generating a large number of random OTs. At a high level, OT extension protocols [IKNP03, KOS15, KKRT16, OOS17] turns a small number of base OTs into a near-arbitrary number of OTs, using only cheap operations. The latest generation of these protocols, initiated in [BCG⁺17], leverages the notion of pseudorandom correlation generators (PCGs) [BCGI18, BCG⁺19b] to enable the construction of extremely efficient OT extension protocols. This line of work recently culminated with the protocols of [BCG⁺19a, SGRR19, WYKW20, YWL⁺20].

Silent OT Extension. While PCGs allow for very efficient constructions of OT extension, this is not their main *claim to fame*: perhaps their most remarkable feature is that they allow the construction of *silent* OT extension protocols. A silent protocol has the property that: after a short interaction, with communication and computation essentially independent of the target number of OTs, the parties can locally store small correlated seeds. Then, the parties can later retrieve these seeds, and *without any further interaction* stretch them into a very large number of OTs. Unfortunately, while the protocols of [BCG⁺19a] enjoy the silent feature, the running time improvements in [SGRR19, WYKW20, YWL⁺20] were achieved at the cost of sacrificing this crucial property.

1.1 Our Results

In this work, we design new protocols for silent oblivious transfer extension and silent vector oblivious linear evaluation (VOLE). The latter is defined over a field \mathbb{F} and allows a receiver with input $x \in \mathbb{F}$ to obtain $x \cdot \mathbf{a} + \mathbf{b}$ from a sender with input vectors (\mathbf{a}, \mathbf{b}) over \mathbb{F} . VOLE is another important building block in some of the most prominent secure computation tasks; e.g. the current most efficient private set intersection [RS21]. We call our (family of) protocols *Silver*, which stands for SILent Vole and oblivious transfER. In addition its silent feature, Silver exhibits extremely good performances, significantly outperforming the most efficient OT extension protocols [IKNP03, YWL⁺20] on all fronts.

At the heart of our results is a radical departure from previous works on secure computation. To put it bluntly, *we decidedly give up on provable security reductions to any well-studied assumption*. Instead, our protocols are based on the conjectured hardness of decoding new, *heuristically designed* linear codes (or, equivalently, the hardness of a new learning parity with noise (LPN) variant). Our approach for building these new linear codes is much closer in spirit to the de facto standard approach for building efficient block ciphers and hash functions in symmetric cryptography: using a general framework that encompasses essentially all known attacks on LPN and syndrome decoding, we identify the core properties that guarantee resistance of our new assumptions against existing attacks. Then, we extract a number of fundamental design criteria which guide

the design of codes with these properties. Eventually, we rely on these design criteria together with extensive simulations to experimentally identify, with good confidence, the codes that exhibit the best properties for our constructions, while plausibly leading to very hard instances of the syndrome decoding problem.

1.2 Philosophy of our Approach

The construction of a cryptographic primitive or protocol can follow two complementary design strategies: a *top-down* approach, which starts from well-established cryptographic assumptions and aims at finding the most efficient construction whose security provably reduces to these assumptions, or a *bottom-up* approach, which tries to find the minimal construction that resists all known attacks, and relies on heuristic design criteria to build an intuition about the concrete security. Traditionally, secure computation has focused on the former, while symmetric cryptography (e.g. block cipher design) followed the latter.

The top-down approach has many attractive features – it deepens our theoretical understanding of the feasibility of cryptographic primitives, enlightens their relation to other primitives, and allows us to spend cryptanalytic efforts on a small set of assumptions. However, this sometimes comes at a huge cost in terms of efficiency: there is often a large gap between the best efficiency which can be achieved from well-established assumptions, and the efficiency which can be achieved with heuristic designs (consider the efficiency gap between SHA-256 and discrete-logarithm-based hash functions). When (our theoretical understanding of) a cryptographic primitive reaches a sufficient level of maturity, it is natural to envision the alternative bottom-up approach, in order to achieve real-world efficiency. This is the position that we advocate for in this work.

In the same way that symmetric cryptography has identified core families of attacks (e.g. linear and differential) and extracted a set of design principles for constructing primitives which plausibly resists them (e.g. substitution-permutation networks), our aim is to initiate the study of the most fundamental MPC primitives, oblivious transfer and its variants, under this angle. Pursuing this approach has the potential of yielding considerable efficiency improvements for MPC and strikes us as a natural next step for putting the efficiency of MPC primitives on par with that of symmetric primitives.

Our work being the first (to our knowledge) to study OT under the lens of heuristic cryptographic design, our constructions should of course be treated with the necessary caution. We invested a considerable effort in developing a rigorous understanding of which design criteria are likely to yield secure and efficient constructions, and relied on extensive experimental simulations to validate that our candidates satisfy these criteria; however, further study is welcomed in order to gain total confidence in their security. Given that Silver withstands the test of time, it will allow for significant improvements for numerous MPC protocols. And if not, we are confident that our analysis will motivate further constructions and analyses from which secure and efficient candidates will emerge.

1.3 Overview of our Methodology

Our starting point is the recent line of work on pseudorandom correlation generators (PCG) [BCG⁺17, BCGH18, BCG⁺19b]. PCGs allow to securely generate

long, pseudorandom correlated strings, using minimal communication. Among the most remarkable achievements of this line of work is *silent oblivious transfer extensions* (SOT extension) [BCG⁺19a, SGRR19]. These protocols have two phases: (1) the two parties interact to distributively generate short correlated seeds with communication/computation essentially independent of the target number of OTs; (2) the parties locally expand the seeds, without any interaction, into a large number of pseudorandom OTs. Afterwards, these OTs can be converted into chosen-input OTs using standard methods. Very recently, efficiency improvements were obtained by [YWL⁺20, WYKW20]. However, this came at the cost of sacrificing the silent feature. In practice, the ability to confine the bulk of the computation to an entirely offline phase, is a crucial efficiency feature.

The SOT Protocol of [BCG⁺19a]. Our approach builds upon the protocol of [BCG⁺19a]. Let us briefly recall its high level intuition:

1. the parties generate additive shares of $x \cdot \mathbf{e}$, where $x \in \mathbb{F}$ is known to the sender, and $\mathbf{e} \in \mathbb{F}^n$ is a random *sparse* vector, known to the receiver.
2. they multiply the shares of $x \cdot \mathbf{e}$ with a public matrix G , obtaining shares of $x \cdot \mathbf{a}$, for $\mathbf{a} = \mathbf{e} \cdot G^\top$. Given a uniform G , \mathbf{a} is pseudorandom under LPN.
3. Optionally, the shares can be hashed to generate pseudorandom OTs.

Generating additive shares of $x \cdot \mathbf{e}$ is extremely efficient, requiring merely two calls to AES for each entry of the vector. The matrix-vector multiplication, however, is the bulk of the computation: in [BCG⁺19a], G is a matrix over $\mathbb{F}_2^{k \times n}$, where k is the target number of OTs and $n = c \cdot k$ for some small constant $c > 1$. MPC protocols commonly require a number of OTs in the millions (if not more), making this step impractical unless G has some structure that allows for fast matrix-vector multiplication. This leads to a tradeoff between efficiency and confidence in the security: when H is a truly random matrix, the multiplication is impractical, but security reduces to the standard syndrome decoding/LPN assumption. Structured matrices give better efficiency, but security reduces to syndrome decoding variants which are less well-understood.

[BCG⁺19a] settled for a reasonable middle ground, by letting G be a random matrix with a quasi-cyclic structure. On the one hand, this structure allows for matrix-vector multiplication in quasilinear time using fast Fourier transform; on the other hand, the underlying assumption (hardness of decoding quasi-cyclic linear codes) has been used in candidate post-quantum code-based cryptographic primitives submitted to the NIST competition, and are therefore relatively well studied. While this choice leads to a reasonably efficient construction, it remains somewhat unsatisfying: it seems very likely that there exists alternative choices for G which have significantly better efficiency, yet still are secure.

However, the particular set of constraints of silent OT extension is very different from all previous coding theory primitives: typically the dimension of the code is minimized, allow high noise rate, and rely on codes with a hidden structure to enable efficient decoding given a secret. In contrast, in the SOT application, the code dimension scales with the target number of OTs (hence typically

millions), the noise rate must remain very low, and no hidden structure or efficient decoding property is required. As a result, there exists no well-established assumption regarding codes tailored for our unusual set of constraints.

Our Approach: a Design Methodology for Constructing G . In this work, we choose to approach the problem differently. Let us call a public matrix $G \in \mathbb{F}_2^{k \times n}$ *SOT-friendly* if it satisfies the following two properties:

- Security: it is infeasible to distinguish $\mathbf{e} \cdot G^\top$ from uniform (for sparse \mathbf{e}).
- Efficiency: the mapping $\mathbf{x} \rightarrow \mathbf{x} \cdot G^\top$ can be computed in *strict linear time*.

We develop a methodology for constructing SOT-friendly matrices by directly identifying some core structural properties of G which guarantee that distinguishing $\mathbf{e} \cdot G^\top$ from random cannot be done using essentially all known attacks on LPN and code-based cryptographic primitives. Yet the mapping $\mathbf{x} \rightarrow \mathbf{x} \cdot G^\top$ can be computed in strict linear time. Our methodology does not “start from zero”: it builds upon well-known results related to breaking these assumptions.

1.4 Our Design Criteria

The first property can be stated in one sentence: *G should generate a code with large minimum distance.* For the second property, we focus on the following sufficient condition: we restrict our attention to matrices G which have a *sparse parity-check matrix* H (i.e., H is a sparse matrix in $\mathbb{F}_2^{m \times n}$ such that $H^\top G = 0$) such that H are in *approximate lower triangular form*.

Large minimum distance and security. Given a matrix G , the problem of distinguishing $\mathbf{e} \cdot G^\top$ from random (for a random sparse vector \mathbf{e}) is the *decisional syndrome decoding problem* with respect to G^\top . The name LPN is commonly used to denote the syndrome decoding assumption in the cryptographic community. As such, we will use both terms interchangeably. It is well-known that distinguishing $\mathbf{e} \cdot G^\top$ reduces to the following problem: given a parity-check matrix H of G , distinguish the distribution $\{\mathbf{b} = \mathbf{x} \cdot H + \mathbf{e}\}$ (where \mathbf{x} is a uniformly random vector over \mathbb{F}_2^m and \mathbf{e} is a random length- n sparse vector) from the uniform distribution (indeed, if \mathbf{b} is indistinguishable from random, then so is $\mathbf{b} \cdot G^\top = (\mathbf{x} \cdot H + \mathbf{e}) \cdot G^\top = \mathbf{e} \cdot G^\top$), which is the learning parity with noise assumption, with dimension n and number of samples m , for the code matrix H . Both LPN and the syndrome decoding problem have been heavily studied in the past decades, and many attacks have been developed. A core observation (which is folklore, and was made explicitly e.g. in [BCG⁺20]) is that essentially all known attacks share a common high level structure: *the distinguisher computes a linear function in the samples \mathbf{b}* (but can depend arbitrarily on the matrix H). But if the code generated by G has large minimum distance d , the distribution $H \cdot \mathbf{x}$ for random \mathbf{x} must be d -wise independent, which implies that no weight- $t \leq d$ linear function $\mathbf{v}^\top \cdot \mathbf{b}$ of $\mathbf{b} = \mathbf{x} \cdot H + \mathbf{e}$ can possibly distinguish it from random. However, if \mathbf{v} has high weight, then the distribution of $\mathbf{v}^\top \cdot \mathbf{e}$ for a random sparse vector \mathbf{e} is close to uniform, and so is $\mathbf{v}^\top \cdot \mathbf{b}$. In this work, we formalize this folklore observation, and use it to derive a concrete heuristic for choosing the parameters of an SOT-friendly matrix. Our concrete heuristic is the following:

If two codes have the same minimum distance & dimensions, their decision syndrome decoding problems likely have the same level of security.

Therefore, when choosing concrete parameters, we will use as a baseline the codes underlying well-studied syndrome decoding variants (e.g. random linear codes in syndrome decoding, or LDPC codes in Alekhnovich’s assumption [Ale03]) and set parameters to achieve the same minimum distance that these codes exhibit. We make two additional comments before moving on to the second property:

- In practice, it is generally very hard to compute the minimum distance of a family of codes. We will provide some efficient concrete choices where provable bounds exist. However, in our most efficient instantiations, we will instead rely on extensive simulations to analyze the minimum distance of the code family using an optimized *minimum distance estimator*, from which we will heuristically derive the minimum distance on large dimensions.
- In existing attacks against LPN/syndrome decoding, the number of noisy coordinates plays a crucial role. However, it has a small impact on the overall efficiency of the SOT: scaling the noise by some factor increases the (very small) amount of communication and computation in the first phase, but has no impact on the second phase. Therefore, even if our hypothesis turns out to be too aggressive, we can actually significantly increase the security level, by increasing the number of coordinates, at a minor cost.

Linear-time encodable LDPC codes. Low-density parity-check codes (LDPC) have a sparse parity-check H , were introduced in the seminal work of Gallager [Gal62], and are among the most well-studied objects in coding theory. Certain random LDPC codes are known to exhibit a good minimum distance [Gal62] and can be decoded efficiently. On the other hand, their encoding time (i.e., the time to evaluate the mapping $\mathbf{x} \rightarrow \mathbf{x} \cdot G$) grows quadratically with the dimension in general. Due to the transposition principle (Section 4), our linear map $\mathbf{x} \rightarrow \mathbf{x} \cdot G^T$ is efficient if and only if LDPC encoding $\mathbf{x} \rightarrow \mathbf{x} \cdot G$ is. Hence, finding LDPC codes whose generating matrix is SOT-friendly boils down to finding linear-time LDPC codes with large distance.

Achieving Fast Encoding and High Minimum Distance. Guided by the above, we therefore seek to construct new families of structured LDPC codes which simultaneously appear to achieve high minimum distance, yet can be encoded extremely efficiently with (our optimized variant of) the g -ALT encoder of Richardson and Urbanke [RU01] as presented in Section 4. Here, we use as a starting point the Tillich-Zémor (TZ) family of codes [TZ06]. TZ codes have appealing features in our setting: they provably achieve *almost* linear minimum distance, and can be encoded in linear time. However, their structure is also sub-optimal in our specific setting: their code is not cache friendly and has sublinear distance due to degree-2 variable nodes. In [TZ06], the presence of these degree-2 variable nodes is motivated by the fact that they allow for high performance iterative decoding. In contrast, our application does not require any decoding property whatsoever. Hence, in Section 6 we refine the TZ codes to tailor them to our setting, improving the concrete minimum distance and encoding time.

We achieve this by iteratively refining our design, using extensive simulations to track the presence of *bad local structures* which, when they show up, lead to worse minimum distance guarantees. We fine-tune the structure of the matrix to minimize the number of cache misses in the encoding algorithm, which have a significant performance impact. To fine-tune the best possible choices of parameters in the low cache-misses setting, we compute, for many randomly generated choices of parameters, the average minimum distance and worst-case minimum distance over 10,000+ random samples of the code matrix.

1.5 Efficiency

After performing this iterative sequence of refinements, we end up with a variety of candidate new LDPC codes, which we call *Silver codes*. We use our Silver codes to instantiate the code matrix in the silent OT extension protocol of [BCG⁺19a], which we also generalize to the setting of VOLE. We implemented Silver, our protocol for SILent Vole and oblivious transfER, using our most optimized code; our implementation is available at libOTe[Rin]. We compare Silver to the best existing OT extension protocols: the standard IKNP protocol [IKNP03], which remains to date the most efficient protocol in the “unlimited bandwidth” setting, the recent protocol of Yang et al. [YWL⁺20], which provides the best concrete performance in natural bandwidth settings (from 10Mbps to 5Gbps), and the silent OT extension protocol of Boyle et al. [BCG⁺19a], which is the most efficient protocol that enjoys the silent feature. When generating 10^7 OTs on one core of a standard laptop, our protocol requires only 300ms of computation and 122KB of communication. In comparison, IKNP requires 58% more computation and $\sim 1300\times$ more communication, [YWL⁺20] requires $\sim 4\times$ more computation and $\sim 14\times$ more communication, and [BCG⁺19a] requires $19\times$ more computation (since our protocol is essentially their SOT with a Silver code, the communication is identical). In a setting with 100Mbps of bandwidth, Silver is at least 50 times more efficient than IKNP even when ignoring all costs beyond those of communication, and at least $4\times$ and $19\times$ more efficient than [YWL⁺20] and [BCG⁺19a] respectively, even when ignoring all communication costs.

2 Preliminaries

Throughout the work we will use $[a, b]$ to denote the set $\{a, \dots, b\}$. $[n]$ is shorthand for $[1, n]$. $=$ will denote mathematical equality while $x := y$ denotes defining x to be equal to y . $|\mathbf{v}|$ denotes the Hamming weight of vector \mathbf{v} . Matrix and vector horizontal concatenation is denoted as $[X|Y]$. Due to space restriction, we defer preliminaries on the silent OT extension protocol of [BCG⁺19a] to Appendix ?? of the Supplementary Material.

2.1 Preliminaries on Bias

Definition 1 (Bias of a Distribution). Given a distribution \mathcal{D} over \mathbb{F}^n and a vector $\mathbf{u} \in \mathbb{F}^n$, the bias of \mathcal{D} with respect to \mathbf{u} , denoted $\text{bias}_{\mathbf{u}}(\mathcal{D})$, is equal to

$$\text{bias}_{\mathbf{u}}(\mathcal{D}) = |\mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[\mathbf{u}^\top \cdot \mathbf{x}] - \mathbb{E}_{\mathbf{x} \sim \mathcal{U}_n}[\mathbf{u}^\top \cdot \mathbf{x}]| = \left| \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}[\mathbf{u}^\top \cdot \mathbf{x}] - \frac{1}{|\mathbb{F}|} \right|,$$

where \mathcal{U}_n denotes the uniform distribution over \mathbb{F}^n . The bias of \mathcal{D} , denoted $\text{bias}(\mathcal{D})$, is the maximum bias of \mathcal{D} with respect to any nonzero vector \mathbf{u} .

Given t distributions $(\mathcal{D}_1, \dots, \mathcal{D}_t)$ over \mathbb{F}_2^n , we denote by $\bigoplus_{i \leq t} \mathcal{D}_i$ the distribution obtained by *independently* sampling $\mathbf{v}_i \xleftarrow{\$} \mathcal{D}_i$ for $i = 1$ to t and outputting $\mathbf{v} \leftarrow \mathbf{v}_1 \oplus \dots \oplus \mathbf{v}_t$. We will use the following bias of the exclusive-or (cf. [Shp09]).

Lemma 2. *Let $t \in \mathbb{N}$ be an integer, and let $(\mathcal{D}_1, \dots, \mathcal{D}_t)$ be t independent distributions over \mathbb{F}_2^n . Then $\text{bias}(\bigoplus_{i \leq t} \mathcal{D}_i) \leq 2^{t-1} \cdot \prod_{i=1}^t \text{bias}(\mathcal{D}_i) \leq \min_{i \leq t} \text{bias}(\mathcal{D}_i)$.*

Eventually, let $\text{Ber}_r(\mathbb{F}_2)$ denote the Bernoulli distribution that outputs 1 with probability r , and 0 otherwise. More generally, we denote by $\text{Ber}_r(\mathbb{F})$ the distribution that outputs a uniformly random element of \mathbb{F} with probability r , and 0 otherwise. We will use a standard simple lemma for computing the bias of a XOR of Bernoulli samples:

Lemma 3 (Piling-up lemma). *For any $0 < r < 1/2$ and any integer n , given n random variables X_1, \dots, X_n i.i.d. to $\text{Ber}_r(\mathbb{F}_2)$, it holds that $\Pr[\bigoplus_{i=1}^n X_i = 0] = 1/2 + (1 - 2r)^n/2$.*

2.2 Syndrome Decoding and Learning Parity with Noise

Our constructions will rely on new variants of the learning parity with noise (LPN) assumption (more accurately, a variant of the syndrome decoding assumption). The LPN assumption over a field \mathbb{F} states, informally, that no adversary can distinguish $(A, A \cdot \mathbf{s} + \mathbf{e})$ from (A, \mathbf{b}) , where A is sampled from the set of generating matrices of some linear code ensemble, \mathbf{s} is a uniform secret vector over \mathbb{F} , \mathbf{e} is a *noise vector* sampled from some distribution over \mathbb{F} -vectors and typically sparse. \mathbf{b} is a uniform vector over \mathbb{F} . More formally, we define the LPN assumption over a ring \mathcal{R} with dimension k , number of samples n , w.r.t. a code generation algorithm \mathbf{C} , and a noise distribution \mathcal{D} :

Definition 4 (Primal LPN). *Let $\mathcal{D}(\mathcal{R}) = \{\mathcal{D}_{k,n}(\mathcal{R})\}_{k,n \in \mathbb{N}}$ denote a family of efficiently sampleable distributions over a ring \mathcal{R} , such that for any $k, n \in \mathbb{N}$, $\text{Im}(\mathcal{D}_{k,n}(\mathcal{R})) \subseteq \mathcal{R}^n$. Let \mathbf{C} be a probabilistic code generation algorithm such that $\mathbf{C}(k, n, \mathcal{R})$ outputs a matrix $A \in \mathcal{R}^{n \times k}$. For dimension $k = k(\lambda)$, number of samples (or block length) $n = n(\lambda)$, and ring $\mathcal{R} = \mathcal{R}(\lambda)$, the (primal) $(\mathcal{D}, \mathbf{C}, \mathcal{R})$ -LPN(k, n) assumption states that*

$$\begin{aligned} \{(A, \mathbf{b}) \mid A \xleftarrow{\$} \mathbf{C}(k, n, \mathcal{R}), \mathbf{e} \xleftarrow{\$} \mathcal{D}_{k,n}(\mathcal{R}), \mathbf{s} \xleftarrow{\$} \mathbb{F}^k, \mathbf{b} \leftarrow A \cdot \mathbf{s} + \mathbf{e}\} \\ \stackrel{c}{\approx} \{(A, \mathbf{b}) \mid A \xleftarrow{\$} \mathbf{C}(k, n, \mathcal{R}), \mathbf{b} \xleftarrow{\$} \mathcal{R}^n\}. \end{aligned}$$

The above definition is very general, and captures in particular not only the standard LPN assumption and its variants, but also assumptions such as LWE or the multivariate quadratic assumption. However, we will typically restrict our attention to assumptions where the noise distribution outputs sparse vectors with high probability. The standard LPN assumption with dimension k , noise

rate r , and n samples is obtained by setting A to be a uniformly random matrix over $\mathbb{F}_2^{n \times k}$, and the noise distribution to be the Bernoulli distribution $\text{Ber}_r^n(\mathbb{F}_2)$, where each coordinate of \mathbf{e} is independently set to 1 with probability r and to 0 with probability $1 - r$. The term “primal” in the above definition comes from the fact that the assumption can come in two equivalent form: the primal form as above, but also a *dual form*: viewing A as the transpose of the parity check matrix H of a linear code generated by G a matrix, i.e. $A = H^\top$, the hardness of distinguishing $H^\top \cdot \mathbf{x} + \mathbf{e}$ from random is equivalent to the hardness of distinguishing $G \cdot (H^\top \cdot \mathbf{x} + \mathbf{e}) = G \cdot \mathbf{e} = \mathbf{e} \cdot G^\top$ from random (since $G^\top \cdot H = 0$).

3 On the Hardness of LPN for Structured LDPC Codes

The learning parity with noise assumption is one of the most fundamental assumptions of cryptography, introduced in the work of [BFKL94]; related problems were used even earlier [McE78]. The hardness of syndrome decoding and its variants (which is equivalent to LPN under our definition – see above) has also been intensely studied in coding theory, starting with the seminal work of Prange [Pra62] (under the name the of syndrome decoding), in learning theory (see e.g. [FGKP09] and references therein), and in random CSP theory (starting with the seminal work of Feige [Fei02]) – all with many follow ups.

Over the past few decades, a tremendous number of attacks against LPN have been proposed. These attacks include, but are not limited to, attacks based on Gaussian elimination and the BKW algorithm [BKW00, Lyu05, LF06, EKM17] and variants based on covering codes [ZJW16, BV16, BTV16, GJL20], information set decoding attacks [Pra62, Ste88, FS09, BLP11, MMT11, BJMM12, MO15, EKM17, BM18], statistical decoding attacks [AJ01, FKI06, Ove06, DAT17], generalized birthday attacks [Wag02, Kir11], linearization attacks [BM97, Saa07], attacks based on finding low weight code vectors [Zic17], or on finding correlations with low-degree polynomials [ABG⁺14, BR17].

A Unified Framework for Attacks against LPN. In light of this situation, it would be excessively cumbersome, when introducing a new variant of LPN, to go over the entire literature of existing attacks and analyze their potential impact on the new variant. The crucial observation, however, is that this is not necessary, as *all the above attacks* (and more generally, essentially all known attacks against LPN and its variants) fit in a common framework, usually denoted the *linear test framework*. Furthermore, the asymptotic resistance of any LPN variant against any attack from the linear test framework can be deduced from two simple properties of the underlying code ensemble and noise distribution. Informally, if

- the code generated by G has high minimum distance, and
- for any large enough subset S of coordinates, with high probability over the choice of $\mathbf{e} \leftarrow \mathcal{D}$, at least one of the coordinates in S of \mathbf{e} will be nonzero,

then the LPN assumption with code matrix G and noise distribution \mathcal{D} cannot be broken by any attack from the linear test framework. We will formalize this and build on it to analyze the asymptotic security of our new LPN variants.

We stress that this crucial observation is not new to our work: a similar observation was explicitly made in previous works [ADI⁺17, BCG⁺20], where it was also used to analyze the security of new LPN variants. Even long before these works, distributions whose outputs look random to linear tests, called *low-bias sample spaces*, have been the subject of a rich and fruitful line of work which was initiated in the seminal work of Naor and Naor [NN90], and the relevance of linear tests to the security analysis LPN assumptions seems to have been at least somewhat folklore. Still, we believe that it will be beneficial and instructive to the reader to present this argument in a unified way with explicit bounds.

3.1 The Linear Test Framework

The common feature of essentially all known attacks against LPN and its variants is that the distinguisher can be implemented as a (nonzero) *linear function of the samples* (the linear test), where the coefficients of the linear combination can depend arbitrarily on the code matrix. Therefore, all these attacks can be formulated as distinguishing LPN samples from random samples by checking whether the output of some linear test (with coefficients depending arbitrarily on the code matrix) is biased away from the uniform distribution. Formally,

Definition 5 (Security against Linear Test). *Let \mathbb{F} be an arbitrary finite field, and let $\mathcal{D} = \{\mathcal{D}_{m,n}\}_{m,n \in \mathbb{N}}$ denote a family of noise distributions over \mathbb{F}^n . Let \mathbf{C} be a probabilistic code generation algorithm such that $\mathbf{C}(m,n)$ outputs a matrix $A \in \mathbb{F}^{n \times m}$. Let $\varepsilon, \delta : \mathbb{N} \mapsto [0, 1]$ be two functions. We say that the $(\mathcal{D}, \mathbf{C}, \mathbb{F})$ -LPN(m, n) assumption with dimension $m = m(\lambda)$ and $n = n(\lambda)$ samples is (ε, δ) -secure against linear tests if for any (possibly inefficient) adversary \mathcal{A} which, on input a matrix $A \in \mathbb{F}^{n \times m}$, outputs a nonzero $\mathbf{v} \in \mathbb{F}^n$, it holds that*

$$\Pr[A \stackrel{\$}{\leftarrow} \mathbf{C}(m, n), \mathbf{v} \stackrel{\$}{\leftarrow} \mathcal{A}(A) : \text{bias}_{\mathbf{v}}(\mathcal{D}_A) \geq \varepsilon(\lambda)] \leq \delta(\lambda),$$

where \mathcal{D}_A denotes the distribution induced by sampling $\mathbf{s} \stackrel{\$}{\leftarrow} \mathbb{F}_2^m$, $\mathbf{e} \leftarrow \mathcal{D}_{m,n}$, and outputting the LPN samples $A \cdot \mathbf{s} + \mathbf{e}$.

The following observation is folklore, and was made explicitly e.g. in [BCG⁺20]:

Observation 1 *Existing attacks against LPN (as listed above) can be cast as instances of the linear test framework. Therefore, none of these attacks can provide a polynomial-time distinguisher against any LPN assumption that is provably (ε, δ) -secure against linear tests, for any negligible functions (ε, δ) .*

[ADI⁺17] went even further and explicitly conjectured that for any LPN variant with a sparse code matrix, the runtime of the best possible attack against LPN is essentially $\text{poly}(1/\varepsilon)$, *i.e.*, the number of times a linear test attack must be repeated until the bias becomes noticeable. See [Assumption 1](#).

3.2 Dual Distance and Security against Linear Tests

Following [ADI⁺17], we call *dual distance* of a matrix M , and write $\text{dd}(M)$, the largest integer d such that every subset of d rows of M is linearly independent. The name “dual distance” stems from the fact that the $\text{dd}(M)$ is also the minimum distance of the dual of the code generated by M (*i.e.*, the code generated by the left null space of M). The following lemma is folklore:

Lemma 6. Let $\mathcal{D} = \{\mathcal{D}_{m,n}\}_{m,n \in \mathbb{N}}$ denote a family of noise distributions over \mathbb{F}^n . Let \mathbf{C} be a probabilistic code generation algorithm s.t. $\mathbf{C}(m, n) \rightarrow A \in \mathbb{F}^{n \times m}$. Then for any $d \in \mathbb{N}$, the $(\mathcal{D}, \mathbf{C}, \mathbb{F})$ -LPN(m, n) assumption with dimension $m = m(\lambda)$ and $n = n(\lambda)$ samples is $(\varepsilon_d, \delta_d)$ -secure against linear tests, where

$$\varepsilon_d = \max_{|\mathbf{v}| > d} \text{bias}_{\mathbf{v}}(\mathcal{D}_{m,n}), \quad \text{and} \quad \delta_d = \Pr_{A \leftarrow \mathbf{C}(m,n)} [\text{dd}(A) \geq d].$$

Proof. The proof is straightforward: fix any integer d . Then with probability at least δ_d , $\text{dd}(A) \geq d$. Consider any (possibly unbounded) adversary \mathcal{A} outputting \mathbf{v} . Two cases can occur:

- Either $|\mathbf{v}| \leq d \leq \text{dd}(A)$. In this case, the bias with respect to \mathbf{v} of the distribution $\{A \cdot \mathbf{s} \mid \mathbf{s} \leftarrow \mathbb{F}^m\}$ is 0 (since this distribution is d -wise independent). Since the bias of the XOR of two distribution is at most the smallest bias among them (see [Lemma 2](#); the same holds for the bias with respect to any fixed \mathbf{v}), we get $\text{bias}(\mathcal{D}_A) = 0$.
- Or $|\mathbf{v}| > d$; in which case, applying [Lemma 2](#) again, $\text{bias}(\mathcal{D}_A) \leq \text{bias}(\mathcal{D}_{m,n})$.

Security of LPN with random codes. An instructive example is to consider the case of LPN with a uniformly random code matrix over \mathbb{F}_2 , and a Bernoulli noise distribution $\mathcal{D}_{m,n} = \text{Ber}_r^n(\mathbb{F}_2)$, for some noise rate r . The probability that d random vectors over \mathbb{F}_2^m are linearly independent is at least

$$\prod_{i=0}^{d-1} \frac{2^m - 2^i}{2^m} \geq (1 - 2^{d-1-m})^d \geq 1 - 2^{2d-m}.$$

Therefore, by a union bound, the probability that a random matrix $A \leftarrow \mathbb{F}_2^{n \times m}$ satisfies $\text{dd}(A) \geq d$ is at least $1 - \binom{n}{d} \cdot 2^{2d-m} \geq 1 - 2^{(2+\log n)d-m}$. On the other hand, for any d and any \mathbf{v} with $|\mathbf{v}| > d$, we have by [Lemma 3](#):

$$\Pr[\mathbf{e} \leftarrow \text{Ber}_r^n(\mathbb{F}_2) : \mathbf{v}^\top \cdot \mathbf{e} = 1] = \frac{1 - (1 - 2r)^d}{2},$$

hence $\text{bias}_{\mathbf{v}}(\text{Ber}_r^n(\mathbb{F}_2)) = (1 - 2r)^d \leq e^{-2rd}$. In particular, setting $d = O(m/\log n)$ suffices to guarantee that with probability at least $\delta_d = 1 - 2^{-O(m)}$, the LPN samples will have bias (with respect to any possible nonzero vector \mathbf{v}) ε_d at most $e^{-O(rm/\log n)}$. Hence, any attack that fits in the linear test framework against the standard LPN assumption with dimension m and noise rate r requires of the order of $e^{O(rm/\log n)}$ iterations. Note that this lower bound still leaves a gap with respect to the best known linear attacks, which require time of the order of $e^{O(rm)}$, $e^{O(rm/\log \log m)}$, and $e^{O(rm/\log m)}$ when $n = O(m)$, $n = \text{poly}(m)$, and $n = 2^{O(m/\log m)}$ respectively [[BKW00](#), [Lyu05](#), [EKM17](#)].

3.3 SOT from Asymptotically Good Linear-Time Encodable Codes

Abstracting out the unnecessary details, recall that the construction of silent oblivious transfer extension introduced in [[BCG⁺19b](#), [BCG⁺19a](#)] and recalled

in Appendix ??, relies on the following assumption: given a large public matrix $G \in F_2^{k \times n}$, is such that $n = c \cdot k$ for some small constant $c > 1$ (e.g. $c = 2$), it should be infeasible to distinguish $\mathbf{e} \cdot G^\top$ from random, where \mathbf{e} is a uniformly random weight- t vector. This corresponds to the dual-LPN assumption, which is equivalent to the primal-LPN assumption with matrix $H \in F_2^{m \times n}$, where H is the parity check for generator G ; *i.e.*, $G^\top \cdot H = 0$.

A selection principle for LPN with structured code. Based on the previous discussions, for any linear code ensemble \mathbf{C} which outputs matrices $H \stackrel{\$}{\leftarrow} \mathbf{C}$ having a large distance w.h.p., it is reasonable to conjecture that the corresponding primal-LPN assumption will hold (since a contradiction would imply a fundamentally different type of attack than existing ones). This conjecture was formally stated in [ADI⁺17] for the case of all *sparse* code ensembles:

Assumption 1 (Assumption 6 in [ADI⁺17]) *For every prime-order field \mathbb{F} , every polynomial $m(\lambda), n(\lambda)$, every constant t , every real $r \in (0, 1/2)$, and every t -sparse matrix $A \in \mathbb{F}^{n \times m}$, the following holds: Any circuit of size $T = \exp(\Omega_r(\text{dd}(A)))$ cannot distinguish $(A \cdot \mathbf{s} + \mathbf{e})$ for $\mathbf{s} \stackrel{\$}{\leftarrow} \mathbb{F}_2^m, \mathbf{e} \leftarrow \text{Ber}_r(\mathbb{F})$ from the uniform distribution with advantage better than $1/T$ ($\Omega_r(x)$ denotes $\Omega(x)$, where the hidden constant may depend on the noise rate r).*

Noise weight versus minimum distance. The above discussions allows to make a simple, yet powerful observation: for typical noise distributions, including the Bernoulli distribution with parameter t/n , the *regular* noise distribution (concatenations of t length- n/t unit vectors), and the *exact* noise distribution (random t -sparse vectors), the running time of linear attacks is lower bounded by a term of the form $e^{c \cdot rd}$ for some constant c , where $r = t/n$ is the noise rate and d is the minimum distance. This suggests the following *safeguard*: if a SOT code exhibits a much worse typical minimum distance behavior than estimated (which in our case would be very surprising but theoretically possible), say, the true distance d is v times shorter than estimated, then same conjectured security level as before can be obtained by *scaling the number of noisy coordinates t by a factor v* . Crucially, in our SOT construction, the impact of this scaling vanishes when the number of OTs is large: it only impacts the complexity of distributing the seed (which increases by a factor v), but has no influence whatsoever, neither on the matrix multiplication part (which is the bulk of the computation) nor on the sparse vector expansion (which is the only other component whose cost scales with the target number of OTs).

Our approach: structured LDPC codes. Asymptotically good families of linear-time encodable codes have been studied in the literature, with probabilistic constructions given in [GDP73, Spi96]. However, these works only targeted asymptotic efficiency. Our aim, on the other hand, is to focus on concrete efficiency, and to find codes with a large concrete minimum distance, and extremely efficient encoding. We choose to focus on structured families of LDPC codes (*i.e.*, codes whose parity-check matrix is sparse), which have been widely studied in the coding theory literature. Our rationale is based on the following observations:

- Most LDPC codes have linear minimum distance;
- Some structured families of LDPC codes admit efficient encoding algorithms;
- Some structured families of LDPC codes provably achieve both fast linear time encoding and *almost* linear minimum distance;
- Structured families of LDPC codes in the literature which do *not* exhibit linear or close-to-linear minimum distance typically satisfy a specific constraint: their Tanner graph contains a large number of degree-2 variable nodes. In contrast, we suggest candidates which admit extremely fast encoding, but do *not* exhibit this structural weakness, and can be experimentally verified to exhibit a very good minimum distance growth.
- For *random* LDPC codes, the corresponding assumption (primal LPN with a random sparse code matrix) is the Alekhovich assumption [Ale03], an important and well-studied assumption.

3.4 Most Sparse Matrices Have Linear Dual Distance

In this section, we show that for any integer $t > 2$, most matrices in $\mathbb{F}_2^{n \times m}$ with rows of Hamming weight t have dual distance linear in m ; more precisely, the fraction of such matrices with dual distance at least $\gamma \cdot k$ (for some constant γ) is at least $1 - m^{2.1-t}$. In coding-theoretic terms, it says that most *column-regular* LDPC codes have linear minimum distance, where the parity check matrix has fixed column weight. Let $W_t(\mathbb{F}_2^n)$ denote the set of all vectors in \mathbb{F}_2^n with Hamming weight exactly t ; we also denote by $W_t(\mathbb{F}_2^{n \times m})$ the set of all matrices in $\mathbb{F}_2^{n \times m}$ with exactly t ones per column.

Theorem 7 (Most sparse matrices have dual distance $O(m)$). *For any constant $c > 1$ and integer $t > 2$, there is a constant $\gamma = \gamma(c, t)$ such that for any large enough m , denoting $n = c \cdot m$,*

$$\Pr \left[A \xleftarrow{\$} W_t(\mathbb{F}_2^{n \times m}) : \frac{\text{dd}(A)}{m} < \gamma \right] \leq 1 - m^{2.1-t}.$$

For completeness, we provide the proof in Appendix ?? of the Supplementary Material; the proof is a direct adaptation to our setting of the analysis of [MST03, Section 5.3]. Building upon our analysis, we also make a key observation: random LDPC codes over *large fields* have linear minimum distance with high probability, *even when their parity-check matrix is randomly sampled with $\{0, 1\}$ entries*. We discuss the implications of this observation for one of our applications, as well as its relation to previous assumptions from the literature, in Appendix ??.

4 Fast LDPC Encoding

We begin with an overview of how to perform fast encoding of LDPC codes by leveraging the sparsity/structure of H . Let us first review the naive encoding method. Recall H defines the code $\mathcal{C} = \{\mathbf{c} \mid H\mathbf{c}^\top = 0\}$. As such, define the systematic form H' for the same code by performing elementary row operations on H to obtain $H' = [-P^\top \mid I_{n-k}]$. Since elementary row operations do not change the null-space, we have $\mathcal{C} = \{\mathbf{c} \mid H\mathbf{c}^\top = 0\} = \{\mathbf{c} \mid H'\mathbf{c}^\top = 0\}$. Although

H is sparse, $P \in \mathbb{F}^{k \times m}$ is likely dense. Let $G := [I_k | P]$ be the symmetric form generator and then encoding can be achieved by computing $\mathbf{c} := \mathbf{x}G$ for $\mathbf{x} \in \mathbb{F}^k$. The cost of this is $O(n^3)$ time to compute P and $O(n^2)$ time to compute $\mathbf{x}P$.

However, we can also use the fact that $H\mathbf{c}^\top = 0$ to encode \mathbf{x} into \mathbf{c} . Recall that $\mathbf{c} = \mathbf{x}G = [\mathbf{x} | \mathbf{c}']$ for $\mathbf{c}' := \mathbf{x}P$. Therefore we can rewrite this as

$$0 = H\mathbf{c}^\top = H[\mathbf{x} | \mathbf{c}']^\top = T\mathbf{x}^\top + S\mathbf{c}'^\top \quad \iff \quad -T\mathbf{x}^\top = S\mathbf{c}'^\top$$

where $H = [T | S]$ and $T \in \mathbb{F}^{m \times k}, S \in \mathbb{F}^{m \times m}$. Given \mathbf{x} , we can compute $\mathbf{y} := -T\mathbf{x}^\top$ in $O(k)$ time since T is sparse. We then solve the sparse system $\mathbf{y} = S\mathbf{c}'^\top$. Using Gaussian elimination, this would naively require $O(m^3) = O(n^3)$ time. However, we can try to leverage the sparsity of H to achieve better efficiency.

Our starting point is the somewhat standard LDPC solving technique known as g -Approximate Lower Triangularization (g -ALT) [RU01, DP15, KS12]. The basic intuition is that this system can be solved in linear time if S is a lower triangular matrix. In particular, the entries along the diagonal should all be set to one. Later we will discuss how to ensure this is the case. The system can be solved by solving each row “independently” starting with row 1 and working down. This idea can be generalized to allow all but the last g rows of H to be triangular (see Figure ?? in Appendix ??). The last g rows are said to be part of the *gap*. As discussed in Appendix ??, a parity check matrix with this form allows for encoding \mathbf{x} as $\mathbf{c} = \mathbf{x}G$ in $O(n + g^2)$ time. Therefore, this remains linear so long as $g = O(\sqrt{n})$. Additionally, we present an optimization in Appendix ?? which reduces this to $O(n)$ at the expense of $O(g)$ communication in the protocol.

Recall that in dual LPN we wish to compute $\mathbf{u} := \mathbf{e} \cdot G^\top$ which is equivalent to primal LPN where $\mathbf{u} := \mathbf{x} \cdot H + \mathbf{e}$. Yet, the encoding algorithm described above is for computing $\mathbf{x} \cdot G$. By the transposition principle [Bor57, IKOS08], we can achieve our goal at effectively the same cost. Roughly, the transformation works by first expressing the circuit which computes $\mathbf{x} \cdot G$ as a series of matrix multiplication, $\mathbf{s}_1 := M_1 \cdot \mathbf{x}, \mathbf{s}_2 := M_2 \cdot \mathbf{s}_1, \dots, \mathbf{e} := M_n \cdot \mathbf{s}_{n-1}$ such that \mathbf{e} is the final output. Any circuit can be expressed in this way. Then $\mathbf{e} \cdot G^\top$ can be computed as $\mathbf{s}_{n-1} := M_n^\top \cdot \mathbf{e}, \mathbf{s}_{n-2} := M_{n-1}^\top \cdot \mathbf{s}_{n-1}, \dots, \mathbf{x} := M_1^\top \cdot \mathbf{s}_1$. Refer to Appendix ?? for a detailed description of all the algorithms discussed in this section.

5 Estimating the Minimum Distance Empirically

Crucial to our construction is the ability to accurately determine the minimum distance of the LDPC matrix H that is employed. Computing the exact minimum distance is known to be NP-Complete [Var97] and typically infeasible for our parameter region, e.g. $n = 2^{20}$. For some LDPC distributions, it is possible to derive an asymptotic bound on the minimum distance; however, many of these have drawbacks in efficiency or distance.

To overcome this, we resort to computational approaches for estimating the minimum distance of an LDPC code ensemble. For relatively small values of n , say less than 200, we compute the exact minimum distance using the approach presented in [HIQO19]. For larger n , say less than 4000, we fall back to a standard heuristic, the noisy impulse method of [BVJD02], for upper bounding the

minimum distance (which we have verified does in fact closely agree with exact minimum distance for smaller values of n). We then extrapolate the asymptotic behavior of the minimum distance for larger values of n .

5.1 Exact Minimum Distance

For computing the exact minimum distance we make use of the so-called Brouwer-Zimmerman algorithm as described in [Gra06], and implemented in [HIQO19]. Loosely speaking, this approach iteratively refines a lower and upper bound until they are equal. First, the generator matrix G is placed in systematic form $G' = [I_k | P]$. Recall that for all $\mathbf{x} \in \mathbb{F}^k \setminus \{0\}$, the corresponding codeword is $\mathbf{c} = [\mathbf{x} | \mathbf{x}P]$ and therefore clearly $|\mathbf{c}| \geq |\mathbf{x}|$. Using this observation, the algorithm proceeds by initializing the lower bound $\ell = 1$ and upper bound $u = m + 1$. All x with $|\mathbf{x}| = \ell$ are encoded as $\mathbf{c} = \mathbf{x}G$ and the upper bound u is replaced as the minimum weight over all codewords considered. While $u \neq \ell$, ℓ is incremented and the process is repeated. See [Gra06, HIQO19] for details.

The running time remains exponential in the size of the code. With careful optimizations, the implementation of [HIQO19] is capable of computing the minimum distance up to about $n = 160$. Since this is relatively small compared to the codes our protocol employs, this approach is primarily used to validate the accuracy of the so-called *noisy impulse method* which we describe next.

5.2 Upper Bounding the Minimum Distance

Our second approach for evaluating the minimum distance of a LDPC family is known as the noise impulse method [BVJD02]. Very roughly speaking, this approach tries to decode the zero codeword when one or more of the bits have been flipped. The intuition is that if the right bits are flipped, then the next closest codeword will correspond to a close-to-minimum weight codeword.

In more details, and including improvements from [XFE04], this approach considers all vectors $\mathbf{c} \in \{0, 1\}^n$ with $|\mathbf{c}| \leq w$ for some small constant w , e.g. 1 or 2. Each \mathbf{c} is input into a belief propagation decoder, typically Min-Sum, which output the decoders estimates on the likelihood that each bit of \mathbf{c} should be error corrected to zero or one. Since at most w bits in \mathbf{c} are one and the actually minimum distance d is almost certainly more than twice w , the most likely codeword will in fact be the original all zero codeword.

However, the likelihood information contained in the decoder output can be leveraged to aid in the search of nearby non-zero codewords. Loosely speaking, belief propagation (BP) decoders work by assigning each bit of \mathbf{c} a likelihood of being zero or one and updating these likelihoods in an iterative process. The initial likelihood values could be that the decoder is 95% certain that each bit is as specified by \mathbf{c} , i.e. an error rate of 0.05. Intuitively, at each iteration the likelihood information for each bit of \mathbf{c} is updated based on how many of the corresponding parity checks pass or fail. An interpretation of this is that it reduces the likelihood values for the zero positions of \mathbf{c} when they are closely related to the positions of \mathbf{c} which were set to one.

The idea is then to sort the positions of \mathbf{c} such that the positions which are most confidently zero are to the right. The same permutation is applied to the

columns of G . Partial Gaussian elimination is applied to G s.t. the left $k \times k$ submatrix is lower triangular with ones along the diagonal. Some of the first k columns are likely linearly dependent, preventing us from making the left k columns of G lower diagonal. In this case, the dependent columns are permuted right and replaced with the next left most column of G . We then consider all (permuted) codewords with the form $\mathbf{c}^* = [\mathbf{c}_1 | \mathbf{c}_2 | 0^{n-k-t}]$ where $\mathbf{c}_2 \in \{0, 1\}^t$ has some maximum weight u , e.g. $t = 50, u = 10$. For each choice of \mathbf{c}_2 , it is possible to compute $\mathbf{c}_1 \in \{0, 1\}^k$ via the left lower diagonal submatrix of G . The estimated upper bound the distance as the minimum weight over all \mathbf{c}^* .

Table 1: The minimum d_{\min} , average d_{avg} and maximum d_{\max} minimum distance obtained over 100 trials for weight 5 uniform LDPC codes.

m	method	w	d_{\min}	d_{avg}	d_{\max}
20	impulse	1	2	5	6
	exact	-	2	5	6
40	impulse	1	4	8.3	10
	exact	-	4	8.3	10
60	impulse	1	8	11.44	14
	impulse	2	8	11.38	14
	exact	-	8	11.38	14
80	impulse	1	12	16.48	20
	impulse	2	12	14.86	18
	exact	-	12	14.86	18
100	impulse	1	16	22.28	26
	impulse	2	14	18.2	20

6 Code Design

Designing an efficient LDPC code for large dimension LPN offers many unique challenges. Our primary two design goals are to achieve large minimum distance, ideally linear in n , and linear time encoding. However, unlike many existing codes from the coding community, we do not care about its decoding performance or other error correcting properties. All our codes have rate $1/2$, i.e. $n/2 = m = k$.

In this section we review two existing LDPC codes, namely uniform and Tillich-Zémor Codes. After describing various benefits and drawbacks of each, we design a new highly efficient LDPC code which achieves an extremely fast linear encoding time and plausibly linear minimum distance.

6.1 Uniform LDPC

As described in [Section 3.4](#), the family $W_t(\mathbb{F}_2^{n \times m})$ of uniform LDPC codes with fixed column weight t are known to have linear minimum distance with good probability. We consider the family of codes parameters by $t \in \{5, 11\}$. While the theoretical bound applies to all $t > 2$, we observe that $t = 3$ experiences very poor concrete minimum distance performance and often do not correspond to a code that can be made systematic. For $t = 5$ we observe a concrete linear minimum distance growth rate of $d_{\text{avg}} = 0.28m, d_{\min} = 0.19m$ over 100 trials. These growth rates were obtained for $n \in [200, 800]$. Since we are interested in the worse case

performance, we are mostly interested in d_{\min} . By increasing the weight to $t = 11$ we obtain a minimum distance growth rate of $d_{\text{avg}} = 0.38m$, $d_{\min} = 0.36m$.

The hardness of syndrome decoding for uniform LDPC codes was the basis of recent proposals [BCGI18, YWL⁺20], and corresponds to the well-established Alekhnovich assumption [Ale03]. While these codes turn out to be inappropriate efficiency-wise in our setting (see below), we will rely on the following heuristic to select the concrete parameters of our new codes: when we experimentally observe, with high confidence, that a distribution over codes achieves a similar average minimum distance, with a similar variance, compared to uniform LDPC codes, we heuristically estimate that the corresponding assumptions should provide a comparable level of hardness. We note that, if it turns out that this heuristic is too optimistic (which would intuitively require finding new attacks radically different from all known attacks), increasing the noise (as described in Section 3.3, Noise weight versus minimum distance) can be used to adjust the hardness level of the underlying assumption without significantly harming efficiency.

Shortcoming of uniform LDPC codes. The choice of basing security on the hardness of decoding uniform LDPC codes was motivated in previous works [BCGI18, YWL⁺20] by the fact that they correspond to the relatively well-established Alekhnovich assumption. However, they turn out to be a relatively poor choice in our setting. At a high level, the reason is that for distributions over random LDPC codes which do not enforce any particular structure beyond guaranteeing some conditions on the number of ones per row and column (*i.e.*, which sample the parity-check matrix uniformly conditioned on constraints on the fractions of variable and check nodes from the Tanner graph which must have a given degree), having a high minimum distance with good probability, and being linear-time encodable with the g -approximate lower triangularization algorithm, appear to be at odd, according to a conjecture of Richardson and Urbanke [RU01] (we will discuss this conjecture in more details in Section 6.2). This effectively justifies moving towards *structured* ensembles of LDPC codes, which also enforce some structure on the *shape* of the parity-check matrix. A prime example of codes achieving a sweet spot between having high minimum distance with good probability, and very fast encoding, is given by the Tillich-Zémor code ensemble.

6.2 Tillich-Zémor Codes

As discussed before, in order to design codes that have efficient encoders as well as good minimum distance, one must move away from random codes and consider more structured codes. As such, structured codes would offer an immediate handle on efficient encoding purely by design. On the other hand, this approaches leaves much for the desire of a more rigorous theoretical understanding of the minimum distance of such structured codes. Such questions have been posed in the past amongst the members of the coding theory and communications communities. One such work is that of Tillich and Zémor [TZ06]. They investigate the minimum distance of structured LDPC codes with two variable nodes of degree-2 per parity-check equation. In the design of LDPC codes with high iterative decoding performance the variable nodes of degree-2 play a very important

to ensure linear encoding complexity is different from the works described in this section, since we actually have zero columns with weight 2. Thus, none the sufficient conditions for sub-linear minimal distance described above are satisfied by our codes. Based on these observations, we conclude that our codes do not provably have sub-linear minimal distance. We leave open the task of formally proving claims regarding the minimum distance of our codes.

6.3 LDPC Silver Codes

We now present our new LDPC constructions, which we dub Silver Codes (codes for SILEnt Vole and oblivious transFER). The goal of these codes is to obtain (plausible) linear minimum distance and extremely efficient encoding. Unlike in the traditional setting, our codes need to perform well (encoding-wise) when n is on the order of millions (but do not need to admit efficient decoding algorithms). Ideally our code would have a very compact representation. If a large preprocessing/sampling procedure must be performed, then the codes will likely need to be stored in memory, possibly requiring more memory than the rest of the protocol. Therefore we aim to design codes with a very succinct description.

Our second goal is to have a very efficient memory access structure. Recall that the encoding algorithm will have to access “memory locations” j and i whenever there is a 1 located at $H_{j,i}$. Therefore we would ideally like H to have some additional structure which maintains some memory locality. For example, having a bounded distance between sequential memory accesses. In the case of TZ codes, for example, the left matrix is uniformly distributed, which significantly harms the performances in terms of memory access. When n is on the order of millions, performing random access into an array of length n can quickly dominate the running time as we will see in [Section 7](#).

Despite this shortcoming, we take TZ as our starting point and iteratively improve it (sacrificing decoding performance, but trying to optimize minimum distance and encoding time) with the (heuristic) guidance of our minimum distance estimators. It will be useful to partition H into left and right halves $[L|R] := H$ which are each of size $m \times m$. For TZ, L is therefore a uniform column/row weight t matrix while R has column/row weight 2 where all ones are effectively on a diagonal band. Recall that we only consider rate 1/2 codes where $k = m = n/2$.

It is also a well known phenomenon that odd column weight t LDPC codes achieve better minimum distance performance (for examples, the bounds on the minimum distance achieved in [\[TZ06\]](#) are much better for odd t). Hence, we restrict ourselves to odd values of t . In particular, we focus on $t \in \{5, 11\}$.

Siv1. Our first observations is that the structure of R in TZ plays a crucial role in the proof of sub-linear distance. For TZ, this structure was desirable as it enables a very efficient linear time encoder. However, using the more general g -ALT encoder we are still able to have linear time encoding for any $g = O(\sqrt{n})$. Our first alteration is then to increase the gap g and ensure all columns of R have weight t . There are several possible values for g and we experimentally settle on $g \in \{24, 32\}$ as they will provide good concrete performance.

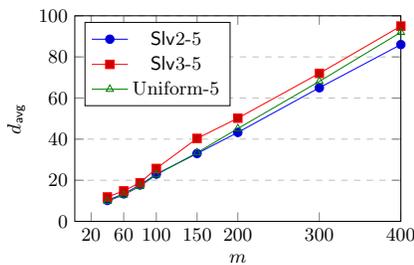
The next question is how should the ones be distributed in R . Our g -ALT encoder require ones along the diagonal which leaves $t - 1$ degrees of freedom

L and R being too similarly distributed. One also might think that to achieve a good expanding property that distributing the diagonals evenly over L would be optimal. However, in this case it is possible for two columns of L to equal.

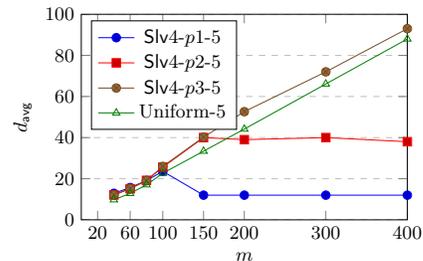
We have experimentally identified that a compromise between these two extremes achieves very good minimum distance performance (both in terms of average distance and variance). In particular, the diagonals should be somewhat evenly distributed while still being irregularly spaced. To identify such distributions we sampled many L at random and evaluate the resulting minimum distance over hundreds of trials and various values of n . An instances of a well performing L with weight $t = 5$ is to distribute the ones of the first column as $\{0m, 0.049m, 0.43m, 0.60m, 0.73m\}$. Other well performing instances have a similar distribution where some diagonals are relatively close while overall they are evenly distributed over the range.

Our methodology for selecting the exact parameters was to evaluate 10,000 random choices at $m \in \{40, 60, 80, 100, 150, 200, 300, 400\}$ and select the top 100 best performing. Out of these, we then ran 100 trials for each $m \in [40, 400]$ with independently sampled R and selected the parameters which maximized d_{\min}/d_{avg} for each m . As such, our selection didn't achieve the highest average distance d_{avg} but instead was "consistently well performing." We note that one has to be careful with the selection of L as a poorly chosen one can result in bad/erratic minimum distance performance. That being said, we observed that most randomly sampled chooses performed well.

The minimum distance performance of this code is depicted in Figure 3a. Interestingly, this code out performs uniform with an average (estimated) minimum distance of $d_{\text{avg}} = 94$ at $m = 400$ compared to $d_{\text{avg}} = 91$ for uniform. Moreover, the variance of this code is quite low, with $d_{\min} = 0.94d_{\text{avg}}$ at $m = 400$ compared to $d_{\min} = 0.91d_{\text{avg}}$ for uniform over 100 trials.



(a) Average minimum distance of Slv2, Slv3-5 vs uniform with $t = 5$.



(b) The negative effect on average minimum distance of having repeats of $p \in \{1, 2\}$ vs $p \geq 3$.

Fig. 3: Performance of Slv3 and Slv4 with $p \in \{1, 2, 3\}$ vs uniform.

Slv4. We now return our attention to improving the distribution of R . Generating R is effectively sampling $O(m)$ random sets of $\binom{g}{t}$, which correspond to the location of the ones on the main diagonal. While linear time, this sampling can

be quite expensive. We therefore experiment with the idea of letting the diagonal repeat over p columns. While one has to be careful with repeated structures in a code, for a sufficiently large p we conjecture and experimentally confirm that it should not harm the minimum distance. We consider a repeat of $p \in \{1, 2, 3, \dots, g\}$ and observe the repeating structure only introduces a weakness for $p \in \{1, 2\}$. The case of $p = 1$ is clearly problematic due to R now effectively being $t + 2$ diagonal lines of width one which structurally is too similar to L . Our experiments reflect this with minimum distances being effectively upper bounded by 12 as seen in Figure 4. For $p = 2$ we observe a similar trend with the distance being upper bounded by 40. However, for $p \geq 3$ we observe no negative effects over all of the trials. To be slightly conservative, we opt to set $p = g$ which for our weight 5 code results in $p = 24$.

We further propose selecting a concrete instance of the diagonal, and validating its performance on the range of experimentally testable values of n . This can in turn give us confidence that the repeating structure does not happen to correspond to a weak instances, e.g. a $p = 1$ instance. Moreover, by selecting a concrete instance, it is possible to hardcode the indices into the program and get a very significant performance improvement.

Slv5. This leads us to our final modification. For the case of $p = g$ we restrict our selection of R such that each row¹ and column has fixed weight $t - 1$ with respect to these random indices. The reason for this alteration is purely to improve the computational efficiency of computing xG^T via the transposed circuit. In particular, the encoding algorithm will process R in a row by row manner. This alteration allows the weight of each row to be not be hard coded improved the performance of the branch predictor, etc. We observe that restricting R to be row regular does not decrease the minimum distance performs. See Appendix ?? for a detailed description.

Eventually, we further consider a variant of Slv5, called Slv5'. This variant is entirely identical, with the sole exception that the parity-check matrix is now viewed as the parity-check matrix *over a field* \mathbb{F} which might not be equal to \mathbb{F}_2 – while the parity-check matrix still has $\{0, 1\}$ entries. We do not use this variant in our main application to silent OT, but it can be used to provide strong efficiency improvements for VOLE over larger fields. We provide support for this modification in Appendix ?? of the Supplementary Material.

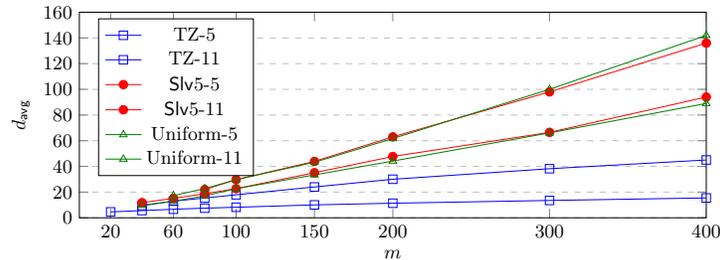


Fig. 4: Average minimum distance of uniform, Slv5 and TZ for weight $t \in \{5, 11\}$.

¹ Excluding edge cases for the first and last set of g rows.

7 Performance Evaluation

We now evaluate the concrete running times of our LDPC codes along with our Silent OT and Vole implementations (available at [Rin]). With respect to our OT protocol we compare with [IKNP03, BCG⁺19a, YWL⁺20]. We also compare our Vole implementation (a direct generalization of [BCG⁺19a] with our LDPC code) with the implementation of [WYKW20]. All implementations target $\kappa = 128$ bits of computational security and $\lambda = 40$ bits of statistical security.

All performance evaluations were performed on a single consumer laptop with an i7 9750H CPU and 16GB of RAM. Networking is performed via localhost. Each party is restricted to a single thread. We note that due to silent property of our protocol, it is very conducive to a multi-threaded implementation but that we only consider single thread performing for simplicity. All numbers reported exclude a setup phase where 128 base OTs are performed.

LDPC Encoding Performance. In previous protocols for silent OT and Vole, the running time was dominated by the compression of the noisy vectors generated in the setup. We now compare our new algorithms with the bit polynomial multiplication encoding used in [BCG⁺19a].

For $n = 2^{20}$, our most optimized code is $31\times$ faster than [BCG⁺19a, CCK⁺18]. This improved running time is not merely due to using an LDPC code as demonstrated by the running time of TZ, which is only between 1.1 and $2\times$ faster than [BCG⁺19a, CCK⁺18]. Moreover, the initial strengthening of the TZ minimum distance by the Slv1 code results in a significant running time increase of $1.5\times$.

The first major performance improvement is achieved by the Slv3 code which changes the distribution of L to have an extremely efficient memory access structure. This change reduces the running time of the L encoding by around $25\times$. The Slv5 code then optimizes the distribution of R to have a repeating structure along with ensuring that it is row regular. These changes allow for very significant memory and system level optimization.

Encoder	weight t	2^{16}	2^{20}	2^{24}
[BCG ⁺ 19a]	-	10.2	194.2	4180
TZ	5	4.5	77.4	1943
	11	4.9	145.2	3971
Slv1	5	7.3	153.2	3632
	11	7.9	241.7	5414
Slv3	5	5.8	88.2	1688
	11	6.1	97	1792
Slv5	5	0.2	6.3	134
	11	0.5	11.3	234

Fig. 5: Running times (ms) of encoding algorithms for LPN with n length vector.

Oblivious Transfer Performance. We now turn our attention to analysing the concrete performance of our OT protocol in comparison to [BCG⁺19a, YWL⁺20,

[IKNP03] as shown in Figure 7. All protocols output m instances of correlated OT where the receiver obtains a per instance bit b and message $m_b \in \{0, 1\}^{128}$ while the sender obtains a global $\Delta \in \{0, 1\}^{128}$ and a per instance message $m_0 \in \{0, 1\}^{128}$ such that $m_b = m_0 + b\Delta$. Random and chosen message OTs can then be obtained via standard techniques. Our protocol is based on that of [BCG⁺19a] and we inherit their $O(\log m)$ communication overhead.

We observe that both our weight 5 and 11 Slv5 codes out perform *all* existing protocol in terms of computational overhead while matching the best communication overhead of [BCG⁺19a]. In particular, our protocol is as much as $1.5\times$ faster than the highly optimized [IKNP03, Rin] protocol which has stood as the most computationally efficient protocol for almost two decades. All this is achieved while communicating exponentially less data. We argue that this is a landmark achievement given the central role OT plays in countless protocols.

The next most efficient protocol is that of Yang et al. [YWL⁺20] which also achieves a sub-linear (but not logarithmic) communication overhead. This protocol is based on Primal LPN and therefore requires a one time setup sub-protocol in which correlated randomness is constructed. Given this, their protocol can then generate correlated OTs on demand. In Figure 7 we distinguish their setup and online protocols as $x+y$ respectively. However, even if only the online protocol is considered, our protocol is more than $4\times$ more efficient in terms of running time and communication. If their setup phase is included then our protocol requires $13\times$ less communication for $m = 10^7$. What is more, their setup phase requires a relatively complicated parameter select procedure which limited us to only performing $m = 10^7$ OTs with their implementation. One reason their only implement this size is that their setup phase has a relatively fixed cost regardless of m . On the other hand, our protocol can easily be executed with any value of m with running times that scales proportionally.

Protocol:	weight t :	Time (ms)				Comm (KB)			
		m				m			
		2^{16}	2^{20}	10^7	2^{24}	2^{16}	2^{20}	10^7	2^{24}
[BCG ⁺ 19a]	-	25	510	5,121	1,0432				
This	5	1	29	268	488	75	94	122	126
	11	2	33	324	591				
[YWL ⁺ 20]	10	-	-	44+1,134	-	-	-	1,130+550	-
[IKNP03]	-	4	45	423	692	1,048	16,777	160,038	268,435

Fig. 6: Single thread running time (ms) and communication (KB) to perform m correlated oblivious transfers in the LAN setting.

Vole Performance. We implement the generalization of [BCG⁺19a] for performing vole. The protocol is largely the same as the OT variant except that f more OTs on strings of length $O(\kappa f)$ need to be performed where f is the log of the field size, i.e. $f = 128$. For our protocol we use the binary Slv5 code while the noise vector is distributed over the whole field. The security of this optimization is discussed in Section ???. We compare with the vole protocol of [WYKW20] (a generalization of [YWL⁺20]) which is based on Primal LPN and therefore

requires a one time setup sub-protocol. We also compare to the 1-out-of- N OT protocol of [OOS17] due to vole also supporting this functionality via hashing.

We observe that our protocol significantly out performs both of these works. Moreover, [WYKW20] performs a vole over a field of size $2^{61} - 1$ while our implementation is for the Galois field of size 2^{128} . As such, this effectively halves their communication. Similarly, [OOS17] has an analogous field size of 2^{79} . Despite working over a larger field, the running time of our protocol $4\times$ faster than [WYKW20] at $m = 4 \times 10^7$ and $22\times$ faster than [OOS17]. Similarly, at $m = 4 \times 10^7$ our protocols requires between 5 to $8\times$ less communication than [WYKW20] depending on if their setup is include and $6200\times$ less than [OOS17].

Protocol:	weight t :	$\log \mathbb{F} $	Time (ms)				Comm (KB)			
			m				m			
			2^{16}	2^{20}	2^{24}	$4 \cdot 10^7$	2^{16}	2^{20}	2^{24}	$4 \cdot 10^7$
This	5 11	128 128	10 11	50 53	616 750	1,390 1,660	339	373	405	409
[WYKW20]	10	61	-	-	-	260+5,699	-	-	-	1,130+2,101
[OOS17]	-	79	15	218	3,499	31,219	4,219	67,137	1,073,832	2,561,551

Fig. 7: Single thread running time (ms) and communication (KB) to perform m voles (or 1-out-of- N OTs for [OOS17]) in the LAN setting.

Applications. The applications of our new protocol are extremely broad. Two of the most compelling are binary triple generation for the GMW[GMW87b] protocol and private set intersection. The former allows generic secure computation of binary circuit at the expense of performing $2|C|$ OTs and sending $2|C|$ bits where $|C|$ is the number of AND gates in the circuit. Due to the extreme efficiency of our protocol, the cost of the OTs is like dominated by the other costs in the GMW protocol, i.e. simply sending the bits. More generally, since our OT protocol is faster than all prior works in effectively all metrics, our protocol should be the de facto choice for generating OTs and binary triples.

The recent semi-honest/malicious secure PSI protocol of [RS21] directly builds on vole and achieved the lowest communication and very fast running times compared to all prior works. This protocol performs a vole of size $2.4n$ where the sets are of size n . Their implementation makes use of the vole protocol of [SGRR19] along with optimizations of [WYKW20]. As such, integrating our vole protocol gives a good example of the speed ups that can be obtained.

For sets of size $n = 2^{16}$ to $n = 2^{24}$ we observe that our vole protocol improves the running time of [RS21] by between 40 and 45 percent and 25 to 1 percent reduction in communication. Concretely, the semi-honest variant of their PSI protocol for $n = 2^{20}$ with our vole implementation would require 3.1 seconds compared to 2.4 seconds of [KKRT16] while at the same time sending $2.5\times$ less data than [KKRT16]. As such, in effectively all real world situation the PSI protocol of [RS21] with our vole is the optimal protocol to use. Moreover, the malicious variant of [RS21] with our vole achieves the fastest running time and lowest communication by a factor of $1.7\times$ and $4\times$ respectively compared to then next most efficient protocol[PTY20].

References

- ABG⁺14. Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. Candidate weak pseudorandom functions in AC^0 *o* MOD_2 . pages 251–260, 2014.
- ADI⁺17. Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. pages 223–254, 2017.
- AJ01. Abdulrahman Al Jabri. A statistical decoding algorithm for general linear block codes. 2001.
- Ale03. Michael Alekhnovich. More on average case vs approximation complexity. pages 298–307, 2003.
- BCG⁺17. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, and Michele Orrù. Homomorphic secret sharing: Optimizations and applications. pages 2105–2122, 2017.
- BCG⁺19a. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. Efficient two-round OT extension and silent non-interactive secure computation. pages 291–308, 2019.
- BCG⁺19b. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. pages 489–518, 2019.
- BCG⁺20. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. pages 1069–1080, 2020.
- BCGI18. Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. pages 896–912, 2018.
- BFKL94. Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. pages 278–291, 1994.
- BJMM12. Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. pages 520–536, 2012.
- BKW00. Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. pages 435–440, 2000.
- BLP11. Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Smaller decoding exponents: Ball-collision decoding. pages 743–760, 2011.
- BM97. Mihir Bellare and Daniele Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. pages 163–192, 1997.
- BM18. Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for LPN security. pages 25–46, 2018.
- Bor57. Jan Lourens Bordewijk. Inter-reciprocity applied to electrical networks. 1957.
- BR17. Andrej Bogdanov and Alon Rosen. Pseudorandom functions: Three decades later. Cryptology ePrint Archive, Report 2017/652, 2017. <http://eprint.iacr.org/2017/652>.
- BTV16. Sonia Bogos, Florian Tramer, and Serge Vaudenay. On solving lpn using bkw and variants. 2016.
- BV16. Sonia Bogos and Serge Vaudenay. Optimization of LPN solving algorithms. pages 703–728, 2016.
- BVJD02. C. Berrou, S. Vaton, M. Jezequel, and C. Douillard. Computing the minimum distance of linear codes by the error impulse method. 2002.

- CCK⁺18. Ming-Shing Chen, Chen-Mou Cheng, Po-Chun Kuo, Wen-Ding Li, and Bo-Yin Yang. Multiplying boolean polynomials with frobenius partitions in additive fast fourier transform, 2018.
- DAT17. Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding. 2017.
- DP15. A. Dutta and A. Pramanik. Modified approximate lower triangular encoding of ldpc codes. 2015.
- DRU06. Changyan Di, Thomas J. Richardson, and Rüdiger L. Urbanke. Weight distribution of low-density parity-check codes. 2006.
- EKM17. Andre Esser, Robert Kübler, and Alexander May. LPN decoded. pages 486–514, 2017.
- Fei02. Uriel Feige. Relations between average case complexity and approximation complexity. pages 534–543, 2002.
- FGKP09. Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. 2009.
- FKI06. Marc PC Fossorier, Kazukuni Kobara, and Hideki Imai. Modeling bit flipping decoding based on nonorthogonal check sums with application to iterative decoding attack of mceliece cryptosystem. 2006.
- FS09. Matthieu Finiasz and Nicolas Sendrier. Security bounds for the design of code-based cryptosystems. pages 88–105, 2009.
- Gal62. Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.
- GDP73. S. I. Gelfand, R. L. Dobrushin, and M. S. Pinsker. On the complexity of coding. 1973.
- GJL20. Qian Guo, Thomas Johansson, and Carl Löndahl. Solving LPN using covering codes. 33(1):1–33, January 2020.
- GMW87a. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. pages 218–229, 1987.
- GMW87b. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design. pages 171–185, 1987.
- Gra06. Markus Grassl. *Searching for linear codes with large minimum distance*. 2006.
- HIQO19. Fernando Hernando, Francisco D. Igual, and Gregorio Quintana-Ortí. Algorithm 994: Fast implementations of the brouwer-zimmermann algorithm for the computation of the minimum distance of a random linear code. 2019.
- IKNP03. Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. pages 145–161, 2003.
- IKOS08. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. pages 433–442, 2008.
- Kil88. Joe Kilian. Founding cryptography on oblivious transfer. 1988.
- Kir11. Paul Kirchner. Improved generalized birthday attack. Cryptology ePrint Archive, Report 2011/377, 2011. <http://eprint.iacr.org/2011/377>.
- KKRT16. Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. pages 818–829, 2016.
- KOS15. Marcel Keller, Emmanuela Orsini, and Peter Scholl. Actively secure OT extension with optimal overhead. pages 724–741, 2015.

- KS12. K. Kobayashi and T. Shibuya. Generalization of lu’s linear time encoding algorithm for ldpc codes. 2012.
- LF06. Éric Levieil and Pierre-Alain Fouque. An improved LPN algorithm. pages 348–359, 2006.
- Lyu05. Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. 2005.
- McE78. Robert J McEliece. A public-key cryptosystem based on algebraic. 1978.
- MMT11. Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. pages 107–124, 2011.
- MO15. Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. pages 203–228, 2015.
- MST03. Elchahan Mossel, Amir Shpilka, and Luca Trevisan. On e-biased generators in NC0. pages 136–145, 2003.
- NN90. Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. pages 213–223, 1990.
- OOS17. Michele Orrù, Emanuela Orsini, and Peter Scholl. Actively secure 1-out-of-N OT extension with application to private set intersection. pages 381–396, 2017.
- OTA07. Ayoub Otmani, Jean-Pierre Tillich, and Iryna Andriyanova. On the minimum distance of generalized LDPC codes. 2007.
- Ove06. Raphael Overbeck. Statistical decoding revisited. pages 283–294, 2006.
- Pra62. Eugene Prange. The use of information sets in decoding cyclic codes. 1962.
- PRTY20. Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from PaXoS: Fast, malicious private set intersection. pages 739–767, 2020.
- Rin. Peter Rindal. libOTe: an efficient, portable, and easy to use Oblivious Transfer Library. <https://github.com/osu-crypto/libOTe>.
- RS21. Peter Rindal and Phillipp Schoppmann. VOLE-PSI: Fast OPRF and circuit-PSI from vector-OLE. In *Eurocrypt*, 2021.
- RU01. Thomas J. Richardson and Rüdiger L. Urbanke. Efficient encoding of low-density parity-check codes. 2001.
- Saa07. Markku-Juhani Olavi Saarinen. Linearization attacks against syndrome based hashes. pages 1–9, 2007.
- SGRR19. Phillipp Schoppmann, Adrià Gascón, Leonie Reichert, and Mariana Raykova. Distributed vector-OLE: Improved constructions and implementation. pages 1055–1072, 2019.
- Shp09. Amir Shpilka. Constructions of low-degree and error-correcting ε -biased generators. 2009.
- Spi96. Daniel A Spielman. Linear-time encodable and decodable error-correcting codes. 1996.
- Ste88. Jacques Stern. A method for finding codewords of small weight. 1988.
- TZ06. Jean-Pierre Tillich and Gilles Zémor. On the minimum distance of structured LDPC codes with two variable nodes of degree 2 per parity-check equation. 2006.
- Var97. A. Vardy. The intractability of computing the minimum distance of a code. 1997.
- Wag02. David Wagner. A generalized birthday problem. pages 288–303, 2002.
- WYKW20. Chenkai Weng, Kang Yang, Jonathan Katz, and Xiao Wang. Wolverine: Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits, 2020.
- XFE04. Xiao-Yu Hu, M. P. C. Fossorier, and E. Eleftheriou. On the computation of the minimum distance of low-density parity-check codes. 2004.

- Yao86. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). pages 162–167, 1986.
- YWL⁺20. Kang Yang, Chenkai Weng, Xiao Lan, Jiang Zhang, and Xiao Wang. Ferret: Fast extension for correlated OT with small communication. pages 1607–1626, 2020.
- Zic17. Lior Zichron. Locally computable arithmetic pseudorandom generators, 2017.
- ZJW16. Bin Zhang, Lin Jiao, and Mingsheng Wang. Faster algorithms for solving LPN. pages 168–195, 2016.