A Framework for Statistically Sender Private OT with Optimal Rate

Pedro Branco[1], Nico Döttling[2], and Akshayaram Srinivasan[3]

[1]Max Planck Institute for Security and Privacy
[2]Helmholtz Center for Information Security (CISPA)
[3]Tata Institute of Fundamental Research

Abstract

Statistical sender privacy (SSP) is the strongest achievable security notion for two-message oblivious transfer (OT) in the standard model, providing statistical security against malicious receivers and computational security against semi-honest senders. In this work we provide a novel construction of SSP OT from the Decisional Diffie-Hellman (DDH) and the Learning Parity with Noise (LPN) assumptions achieving (asymptotically) optimal amortized communication complexity, i.e. it achieves rate 1. Concretely, the total communication complexity for k OT instances is $2k(1 + o(1))$, which (asymptotically) approaches the information-theoretic lower bound. Previously, it was only known how to realize this primitive using heavy rate-1 FHE techniques [Brakerski et al., Gentry and Halevi TCC'19].

At the heart of our construction is a primitive called statistical co-PIR, essentially a a public key encryption scheme which statistically erases bits of the message in a few hidden locations. Our scheme achieves nearly optimal ciphertext size and provides statistical security against malicious receivers. Computational security against semi-honest senders holds under the DDH assumption.

1 Introduction

Oblivious Transfer Oblivious transfer (OT) [Rab05] is one of the central objects of study in secure computation: OT is complete for secure two- and multiparty computation in the sense that given a secure OT protocol, any distributed function can be computed securely. In its basic form, OT is a protocol between a receiver, holding a bit $b \in \{0,1\}$ and a sender holding two bits $m_0, m_1 \in \{0,1\}$. It allows the receiver to retrieve the bit m_b in such a way that the sender learns nothing about b, while the receiver learns nothing about m_{1-b}.

In the two party setting, OT protocols [1] cannot be information-theoretically secure against both parties, hence a cryptographic communication overhead of size $\mathsf{poly}(\lambda)$ is necessary. To amortize this overhead, one of the protocol parameters has to grow polynomially. In string-OT, the sender transfers (potentially) long message strings $\mathbf{m}_0, \mathbf{m}_1 \in \{0,1\}^{\mathsf{poly}(\lambda)}$.

The batch-setting offers a different angle on amortization: Rather than increasing the length of the messages to size $\mathsf{poly}(\lambda)$, one can bundle the joint execution of many (say, k) OT-instances into a single protocol.

Oblivious Transfer with High Rate. Minimizing the communication overhead of OT in the string and batch setting has received considerable attention in recent years [IKNP03, BCGI18, BCG+19b, BCG+19a, BCG+19b, DGI+19, BCG+20a, BCG+20b, GHO20, CGH+21, ADD+22]. This line of research has culminated in constructions of OT protocols from several hardness assumptions [BDGM19, GH19, BBDP22]

[1]without the help of additional trust assumptions such as e.g. secure hardware

achieving optimal communication of $2k(1 + o(1))$, which approaches the information theoretic lower-bound $2k$.[2] The rate of an OT protocol in the batch setting is defined as the ratio between the information-theoretic lower bound and the overall communication of a given protocol, hence protocols with communication $2k(1 + o(1))$ have rate $1/(1 + o(1)) = 1 - o(1)$ which approaches 1.

Compared to "low-rate" OT protocols, say protocols with rate $\leq 1/2$, such high-rate OT protocols have been notoriously hard to build and there is a fundamental reason for this. Achieving communication seems to involve a *phase-transition* and should be viewed as more than a *constant factor* improvement. For example, (two-message) OT with download-rate beyond this threshold implies the existence of lossy trapdoor functions [DGI+19][3] or succinct two-round protocols for branching program evaluation and PIR schemes [IP07, DGI+19].

The techniques developed in this context found surprising applications, and where instrumental in several constructions of correlation-intractable hash functions, which gave rise to non-interactive zero-knowledge protocols [BKM20, JJ21] or batch arguments [KLVW22] from weaker assumptions.

The goal of this paper is to study what is the strongest notion of security we can achieve for OT in the plain model (i.e., without any trusted setup assumptions) while preserving optimal communication.

Statistical Sender Privacy. The standard (simulation-based) security definition of OT is given with respect to semi-honest adversaries. To achieve simulation-based security against malicious parties, one has to either rely on trusted setup assumptions (or random oracles), or increase the round complexity of the protocol. Statistical sender privacy provides a meaningful relaxation to the standard notion of malicious security and has been shown to be achievable with *two-message protocols without making use of setup assumptions* [NP01, AIR01]. Since then, SSP OT was built from several assumptions such as DDH [NP01, AIR01, ADD+22], QR (or DCR) [HK12], lattice based assumptions [BD18, MS20], or LPN together with a derandomization assumption [BF22].

SSP OT protocols provide security against a computationally bounded sender, but a strong statistical security notion against malicious and unbounded receivers. In essence, this notion requires the existence of an *unbounded* simulator which extracts the receiver's choice bit b^* from his (potentially malformed) protocol message, and simulates the response of an honest sender given only the message m_{b^*} the receiver is supposed to obtain.

SSP OT has been the critical ingredient in the construction of several strong primitives, most notably the construction of malicious circuit-private FHE [OPP14], two-round statistical zaps [JKKR17, BGI+17a, KKS18, BFJ+20, GJJM20], non-malleable commitments [KS17] or two-party computation with statistical security [KM20, BPS22].

SSP OT with Rate 1. The rate-1 and SSP properties are not *orthogonal*, but intricately connected. Specifically, SSP OT can be constructed from any rate 1 OT [BGI+17a, DGI+19]. The catch is, however, that this transformation (and in fact any such generic transformation) does not preserve rate 1, but necessarily makes the rate drop below 1/2.

Further note that rate 1 (batch or string) OT is by no means automatically SSP. For string OT a malicious receiver might learn half of the bits of $\mathbf{m}_0$ and $\mathbf{m}_1$, whereas for batch OT a malicious receiver might learn both message bits m_0 and m_1 for some of the batch instances. Indeed, with no safeguards against such attacks in place, it is relatively straightforward to construct malformed receiver messages to implement this attack e.g. against the scheme of Brakerski et al. [BBDP22].

One may thus raise the question whether rate 1 and SSP security can actually be achieved simultaneously. Indeed, this question was answered affirmatively by Brakerski et al. [BDGM19] and by Gentry and Halevi [GH19] under the LWE assumption, whose constructions of rate-1 statistically circuit private FHE give rise to rate-1 SSP OT, both in the string OT and the batch OT setting.

[2] The communication complexity of any, even insecure, OT is at least $2k$. We can achieve this by sacrificing sender's security (the sender sends both (m_0, m_1)) or receiver's security (the receiver sends b and obtains m_b from the sender).

[3] This is the main reason for why one could expect such a protocol to inherently be heavier on public-key operations.

Their constructions, however, relies on heavy FHE machinery and on specifics of the LWE problem which allow for *ciphertext compression*. As a consequence, their result should be seen as a feasibility result. Furthermore, due to its heavy reliance on FHE-specific (non-black-box) techniques their result does not provide a recipe on how to construct high-rate SSP OT from a wider range of assumptions, or how to realize this primitive with concrete efficiency. This is unsatisfactory both from a theoretical and an applied perspective: the foundation of rate 1 SSP OT is as narrow as that of FHE, and the heavy non-black-box machinery in their construction constitutes a serious roadblock for actually using this primitive.

More recently, in the string OT setting, Aggarwal et al [ADD+22] showed that *download* rate-1 SSP string OT (for asymptotically long strings) can be achieved from the standard DDH assumption via a fully black box construction.

However, in the setting of rate-1 SSP batch OT, the FHE-based constructions of [BDGM19, GH19] are currently the only candidates.

1.1 Our Results

A framework for SSP OT with optimal rate. Our main result is a new framework for the construction of SSP batch OT schemes with optimal communication complexity in the plain model (i.e., no random oracles or common reference string).

Statistical co-PIR. Our framework is a refinement of the blueprint of [BBDP22]. One of the key tools in [BBDP22] is a primitive called co-PIR. On a high level, a co-PIR scheme can be seen as a rate-1 public key encryption scheme (for long messages) which erases some bits of the sender's message in a way such that they are not recoverable by the receiver.

We identify the construction of co-PIR in [BBDP22] as one of the main bottlenecks towards an SSP secure rate 1 batch OT construction. The main reason is that their construction is only computationally secure, and the *lost bits* of the sender's message are only computationally hidden from the receiver.

Our first contribution is a statistically secure construction of co-PIR, which guarantees that the lost bits of the sender's message are *statistically hidden* from the receiver. As such, a statistical co-PIR scheme can be seen *somewhere statistically hiding* encryption scheme.

Theorem 1 (Informal)**.** *There exists a co-PIR scheme that is statistically secure against malicious receivers and computationally secure against semi-honest senders assuming the DDH assumption. Additionally, the scheme fulfills the following efficiency properties:*

- *The sender's computational complexity is subquadratic in the size of the database,* $|\mathbf{D}|^{1+\varepsilon} \cdot t \cdot \mathsf{poly}(\lambda)$ *where* $\mathbf{D}$ *represents the input of the sender and* t *represents the size of the receiver's input and* $1 > \varepsilon > 0$*. The receiver's computational complexity is* $|\mathbf{D}| \cdot t \cdot \mathsf{poly}(\lambda)$*.*
- *The size of the sender's message is* $|\mathbf{D}| + o(|\mathbf{D}|)$ *when* $t = o(|\mathbf{D}|)$*.*

We further provide a generic construction of a co-PIR scheme from any (statistically sender private) rate-1 block-PIR scheme, i.e. a PIR scheme which transfers large blocks. Such rate-1 block PIR schemes can be constructed from the DDH assumption [ADD+22]. A comparison between different co-PIR schemes is presented in Table 1.

Rate-1 SSP batch OT from DDH and LPN We provide instantiations of the primitives of our framework from the DDH assumption. We thus obtain a rate-1 SSP batch OT scheme from the DDH and LPN (with inverse polynomial noise rate) assumptions. Specifically, to execute k independent OTs, the overall communication complexity required by our protocol is $2k(1 + o(1))$.

Theorem 2 (Informal)**.** *There exists a SSP OT scheme with optimal rate where security for the receiver holds assuming both DDH and LPN (with inverse polynomial noise-rate) assumptions.*

	Security	Hardness Assumption	Sender's Work
[BBDP22]	Semi-honest	DDH, QR, LWE	$\lvert\mathbf{D}\rvert \cdot t \cdot \mathsf{poly}(\lambda)$
This work	SSP	DDH	$\lvert\mathbf{D}\rvert^{1+\varepsilon} \cdot t \cdot \mathsf{poly}(\lambda)$
This work Appendix B	SSP	Rate-1 PIR	$\lvert\mathbf{D}\rvert^{2} \cdot t \cdot \mathsf{poly}(\lambda)$

Table 1: Comparison between existing co-PIR schemes. Here, $\mathbf{D}$ represents the input of the sender (i.e., the database), t represents the size of the receiver's input (i.e., how many indices will be erased) and $1 > \varepsilon > 0$. SSP stands for statistical sender privacy. All schemes achieve (asymptotic) download-rate 1 and the receiver's message is of size $\mathsf{poly}(t, \lambda) \cdot \mathsf{polylog}(|\mathbf{D}|)$. For all schemes, the receiver's computational complexity is $|\mathbf{D}| \cdot t \cdot \mathsf{poly}(\lambda)$.

Our result improves upon Brakerski et al.[BBDP22] (henceforth, denoted as BBDP scheme) in terms of security: Our work achieves a stronger notion of security, namely we prove security against unbounded malicious receivers whereas the BBDP scheme achieves only computational security against semi-honest receivers. We stress that, although the BBPD scheme achieves download rate 1, it only provides *computational* (instead of statistical) security against semi-honest receivers. This is because a computationally unbounded semi-honest receiver has enough information to break a subset of the OTs. A comparison with BBDP is given in Table 2

	Security	Hardness Assumption	Sender's Work
[BBDP22]	Semi-honest	$\left\{\begin{array}{c}\text{DDH, QR,}\\ \text{LWE}\end{array}\right\}$ + LPN	$k^{1+\varepsilon_0} \cdot \mathsf{poly}(\lambda)$
This work Section 10	SSP	DDH + LPN	$k^{1+\varepsilon_1} \cdot \mathsf{poly}(\lambda)$

Table 2: Comparison between existing optimal-rate OT schemes. Here, k represents the number of OT executions, and $1 > \varepsilon_0, \varepsilon_1 > 0$. SSP stands for statistical sender privacy. For all schemes, the receiver's computational complexity is slightly superlinear $k^{1+\varepsilon} \cdot \mathsf{poly}(\lambda)$.

Communication-efficient 2PC. As an application of main result, we give a construction of a 2PC protocol that has statistical security against one of the parties and has constant communication overhead. We obtain this protocol by instantiating the GMW protocol [GMW87] using our SSP OT scheme. An informal statement of this result is given below.

Theorem 3 (Informal)**.** *There exists a two-party secure computation scheme with communication complexity of* $\mathcal{O}(|\mathcal{C}|) + |x| + |y| + \mathsf{poly}(\lambda)$ *where* $\mathcal{C}$ *is the circuit being computed and* x, y *are the inputs of the parties. The scheme achieves statistical security against one of the parties and computational security against the other one (assuming both DDH and LPN assumptions) in the semi-honest setting.*

Previously, 2PC protocols for general circuits with constant overhead in the size of the circuit (or, better) and which provide semi-honest statistical security against one of the parties were known either from circuit-private FHE [OPP14] or SSP OT along with PRGs in NC^0 [IKOS08].

2 Technical Overview

Throughout this technical overview, we refer to the ratio between the size of the receiver's protocol message and the size of the receiver's input as the *upload rate*, and the ratio between the size of the sender's protocol

message and the size of the receiver's output as the *download rate*.

2.1 Optimal-Rate OT Secure Against Semi-Honest Adversaries

Our starting point is the recent construction of *semi-honestly secure* rate-1 batch OT by Brakerski et al. [BBDP22]. In their construction semi-honest security against both senders and receivers holds computationally.

The core idea in [BBDP22] is the following: The receiver encrypts his choice-bits under a specific rate-1 private key encryption scheme, and encrypts the (short) keys under a linearly homomorphic public-key encryption (LHE) scheme. The private-key encryption scheme has the feature that approximate decryption is a linear operation. This allows the sender to decrypt the private-key ciphertext *under the hood* of the LHE an thus obtain a noisy LHE encryption of the receiver's choice. The actual OT function $f(\mathbf{x}) = (\mathbf{m}_1 - \mathbf{m_0}) \odot \mathbf{x} + \mathbf{m}_0$ (where $(\mathbf{m}_0, \mathbf{m}_1)$ is the sender's input) can now be evaluated homomorphically on the receiver's choice bits [4]. Here, $\odot$ denotes the component-wise multiplication.

Given that the LHE scheme supports *post-evaluation ciphertext compression*, the ciphertext thus generated can be compressed into a rate-1 ciphertext, which is then sent to the receiver.

In [BBDP22], the private-key scheme is instantiated from the LPN assumption and the LHE with post-evaluation compression mechanism using decisional Diffie-Hellman (DDH), quadratic residuosity (QR) or learning with errors (LWE) by adapting the ciphertext compression techniques from [BGI16, DGI+19, BBD+20, BBDP22].

Co-Private Information Retrieval. While the basic outline of the above scheme intuitively makes sense, there is a subtle issue we have glossed over: the decryption of the private key scheme is only approximate. Consequently, this will lead to a correctness error which causes the receiver to learn outputs he was not supposed to learn.

More concretely, the private-key encryption scheme in [BBDP22] is realized as a basic LPN encryption scheme, where $\mathbf{A} \leftarrow\$ \mathbb{F}_2^{n \times m}$ is a public random matrix, $\mathbf{s} \in \mathbb{F}_2^n$ is the secret key, and ciphertexts are of the form $\mathbf{c} = \mathbf{sA} + \mathbf{e} + \mathbf{b}$, where $\mathbf{e} \in \mathbb{F}_2^m$ is a sparse random noise term, and $\mathbf{b}$ is the vector of choice bits . The noisy plaintext can be recovered by computing $\mathbf{c} - \mathbf{sA} = \mathbf{b} + \mathbf{e}$.

Consequently, in positions i where $\mathbf{e}_i = 1$, the receiver will obtain the wrong OT output, namely $\vec{m}_{1-\vec{b}_1,i}$. Note that this does not just constitute a correctness issue, but a security issue as the receiver is not supposed to learn this value.

To address this issue, [BBDP22] introduced a new primitive called co-Private Information Retrieval (co-PIR). A co-PIR scheme allows a receiver to retrieve a database from a sender with the guarantee that some positions (unknown to the sender) are erased. More precisely, in a co-PIR scheme, the receiver starts by choosing a subset of indices $S \subset [m]$ and computes a first message copir_1. S denotes the set of indices that the receiver wants to be erased. The sender, with input a database $\mathbf{D} \in \{0,1\}^m$, computes a second message copir_2 that allows the receiver to retrieve a database $\tilde{\mathbf{D}}$. The correctness property guarantees that $\mathbf{D}$ and $\tilde{\mathbf{D}}$ coincide for all the locations $[m] \setminus S$. For security, we require that the sender obtains no information about the receiver's input and the receiver in turn learns nothing about the positions $\mathbf{D}_i$ for $i \in S$.[5] In terms of efficiency, we require that the size of the receiver's message copir_1 to only grow polylogarithmically in the size of the database m and polynomially on the size of S. Moreover, the size of sender's message copir_2 should be $\mathbf{D} + o(\mathbf{D})$ and we call such a co-PIR scheme to have near optimal download rate.

[BBDP22] provided a computationally secure construction of co-PIR from puncturable pseudorandom functions and PIR, or alternatively GGM PRFs [GGM86] and (low-rate) OT following [BCG+19a] (also known as punctured OT [BGI17b]). Moreover, these constructions achieve near optimal download rate. From a technical perspective, these constructions are *inherently* limited to computational security due to the way they use puncturable PRFs.

[4] For subtle technical reasons, the decryption and OT functions are combined into a single linear function

[5] Co-PIR can be seen as the opposite of PIR: In a PIR the receiver retrieves the positions of the database that it is asking for, whereas in a co-PIR it gets the entire database except for those positions.

The above-mentioned issue can now be adressed as follows using both co-PIR and a (sender-private) PIR: The receiver generates a co-PIR message which erases all the locations corresponding to LPN errors (note that the error-locations are known to the receiver). Furthermore, he generates PIR instances which retrieve information at the locations corresponding to LPN errors. The sender will now transmit the compressed LHE ciphertext using the co-PIR, and use the PIR scheme to transmit the correct outputs at the erased positions.

2.2 Towards Statistical Sender Privacy

In order to adapt the BBDP-framework to the setting of statistical sender privacy *while preserving rate-1*, we encounter the following challenges.

1. **Statistical co-PIR.** Clearly, the biggest issue with the BBDP construction with respect to SSP security lies in the fact that their co-PIR scheme offers only computational security, *even against semi-honest receiver*. Hence it seems inevitable that we have to take a different route to construct statistical co-PIR. Additionally, we need this statistical secure co-PIR scheme to have near optimal download rate.

2. **Consistency of Inputs.** We need the input set S sent by the receiver as part of PIR and co-PIR messages to be the same. Otherwise, a malicious receiver can cheat and learn both messages $m_{0,i}$ and $m_{1,i}$ for some position i and thus, breaking the sender security of OT.

3. **Well-formedness of ciphertexts.** The protocol described above assumes that the ciphertext ct (encrypting the LPN secret) generated by the receiver is well-formed. Namely, this ciphertext should encrypt bits and have a special structure that allows for packing of ciphertexts.

In the following, we will outline our approach to deal with these issues.

2.3 Statistical co-PIR

Our main challenge is to construct a co-PIR scheme that provides statistical security against malicious receivers and has near optimal download rate. We build such a co-PIR scheme in a sequence of steps:

1. We start by building a one-query statistical semi-honest co-PIR, which erases only a single block of bits and provides semi-honest security for both the receiver and the sender.

2. We then show how to achieve a one-query statistical semi-honest co-PIR that erases a single bit (instead of an entire block).

3. In the next transformation, we show how to bootstrap a co-PIR that only allows to erase one bit into one where multiple bits are erased.

4. Finally, we show how to achieve statistical sender privacy.

2.3.1 One-Query Statistical Semi-Honest Co-PIR

We first tackle the simpler task of constructing a statistical co-PIR which only erases one position of the database and the security is required to hold only against semi-honest adversaries. We will call this primitive a one-query semi-honest co-PIR.

One-query Semi-honest co-PIR from PIR. One-query semi-honest co-PIR can be constructed in a generic way from PIR. The receiver's input to the PIR corresponds to the position that it wants to be erased. The sender's input to the PIR corresponds to vectors $\hat{\mathbf{D}}_1, \ldots, \hat{\mathbf{D}}_m$ where each $\hat{\mathbf{D}}_i$ corresponds to the database $\mathbf{D}$ with the i-th position erased. By the correctness of the PIR, the receiver obtains $\hat{\mathbf{D}}_i$.

For the resulting co-PIR scheme to be rate-1 and provide statistical security for the sender, we need that the underlying PIR to fulfill these requirements. Such a PIR scheme was recently constructed in the work of Aggarwal et al. [ADD+22].

However, a drawback of this construction is that the sender's work is proportional to $|\mathbf{D}|^2$, which is the size of the sender's input to the PIR, whereas the receiver's work is proportional to $|\mathbf{D}|$. We now explain how to build a co-PIR scheme which achieves better efficiency for the sender.

All-but-one lossiness. Our first observation is that a one-query statistical co-PIR resembles a primitive called *all-but-one trapdoor lossy function* (ABO-TDF) [PW08]. Loosely speaking, an ABO-TDF is a function parametrized by some public key and which is invertible everywhere except for some specified one branch where it loses information. Crucially, the public key should not reveal about the lossy branch.

Peikert and Waters provide a simple construction of ABO-TDF from a linear homomorphic scheme LHE such as El Gamal: to generate an ABO-TDF public key which is lossy on a branch i^*, one first generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda)$ and encrypts $\mathsf{ct} \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, i^*)$. The new ABO-TDF public key is composed by $(\mathsf{pk}, \mathsf{ct})$. To encrypt a message $m \in \mathbb{Z}_2$ under index i, we homomorphically compute the function $f(x) = (i - x) \cdot m$ and obtain a new ciphertext $\tilde{\mathsf{ct}}$.

It is easy to see that for all $i \neq i^*$, decryption can recover $m = \mathsf{LHE.Dec}(\mathsf{sk}, \tilde{\mathsf{ct}}) \cdot (i - i^*)^{-1}$. However, when $i = i^*$ all information about m is statistically hidden, assuming that LHE is function private.

A simple statistical co-PIR with large computation. In the following, let p be the order of a DDH group and LHE be a function private LHE scheme over a smaller field $\mathbb{Z}_q$ for $q = \mathsf{poly}(\lambda)$, such as the one presented in [BBDP22]. Let $\mathbf{D} \in \mathbb{Z}_q^m$ be the database of the sender, where q will be later defined. As a first approach consider the following protocol for co-PIR:

- The receiver creates a pair of public/secret keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda)$ and encrypts $\mathsf{ct} \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, i^*)$ for $i^* \in [m]$.
- For all $i \in [m]$ the sender homomorphically computes $f_i(x) = (i-x) \cdot \mathbf{D}_i$ over ct and obtains ciphertexts $\tilde{\mathsf{ct}}_1, \ldots, \tilde{\mathsf{ct}}_m$.
- For all $i \neq i^*$ the receiver obtains $\mathbf{D}_i \leftarrow \mathsf{LHE.Dec}(\mathsf{sk}, \tilde{\mathsf{ct}}_i)$.

It is easy to see that correctness holds for all $i \neq i^*$. Semi-honest security for the receiver follows from the IND-CPA of LHE and semi-honest statistical security for the sender follows from the statistical function privacy of LHE and from the fact that $(i^* - i^*) \cdot m = 0 \cdot m = 0$.

In terms of efficiency, the receiver's message is composed by a public key and an encryption and hence its size is independent of $|\mathbf{D}|$. However, the sender's message is composed by m uncompressed ciphertexts. So the scheme does not achieve near optimal download rate.

To achieve near optimal download rate, we will use the ciphertext compression technique for El Gamal presented in [BBDP22] (which is itself based on previous works [BGI16, DGI+19, BBD+20]). These techniques are specially designed for packed El Gamal and to use these packing techniques, we need the following two conditions to hold.

1. The receiver's message needs to encrypt a matrix rather than a single value i^*, in order for packing to be possible. That is, $\mathsf{ct} \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, i^* \cdot \mathbf{I})$ where $\mathbf{I}$ is the identity matrix of size k.
2. We need $\mathbf{D}_i$ to be in $\mathbb{Z}_q^k$ for large enough k in order to amortize the size of the ciphertext for a single block. Moreover, we need that $q > m$. The latter condition comes from the fact that the operation $(i - i^*)$ needs to be performed over a modulus greater than m. If that was not the case, then it might happen that $(i - i^*) = 0 \mod q$ for $i \neq i^*$ over the integers and we will loose correctness.

This gives us a statistical semi-honest one-query co-PIR for databases of size $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ where each $\mathbf{D}_i \in \mathbb{Z}_q^k$ for $q > m$.

In terms of computation, the scheme still incurs a quadratic blowup for both the sender and the receiver. All ciphertext compression mechanisms for DDH [BGI16, DGI+19, BBD+20, BBDP22] have computational complexity scaling with q for both the sender and the receiver. Since $q > m$, for each block, both parties have to spend computational work proportional to m. Since there are m blocks, we end up with computational complexity proportional to at least m^2. Thus, we have not achieved any significant gains over the simple solution from PIR.

An efficient statistical co-PIR. To improve the computational complexity of the protocol, we need a way to make the complexity of encrypting and decrypting each block independent of m. Towards this goal, we use a standard trick of embedding the underlying messages in an extension field.

To be a bit more specific, our idea is to parse the database $\mathbf{D}$ as a vector over an extension field $\mathbb{F}_{2^\mu}$ where $\mu = \lceil \log m \rceil$. That is, we parse $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m) \in \mathbb{F}_{2^\mu}^{k \cdot m}$ where each $\mathbf{D}_i \in \mathbb{F}_{2^\mu}^k$.

We rely on the fact that for any two elements $\hat{x}, \hat{a} \in \mathbb{F}_{2^\mu}$, where $\hat{x} = x_1 + x_2\alpha + \cdots + x_\mu\alpha^{\mu-1}$ for some symbol α, each coefficient of the product $\hat{x} \cdot \hat{a}$ can be expressed as a linear function depending only on $\hat{a}$. That is,

$$\hat{x} \cdot \hat{a} = f_{1,\hat{a}}(\mathbf{x}) + f_{2,\hat{a}}(\mathbf{x})\alpha + \cdots + f_{\mu,\hat{a}}(\mathbf{x})\alpha^{\mu-1}$$

where $\mathbf{x} = (x_1, \ldots, x_\mu)$ and each $f_{i,\hat{a}}$ is a $\mathbb{Z}_2$-linear function that depends solely on $\hat{a}$.

Given this, the new scheme with improved computational complexity can be obtained as follows:

- Given an index $i^* \in [m]$, the receiver first decomposes it into its binary decomposition $\mathbf{i}^* = (i_1^*, \ldots, i_\mu^*)$. Then, it creates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda)$ and encrypts each i_1^*, that is, $\mathsf{ct}_j \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, i_j^* \cdot \mathbf{I})$ where $\mathbf{I}$ is the identity matrix of size k. It sends pk and the ciphertexts ct_j to the sender.

- The sender parses $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m) \in \mathbb{F}_{2^\mu}^{k \cdot m}$ where each $\mathbf{D}_i \in \mathbb{F}_{2^\mu}^k$. For each $\ell \in [m]$ it evaluates the function $f_i(\hat{\mathbf{X}}) = (\hat{\ell} \cdot \mathbf{I} - \hat{\mathbf{X}}) \cdot \mathbf{D}_\ell$ over $\mathbb{F}_{2^\mu}$ where $\hat{\ell}$ is the embbeding of ℓ in $\mathbb{F}_{2^\mu}$ (by first converting into its binary decomposition and then interpreting it as a $\mathbb{F}_{2^\mu}$ element). As we have seen, f_i can be expressed as a $\mathbb{Z}_2$-linear function. Let $\tilde{\mathsf{ct}}_\ell$ be the resulting ciphertexts. It compresses the ciphertexts $\tilde{\mathsf{ct}}_\ell$ to make them rate 1.

- Finally, the receiver decrypts each $\tilde{\mathsf{ct}}_\ell$ for $\ell \neq i^*$, interprets the result as a $\mathbb{F}_{2^\mu}^k$ vector $\mathbf{u}$ and computes $\tilde{\mathbf{D}}_\ell = (\hat{\ell} - \hat{i}^*)^{-1} \cdot \mathbf{u}$ over $\mathbb{F}_{2^\mu}$.

Correctness, semi-honest security for the receiver and semi-honest statistical sender security hold as in the protocol above.

Computational complexity. Unlike the previous protocol, the sender needs to compress m ciphertexts of size $\mu \cdot k$. However, now all ciphertexts encrypt bits instead of messages over a larger modulus. Hence, the sender's computational complexity grows only linearly with the size $m \cdot k \cdot \mu$ of the database. Similarly, the receiver needs to decrypt the ciphertexts (encrypting bits) sent by the sender, hence the computational complexity for the receiver grows only linearly with the size of the database.

2.3.2 Full-Fledged Statistical Co-PIR

Until now we have discussed how to obtain a semi-honest statistical one-query co-PIR. However, for our OT application, the co-PIR needs to i) provide statistical security against a malicious receiver, and ii) support more than one query. Additionally, to obtain a bit-OT, we need our co-PIR to erase a single bit of the database whereas the construction presented above only works for large erased blocks.

Bit co-PIR from block co-PIR. We start by the simplest task of turning a co-PIR which erases an entire block, or block co-PIR, into one that erases a single bit, or bit co-PIR.

Assume that the receiver wants to erase the j^* bit of the i^* block. We show that a bit co-PIR can be built from a block co-PIR by additionally assuming the existence of a PIR scheme. The block co-PIR will erase an entire block $\mathbf{D}_{i^*} = (D_{i^*,1}, \ldots, D_{i^*,k})$. The remaining positions $D_{i^*,j}$ for $j \neq j^*$ can be sent to the receiver via a PIR. The resulting scheme can be seen as a (one-query) bit co-PIR as only D_{i^*,j^*} is erased from the perspective of the receiver. Importantly, we show that this scheme preserves security even against malicious receivers. This is because if the PIR message points to another block, then the malicious receiver obtains strictly lesser information and this does not violate the privacy of the honest sender.

Additionally, in terms of communication the scheme preserves i) short message from the receiver as the PIR receiver's message is small compared to the size of the database $|\mathbf{D}| = mk$, and ii) near optimal download rate as the sender's PIR message only grows with k.

In terms of computation, the scheme preserves the computational complexity for the receiver. However, the sender now as to run k PIR second message which makes its work to grow proportionally with $|\mathbf{D}| \cdot k$. Setting k to be sublinear in m yields subquadratic work in the size of the database for the sender.

Multiple-query co-PIR via recursion. We now discuss how to obtain a co-PIR where the receiver's input is a set S of indices instead of a single index.

Assume that the message of the sender can be decomposed into bit ciphertexts. That is, copir_2 can be decomposed into $(h, \alpha_1, \ldots, \alpha_m)$ where h is a header of size $\mathsf{poly}(\lambda)$ and each α_i decrypts to $D_i \in \{0,1\}$, where $\mathbf{D} = (D_1, \ldots, D_m)$ is the sender's input.

The crucial idea of this transformation is that, since the underlying one-query co-PIR has near optimal download rate, the sender can recurse the co-PIR without any blowup in the communication. Concretely the protocol works as follows:

- The receiver sends t first one-query co-PIR messages $\{\mathsf{copir}_{1,i}\}_{i\in[t]}$ to the sender, each one encoding an index a_i to be erased.
- The sender computes the first one-query co-PIR message $\mathsf{copir}_{2,1}$ using the input database $\mathbf{D} \in \{0,1\}^m$ and $\mathsf{copir}_{1,1}$. Recall that $\mathsf{copir}_{2,1}$ can be decomposed into $(h_1, \alpha_1^{(1)}, \ldots, \alpha_m^{(1)})$. The sender now creates a second $\mathsf{copir}_{2,2}$ using a new database $\mathbf{D}_1 = (\alpha_1^{(1)}, \ldots, \alpha_m^{(1)})s$ and $\mathsf{copir}_{1,2}$. The sender repeats this process until it obtains $\mathsf{copir}_{2,t}$ (together with all previous headers) and sends this to the receiver.
- The receiver can recursively decrypt each $\mathsf{copir}_{2,t+1-i}$ for $i \in [t]$. At each step, the a_{t+1-i} position of $(\alpha_1^{(t+1-i)}, \ldots, \alpha_m^{(t+1-i)})$ is erased and information about $D_{a_{t+1-i}}$ is statistically erased.

Since the underlying one-query co-PIR has near optimal download rate, each iteration of the recursion maintains this property as long as t is sublinear in the size of the initial database $\mathbf{D}$. Furthermore, if the starting co-PIR is statistically secure against malicious receivers, then so is the transformed co-PIR.

Achieving statistical sender privacy against malicious receivers. So far we have only discussed how to achieve semi-honest statistical security. It remains to show how to turn the protocol statistically secure for the sender against malicious receivers who might send malformed first round messages.

If we are able to guarantee that the receiver's message is well-formed, then we can use the semi-honest (statistical) security to argue malicious (statistical) security. As a first approach, we will discuss how to use a statistically sender secure *conditional disclosure of secrets* (CDS) to achieve the stronger notion of security.

Let $\mathcal{L}$ be an NP language. Recall that in a CDS scheme, the receiver holding a witness w for a statement x, sends a first message cds_1 to the sender that commits to w. The sender holding a message m computes a second CDS message cds_2 which allows the receiver to retrieve m iff $x \in \mathcal{L}$ and w is a valid witness. In terms of security, we want that if $x \notin \mathcal{L}$, then m is statistically hidden from the receiver.

It is well-known that statistically sender secure CDS schemes for NC1 can be constructed using (low-rate) SSP OT and information theoretic garbled circuits [IK00, AIK04, App17]. Moreover, any NP language can be verified by a NC1 circuit [GGH+13].

In order to achieve statistical sender security against malicious receivers, we will use a CDS that guarantees that the receiver's message is well-formed. Consider the following language

$$\mathcal{L}_{\mathsf{CoPIR}'} = \{\mathsf{copir}_1 : \exists (S, r) \text{ s.t. } \mathsf{copir}_1 \leftarrow \mathsf{CoPIR}'.\mathsf{Query}(S; r)\}$$

that is the language of well formed receiver's messages, where CoPIR' is the semi-honest protocol[6] described in the previous sections. Additionally, recall that our co-PIR scheme has a decomposability feature that the sender's message copir_2 can be decomposed into $(\alpha_1, \ldots, \alpha_m)$ where each α_i encodes $\mathbf{D}_i$.

The statistically secure protocol can be roughly described as follows:

- The receiver sends a co-PIR message $\mathsf{copir}_1 \leftarrow \mathsf{CoPIR}'.\mathsf{Query}(S; r)$ for a set of indices S of size t using random coins r. It additionally sends a first message cds_1 for language $\mathcal{L}_{\mathsf{CoPIR}'}$ using (S, r) as the witness.
- The sender computes $\mathsf{copir}_2 \leftarrow \mathsf{CoPIR}'.\mathsf{Send}(\mathsf{copir}_1, \mathbf{D})$ and decomposes copir_2 into $(\alpha_1, \ldots, \alpha_m)$. It now samples random $\beta_1, \ldots, \beta_t$ and computes

  $$\mathbf{v} = (\alpha_1, \ldots, \alpha_m) + (\beta_1, \ldots, \beta_t, 0, \ldots, 0)$$

  that is the first t coordinates of copir_2 are hidden using $\beta_1, \ldots, \beta_t$. It now sends a CDS message $\mathsf{cds}_2 \leftarrow \mathsf{CDS}.\mathsf{Send}(\mathsf{cds}_1, (\beta_1, \ldots, \beta_t))$ encrypting the values $(\beta_1, \ldots, \beta_t)$.

If copir_1 is well-formed then the receiver can retrieve the values $\beta_1, \ldots, \beta_t$, recover copir_2 and retrieve $(\alpha_1, \ldots, \alpha_m)$. In this case, we can use the semi-honest (statistical) security of the underlying CoPIR' to argue that the scheme is statistically secure. On the other hand, if copir_1 is malformed, then the values $\beta_1, \ldots, \beta_t$ are statistically hidden from the receiver given that CDS is statistically secure. In this case, the values $\alpha_1, \ldots, \alpha_t$ are statistically hidden from the receiver's point of view and thus, the first t positions of $\mathbf{D}$ are statistically hidden.

In terms of communication, the scheme has near optimal download rate as the CDS communication only depends on t (i.e., the size of the receiver's message) and this is typically set to be sub-linear in the size of the database.

While this gives us a generic solution to achieve SSP co-PIR, it incurs in a huge overhead as we need to make non black-box use of the underlying semi-honest co-PIR.

A black-box solution. To achieve a better concrete efficiency, we show how to build a black-box CDS scheme specifically for our purposes.

Recall that the receiver's message is composed by a ciphertext encrypting a square matrix of size k.[7] That is, a well formed receiver's message consists of $\mathsf{ct} \leftarrow \mathsf{LHE}.\mathsf{Enc}(\mathsf{pk}, b \cdot \mathbf{I})$ where $b \in \{0, 1\}$ and $\mathbf{I}$ is the identity matrix. However, if the receiver behaves maliciously, then it can encrypt any matrix $\mathbf{A}$ so that it learns partial (or total) information about the erased block.

Algebraic restriction codes. This is where algebraic restriction (AR) [ADD+22] codes come into play. Roughly speaking, an AR code restricts the class of functions that an adversary can apply over an encoded value.

More precisely, let $\hat{\mathbf{y}}_i \leftarrow \mathsf{AR}.\mathsf{Encode}(\mathbf{y}_i)$ for $i = 1, 2$. The work of [ADD+22] provides a construction of AR codes that restrict the class of any linear function $g(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2) = \hat{\mathbf{y}}_1 \cdot \mathbf{A} + \hat{\mathbf{y}}_2$ over the encoded values to the class of $f(\mathbf{y}_1, \mathbf{y}_2) = \mathbf{y}_1 \cdot (c \cdot \mathbf{I}) + \mathbf{y}_2$ where $\mathbf{I}$ is the identity matrix and $c \in \mathbb{Z}_p$. The security of AR codes allow to statistically simulate the evaluation of g over two encoded values $\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2$ given just the output of the decoding $f(\mathbf{y}_1, \mathbf{y}_2)$ where f is a function that depends on g.

[6] Technically speaking, we need the co-PIR scheme to be semi-malicious secure but for the sake of this overview, we will ignore this difference. Our co-PIR scheme constructed before satisfies semi-malicious security.

[7] The receiver's message is actually composed by several of these ciphertexts but for simplicity we assume that we only have one ciphertext.

A CDS from AR codes. Our main idea is to recast the construction of [ADD+22] in terms of CDS. Specifically, the sender sets

$$\mathbf{y}_1 = \begin{pmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \end{pmatrix} \text{ and } \mathbf{y}_1 = \begin{pmatrix} \mathbf{m} \\ \mathbf{m} - \mathbf{r}_1 \end{pmatrix}$$

where $\mathbf{r}_0, \mathbf{r}_1 \leftarrow\$ \mathbb{Z}_p^k$ and $\mathbf{m}$ is the message that the sender wants to send.

However, instead of evaluating $F(\mathbf{X}) = \mathbf{y}_1 \cdot \mathbf{X} + \mathbf{y}_2$ (that depends on $\mathbf{y}_1, \mathbf{y}_2$) over ct, it first AR encodes both $\mathbf{y}_1, \mathbf{y}_2$, applies $G(\mathbf{X}) = \hat{\mathbf{y}}_1 \cdot \mathbf{X} + \hat{\mathbf{y}}_2$ and finally homomorphically decodes the result.[8]

The AR codes security guarantees that the receiver will decrypt to something of the form

$$\mathbf{y}_1 \cdot (c\mathbf{I}) + \mathbf{y}_2 = \begin{pmatrix} c\mathbf{r}_0 + \mathbf{m} \\ c\mathbf{r}_1 + \mathbf{m} - \mathbf{r}_1 \end{pmatrix}.$$

If $c = 0, 1$ then the receiver can retrieve $\mathbf{m}$. However, if $c \neq 0, 1$ then the message $\mathbf{m}$ is statistically hidden from the receiver.

2.4 Consistency of Inputs

Recall that we need a mechanism to ensure that the same set is used by a malicious receiver to generate a PIR and a co-PIR message. First, note that the underlying PIR also needs to be statistically sender secure, and this can be instantiated using the scheme of [ADD+22]. Our second crucial observation is that the co-PIR receiver message comprises of a public key pk and encryptions of $a_i \cdot \mathbf{I}$ for $a_i \in \{0, 1\}$ which means that the co-PIR messages are identical to the PIR messages of the scheme of [ADD+22]![9] This means that the receiver does not have to send separate PIR messages: the sender can just interpret the co-PIR messages as PIR messages and this guarantees consistency of inputs.

2.5 Well-formedness of Ciphertexts

The last missing piece is how to ensure that the ciphertexts encrypting the LPN secrets are well-formed. These ciphertexts need to have a special structure i.e., they need to be encrypting bits, but in the current form, there is nothing that prevents the adversary from sending malformed ciphertexts and learn additional information.

Unfortunately, we cannot use a generic CDS protocol as we will lose optimal sender's message length or statistical security against the receivers. This is where we use rate-1 CDS. A rate-1 CDS is a standard statistical sender secure CDS with one additional efficiency property: we require the size of the sender's message to be $|m| + o(|m|)$ for sufficiently long m (which is larger than the size of the NP verification).

Assume for now that we have a download rate-1 CDS scheme which is statistically secure against malicious receivers. The sender encrypts its OT message using this CDS, and this message will be released to the receiver iff the ciphertexts are well-formed. Since the CDS scheme is rate-1, there is no blowup in the size of the sender's message. Moreover, the receiver's CDS message is small as its size only depends on the size of the LPN secrets and the size of the NP relation to be verified is independent of the size of the sender's input.

To construct such a rate-1 CDS scheme, we plug the (download rate-1) OT scheme of [ADD+22] together with the encryption scheme of [IP07] which yields a CDS scheme for branching programs (which contains NC1 circuits) and this is sufficient for our purposes.

2.6 Future Directions

Except for the use of the general purpose rate-1 CDS scheme used in the last step, our scheme uses only black-box techniques in the sense that it does not use explicit circuit-level description of cryptographic

[8] For this to work, we need the decoding function of the AR codes to be a linear function and this is indeed the case for AR codes from [ADD+22].

[9] Our actual co-PIR scheme is a bit more complex as it also contains PIR messages as a result of the block-to-bit transformation. However, our co-PIR scheme is still "PIR-compatible" for a variant of the PIR scheme of [ADD+22].

primitives. Coming up with a black-box technique that guarantees well-formedness of the ciphertexts of the receiver is a interesting open problem.

3 Preliminaries

The acronym PPT denotes "probabilistic polynomial time". Throughout this work, λ denotes the security parameter. By $\mathsf{negl}(\lambda)$, we denote a negligible function in λ, that is, a function that vanishes faster than any inverse polynomial in λ. Let $n \in \mathbb{N}$. Then, $[n]$ denotes the set $\{1, \ldots, n\}$. If $\mathcal{A}$ is an algorithm, we denote by $y \leftarrow \mathcal{A}(x)$ the output y after running $\mathcal{A}$ on input x. If S is a (finite) set, we denote by $x \leftarrow\!\!\$\, S$ the experiment of sampling uniformly at random an element x from S. If D is a distribution over S, we denote by $x \leftarrow\!\!\$\, D$ the element x sampled from S according to D. We denote by $S[i]$ the i-th element of S (where the elements are ordered by ascending order except when explicitly stated otherwise)

For two probability distributions X, Y, we use the notation $X \approx_s Y$ to state that the distributions are statistically indistinguishable.

For two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{F}^n$ over a finite field $\mathbb{F}$, we denote by $\mathbf{u} \odot \mathbf{v}$ their component-wise multiplication. We denote by $\mathsf{Supp}(\mathbf{u})$ the support of $\mathbf{u}$, that is, the set of indices where $\mathbf{u}$ is different from 0.[10] For $S \subseteq [n]$, $\mathbf{u}_S$ denotes the vector $(\mathbf{u}_i)_{i \in S}$. Finally, $\mathbf{u}^T$ denotes the transpose of $\mathbf{u}$ and $\mathsf{hw}(\mathbf{u})$ denotes the Hamming weight of $\mathbf{u}$ (that is, the number of coordinates of $\mathbf{u}$ different from 0).

Let $\mathsf{Diag}(n, \mathbf{v})$ be the algorithm that takes a vector $\mathbf{v} = (v_1, \ldots, v_n) \in \{0,1\}^n$ and outputs a matrix

$$\mathbf{D} = \begin{pmatrix} v_1 & & 0 \\ & \ddots & \\ 0 & & v_n \end{pmatrix} \in \{0,1\}^{n \times n},$$

i.e. $\mathbf{D} \in \{0,1\}^{n \times n}$ is a diagonal matrix with the components of $\mathbf{v}$ on its diagonal.

Addtionally let $\mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{v})$ be the algorithm that takes $i \in [\ell]$ and a row-vector $\mathbf{v} \in \{0,1\}^n$ and outputs a matrix

$$\mathbf{V} = \begin{pmatrix} 0 & \ldots & 0 \\ \vdots & & \vdots \\ 0 & \ldots & 0 \\ — & \mathbf{v} & — \\ 0 & \ldots & 0 \\ \vdots & & \vdots \\ 0 & \ldots & 0 \end{pmatrix} \in \{0,1\}^{\ell \times n},$$

i.e. the i-th row of $\mathbf{V}$ is $\mathbf{v}$, but $\mathbf{V}$ is 0 everywhere else.

3.1 Algebraic Restriction Codes

We recall the definition of algebraic restriction (AR) codes presented in [ADD+22]. In essence, an AR code restricts an adversary to evaluate a specific function over an encoding.

Definition 1. *An algebraic restriction (AR) code for a function family $\mathcal{F}$ is composed by a tuple of algorithms* $(\mathsf{Encode}, \mathsf{Eval}, \mathsf{Decode})$ *such that*

- $\mathsf{Encode}(s, x)$ *takes as input a seed s and an input x. it outputs a codeword c.*
- $\mathsf{Eval}(c, f)$ *takes as input a codeword c and a function $f \in \mathcal{F}$. It outputs a new codeword d.*
- $\mathsf{Decode}(s, d)$ *takes as input a seed s and a codeword d. It outputs a value y.*

[10] If there is only one index different from zero, $\mathsf{Supp}(\mathbf{u})$ denotes this index.

An AR code should fulfill the following properties.

Definition 2 (Correctness)**.** *An AR code* $\mathsf{AR} = (\mathsf{Encode}, \mathsf{Eval}, \mathsf{Decode})$ *is said to be correct if for all seeds* s*, all inputs* x *and all functions* $f \in \mathcal{F}$ *we have that*

$$\Pr\left[\mathsf{Decode}(s, \mathsf{Eval}(\mathsf{Encode}(s, x), f)) = f(x)\right] = 1.$$

Let $\mathcal{G}$ and $\mathcal{F}$ be two classes of functions. AR codes gives us the guarantee that if we evaluate a function from $\mathcal{G}$ then we learn no extra information as if we had evaluate some function from $\mathcal{F}$. This intuition is captured by a simulation security definition.

Definition 3 (Restriction security)**.** *An AR code* $\mathsf{AR} = (\mathsf{Encode}, \mathsf{Eval}, \mathsf{Decode})$ *is* $\mathcal{G}$*-*$\mathcal{F}$ *restriction secure if there exists an extractor* Ext *and a simulator* Sim *such that for every* x *and every function* $g \in \mathcal{G}$ *we have that for all adversaries* $\mathcal{A}$

$$\left|\Pr\left[1 \leftarrow \mathcal{A}(s, g(\mathsf{Encode}(s, x)), \mathsf{aux})\right] - \Pr\left[1 \leftarrow \mathcal{A}(s, \mathsf{Sim}(s, \mathsf{aux}, f(x)), \mathsf{aux})\right]\right| \leq \mathsf{negl}(\lambda)$$

where s *is chosen uniformly at random and* $(f, \mathsf{aux}) \leftarrow \mathsf{Ext}(g)$*.*

The work of [ADD+22] constructs AR codes that restricts the adversary from arbitrary linear functions to simple linear functions.

Lemma 1 ([ADD+22])**.** *Let* $q, n > 0$*,* $m > 2n + 2 + 2\lambda$ *and* $\mathbb{F}_q$ *be a field of order* q*. Consider the following classes of functions:*

- $\mathcal{F}$ *consists of all functions* $f : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{F}_q^n$ *of the form* $f(\mathbf{x}, \mathbf{y}) = a \cdot \mathbf{x} + \mathbf{y}$ *where* $a \in \mathbb{F}_q$*.*
- $\mathcal{G}$ *consists of all functions* $g : \mathbb{F}_q^m \times \mathbb{F}_q^m \to \mathbb{F}_q^m$ *of the form* $g(\mathbf{x}, \mathbf{y}) = \mathbf{x}\mathbf{A} + \mathbf{y}$ *where* $\mathbf{A} \in \mathbb{F}_q^{m \times m}$*.*

Then, there exists an AR code AR *that statistically restricts* $\mathcal{G}$ *to* $\mathcal{F}$*. Importantly, the decoding algorithm of the construction can be described as a linear function. That is, given a codeword* $\mathbf{c} \leftarrow \mathsf{AR.Encode}(s, \mathbf{x})$*, the output of the algorithm* Decode *can be obtained by computing* $\mathbf{x} = \mathbf{R}_s \mathbf{c}^T$ *for some matrix* $\mathbf{R}_s$ *defined by the seed* s*.*

3.2 Learning Parity with Noise

Informally, the LPN assumption states that it is hard to find a solution for a noisy system of linear equations over $\mathbb{Z}_2$. We now give the precise definition of the assumption.

Definition 4 (LPN assumption)**.** *Let* $n, m, t \in \mathbb{N}$ *such that* $n \in \mathsf{poly}(\lambda)$ *and let* $\chi_{m,t}$ *be uniform distribution over the set of error vectors of size* m *and hamming weight* t*. The* Learning Parity with Noise *(LPN) assumption* $\mathsf{LPN}(n, m, \rho)$ *holds if for any PPT adversary* $\mathcal{A}$ *we have that*

$$\left|\Pr\left[1 \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{s}\mathbf{A} + \mathbf{e}) : \begin{array}{c}\mathbf{A} \leftarrow\!\!\$\, \{0,1\}^{n \times m} \\ \mathbf{s} \leftarrow\!\!\$\, \{0,1\}^{n} \\ \mathbf{e} \leftarrow\!\!\$\, \chi_{m,t}\end{array}\right] - \Pr\left[1 \leftarrow \mathcal{A}(\mathbf{A}, \mathbf{y}) : \begin{array}{c}\mathbf{A} \leftarrow\!\!\$\, \{0,1\}^{n \times m} \\ \mathbf{y} \leftarrow\!\!\$\, \{0,1\}^{m}\end{array}\right]\right| \leq \mathsf{negl}(\lambda)$$

where $\rho = m/t$ *(*ρ *is called the noise rate).*

In this work, we assume that the noise rate ρ is $m^{1-\varepsilon}$ for any constant $\varepsilon > 0$. The LPN assumption is believed to be hard for that noise rate (see e.g. [BCG+19a] and references therein).

We also use the LPN over large fields assumption $\mathsf{LPN}(n, m, \rho, q)$ where the all components come from a larger field $\mathbb{Z}_q$ instead of $\{0, 1\}$. This assumption has been extensively used in previous works (e.g. [BCG+19a, JLS21]).

3.3 Cryptographic Primitives

3.3.1 Rate-1 Circuit-Private Linearly Homomorphic Encryption

Here we present the definition of packed LHE. The LHE that we need in this work should fulfill circuit privacy and have ciphertexts with rate 1.

Definition 5. *A (packed)* linearly homomorphic encryption *(LHE) scheme* LHE *over a finite group* $\mathbb{G}$ *is composed by a tuple of algorithms* $(\mathsf{Keygen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Shrink}, \mathsf{DecShrink})$ *such that:*

- $\mathsf{KeyGen}(1^\lambda, k)$ *takes as input a security parameter* λ *and* $k \in \mathbb{N}$*. It outputs a pair of public and secret keys* $(\mathsf{pk}, \mathsf{sk})$*.*
- $\mathsf{Enc}(\mathsf{pk}, \mathbf{m} = (m_1, \ldots, m_k))$ *takes as input a public key* pk *and a message* $\mathbf{m} = (m_1, \ldots, m_k) \in \mathbb{G}^k$*. It outputs a ciphertext* ct*.*
- $\mathsf{Eval}(\mathsf{pk}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$ *takes as input a public key* pk*, a linear function* $f : (\mathbb{G}^k)^\ell \to \mathbb{G}^k$ *and* ℓ *ciphertexts* $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$*. It outputs a new ciphertext* $\tilde{\mathsf{ct}}$*.*
- $\mathsf{Shrink}(\mathsf{pk}, \mathsf{ct})$ *takes as input a public key* pk *and a ciphertext* ct*. It outputs a new shrunken ciphertext* ct'*.*
- $\mathsf{DecShrink}(\mathsf{sk}, \mathsf{ct})$ *takes as input a secret key* sk *and a shrunken ciphertext* ct*. It outputs a message* $\mathbf{m}$*.*

For simplicity, we define the algorithm $\mathsf{Eval\&Shrink}(\mathsf{pk}, f, (\mathsf{ct}_1 \ldots, \mathsf{ct}_\ell))$ which outputs a ciphertext $\tilde{\mathsf{ct}}$ and is defined as

$$\mathsf{Eval\&Shrink}(\mathsf{pk}, f, (\mathsf{ct}_1 \ldots, \mathsf{ct}_\ell)) = \mathsf{Shrink}(\mathsf{pk}, \mathsf{Eval}(\mathsf{pk}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)))$$

for any linear function f.

We require the following properties from a (circuit-private) packed LHE: Correctness, semantic security, compactness and circuit-privacy.

Definition 6 (Correctness)**.** *A packed LHE scheme* LHE *is said to be correct if for any* $\ell \in \mathbb{N}$*, any messages* $\mathbf{m}_1, \ldots, \mathbf{m}_\ell$ *and any linear function* $f : (\mathbb{G}^k)^\ell \to \mathbb{G}^k$ *we have that*

$$\Pr\left[\tilde{\mathbf{m}} \leftarrow \mathsf{DecShrink}(\mathsf{sk}, \tilde{\mathsf{ct}}) : \begin{array}{c}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k) \\ \mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{m}_i) \text{ for } i \in [\ell] \\ \tilde{\mathsf{ct}} \leftarrow \mathsf{Eval\&Shrink}(\mathsf{pk}, , f, (\mathsf{ct}_1 \ldots, \mathsf{ct}_\ell))\end{array}\right] = 1$$

where $\tilde{\mathbf{m}} \leftarrow f(\mathbf{m}_1, \ldots, \mathbf{m}_\ell)$*.*

Definition 7 (Semantic Security)**.** *A packed LHE scheme* LHE *is said to be semantically secure if for all PPT* $\lambda \in \mathbb{N}$*, all* $k = \mathsf{poly}(\lambda)$ *and all adversaries* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ *we have that*

$$\left|\Pr\left[b \leftarrow \mathcal{A}_1(\mathsf{st}, \mathsf{ct}) : \begin{array}{c}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k) \\ (\mathbf{m}_0, \mathbf{m}_1, \mathsf{st}) \leftarrow \mathcal{A}_0(\mathsf{pk}) \\ b \leftarrow\!\!\$ \{0,1\} \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{m}_b)\end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda).$$

Definition 8 (Compactness)**.** *We require that a packed LHE scheme* LHE *has the following compactness properties:*

- *For* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k)$*, the size of the public key* $|\mathsf{pk}|$ *is bounded by* $k \cdot \mathsf{poly}(\lambda)$*.*
- *For any linear function* $f : (\mathbb{G}^k)^\ell \to \mathbb{G}^k$ *and any* $(\mathbf{m}_1, \ldots, \mathbf{m}_\ell) \in (\mathbb{G}^k)^\ell$ *we have that*

$$\lim_{\lambda \to \infty} \inf \frac{|f(\mathbf{m}_1, \ldots, \mathbf{m}_\ell)|}{|\mathsf{Eval\&Shrink}(\mathsf{pk}, , f, (\mathsf{ct}_1 \ldots, \mathsf{ct}_\ell))|} \to 1$$

for sufficiently large k*, where* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k)$ *and* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{m}_i)$ *for* $i \in [\ell]$*. In this case, we say that the scheme has rate 1.*

We also need that the packed LHE scheme fulfills circuit privacy in the malicious case.

Definition 9 (Circuit privacy)**.** *A packed LHE scheme* LHE *is said to be statistically circuit-private if for all messages* $(\mathbf{m}_1, \ldots, \mathbf{m}_\ell) \in (\mathbb{G}^k)^\ell$ *and all linear functions* $f : (\mathbb{G}^k)^\ell \to \mathbb{G}^k$*, there exists a simulator* Sim *such that for all public keys* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k)$ *and all ciphertexts* $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{m}_1)$ *we have that*

$$\left\{(\mathsf{pk}, \mathsf{sk}, \tilde{\mathsf{ct}}) : \tilde{\mathsf{ct}} \leftarrow \mathsf{Eval\&Shrink}(\mathsf{pk}, , f, (\mathsf{ct}_1 \ldots, \mathsf{ct}_\ell))\right\} \approx_s \left\{(\mathsf{pk}, \mathsf{sk}, \tilde{\mathsf{ct}}) : \tilde{\mathsf{ct}} \leftarrow \mathsf{Sim}(\mathsf{pk}, \tilde{\mathbf{m}})\right\}.$$

In other words, since Sim does not use f to compute $\tilde{\mathsf{ct}}$, no information about it is leaked from $\tilde{\mathsf{ct}}$ (apart from what is trivially leaked by f).

Encryption of matrices. Above, we defined LHE that supports encryption of vectors $\mathbf{m} \in \mathbb{G}^k$. We can easily extend the definition to support encryption of matrices $\mathbf{M} \in \mathbb{G}^{k \times \alpha}$ for any $\alpha = \mathsf{poly}(\lambda)$: Given a public key pk, an encryption $\mathsf{Enc}(\mathsf{pk}, \mathbf{M})$ of $\mathbf{M}$ is defined as

$$\mathsf{Enc}(\mathsf{pk}, \mathbf{M}) = \begin{pmatrix} | & & | \\ \mathsf{Enc}\left(\mathsf{pk}, \mathbf{m}^{(1)}\right) & \ldots & \mathsf{Enc}\left(\mathsf{pk}, \mathbf{m}^{(\alpha)}\right) \\ | & & | \end{pmatrix}$$

where $\mathbf{m}^{(i)}$ is the i-th column of $\mathbf{M}$.

Instantiations. The work of [BBDP22] presents a construction that fulfills the requirements above from the DDH assumption where $\mathbb{G} = \mathbb{Z}_q$ for any $q = \mathsf{poly}(\lambda)$.

3.3.2 Conditional Disclosure of Secrets

We now define the sintax of CDS.

Definition 10 (Conditional disclosure of secrets)**.** *A conditional disclosure of secrets (CDS) scheme* CDS *for a language* $\mathcal{L}$ *is composed by the following algorithms:*

- $\mathsf{Enc}_{\mathcal{L}}(1^\lambda, w)$ *takes as input the security parameter* λ *and a witness* $w \in \mathcal{W}$*. It outputs a first message* cds_1 *and a state* st*.*
- $\mathsf{Send}_{\mathcal{L}}(\mathsf{cds}_1, x, m)$ *takes as input a first message* cds_1*, a statement* $x \in \mathcal{X}$ *and a message* m*. It outputs a second message* cds_2*.*
- $\mathsf{Release}(\mathsf{cds}_2, \mathsf{st})$ *takes as input a second message* cds_2 *and a private state* st*. It outputs a message* m*.*

Definition 11 (Correctness)**.** *A CDS scheme* CDS *is said to be correct if for any* $x \in \mathcal{L}$ *and any message* m *we have that*

$$\Pr\left[m \leftarrow \mathsf{Release}(\mathsf{cds}_2, \mathsf{st}) : \begin{array}{c}(\mathsf{cds}_1, \mathsf{st}) \leftarrow \mathsf{Enc}(1^\lambda, w) \\ \mathsf{cds}_2 \leftarrow \mathsf{Send}(\mathsf{cds}_1, x, m)\end{array}\right] = 1.$$

Definition 12 (Receiver security)**.** *A CDS scheme* CDS *is said to be receiver secure if for any adversary* $\mathcal{A}$ *and any pair of* $(w_0, w_1) \in \mathcal{W}^2$ *we have that*

$$\left| \begin{array}{c}\Pr\left[1 \leftarrow \mathcal{A}(\mathsf{cds}_1) : (\mathsf{cds}_1, \mathsf{st}) \leftarrow \mathsf{Enc}(1^\lambda, w_0)\right] - \\ \Pr\left[1 \leftarrow \mathcal{A}(\mathsf{cds}_1) : (\mathsf{cds}_1, \mathsf{st}) \leftarrow \mathsf{Enc}(1^\lambda, w_1)\right]\end{array} \right| \leq \mathsf{negl}(\lambda).$$

Definition 13 (Statistical sender security)**.** *A CDS scheme* CDS *is said to be statistical sender secure if there is a (possibly computationally inefficient) extractor* $\mathsf{CDS.Ext}$ *such that for any message* cds_1 *and any pair of messages* (m_0, m_1)*, if* $(x, w) \notin \mathcal{R}_{\mathcal{L}}$ *(where* $\mathcal{R}_{\mathcal{L}}$ *is the relation of* $\mathcal{L}$*) then*

$$\mathsf{Send}(\mathsf{cds}_1, x, m_0) \approx_s \mathsf{Send}(\mathsf{cds}_1, x, m_1)$$

where $w \leftarrow \mathsf{CDS.Ext}(\mathsf{cds}_1)$*.*

Recall that any NP verification algorithm can be described as a NC1 circuit [GGH+13]. Then we have the following lemma.

Lemma 2 ([IK02, OPP14])**.** *Assuming the existence of SSP-OT together with an information-theoretical version of Yao's garbling, there exists a statistically sender secure CDS for functions in NC1.*

3.3.3 Rate-1 Statistically Sender Secure Conditional Disclosure of Secrets

In this section we show how we can recast the results in [ADD+22] as a rate-1 CDS. A rate-1 CDS is a CDS scheme where the ratio between the sender's message and sender's input is close to 1. Moreover the receiver's CDS message is independent of the size of the sender's input.

Recall that [ADD+22] gives an download rate-1 SSP OT which together with [IP07] gives us a two-round protocol (or homomorphic encryption for branching programs) with the following properties: i) statistical branching program privacy, and ii) semi-compactness which means that the size of the evaluated ciphertext grows with the depth of the branching program but not with its size. Let $\mathsf{2PC} = (\mathsf{Enc}, \mathsf{Eval}, \mathsf{Receive})$ be such a protocol where

- $\mathsf{Enc}(1^\lambda, x)$ takes as input a value x. It outputs ct and a private state st.
- $\mathsf{Eval}(\mathsf{ct}, \mathsf{B})$ takes as input a ciphertext and a branching program B. It outputs an evaluated ciphertext $\tilde{\mathsf{ct}}$.
- $\mathsf{Receive}(\tilde{\mathsf{ct}}, \mathsf{st})$ takes as input an evaluated ciphertext $\tilde{\mathsf{ct}}$ and a private state st. It outputs a value y.

In terms of correctness, we want that $y = \mathsf{B}(x)$ with overwhelming probability. In terms of security, we want that i) receiver security holds if $\mathsf{Enc}(1^\lambda, x_1)$ and $\mathsf{Enc}(1^\lambda, x_2)$ are computationally indistinguishable for any two x_1, x_2; and ii) statistical sender security holds if there exists an extractor Ext such that for any two branching programs $\mathsf{B}_1, \mathsf{B}_2$ such that $\mathsf{B}_1(x) = \mathsf{B}_2(x)$ then $\mathsf{Eval}(\mathsf{ct}, \mathsf{B}_1) \approx_s \mathsf{Eval}(\mathsf{ct}, \mathsf{B}_2)$ for any ct and where $x \leftarrow \mathsf{Ext}(\mathsf{ct})$.

Lemma 3 ([IP07, ADD+22])**.** *There exists a statistically sender secure homomorphic encryption scheme* $\mathsf{2PC} = (\mathsf{Enc}, \mathsf{Eval}, \mathsf{Receive})$ *as described above such that*

- $|\mathsf{ct}| = \mathsf{poly}(|x|, \lambda)$ *where* $\mathsf{ct} \leftarrow \mathsf{Enc}(1^\lambda, x)$.
- $|\tilde{\mathsf{ct}}| = |\mathsf{B}(x)| + \mathsf{poly}(D, \lambda)$ *where* $\tilde{\mathsf{ct}} \leftarrow \mathsf{Eval}(\mathsf{ct}, \mathsf{B})$ *and* D *is the depth of* B.

Recall that any verification circuit can be implemented by an NC1 circuit [GGSW13] and that any NC1 can be described as a branching program whose depth depends polynomially on the size of the input (which for a NP verification circuit depends polynomially on the size of the statement and the size of the witness).

Corollary 1. *There exists a two-round CDS scheme for any NP language where:*

- *semi-honest receiver security holds under the DDH assumption and malicious sender security holds statistically.*
- *The size of* cds_1 *is* $\mathsf{poly}(|x|, |w|, \lambda)$ *and the size of* cds_2 *is* $|\mathbf{m}| + \mathsf{poly}(|x|, |w|, \lambda)$ *where* $\mathsf{cds}_1 \leftarrow \mathsf{CDS.Enc}(1^\lambda, w)$, $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, x, \mathbf{m})$, *$x$ is the statement and w is the witness.*

3.3.4 Private Information Retrieval

Private Information Retrieval (PIR) schemes [CGKS95] allow a user to retrieve the i-th bit of an n-bit database, without revealing to the database holder the value of i. We use PIR schemes with sender privacy [DMO00, ADD+22].

Definition 14 (PIR)**.** *A private information retrieval (PIR) scheme* PIR *is composed by the following algorithms:*

- $\mathsf{Query}(n, i)$ *takes as input an index* $i \in [n]$*. It outputs a query* q *and a state* st_i*.*
- $\mathsf{Send}(\mathbf{D}, \mathsf{q})$ *takes as input a database* $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_n) \in (\{0,1\}^B)^n$ *(where each* $\mathbf{D}_i \in \{0,1\}^B$*) and a message* q*. It outputs a response* r*.*
- $\mathsf{Retrieve}(\mathsf{r}, \mathsf{st}_i)$ *takes as input a response* r *and a state* st_i*. It retrieves the entry* $\mathbf{D}_i$*.*

We will slightly overload the notation and write $\mathsf{Query}(n, S)$ where S is a set of size t to denote $\{\mathsf{Query}(n, S[i])\}_{i\in[t]}$.

Definition 15 (Correctness)**.** *A PIR scheme* PIR *is said to be correct if for any* $n = \mathsf{poly}(\lambda)$*,* $\mathbf{D} \in (\{0,1\}^B)^n$ *and* $i \in [n]$*, we have that*

$$\Pr\left[\mathbf{D}_i = \mathsf{Retrieve}(\mathsf{st_i}, \mathsf{r}) : \begin{array}{c}(\mathsf{st_i}, \mathsf{q}) \leftarrow \mathsf{Query}(n, i) \\ \mathsf{r} \leftarrow \mathsf{Send}(\mathbf{D}, \mathsf{q})\end{array}\right] = 1.$$

Definition 16 (User privacy)**.** *A PIR scheme* PIR *is said to be user private if for any PPT adversary* $\mathcal{A}$*, any* $n, \lambda \in \mathbb{N}, \mathbf{D} \in (\{0,1\}^B)^n$ *and* $i, j \in [n]$*, we have that*

$$\left|\begin{array}{c}\Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathbf{D}, \mathsf{q_i}) : (\mathsf{st_i}, \mathsf{q_i}) \leftarrow \mathsf{Query(n, i)}] - \\ \Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathbf{D}, \mathsf{q_j}) : (\mathsf{st_j}, \mathsf{q_j}) \leftarrow \mathsf{Query(n, j)}]\end{array}\right| \leq \mathsf{negl}(\lambda).$$

Definition 17 (Statistical sender privacy)**.** *A PIR scheme* PIR *is said to be statistical sender private if there exists an extractor* Ext *such that for any* $\lambda \in \mathbb{N}$*, any* $n = \mathsf{poly}(\lambda)$*, any first message* q_i *we have that*

$$\{\mathsf{r}_i : \mathsf{r}_i \leftarrow \mathsf{Send}(\mathbf{D}^x, \mathsf{q}_i)\} \approx_s \{\mathsf{r}_i : \mathsf{r}_i \leftarrow \mathsf{Send}(\mathbf{D}^y, \mathsf{q}_i)\}$$

where $i \leftarrow \mathsf{Ext}(\mathsf{q}_i)$ *and* $\mathbf{D}^x, \mathbf{D}^y \in \{0,1\}^n$ *are two databases such that* $\mathbf{D}^x_i = \mathbf{D}^y_i$*.*

The ADD+ scheme. A scheme fulfilling these properties was presented in [ADD+22] and its security is based on DDH. We will refer to such scheme as ADD+ scheme. Additionally, this scheme achieves the following communication complexity: Let B be the size of the block received by the receiver and let $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m) \in \{0,1\}^{mB}$ where each $\mathbf{D}_i \in \{0,1\}^B$. Of course the size of the block retrieved by the receiver can be made arbitrarily large by making the sender to reuse the receiver's message.

- **Receiver's message.** The bitsize of q is upper-bounded by $B \cdot \mathsf{poly}(\lambda)$.
- **Sender's message.** The bitsize of r is upper-bounded by $|\mathbf{D}| + \mathsf{poly}(\lambda) \cdot \mathsf{polylog}(|\mathbf{D}|)$.

The [ADD+22] scheme makes use of a LHE scheme LHE with rate-1 ciphertexts. Moreover, the first message sent by the receiver for an input index $j \in [n]$ is composed by $\left(\mathsf{pk}, \{\mathsf{CT}_i\}_{i\in[mu]}\right)$ where each component is composed in the following way: i)$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, B)$, and ii) each $\mathsf{CT}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, j_i \cdot \mathbf{I}_B)$ where $(j_1, \ldots, j_\mu)$ is the binary decomposition of i and $\mathbf{I}_B$ is the identity matrix of size B.

3.3.5 Two-Round Statistical Sender Private Oblivious Transfer

Oblivious transfer (OT) allows the receiver to encode a bit b and send it to the sender. The sender creates a response given this encoding and a pair of messages (m_0, m_1). Finally, the receiver recovers m_b.

Definition 18. *A two-round statistical sender private oblivious transfer (SSP OT) scheme* OT *is composed by the following algorithms:*

- $\mathsf{OTR}(1^\lambda, \mathbf{b})$ *takes as input the security parameter* λ *and a vector* $\mathbf{b} = (b_1, \ldots, b_n) \in \{0,1\}^n$*. It outputs a message* ot_1 *and a private state* st*.*

- $\mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_0, \mathbf{m}_1)$ *takes as input a first OT message* ot_1 *and two vectors* $\mathbf{m}_0, \mathbf{m}_1 \in \{0,1\}^n$.[11] *It outputs a second OT message* ot_2.
- $\mathsf{OTD}(\mathsf{ot}_2, \mathsf{st})$ *takes as input a second OT message* ot_2 *and a private state* st. *It outputs a vector* $\tilde{\mathbf{m}}$.

An OT should be correct, sender secure and receiver secure. Here, we consider a special sender security definition called statistical sender security.

Definition 19 (Correctness)**.** *An OT scheme* OT *is said to be correct if for any* $n \in \mathsf{poly}(\lambda)$*, any pair of messages* $(\mathbf{m}_0, \mathbf{m}_1) \in (\{0,1\}^n)^2$ *and any* $b \in \{0,1\}$*, we have that*

$$\Pr\left[\mathbf{m}_b = \mathsf{OTD}(\mathsf{ot}_2, \mathsf{st}) : \begin{array}{c}(\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, b) \\ \mathsf{ot}_2 \leftarrow \mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_0, \mathbf{m}_1)\end{array}\right] = 1.$$

Definition 20 (Statistical sender privacy)**.** *An OT scheme* OT *is said to be statistical sender private if for any* $\lambda \in \mathbb{N}$*, any* $n = \mathsf{poly}(\lambda)$*, there exists an extractor* $\mathsf{OT.Ext}$ *such that for any two messages* $\mathbf{m}_0, \mathbf{m}_1 \in \{0,1\}^n$ *and any* ot_1 *we have that*

$$\mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_0, \mathbf{m}_1) \approx_s \mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_{b'}, \mathbf{m}_{b'})$$

where $b' \leftarrow \mathsf{OT.Ext}(\mathsf{ot}_1)$.

Definition 21 (Receiver security)**.** *An OT scheme* OT *is said to be receiver secure if for any PPT adversary* $\mathcal{A}$*,* λ *and any* $b, b' \in \{0,1\}$ *we have that*

$$\left|\begin{array}{c}\Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathsf{ot}_1) : (\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, b)] - \\ \Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathsf{ot}_1) : (\mathsf{ot}_1, \mathsf{st}) \leftarrow \mathsf{OTR}(1^\lambda, b')]\end{array}\right| \leq \mathsf{negl}(\lambda).$$

Efficiency. We consider two-round OT protocols where ot_1 is the message sent by the receiver and ot_2 is the message sent by the sender. Let $\mathbf{b} = (b_1, \ldots, b_n)$ be the input of the receiver and let $\mathbf{m_b}$ be its output.

We call *upload rate* to the value $\lim_{\lambda\to\infty} \inf \frac{|\mathsf{ot}_1|}{|\mathbf{b}|}$ and *download rate* to the value $\lim_{\lambda\to\infty} \inf \frac{|\mathsf{ot}_1|}{|\mathbf{m_b}|}$. When both values tend to 1 then we say that the scheme has overall rate 1.

4 Conditional Disclosure of Secrets for DDH-based Encryption

In this section we present a black-box construction of a CDS for a specific language. Namely, our scheme guarantees that an El-Gamal public key is well-formed and that a certain ciphertext encrypts a bit. The scheme fulfills statistical sender privacy.

Before presenting our CDS scheme, we extend the definition of packed LHE of Definition 5. This definition only considers operations over a polynomial size field. However, the El Gamal encryption also allows for operations over $\mathbb{Z}_p$ and this is captured by the algorithm $\mathsf{AltEval}_1$. The second algorithm $\mathsf{AltEval}_2$ captures a different type of homomorphism accross slots. After using this evaluation algorithm, the applied function needs to be known in order to decrypt the resulting ciphertext.

Definition 22 (Alternative Eval algorithm for LHE)**.** *Consider a rate-1 circuit-private LHE scheme* $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Shrink}, \mathsf{DecShrink})$ *that supports evaluation of ciphertexts over* $\mathbb{Z}_2$ *as described in Definition 5. We say that* LHE *supports alternative linear homomorphism if there are algorithms* $(\mathsf{AltEval}_1, \mathsf{AltEval}_2, \mathsf{AltDec})$ *such that*

- $\mathsf{AltEval}_1(\mathsf{pk}, (\mathbf{a}, \mathbf{b}), \mathsf{CT})$ *takes as input a matrix* CT *of ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_m$ *of size* n*,* $\mathbf{a} \in \mathbb{Z}_p^m$ *and* $\mathbf{b} \in \mathbb{Z}_p^n$*. It outputs a new ciphertext* ct.

[11] We use the term bit OT to denote the case where $n = 1$.

- $\mathsf{AltEval}_2(\mathsf{pk}, \mathbf{R} \in \mathbb{Z}_p^{m \times n}, \mathsf{ct})$ *takes as input a public key* pk*, a matrix* $\mathbf{R}$ *and a ciphertext* ct*. It outputs a new ciphertext* $\tilde{\mathsf{ct}}$*.*
- $\mathsf{AltDec}(\mathsf{sk}, \mathbf{R}, \tilde{\mathsf{ct}})$ *takes as input a secret key* sk*, a matrix* $\mathbf{R}$ *and a ciphertext* ct*. It outputs a message* $\mathbf{m}$*.*

The correctness requirement we need is the following: for all matrices $\mathbf{M} \in \mathbb{Z}_2^{n \times m}, \mathbf{a} \in \mathbb{Z}_p^m, \mathbf{b} \in \mathbb{Z}_p^n, \mathbf{R} \in \mathbb{Z}_2^{m \times n}$

$$\Pr\left[\mathbf{R} \cdot (\mathbf{M} \cdot \mathbf{a}^T + \mathbf{b}^T) = \mathsf{AltDec}(\mathsf{sk}, \mathbf{R}, \tilde{\mathsf{ct}}) : \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda, k) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathbf{M}) \\ \mathsf{ct} \leftarrow \mathsf{AltEval}_1(\mathsf{pk}, (\mathbf{a}, \mathbf{b}), \mathsf{CT}) \\ \mathsf{ct}^* \leftarrow \mathsf{AltEval}_2(\mathsf{pk}, \mathbf{R}, \mathsf{ct}) \\ \tilde{\mathsf{ct}} \leftarrow \mathsf{Shrink}(\mathsf{pk}, \mathsf{ct}^*) \end{array}\right] = 1$$

if each coordinate of $\mathbf{R} \cdot (\mathbf{M} \cdot \mathbf{a}^T + \mathbf{b}^T)$ *is in* $\mathsf{poly}(\lambda)$*.*

It is shown in [ADD+22] that the standard El Gamal encryption scheme supports these two types of homomorphism. We present the scheme in Appendix A. Additionally the scheme fulfills statistical malicious circuit privacy.

Construction. We will now focus on the construction. Let $\mu > 0$. In the following, consider the following language parametrized by a public key pk in the range of $\mathsf{LHE.KeyGen}$

$$\mathcal{L}_{\mathsf{pk}} = \left\{ \{\mathsf{ct}_i\}_{i \in [\mu]} : \begin{array}{c} \mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, a_i \cdot \mathbf{I}_\ell; r_i) \\ a_i \in \{0,1\} \end{array} \right\}$$

where $\mathbf{I}_\ell$ is the identity matrix of dimension ℓ.

Ingredients. We will need the following ingredients:

- Let $\mathsf{AR} = (\mathsf{Encode}, \mathsf{Eval}, \mathsf{Decode})$ be an AR code with linear decoding as the one presented in Lemma 1. That is, to decode a codeword $\mathbf{y}$, one computes $\mathbf{R}_s \cdot \mathbf{y}^T$ for a matrix $\mathbf{R}_s \in \mathbb{Z}_q^{2n \times \ell}$ specified by the seed s.[12]
- Let $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, , \mathsf{Shrink}, \mathsf{DecShrink})$ be a packed rate-1 circuit-private LHE with alternative evaluation algorithms $(\mathsf{AltEval}_1, \mathsf{AltEval}_2, \mathsf{AltDec})$ as described in Definition 22.
- Let $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, \ell)$.

Construction 1. *We now show how to build a CDS scheme for the languange* $\mathcal{L}_{\mathsf{pk}}$*.*

$\mathsf{Enc}(1^\lambda, w, \mathsf{aux} = \mathsf{sk})$**:**

- *Parse* w *as* $a_1, \ldots, a_\mu$*, where each* $a_i \in \{0,1\}$*.*
- *Output* $\mathsf{cds}_1 = \perp$ *and* $\mathsf{st} = (w, \mathsf{sk})$*.*

[12] Recall that AR codes restricts the class of functions that the adversary can evaluate evaluating. In our case, we use the construction of Lemma 1 which restricts the adversary to evaluating scalar functions of the form $f(\mathbf{x}, \mathbf{y}) = a \cdot \mathbf{x} + \mathbf{y}$ or equivalently $f(\mathbf{x}, \mathbf{y}) = \mathbf{I} \cdot \mathbf{x} + \mathbf{y}$ where $\mathbf{I}$ is the identity matrix.

$\mathsf{Send}(\mathsf{cds}_1, \mathcal{L}_{\mathsf{pk}}, \mathbf{d} \in \{0,1\}^n)$**:**

- *Sample* $\mathbf{d}_1, \ldots, \mathbf{d}_\mu \leftarrow\!\!\$\, \{0,1\}^n$ *such that* $\mathbf{d} = \sum_{i=1}^{\mu} \mathbf{d}_i$.
- *Sample a seed* $\mathbf{R}_s \leftarrow\!\!\$\, \mathbb{Z}_p^{2n \times \ell}$ *for the AR code.*
- *For all* $i \in [\mu]$*, sample* $\mathbf{r}_{0,i}, \mathbf{r}_{1,i} \leftarrow\!\!\$\, \mathbb{Z}_p^n$*. Set* $\mathbf{y}_{1,i} = \begin{pmatrix} \mathbf{r}_{0,i} \\ \mathbf{r}_{1,i} \end{pmatrix}$ *and* $\mathbf{y}_{2,i} = \begin{pmatrix} \mathbf{d}_i \\ \mathbf{d}_i - \mathbf{r}_{1,i} \end{pmatrix}$.
- *For all* $i \in [\mu]$*, compute* $\hat{\mathbf{y}}_{1,i} \leftarrow \mathsf{AR.Encode}(s, \mathbf{y}_{1,i})$ *and* $\hat{\mathbf{y}}_{2,i} \leftarrow \mathsf{AR.Encode}(s, \mathbf{y}_{2,i})$.
- *For all* $i \in [\mu]$*, let* $G_i(\mathbf{X}) = \mathbf{X} \cdot \hat{\mathbf{y}}_{1,i}^T + \hat{\mathbf{y}}_{2,i}^T$*. Compute* $\hat{\mathsf{ct}}_i \leftarrow \mathsf{LHE.AltEval}_1(\mathsf{pk}, G_i, \mathsf{ct}_i)$.
- *For all* $i \in [\mu]$*,compute* $\tilde{\mathsf{ct}}_i \leftarrow \mathsf{LHE.AltEval}_2(\mathsf{pk}, \mathbf{R}_s, \hat{\mathsf{ct}}_i)$
- *Output* $\mathsf{cds}_2 = (\{\tilde{\mathsf{ct}}_i\}_{i \in [\mu]}, \mathbf{R}_s)$.

$\mathsf{Release}(\mathsf{cds}_2, \mathsf{st})$**:**

- *Parse* cds_2 *as* $(\{\tilde{\mathsf{ct}}_i\}_{i \in [\mu]}, \mathbf{R}_s)$ *and* $\mathsf{st} = (w, \mathsf{sk})$ *where* $w = a_1, \ldots, a_\mu$.
- *For all* $i \in [\mu]$*, compute* $\mathbf{w}_i \leftarrow \mathsf{LHE.AltDec}(\mathsf{sk}, \mathbf{R}_s, \tilde{\mathsf{ct}}_i)$*. Parse* $\mathbf{w}_i = \begin{pmatrix} \mathbf{z}_{i,0} \\ \mathbf{z}_{i,1} \end{pmatrix}$ *and set* $\tilde{\mathbf{d}}_i = \mathbf{z}_{i,a_i}$.
- *Output* $\tilde{\mathbf{d}} = \sum_{i=1}^{\mu} \tilde{\mathbf{d}}_i$.

Lemma 4 (Correctness)**.** *The scheme presented in Construction 1 is correct assuming that* LHE *and* AR *are correct.*

Proof. By the correctness of the alternative evaluation algorithms, we have that for all $i \in [\mu]$

$$\begin{aligned}
\mathbf{w}_i = \begin{pmatrix} \mathbf{z}_{i,0} \\ \mathbf{z}_{i,1} \end{pmatrix} &= \mathsf{LHE.AltDec}(\mathsf{sk}, \mathsf{R}, \tilde{\mathsf{ct}}_i) \\
&= \mathbf{R}_s \cdot \left((a_i \cdot \mathbf{I}_\ell) \cdot \hat{\mathbf{y}}_{1,i}^T + \hat{\mathbf{y}}_{2,i}^T \right) = \mathbf{R}_s \cdot \left(a_i \cdot \hat{\mathbf{y}}_{1,i}^T + \hat{\mathbf{y}}_{2,i}^T \right) \\
&= a_i \cdot \mathbf{y}_{1,i}^T + \mathbf{y}_{2,i}^T = a_i \cdot \begin{pmatrix} \mathbf{r}_{0,i} \\ \mathbf{r}_{1,i} \end{pmatrix} + \begin{pmatrix} \mathbf{d}_i \\ \mathbf{d}_i - \mathbf{r}_{1,i} \end{pmatrix} \\
&= \begin{pmatrix} a_i \cdot \mathbf{r}_{0,i} + \mathbf{d}_i \\ a_i \cdot \mathbf{r}_{1,i} + \mathbf{d}_i - \mathbf{r}_{1,i} \end{pmatrix}
\end{aligned}$$

where the fourth equality follows from the correcness of the AR code. Hence, if $a_i = 0$ then $\tilde{\mathbf{d}}_i = \mathbf{z}_{i,0} = \mathbf{d}_i$. Else if $a_i = 1$ then $\tilde{\mathbf{d}}_i = \mathbf{z}_{i,1} = \mathbf{d}_i$.

Since $\tilde{\mathbf{d}}_i = \mathbf{d}_i$ for all $i \in [\mu]$, then we have that $\sum_{i=1}^{\mu} \tilde{\mathbf{d}}_i = \mathbf{d}$. □

Lemma 5 (Receiver security)**.** *The scheme presented in Construction 1 is receiver secure.*

This follows immediately from the fact that the receiver sends no additional information to the sender.

Lemma 6 (Statistical sender security)**.** *The scheme presented in Construction 1 is statistical sender secure assuming that* LHE *is statistically malicious circuit private and* AR *restricts functions of the form* $g(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2) = \mathsf{LHE.AltEval}_1(\mathsf{pk}, G_i, \mathsf{ct}_i)$ *to linear functions of the form* $f(\mathbf{y}_1, \mathbf{y}_2) = a\mathbf{y}_1 + \mathbf{y}_2$ *or* $f(\mathbf{y}_1, \mathbf{y}_2) = \mathbf{y}_1$.

Proof. We first describe the extractor CDS.Ext. This algorithm uses the AR extractor AR.Ext as a subroutine.

$\mathsf{CDS.Ext}(\mathsf{cds}_1)$**:**

- Run $\mathbf{A}_1, \ldots, \mathbf{A}_\mu \leftarrow \mathsf{LHE.Ext}(\mathsf{pk}, \mathsf{ct}_1, \ldots, \mathsf{ct}_\mu)$ where each $\mathbf{A} \in \mathbb{Z}_p^{\ell \times \ell}$.
- For all $i \in [\mu]$ define $g_i(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2) = \mathbf{A} \cdot \hat{\mathbf{y}}_1^T + \hat{\mathbf{y}}_2^T$.
- For all $i \in [\mu]$ run $(f_i, \mathsf{aux}_i) \leftarrow \mathsf{AR.Ext}(g_i)$.
- If f_i is of the form $f_i(\mathbf{y}_1, \mathbf{y}_2) = \mathbf{y}_1$, output $a_i = \perp$. Else if f_i is of the form $f_i(\mathbf{y}_1, \mathbf{y}_2) = a_i \mathbf{y}_1 + \mathbf{y}_2$, output a_i.

Hybrid $\mathcal{H}_0$. This is the real game.

Hybrid $\mathcal{H}_{1,i}$. This hybrid is identical to the previous one except that the simulator first extracts $\mathbf{A}_1, \ldots, \mathbf{A}_\mu \leftarrow \mathsf{LHE.Ext}(\mathsf{pk}, \mathsf{ct}_1, \ldots, \mathsf{ct}_\mu)$ where each $\mathbf{A} \in \mathbb{Z}_p^{\ell \times \ell}$. Then it computes $\mathbf{z}_i = \mathbf{R}_s \cdot \mathbf{A}_i \cdot \hat{\mathbf{y}}_{1,i}^T + \hat{\mathbf{y}}_{2,i}^T$ and computes $\tilde{\mathsf{ct}}_i \leftarrow \mathsf{LHE.Sim}(\mathsf{pk}, \mathbf{z}_i)$. This hybrid is defined for $i = 1, \ldots, \mu$.

Statistical indistinguishability of hybrids follows from the statistical malicious circuit privacy of the underlying LHE.

Hybrid $\mathcal{H}_{2,i}$. This hybrid is identical to the previous one except that the simulator extracts $(f_i, \mathsf{aux}_i) \leftarrow \mathsf{AR.Ext}(g_i)$. This hybrid is defined for $i = 1, \ldots, \mu$.

Now assume that $a_i \neq 0, 1$. In the following, let $\mathsf{AR.Sim}$ be the simulator of the AR code. Assume that f_i is of the form $f_i(\mathbf{y}_1, \mathbf{y}_2) = a_i \mathbf{y}_1 + \mathbf{y}_2$ where $a_i \neq 0, 1$. Then note that

$$\begin{aligned} f(\mathbf{y}_{1,i}\mathbf{y}_{2,i}) &= a_i \cdot \begin{pmatrix} \mathbf{r}_{0,i} \\ \mathbf{r}_{1,i} \end{pmatrix} + \begin{pmatrix} \mathbf{d}_i \\ \mathbf{d}_i - \mathbf{r}_{1,i} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{d}_i \\ \mathbf{d}_i \end{pmatrix} + \begin{pmatrix} a_i \mathbf{r}_{0,i} \\ (a_i - 1)\mathbf{r}_{1,i} \end{pmatrix} \\ &\approx_s \begin{pmatrix} \mathbf{r}_{0,i} \\ \mathbf{r}_{1,i} \end{pmatrix} \end{aligned}$$

where the last step follows from the fact that $a_i \neq 0, 1$ and both $\mathbf{r}_{0,i}, \mathbf{r}_{1,i}$ are uniformly distributed. Again, by the security of AR codes, we can replace $\hat{\mathsf{ct}}_i$ by $\mathsf{AR.Sim}(\mathsf{aux}, \mathbf{r})$ and this change goes unnoticed except with negligible probability. □

Communication complexity. We now analyze the communication complexity of the scheme. The sender's message cds_2 is composed by $\{\tilde{\mathsf{ct}}_i\}_{i \in [\mu]}, \mathbf{R}_s$ where

- $|\{\tilde{\mathsf{ct}}_i\}_{i \in [\mu]}| = \mu \cdot n \cdot \mathsf{poly}(\lambda)$
- $|\mathbf{R}_s| = 2n \cdot \ell \cdot \mathsf{poly}(\lambda)$.

Thus, the total communication complexity is $\mu \cdot n \cdot \mathsf{poly}(\lambda) + 2n \cdot \ell \cdot \mathsf{poly}(\lambda) = \mathsf{poly}(\mu, |\mathbf{d}|, \ell, \lambda)$.

Remark 1. *A CDS protocol can be seen as a two-round witness encryption scheme. We remark that the protocol presented in Construction 1 can also be seen as a witness encryption for the language $\mathcal{L}_{\mathsf{pk}}$ (where the witness that allows for decryption is the secret key associated with pk) since the first message by the receiver cds_1 is just the empty string.*

5 Definition of Co-Private Information Retrieval

We start by defining co-PIR in an identical way as in [BBDP22]. A co-PIR is a primitive that allows a sender to transmit a database to a receiver, except for some positions which will be (statistically) erased.

Definition 23 (Co-PIR)**.** *Let $\mathbb{H}$ be a group. A co-private information retrieval (co-PIR) scheme is parametrized by a integer $m = \mathsf{poly}(\lambda)$ and is given by a tuple of algorithms $(\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ with the following sintax:*

- $\mathsf{Query}(1^\lambda, S)$ *takes as input the security parameter λ and a subset of indices $S = \{i_1, \ldots, i_t\} \subseteq [m]$ of size t. It outputs a first co-PIR message copir_1 and a private state st.*
- $\mathsf{Send}(\mathsf{copir}_1, \mathbf{D})$ *takes as input a first co-PIR message copir_1 and a database $\mathbf{D} \in \mathbb{H}^m$.*[13] *It outputs a second co-PIR message copir_2.*
- $\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$ *takes as input a second co-PIR message copir_2 and a private state st. It outputs a database $\tilde{\mathbf{D}} \in \mathbb{H}^m$.*

A co-PIR scheme should fulfill the following properties.

Definition 24 (Correctness)**.** *A co-PIR scheme is said to be correct if for any $m = \mathsf{poly}(\lambda)$ and $S \subseteq [m]$*

$$\Pr\left[\mathbf{D}_{\bar{S}} = \tilde{\mathbf{D}}_{\bar{S}} : \begin{array}{c}(\mathsf{copir}_1, \mathsf{st}) \leftarrow \mathsf{Query}(1^\lambda, S) \\ \mathsf{copir}_2 \leftarrow \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}) \\ \tilde{\mathbf{D}} \leftarrow \mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})\end{array}\right] = 1$$

where $\bar{S} = [m] \setminus S$. In other words, $\mathbf{D}_i = \tilde{\mathbf{D}}_i$ for all $i \notin S$.

We also define a slightly stronger notion of security that we call locally correct.

Definition 25 (Locally correctness)**.** *A co-PIR scheme is locally correct if the following holds: i) copir_2 is of the form $(\alpha_1, \ldots, \alpha_m)$, and ii) The Rec algorithm can be divided into subalgorithms Rec_i such that $\mathbf{D}_i \leftarrow \mathsf{Rec}_i(\alpha_i, \mathsf{st})$ for all $i \notin S$.*

Definition 26 (Efficiency)**.** *A co-PIR scheme is said to be efficient if the it fulfills the following requirements:*

- $|\mathsf{copir}_1| = \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda, |S|)$.[14]
- ***Download rate 1:*** *If t is sublinear in $|\mathbf{D}|$ (that is $t = o(|\mathbf{D}|)$) then $\lim_{\lambda\to\infty} \sup \frac{|\mathsf{copir}_2|}{|\mathbf{D}|} \to 1$ for sufficiently large $|\mathbf{D}|$ where $\mathbf{D} \in \mathbb{H}^m$ and $\mathsf{copir}_2 \leftarrow \mathsf{Send}(\mathsf{copir}_1, \mathbf{D})$.*[15]

Definition 27 (Receiver security)**.** *A co-PIR scheme CoPIR is said to be receiver secure if for all $m = \mathsf{poly}(\lambda)$, any subsets $S_1, S_2 \subseteq [m]$ we have that for any adversary $\mathcal{A}$*

$$\left|\begin{array}{l}\Pr\left[1 \leftarrow \mathcal{A}(k, \mathsf{copir}_1) : (\mathsf{copir}_1, \mathsf{st}) \leftarrow \mathsf{Query}(1^\lambda, S_1)\right] - \\ \Pr\left[1 \leftarrow \mathcal{A}(k, \mathsf{copir}_1) : (\mathsf{copir}_1, \mathsf{st}) \leftarrow \mathsf{Query}(1^\lambda, S_2)\right]\end{array}\right| \leq \mathsf{negl}(\lambda).$$

[13] We use the term bit co-PIR to denote the case when $\mathbb{H} = \{0, 1\}$. Otherwise, we use the term block co-PIR.

[14] We usually consider co-PIR protocols where the first message depends polylogarithmically on the size of $\mathbf{D}$, similarly to PIR protocols. However, for our OT application in Section 10, it is enough to consider copir_1 to depend sublinearly on the size of $\mathbf{D}$.

[15] In a co-PIR, we allow the sender's message to be of the same size of the sender's input (or even slightly larger by an additive term depending on t) instead of the usual rate-1 definition which compares the sender's message with the receiver's input. This is the reason why we e define a co-PIR to be rate-1 only for $t = o(\mathbf{D})$ erased positions, which is enough for our applications.

Sender security. We define two notions for statistical sender security. The first one, which is the strongest one, considers malicious and computationally unbounded receivers.

Definition 28 (Statistical sender security)**.** *A co-PIR is said to be statistically sender secure if there is a (possibly computationally inefficient) extractor* $\mathsf{CoPIR.Ext}$ *such that for any message* copir_1 *and any pair of databases* $(\mathbf{D}, \mathbf{D}')$

$$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D}) \approx_s \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}')$$

where $S \leftarrow \mathsf{CoPIR.Ext}(\mathsf{copir}_1)$ *and* $\mathbf{D}'_i = \mathbf{D}_i$ *for* $i \notin S$*. Here,* S *is a set of size at most* t*.*

We also consider a relaxation of sender security that only considers semi-honest and computationally unbounded receivers.

Definition 29 (Semi-honest statistical sender security)**.** *A one-query co-PIR scheme* CoPIR *is said to be semi-honest sender secure if for all* $S \subseteq [m]$ *of size* t *we have that*

$$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D}) \approx_s \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}')$$

for all $\mathbf{D}, \mathbf{D}' \in \mathbb{H}^m$ *such that* $\mathbf{D}'_i = \mathbf{D}_i$ *for* $i \notin S$*, and all* $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, S)$*.*

A co-PIR fulfilling the latter security definition is called a semi-honest co-PIR.

One-query co-PIR. We also define a one-query co-PIR scheme, that is, a co-PIR where the receiver's query is composed by a single index.

Definition 30 (One-query co-PIR)**.** *A one-query co-PIR scheme is identical to a co-PIR except that the input of the receiver is composed by a single index. That is, the set* S *in Definition 23 is of the form* $S = \{i^*\}$*. Correctness, statistical sender security and receiver security are defined in an analogous way.*

Self-reducibility. Another property that we will need is self-reducibility.[16] This ensures that the output of a one-query co-PIR has the same form as the database and thus can be input into a new one-query co-PIR.

Definition 31 (Self-reducibility)**.** *A one-query co-PIR scheme is said to be self-reducible if the sender's message is of the form* $\mathsf{copir}_2 = (\mathsf{head}, \alpha_1, \ldots, \alpha_m)$*. Moreover, for any* $i^* \in [m]$*, any two databases* $\mathbf{D}, \mathbf{D}'$ *such that* $\mathbf{D}_i = \mathbf{D}'_i$ *for all* $i \neq i^*$ *and any* copir_1 *message we have that*

$$(\mathsf{head}, \alpha_1, \ldots, \alpha_{i^*-1}, \alpha_{i^*+1}, \ldots, \alpha_m) \approx_s (\mathsf{head}', \alpha'_1, \ldots, \alpha'_{i^*-1}, \alpha'_{i^*+1}, \ldots, \alpha'_m)$$

where $\mathsf{copir}_2 = (\mathsf{head}, \alpha_1, \ldots, \alpha_m) \leftarrow \mathsf{Send}(\mathsf{copir}_1, \mathbf{D})$*,* $\mathsf{copir}'_2 = (\mathsf{head}', \alpha'_1, \ldots, \alpha'_m) \leftarrow \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}')$*.*

In other words, self-reducibility states that all information about the block/bit D_i of the database is contained in a single block/bit of the copir_2 message. This property will be essential for recursion.[17]

PIR compatibility. Let PIR be a PIR scheme and CoPIR be a co-PIR scheme. We say that CoPIR is PIR-compatible if the first message $\mathsf{copir}_1 \leftarrow \mathsf{CoPIR.Query}(1^\lambda, S)$ can be used as a first message of the PIR scheme. That is, we can parse copir_1 as q and compute $\mathsf{r} \leftarrow \mathsf{PIR.Send}(\mathbf{D}, \mathsf{q})$ while preserving correctness, receiver security and (statistical) sender security.

[16] The work of [BCM22] defines an identical property for OT.

[17] In this definition we assume that the i-th block/bit of copir erases D_i. However, this does not need to be the case in general: it might happen that the i-th block/bit of copir erases D_j, which is what happens with our construction in Section 7. However, both definitions are equivalent up to a reordering of the database.

6 Semi-Honest One-Query Co-PIR

We first present a construction of a one-query co-PIR that achieves semi-honest statistical sender security.

Before presenting our scheme we show how we can convert a LHE scheme over $\mathbb{Z}_2$ that supports ciphertext shrinking into an LHE scheme over $\mathbb{F}_q$ where $q = 2^\mu$ for some $\mu = \mathsf{poly}(\lambda)$. Here, we rely on the fact that multiplication over $\mathbb{F}_q$ can be expressed as a linear function over the field $\mathbb{Z}_2$. That is, suppose that an element $x \in \mathbb{F}_q$ is of the form $x = x_1 + x_2\alpha + \cdots + x_\mu\alpha^{\mu-1}$ where each $x_i \in \mathbb{Z}_2$ and α is a symbol. Then, for elements $a, x \in \mathbb{F}_q$ the product

$$xa = f_{1,a}(\mathbf{x}) + f_{2,a}(\mathbf{x})\alpha + \cdots + f_{\mu,a}(\mathbf{x})\alpha^{\mu-1}$$

where $\mathbf{x} = (x_1, \ldots, x_\mu)$ and each $f_{i,a} : \mathbb{Z}_2^\mu \to \mathbb{Z}_2$ is a $\mathbb{Z}_2$-linear function which depends solely on a. This means that there is a square matrix $\mathbf{A}$ (determined by a) such that the coefficients of the product $x \cdot a$ over $\mathbb{F}_q$ are the coefficients of $\mathbf{x} \cdot \mathbf{A}$ over $\mathbb{Z}_2$.

We now define the following functions.

- $\mathsf{FieldMult}(a \in \mathbb{F}_q, \mu)$ takes as input an element $a \in \mathbb{F}_q$ where $q = 2^\mu$. It outputs a matrix $\mathbf{A} \in \{0,1\}^{\mu\times\mu}$ such that the coefficients of $\mathbf{xA} \in \{0,1\}^\mu$ correspond to the coefficients of $x \cdot a$ over $\mathbb{F}_q$ (here $\mathbf{x}$ is a binary vector whose coefficients are the ones of $x \in \mathbb{F}_q$).

Ingredients. We need the following ingredients: Let $k, m \in \mathsf{poly}(\lambda)$, $\mu > \lceil \log m \rceil$ and $q = 2^\mu$. Let

- $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Shrink}, \mathsf{DecShrink})$ be a rate-1 packed LHE scheme over $\mathbb{Z}_2$.
- $\mathsf{bin} : [m] \to \{0,1\}^\mu$ be the function that outputs the binary decomposition.

Construction 2. *We now present the full construction.*

$\mathsf{Query}(1^\lambda, i^* \in [m])$**:**

- *Compute* $\mathbf{i}^* = (i_1^*, \ldots, i_\mu^*) \leftarrow \mathsf{bin}(i^*)$.
- *For all* $\ell \in [\mu]$ *set*

$$\mathbf{T}_\ell^* = i_\ell^* \cdot \mathbf{I}_\ell = \begin{pmatrix} i_\ell^* & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & i_\ell^* & \ldots & \mathbf{0} \\ \vdots & & \ddots & \vdots \\ \mathbf{0} & \ldots & \mathbf{0} & i_\ell^* \end{pmatrix} \in \{0,1\}^{k\times k}$$

  where $\mathbf{I}_k$ *is the identity matrix of size* k.
- *Create* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, k)$.
- *For all* $\ell \in [\mu]$ *compute* $\mathsf{ct}_\ell \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{T}_\ell^*)$.[18]
- *Output* $\mathsf{copir}_1 = (\mathsf{pk}, \{\mathsf{ct}_\ell\}_{\ell\in[\mu]})$ *and* $\mathsf{st} = (\mathsf{sk}, i^*)$.

$\mathsf{Send}\left(\mathsf{copir}_1, \mathbf{D} \in \left(\mathbb{F}_q^k\right)^m\right)$**:**

- *Parse* copir_1 *as* $(\mathsf{pk}, \mathsf{ct})$. *Additionally, parse* $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ *where each* $\mathbf{D}_i = (d_{i,1}, \ldots, d_{i,k}) \in \mathbb{F}_q^k$ *and* $d_{i,j} \in \mathbb{F}_q$ *for all* $j \in [k]$.
- *For all* $i \in [m]$ *and* $j \in [k]$ *determine* $\mathbf{A}_{i,j} \leftarrow \mathsf{FieldMult}(d_{i,j}, \mu)$. *Parse* $\mathbf{A}_{i,j} = \begin{pmatrix} - & \mathbf{a}_{i,j,1} & - \\ & \vdots & \\ - & \mathbf{a}_{i,j,\mu} & - \end{pmatrix} \in \{0,1\}^{\mu\times\mu}$.

[18] Recall that an ecryption of a matrix is defined as individual packed encryptions of each column.

- *For all* $i \in [m]$, $j \in [k]$ *and* $\ell \in [\mu]$ *compute* $\mathbf{C}_{i,j,\ell} \leftarrow \mathsf{SingleRowMatrix}(k, \mu, j, \mathbf{a}_{i,j,\ell})$.
- *For all* $i \in [m]$, *compute* $\mathbf{e}_i = (e_{i,1}, \ldots, e_{i,\mu}) \leftarrow \mathsf{bin}(i)$. *Additionally for all* $\ell \in [\mu]$ *set* $\mathbf{U}_{i,\ell} = \mathsf{Diag}(k, e_{i,\ell})$.
- *For all* $i \in [m]$ *consider the following* $\mathbb{Z}_2$ *function* $f_i : (\{0,1\}^{k \times k})^\mu \to \{0,1\}^{k \times \mu}$ *defined by*

$$f_i(\mathbf{X}_1, \ldots, \mathbf{X}_\mu) = \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} (\mathbf{U}_{i,\ell} - \mathbf{X}_\ell) \cdot \mathbf{C}_{i,j,\ell}.$$

- *For all* $i \in [m]$ *compute* $\tilde{\mathsf{ct}}_i \leftarrow \mathsf{LHE.Eval\&Shrink}(\mathsf{pk}, f_i, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\mu))$.
- *Output* $\mathsf{copir}_2 = \{\tilde{\mathsf{ct}}_i\}_{i \in [m]}$.

$\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$**:**

- *Parse* copir_2 *as* $\{\tilde{\mathsf{ct}}_i\}_{i \in [m]}$ *and* st *as* (sk, i^*).
- *For all* $i \in [m] \setminus \{i^*\}$ *compute* $\mathbf{W}_i \leftarrow \mathsf{LHE.Dec}(\mathsf{sk}, \tilde{\mathsf{ct}}_i)$. *For each* $j \in [k]$ *parse each row* $\mathbf{w}_{i,j} \in \{0,1\}^\mu$ *of* $\mathbf{W}_i$ *as an element* $w_{i,j} \in \mathbb{F}_q$.
- *For all* $i \in [m] \setminus \{i^*\}$, *set* $\mathbf{e}_i \leftarrow \mathsf{bin}(i)$. *Parse* $\mathbf{e}_i$ *and* $\mathbf{i}^*$ *as* $\mathbb{F}_q$ *elements (that is,* $\hat{e}_i, \hat{i}^* \in \mathbb{F}_q$ *are the elements whose coefficients correspond to the coefficients of* $\mathbf{e}_i, \mathbf{i}^* \in \{0,1\}^\mu$*). Compute* $\tilde{\mathbf{D}}_i = (\hat{e}_i - \hat{i}^*)^{-1} \cdot (w_{i,1}, \ldots, w_{i,k})$ *over* $\mathbb{F}_q$. *Note that* $\tilde{\mathbf{D}}_i \in \mathbb{F}_q$.
- *Output* $\tilde{\mathbf{D}} = (\tilde{\mathbf{D}}_1, \ldots, \tilde{\mathbf{D}}_{i^*-1}, \mathbf{0}, \tilde{\mathbf{D}}_{i^*+1}, \ldots, \tilde{\mathbf{D}}_m)$.

Lemma 7 (Locally correctness)**.** *The scheme presented in Construction 2 is locally correct given that* LHE *is correct.*

Proof. By the correctness of the LHE we have that for all $i \in [m] \setminus \{i^*\}$

$$\begin{aligned}
\mathbf{W}_i &= \mathsf{LHE.Dec}(\mathsf{sk}, \mathsf{ct}_i) \\
&= \mathsf{LHE.Dec}(\mathsf{sk}, \mathsf{LHE.Eval\&Shrink}(\mathsf{pk}, f_i, (\mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{T}^*_1), \ldots, \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{T}^*_\mu))) \\
&= f_i(\mathbf{T}^*_1, \ldots, \mathbf{T}^*_\mu) = \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} (\mathbf{U}_{i,\ell} - \mathbf{T}^*_\ell) \cdot \mathbf{C}_{i,j,\ell} = \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} \mathsf{Diag}(k, e_{i,\ell} - i^*_\ell) \cdot \mathbf{C}_{i,j,\ell} \\
&= \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} (e_{i,\ell} - i^*_\ell) \cdot \mathsf{SingleRowMatrix}(k, \mu, j, \mathbf{a}_{i,j,\ell}) \\
&= \begin{pmatrix} \sum_{\ell=1}^{\mu} (e_{i,\ell} - i^*_\ell) \cdot \mathbf{a}_{i,1,\ell} \\ \vdots \\ \sum_{\ell=1}^{\mu} (e_{i,\ell} - i^*_\ell) \cdot \mathbf{a}_{i,k,\ell} \end{pmatrix} = \begin{pmatrix} (\mathbf{e}_i - \mathbf{i}^*) \cdot \mathbf{A}_{i,1} \\ \vdots \\ (\mathbf{e}_i - \mathbf{i}^*) \cdot \mathbf{A}_{i,k} \end{pmatrix}
\end{aligned}$$

Let $\mathbf{w}_{i,j}$ be the j-th row of $\mathbf{W}_i$ and parse it as an element $w_{i,j} \in \mathbb{F}_q$. Then $w_{i,j} = (\hat{e}_i - \hat{i}^*) \cdot d_{i,j}$ over $\mathbb{F}_q$ where $\hat{e}_i, \hat{i}^* \in \mathbb{F}_q$ are the vectors $\mathbf{e}_i, \mathbf{i}^* \in \{0,1\}^\mu$ parsed as elements in $\mathbb{F}_q$. This follows from the definition of the algorithm $\mathsf{FieldMult}$.

Thus

$$\begin{aligned}
\tilde{\mathbf{D}}_i &= (\hat{e}_i - \hat{i}^*)^{-1} \cdot (w_{i,1}, \ldots, w_{i,k}) \\
&= (\hat{e}_i - \hat{i}^*)^{-1} \cdot ((\hat{e}_i - \hat{i}^*) \cdot d_{i,1}, \ldots, (\hat{e}_i - \hat{i}^*) \cdot d_{i,k}) \\
&= (d_{i,1}, \ldots, d_{i,k}) = \mathbf{D}_i
\end{aligned}$$

for all $i \in [m] \setminus \{i^*\}$.

Finally, note that each block $i \in [m] \setminus \{i^*\}$ can be recovered independently. Hence, we have local correctness. □

Lemma 8 (Receiver security)**.** *The scheme presented in Construction 2 is receiver secure given that* LHE *is IND-CPA secure.*

Proof. The receiver's message is composed by $(\mathsf{pk}, \mathsf{ct})$ where $\mathsf{ct} \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{T})$. We can build a reduction that breaks the IND-CPA security of LHE given that there is an adversary that breaks the receiver security of the scheme. The reduction, after receiving a public key pk, simply sets $m_0 = \mathbf{T}_0^*$ and $m_1 = \mathbf{T}_1^*$ where each $\mathbf{T}_b^*$ corresponds to the matrix encoding an index i_b^*. Upon receiving ct from the challender, it sends $(\mathsf{pk}, \mathsf{ct})$ to the adversary that outputs whatever the adversary outputs. □

Lemma 9 (Semi-honest statistical sender security)**.** *The scheme presented in Construction 2 is semi-honest statistically sender secure given that* LHE *is statistically circuit-private.*

Proof. Let $\mathsf{copir}_1 = (\mathsf{pk}, \mathsf{ct})$ be a message in the range of $\mathsf{Query}(1^\lambda, i^*)$. Additionally, let LHE.Sim be the simulator of Definition 9. The proof follows from the following sequence of hybrids.

Hybrid $\mathcal{H}_0$. This is the real protocol.

Hybrid $\mathcal{H}_1$. In this hybrid, we replace ct_{i^*} by $\mathsf{ct}_{i^*} \leftarrow \mathsf{LHE.Sim}(\mathsf{pk}, \mathbf{0})$. First note that

$$\begin{aligned} f_{i^*}(\mathbf{T}_1^*, \ldots, \mathbf{T}_\mu^*) &= \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} (\mathbf{U}_{i^*,\ell} - \mathbf{T}_\ell^*) \cdot \mathbf{C}_{i^*,j,\ell} = \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} \mathsf{Diag}(k, e_{i^*,\ell} - i_\ell^*) \cdot \mathbf{C}_{i^*,j,\ell} \\ &= \sum_{j=1}^{k} \sum_{\ell=1}^{\mu} \mathsf{Diag}(k, 0) \cdot \mathbf{C}_{i^*,j,\ell} = \mathbf{0} \end{aligned}$$

where the third equality holds since for $i = i^*$ we have that $\mathbf{e}_i = \mathbf{i}^*$, Hence statistical indistinguishability between the hybrids follows from the function privacy of the LHE.

Note that, in this hybrid, $\mathbf{D}_{i^*}$ is not used. Hence, no information about it is leaked to the receiver. □

Self-reducibility. We show that the scheme presented in Construction 2 is self-reducible. First, note that $\mathsf{copir}_2 = (\mathsf{ct}_1, \ldots, \mathsf{ct}_m)$ hence it has the prescribed form. Furthermore, each ct_i is computed using only $\mathbf{D}_i$ (and not other block $\mathbf{D}_j$ for $j \neq i$). Hence, we have self-reducibility.

PIR-compatibility. The first message of the scheme presented in Construction 2 is composed by $\mathsf{copir}_1 = (\mathsf{pk}, \{\mathsf{ct}_j\}_{j \in [\mu]})$ where each ct_j encrypts $i_j^* \cdot \mathbf{I}_\ell$. Hence, copir_1 has the same form as a first message PIR scheme ADD+ (sketched in Section 3.3.4).

Communication complexity. We now analyze the communication complexity of the scheme.

- **Size of copir_1.** The message copir_1 is composed by $(\mathsf{pk}, \mathsf{ct})$. We have that
  - $|\mathsf{pk}| = k \cdot \mathsf{poly}(\lambda)$
  - $\mathsf{ct} = \mu \cdot k^2 \cdot \mathsf{poly}(\lambda)$.

  Hence $|\mathsf{copir}_1| = k \cdot \mathsf{poly}(\lambda) + \mu \cdot k^2 \cdot \mathsf{poly}(\lambda) = \mathsf{poly}(k, \lambda)$ which is independent of m as required.

- **Download rate.** Let $1 + \rho_{\mathsf{LHE}}$ be the rate of LHE for encryptions of size k. The message copir_2 is composed by m ciphertexts ct_i encrypting $k \cdot \mu$ bits each. The bitsize of of the sender's message is

$$|\mathsf{copir}_2| = m \cdot k \cdot \mu \cdot (1 + \rho_{\mathsf{LHE}}).$$

  This means that the rate of the scheme is

$$\rho_{down} = \frac{m \cdot k \cdot \mu \cdot (1 + \rho_{\mathsf{LHE}})}{mk\mu} = 1 + \rho_{\mathsf{LHE}}.$$

  For large enough k and $\mu = k^{1-\varepsilon}$ (for $\varepsilon > 0$, we can instantiate LHE such that $\rho_{\mathsf{LHE}} = 1/\mathsf{poly}(k)$ and we can upperbound ρ_{down} by $1 + 1/\mathsf{poly}(k)$.

Computational complexity. We now analyze the computational complexity of the scheme in terms of the size of $\mathbf{D}$.

- **Receiver.** The Query algorithm is independent of m. In the Rec algorithm, one needs to perform a decryption for each element in $[m]$. Hence, the receiver's computational complexity can be bounded by $|\mathbf{D}| \cdot \mathsf{poly}(\lambda)$.
- **Sender.** Similarly, for each element in $[m]$ the Send algorithm needs to perform an evaluation over a ciphertext and shrink the resulting ciphertext (which takes work $\mathcal{O}(k \times \mu)$). Hence, the sender's computational complexity can be bounded by $m \cdot \mathcal{O}(k \cdot \mu) = |\mathbf{D}| \cdot \mathsf{poly}(\lambda)$.

7 Bit One-Query Co-PIR from Block One-Query Co-PIR

We now show how we can build a co-PIR that erases only one bit from the database (which we will refer to as bit co-PIR) from a co-PIR that erases an entire block from the database.

The main idea behind is that part of the erased block can be transmitted to the receiver via a PIR without incuring in additional communication. A bit care needs to be taken in order to preserve the PIR-compatibility of the scheme. We prove that the transformation preserves semi-honest security but it also preserves statistical sender security if the underlying PIR is statistical sender secure.

Ingredients. We will need the following ingredients:

- A block PIR scheme $\mathsf{PIR}_k = (\mathsf{Query}, \mathsf{Send}, \mathsf{Retrieve})$ parametrized by k.
- A block PIR scheme $\mathsf{PIR}_m = (\mathsf{Query}, \mathsf{Send}, \mathsf{Retrieve})$ parametrized by m.
- A block co-PIR scheme $\mathsf{1QCoPIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ for databases in $(\{0,1\}^k)^m$ parametrized by m that is PIR_m-compatible.

Construction 3 (Bit co-PIR)**.** *We now described the scheme in full detail.*

$\mathsf{Query}(1^\lambda, (i^*, j^*) \in [m] \times [k])$**:**

- *Compute* $(\mathsf{copir}'_1, \mathsf{st}') \leftarrow \mathsf{1QCoPIR.Query}(1^\lambda, i^*)$.
- *Compute* $(\mathsf{q}, \mathsf{st}'') \leftarrow \mathsf{PIR}_k.\mathsf{Query}(1^\lambda, j^*)$.
- *Output* $\mathsf{copir}_1 = (\mathsf{copir}'_1, \mathsf{q})$ *and* $\mathsf{st} = (\mathsf{st}'\mathsf{st}'')$.

$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D} \in (\{0,1\}^k)^m)$**:**

- *Parse* $\mathbf{D} = (\mathbf{D}_1), \ldots, \mathbf{D}_m)$ *where each* $\mathbf{D}_i \in \{0,1\}^k$, copir_1 *as* $(\mathsf{copir}'_1, \mathsf{q})$. *Addtionally parse* copir'_1 *as a* PIR_m *first message* q'.
- *Compute* $\mathsf{copir}'_2 \leftarrow \mathsf{1QCoPIR.Send}(\mathsf{copir}'_1, \mathbf{D})$.
- *For all* $i \in [m]$ *and all* $j \in [k]$ *consider* $\mathbf{C}_{i,j} \in \{0,1\}^k$ *which is equal to* $\mathbf{D}_i$ *except for the* j*-th position which is equal to* 0.
- *For all* $i \in [m]$ *compute* $\mathsf{r}_i \leftarrow \mathsf{PIR}_k.\mathsf{Send}(\mathsf{q}, (\mathbf{C}_{i,1}, \ldots, \mathbf{C}_{i,k}))$.
- *Compute* $\mathsf{r}' \leftarrow \mathsf{PIR}_m.\mathsf{Send}(\mathsf{q}', (\mathsf{r}_1, \ldots, \mathsf{r}_m))$.
- *Output* $\mathsf{copir}_2 = (\mathsf{copir}'_2, \mathsf{r}')$.

$\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$**:**

- *Parse* copir_2 *as* $(\mathsf{copir}'_2, \mathsf{r}')$ *and* st *as* $(\mathsf{st}', \mathsf{st}'')$.
- *Compute* $\hat{\mathbf{D}} \leftarrow \mathsf{1QCoPIR.Rec}(\mathsf{copir}'_2, \mathsf{st}')$.
- *Compute* $\mathsf{r} \leftarrow \mathsf{PIR}_m.\mathsf{Retrieve}(\mathsf{st}', \mathsf{r}')$.
- *Compute* $\mathbf{C} \leftarrow \mathsf{PIR}_k.\mathsf{Retrieve}(\mathsf{st}'', \mathsf{r})$ *where* $\mathbf{C} \in \{0,1\}^k$.
- *Set* $\tilde{\mathbf{D}}$ *such that*

$$\tilde{\mathbf{D}}_{i,j} = \begin{cases} \hat{\mathbf{D}}_{i,j} \ \textit{if } i \neq i^* \\ \mathbf{C}_{i,j} \ \textbf{\textit{if}} \ i = i^* \wedge j \neq j^* \end{cases}$$

- *Output* $\tilde{\mathbf{D}}$.

Lemma 10 (Correctness)**.** *The scheme presented in Construction 3 is correct assuming that both* 1QCoPIR *and* PIR *are correct.*

Proof. By the correctness of the CoPIR scheme we have that for all blocks $i \neq i^*$ $\tilde{\mathbf{D}}_i = \mathbf{D}_i$.

Moreover, for $i = i^*$ we have that $\mathsf{r} = \mathsf{r}_{i^*}$ by the correctness of PIR_m. Hence $\mathsf{PIR}_k.\mathsf{Retrieve}(\mathsf{st}, \mathsf{r}_{i^*}) = \mathbf{C}$ where $\mathbf{C}_j = \mathbf{D}_{i^*,j}$ for all $j \neq j^*$. □

Lemma 11 (Receiver security)**.** *The scheme presented in Construction 3 is receiver secure given that both* 1QCoPIR *and* PIR_k *are receiver secure.*

Proof. Let $(i_1^*, j_1^*), (i_2^*, j_2^*) \in [m] \times [k]$. The proof follows from the sequence of hybrids:

Hybrid $\mathcal{H}_0$**.** In this game, $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, (i_1^*, j_1^*))$.

Hybrid $\mathcal{H}_1$**.** This hybrid is identical to the previous one except that we compute $(\mathsf{copir}'_1, \mathsf{st}') \leftarrow \mathsf{1QCoPIR.Query}(1^\lambda, i_2^*)$.

Indistinguishability of hybrids $\mathcal{H}_1$ and $\mathcal{H}_0$ follows from the receiver security of the underlying 1QCoPIR.

Hybrid $\mathcal{H}_2$**.** This hybrid is identical to the previous one except that we compute $(\mathsf{q}, \mathsf{st}'') \leftarrow \mathsf{PIR}_k.\mathsf{Query}(1^\lambda, j_2^*)$.

Indistinguishability of hybrids $\mathcal{H}_2$ and $\mathcal{H}_1$ follows from the receiver security of the underlying PIR. Note that this hybrid corresponds to $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, (i_2^*, j_2^*))$. □

Lemma 12 (Statistical sender security)**.** *The scheme presented in Construction 3 is semi-honest sender secure given that* 1QCoPIR *and* PIR *are semi-honest sender secure.*

Proof. Let $\mathbf{D}$ and $\mathbf{D}'$ be two different databases such that $\mathbf{D}_{i,j} = \mathbf{D}'_{i,j}$ for all pairs $(i,j) \neq (i^*, j^*)$. The proof follows the following sequence of hybrids.

Hybrid $\mathcal{H}_0$. This is the real game.

Hybrid $\mathcal{H}_1$. In this hybrid, we replace $\mathbf{D}_{i,j}$ for $\mathbf{D}'_{i,j}$ for $i = i^*$ and for all $j \neq j^*$. Security follows from PIR_k.

Hybrid $\mathcal{H}_2$. In this hybrid, we replace r_i for all $i \neq i^*$ for $\mathsf{r}_i \leftarrow \mathsf{PIR}_k.\mathsf{Send}(\mathsf{q}, \mathbf{D}_{i,j^*}$. Security follows from the sender security of PIR_m.

Hybrid $\mathcal{H}_3$. In this hybrid, we replace $\mathsf{copir}'_1 \leftarrow \mathsf{1QCoPIR.Send}(1^\lambda, \mathbf{D}')$. Security follows from the security of $\mathsf{1QCoPIR}$. □

Communication complexity. We now analyze the communication complexity of our scheme. In the following, let $1 + \rho$ be the rate of the underlying CoPIR scheme.

- **Size of copir_1.** The message copir_1 is composed by $(\mathsf{copir}'_1, \mathsf{q})$. We have that
  - $|\mathsf{copir}'_1| = \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)$
  - $|\mathsf{q}| = |\mathsf{r}_i| \cdot \mathsf{polylog}(m|\mathsf{r}_i|) \cdot \mathsf{poly}(\lambda)$ where $|\mathsf{r}_i| = k \cdot \mathsf{poly}(\lambda)$.

  Thus, the total size of $|\mathsf{copir}_1|$ is $\mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda, k)$ as required.
- **Download rate.** The message copir_1 is composed by $(\mathsf{copir}'_2, \{\mathsf{q}_{i_i^*,j}\}_{i\in[t]}, j \in [k] \setminus \{j_i^*\})$. We have that
  - $|\mathsf{copir}'_2| = |\mathbf{D}| \cdot (1 + \rho)$.
  - $|\mathsf{r}| = k \cdot \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)$.

  Hence, the download rate of the scheme in Construction 3 is

$$\rho_{\mathsf{down}} = \frac{|\mathbf{D}| \cdot (1 + \rho) + \cdot k \cdot \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)}{|\mathbf{D}|} = 1 + \rho + \frac{\cdot k \cdot \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)}{|\mathbf{D}|}$$

  which tends to 1 for large enough $|\mathbf{D}|$.

PIR-compatibility. The receiver's message is composed by $\mathsf{copir}'_1, \mathsf{q}$ where copir'_1 can be parsed as a (k-block) PIR_m first message. Then the scheme is PIR'-compatible where PIR' is the scheme which first applies q to every k-block of $\mathbf{D}$ and then these m responses are fed into PIR_m.

Self-reducibility. The final co-PIR sender's message is composed by $\mathsf{copir}_2 = (\mathsf{copir}'_2, \mathsf{r})$. Assume that the underlying block co-PIR and PIR are both self-reducible. Then the new bit co-PIR protocol is self-reducible modulo a reordering of the database. Let (i^*, j^*) be the input of the receiver, i.e., the position that is going to be erased. Since the underlying block co-PIR is statistically secure then all information about the block $\mathbf{D}_{i^*}$ is contained in r. Now, assuming that the underlying PIR is self-reducible then so is the final protocol.

Looking ahead, the final copir_2 is supposed to have this structure to be compatible with recursion. This structure is compatible with recursion modulo a reordering of the database.

Computational complexity. We analyze the computational complexity for each of the parties individually. Let $\mathsf{CC}^{\mathsf{P}}_{\mathsf{Pr}}$ be the computational complexity of party P in protocol Pr.

- **Receiver.** The receiver's computational complexity is equal to $\mathsf{CC}^{\mathsf{R}}_{\mathsf{1QCoPIR}} + \mathsf{CC}^{\mathsf{R}}_{\mathsf{PIR_m}} + \mathsf{CC}^{\mathsf{R}}_{\mathsf{PIR_k}}$.
- **Sender.** The sender's computational complexity is equal to $\mathsf{CC}^{\mathsf{S}}_{\mathsf{1QCoPIR}} + \mathsf{CC}^{\mathsf{S}}_{\mathsf{PIR_m}} + m \cdot \mathsf{CC}^{\mathsf{S}}_{\mathsf{PIR_k}}$.

Instantiating $\mathsf{1QCoPIR}$ with the scheme from Section 6 and PIR_k with $\mathsf{ADD}+$ scheme we obtain a scheme where the receiver's computational complexity is $|\mathbf{D}|\mathsf{poly}(\lambda)$ and the sender's computational complexity is $|\mathbf{D}| \cdot k \cdot \mathsf{poly}(\lambda)$.

8 Semi-Honest Co-PIR from Semi-Honest One-Query Co-PIR

We now show how to bootstrap a one-query co-PIR into a multiple-query co-PIR. This construction works by recursing the one-query co-PIR multiple times and, at each step, one position of the database is erased. Since the underlying one-query co-PIR has rate 1, then there is no blowup in communication when we recurse it.

Construction 4. *Let* $\{\mathbb{H}_i\}_{i\in[t]}$ *be groups. For* $i \in [t]$ *let* $\mathsf{1QCoPIR}^{(i)} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ *be a one-query co-PIR scheme such that the outputs of* $\mathsf{1QCoPIR}^{(i)}.\mathsf{Send}$ *are of the form* $(\mathsf{head}, \alpha_1, \ldots, \alpha_m)$ *where* $\alpha_j \in \mathbb{H}_i$.

$\mathsf{Query}(1^\lambda, S)$**:**

- *Parse* $S = \{a_1, \ldots, a_t\}$.
- *For* $i \in [t]$ *compute* $(\mathsf{copir}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{1QCoPIR}^{(i)}.\mathsf{Query}(1^\lambda, a_i)$.
- *Output* $\mathsf{copir}_1 = \{\mathsf{copir}_{1,i}\}_{i\in[t]}$ *and* $\mathsf{st} = \{\mathsf{st}_i\}_{i\in[t]}$.

$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D} \in \mathbb{H}^m)$**:**

- *Parse* copir_1 *as* $\{\mathsf{copir}_{1,i}\}_{i\in[t]}$. *Set* $\mathsf{DB}_0 = \mathbf{D}$.
- *For* $i \in [t]$ *do the following:*
  - *Compute* $\mathsf{copir}_{2,i} \leftarrow \mathsf{1QCoPIR}^{(i)}.\mathsf{Send}(\mathsf{copir}_{1,i}, \mathsf{DB}_{i-1})$.
  - *Parse* $\mathsf{copir}_{2,i}$ *as* $(\mathsf{head}_i, \alpha_{i,1}, \ldots, \alpha_{i,m})$.
  - *Set* $\mathsf{DB}_i = (\alpha_{i,1}, \ldots, \alpha_{i,m})$.
- *Set* $\mathsf{DB}^* = \mathsf{DB}_t$.
- *Output* $\mathsf{copir}_2 = (\mathsf{DB}^*, \mathsf{head}_1, \ldots, \mathsf{head}_t)$.

$\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$**:**

- *Parse* copir_2 *as* DB^* *and* st *as* $\{\mathsf{st}_i\}_{i\in[t]}$.
- *Set* $\mathsf{DB}'_t = \mathsf{DB}^*$.
- *For* $i = t$ *to* 1, *set* $\mathsf{copir}'_{2,i} = (\mathsf{head}_i, \mathsf{DB}'_i)$ *and compute* $\mathsf{DB}'_{i-1} \leftarrow \mathsf{1QCoPIR}^{(i)}.\mathsf{Rec}(\mathsf{copir}'_{2,i}, \mathsf{st}_i)$.
- *Output* $\tilde{\mathbf{D}} = \mathsf{DB}'_0$.

Lemma 13 (Correctness)**.** *The scheme presented in Construction 4 is correct assuming that* 1QCoPIR *is locally correct.*

Proof. For all $i = t$ to 1, we have that $\alpha'_{i-1,j} = \alpha_{i-1,j}$ where $\alpha'_{i-1,j} \leftarrow \mathsf{1QCoPIR}^{(i)}.\mathsf{Rec}(\alpha'_{i,j}, \mathsf{st}_i)$ for all $j \notin \{a_t, \ldots, a_i\}$ given that $\mathsf{1QCoPIR}^{(i)}$ is locally correct. This means that $\mathsf{DB}'_{i-1,j} = \mathsf{DB}_{i-1,j}$ for all $j \in [t]$ such that $j \neq\notin \{a_t, \ldots, a_i\}$, where $\mathsf{DB}'_{i-1} = (\alpha'_{i-1,1}, \ldots, \alpha'_{i-1,m})$.

Thus $\mathsf{DB}'_0 = \mathsf{DB}_0$ for all positions $j \notin \{a_t, \ldots, a_1\}$. Thus $\mathbf{D}'_j = \mathbf{D}_j$ for all $j \notin \{a_t, \ldots, a_1\}$. □

Lemma 14 (Receiver security)**.** *The scheme presented in Construction 4 is receiver secure given that* 1QCoPIR *is receiver secure.*

Proof. Let $S_1, S_2 \subseteq [m]^t$ be any two sets where $S_1 = \{a_1, \ldots, a_t\}$ and $S_2 = \{a'_1, \ldots, a'_t\}$. The proof follows from the sequence of hybrids:

Hybrid $\mathcal{H}_0$. In this game, $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, S_1)$.

Hybrid $\mathcal{H}_{1,i}$. This hybrid is identical to the previous one except that we compute

$$(\mathsf{copir}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{1QCoPIR.Query}(1^\lambda, a'_i; r'_i)$$

using random coins $r'_i \in \{0,1\}^\lambda$. This hybrid is defined for $i = 1, \ldots, t$.

Indistinguishability of hybrids $\mathcal{H}_{1,i-1}$ and $\mathcal{H}_{1,i}$ follows from the receiver security of the underlying $\mathsf{1QCoPIR}$ for all $i = 1, \ldots, t$ where ($\mathcal{H}_{1,0} = \mathcal{H}_0$). Note that this hybrid corresponds to $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, S_2)$. □

Lemma 15 (Semi-honest sender security)**.** *The scheme presented in Construction 4 is semi-honest staristically sender secure given that* $\mathsf{1QCoPIR}$ *is self-reducible and semi-honest statistically sender secure.*

Proof. The proof follows the sequence of hybrids.

Hybrid $\mathcal{H}_0$. This is the real experiment.

For all $i = 1, \ldots, t$ consider the following hybrids.

Hybrid $\mathcal{H}_{i,1}$. This hybrid is identical to the previous one except that we replace $\alpha_{t-i,a_{t+1-i}}$ by 0.

Indistinguishability of hybrids $\mathcal{H}_{i,1}$ and $\mathcal{H}_{i,0}$ (where $\mathcal{H}_{i,0} = \mathcal{H}_0$ for $i = 1$ and $\mathcal{H}_{i,0} = \mathcal{H}_{i-1,t+1-i}$ for $i > 1$) follows from the semi-honest sender security of the underlying $\mathsf{1QCoPIR}$.

Hybrid $\mathcal{H}_{i,j}$. This hybrid is identical to the previous one except that we replace $\alpha_{t-i-j,a_{t+1-i}}$ by 0. This hybrid is defined for $j = 1, \ldots, t+1-i$.

Indistinguishability of hybrids $\mathcal{H}_{i,j}$ and $\mathcal{H}_{i,j-1}$ (where $\mathcal{H}_{i,0} = \mathcal{H}_{i-1,t+1-i}$) follows from the self-reducibility of the underlying $\mathsf{1QCoPIR}$.

Note that in the last hybrid we set $\alpha_{0,a_i} = 0$ for all $i \in [t]$. This means that we set the input database $\mathbf{D}$ to be 0 for all coordinates in S. □

Communication complexity. We now analyze the communication complexity of our scheme. Let $1+\rho$ be the download rate of the underlying $\mathsf{1QCoPIR}$ scheme.

- **Size of copir_1.** The message copir_1 is composed by $\{\mathsf{copir}_{1,i}\}_{i\in[t]}$. For some $i \in [t]$ we have that $|\mathsf{copir}_{1,i}| = \mathsf{polylog}(|\mathbf{D}|(1+\rho)^{i-1})) \cdot \mathsf{poly}(\lambda)$ since the database increases in size by a factor of $(1+\rho)$ at each step. Hence

  $$|\mathsf{copir}_{1,i}| = \mathsf{polylog}(|\mathbf{D}|(1+\rho)^{i-1})) \cdot \mathsf{poly}(\lambda) \leq \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(i, \lambda).$$

  Thus, the total size of $|\mathsf{copir}_1|$ is $\mathsf{polylog}(\mathbf{D}) \cdot \mathsf{poly}(\lambda, t)$, which depends only polylogarithmically on m as required.

- **Dowload rate.** The message copir_2 is composed by DB^*. We have that $|\mathsf{DB}^*| = |\mathbf{D}|(1+\rho)^t \leq |\mathbf{D}|(1+t\rho)$ (by the Bernoulli inequality).

  Hence, the download rate of the scheme is

  $$\rho_{\mathsf{down}} = \frac{|\mathsf{copir}_2|}{\mathbf{D}} \leq \frac{|\mathbf{D}|(1+t\rho) + \mathsf{poly}(t,\lambda)}{|\mathbf{D}|} = 1 + t\rho + \frac{\mathsf{poly}(t,\lambda)}{|\mathbf{D}|}.$$

  If $t = \rho^{-1+\varepsilon}$ for some $0 < \varepsilon < 1$ then we can upperbound the download rate by $1 + \rho^\varepsilon + \frac{\mathsf{poly}(t,\lambda)}{|\mathbf{D}|}$.

Self-reducibility. First, note that the sender's message is of the form $(\mathsf{head}, \alpha_1, \ldots, \alpha_m)$ where $\mathsf{head} = (\mathsf{head}_1, \ldots, \mathsf{head}_t)$ and $\mathsf{DB}^* = (\alpha_1, \ldots, \alpha_m)$. Second, if the underlying one query co-PIR $\mathsf{1QCoPIR}$ is self-reducible then so is the resulting (multiple query) co-PIR. To see this, note that if we erase the i-th position of DB^*, then by self-reducibility of $\mathsf{1QCoPIR}$ all information about $\mathbf{D}_i$ is erased since all information about the i-th position of all intermediate DB_j is erased).

PIR-compatibility. Let PIR be a PIR scheme. If the underlying $\mathsf{1QCoPIR}$ is PIR compatible then so is the final construction since the receiver's message is composed by t first message co-PIR $\mathsf{copir}_{1,i}$. Hence, the message copir_1 can be used as a PIR message with the same input set S.

Computational complexity. We now analyze the computational complexity of the scheme in terms of the size of $\mathbf{D}$ and t. Let CC_R and CC_S be the computational complexity of the receiver and of the sender in the underlying $\mathsf{1QCoPIR}$ scheme.

- **Receiver.** The Query algorithm is independent of m and it only depends on t. In the Rec algorithm, one needs to run the $\mathsf{1QCoPIR.Rec}$ for each element in $[t]$. Hence, the receiver's computational complexity can be bounded by $t \cdot \mathsf{CC}_\mathsf{R}$.
- **Sender.** Similarly, the Send algorithm needs to run $\mathsf{1QCoPIR.Send}$ for each element in $[t]$ Hence, the sender's computational complexity can be bounded by $t \cdot \mathsf{CC}_\mathsf{S}$.

Instantiation. The scheme from Construction 3 is a one-query co-PIR (for bits) that is semi-honest, self-reducible and PIR-compatible for some PIR scheme (in particular it is compatible with some modified version of the $\mathsf{ADD+}$ scheme, see Section 7). Instantiating the scheme presented in Construction 4 with the scheme from Construction 3, we obtain a (multiple query) co-PIR for bits which is semi-honest, self-reducible and PIR-compatible for some PIR scheme.

9 Statistical Sender Secure Co-PIR

In this section we present a scheme for statistical sender secure co-PIR. Our scheme works by bootstrapping a semi-honest co-PIR into a statistical sender secure one. We also show in Appendix B an alternative construction for SSP co-PIR from SSP PIR, albeit at the cost of slightly worse overall communication complexity.

We now show how to boostrap a semi-honest co-PIR into a statistical sender secure co-PIR using a CDS. Essentially, the CDS will ensure that the first message of the receiver is well-formed.

Let CoPIR be a semi-honest co-PIR scheme parametrized by m. Consider the following languange $\mathcal{L}_{\mathsf{CoPIR}}$ parametrized by CoPIR

$$\mathcal{L}_{\mathsf{CoPIR}} = \left\{\mathsf{copir}_1 : \exists (S, r) \in [m]^t \times \{0,1\}^\lambda \text{ s.t. } \mathsf{copir}_1 \leftarrow \mathsf{CoPIR.Query}(1^\lambda, S; r)\right\}.$$

Clearly this is a NP languange thus there exists a statistical sender secure CDS scheme for this particular languange [IK02, OPP14].

Ingredients. Let $\mathbb{H}$ be a group. Let

- $\mathsf{CoPIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ be a co-PIR scheme where the outputs of $\mathsf{CoPIR.Send}$ are of the form $(\alpha_1, \ldots, \alpha_m)$ where $\alpha_i \in \mathbb{H}$.
- $\mathsf{CDS} = (\mathsf{Enc}, \mathsf{Send}, \mathsf{Release})$ be a statistical sender secure CDS scheme for the the language $\mathcal{L}_{\mathsf{CoPIR}}$. Looking ahead, we will use the CDS construction from Section 4 to obtain a black-box construction.

Construction 5. *We now describe the construction in full detail.*

$\mathsf{Query}(1^\lambda, S)$**:**

- *Parse* $S = \{a_1, \ldots, a_t\}$.
- *Compute* $(\mathsf{copir}'_1, \mathsf{st}') \leftarrow \mathsf{CoPIR.Query}(1^\lambda, S; r)$ *using random coins* $r \in \{0,1\}^\lambda$.
- *Compute* $(\mathsf{cds}_1, \mathsf{st}'') \leftarrow \mathsf{CDS.Enc}(1^\lambda, (S, r))$.
- *Output* $\mathsf{copir}_1 = (\mathsf{copir}'_1, \mathsf{cds}_1)$ *and* $\mathsf{st} = (\mathsf{st}', \mathsf{st}'')$.

$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D} \in \mathbb{G}^m)$**:**

- *Parse* copir_1 *as* $(\mathsf{copir}'_1, \mathsf{cds}_1)$.
- *Compute* $\mathsf{copir}'_2 \leftarrow \mathsf{CoPIR.Send}(\mathsf{copir}_1, \mathbf{D})$.
- *Parse* copir'_2 *as* $(\alpha_1, \ldots, \alpha_m)$ *and set* $\mathsf{DB} = (\alpha_1, \ldots, \alpha_m)$.
- *For all* $i \in [t]$ *sample* $\beta_i \leftarrow\!\!\$\, \mathbb{H}$.
- *Set* $\mathsf{DB}^* = \mathsf{DB} + (\beta_1, \ldots, \beta_t, 0 \ldots, 0)$.
- *Compute* $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, \mathcal{L}_{\mathsf{CoPIR}}, (\beta_1, \ldots, \beta_t))$.
- *Output* $\mathsf{copir}_2 = (\mathsf{DB}^*, \mathsf{cds}_2)$.

$\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$**:**

- *Parse* copir_2 *as* $(\mathsf{DB}^*, \mathsf{cds}_2)$ *and* st *as* $(\mathsf{st}', \mathsf{st}'')$.
- *Compute* $(\beta'_1, \ldots, \beta'_t) \leftarrow \mathsf{CDS.Release}(\mathsf{cds}_2, \mathsf{st}'')$.
- *Compute* $\mathsf{DB}' \leftarrow \mathsf{DB}^* - (\beta'_1, \ldots, \beta'_t, 0, \ldots, 0)$.
- *Set* $\mathsf{copir}'_2 = \mathsf{DB}'$ *and compute* $\tilde{\mathbf{D}} \leftarrow \mathsf{CoPIR.Rec}(\mathsf{copir}'_2, \mathsf{st}')$.
- *Output* $\tilde{\mathbf{D}}$.

We now prove correctness and security of our scheme.

Lemma 16 (Correctness)**.** *The scheme presented in Construction 5 is correct assuming that both* CoPIR *and* CDS *are correct.*

Proof. By the correctness of CDS, we have that $(\beta'_1, \ldots, \beta'_t) = (\beta_1, \ldots, \beta_t)$ where $(\beta'_1, \ldots, \beta'_t) \leftarrow \mathsf{CDS.Release}(\mathsf{cds}_2, \mathsf{st}')$. Thus

$$\begin{aligned}\mathsf{DB}' &= \mathsf{DB}^* - (\beta'_1, \ldots, \beta'_t, 0, \ldots, 0)\\ &= \mathsf{DB} + (\beta_1, \ldots, \beta_t, 0 \ldots, 0) - (\beta'_1, \ldots, \beta'_t, 0, \ldots, 0)\\ &= \mathsf{DB}.\end{aligned}$$

Now, by the correctness of the underlying CoPIR scheme, we have that $\mathbf{D}_i = \tilde{\mathbf{D}}_i$ where $\tilde{\mathbf{D}} \leftarrow \mathsf{CoPIR.Rec}(\mathsf{DB}', \mathsf{st}')$, for all $i \notin S$. □

Lemma 17 (Receiver security)**.** *The scheme presented in Construction 5 is receiver secure given that both* CoPIR *and* CDS *are receiver secure.*

Proof. Let $S_1, S_2 \subseteq [m]^t$ be any two sets. The proof follows from the sequence of hybrids:

Hybrid $\mathcal{H}_0$**.** In this game, $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, S_1)$.

Hybrid $\mathcal{H}_1$. This hybrid is identical to the previous one except that we compute $(\mathsf{cds}_1, \mathsf{st}') \leftarrow \mathsf{CDS.Enc}(1^\lambda, (0,0))$.

Indistinguishability of hybrids follows from the receiver security of the underlying CDS.

Hybrid $\mathcal{H}_2$. This hybrid is identical to the previous one except that we compute $(\mathsf{copir}'_1, \mathsf{st}') \leftarrow \mathsf{CoPIR.Query}(1^\lambda, S_2; r_2)$ using random coins $r_2 \in \{0,1\}^\lambda$.

Indistinguishability of hybrids $\mathcal{H}_2$ and $\mathcal{H}_1$ follows from the receiver security of the underlying CoPIR.

Hybrid $\mathcal{H}_3$. This hybrid is identical to the previous one except that we compute $(\mathsf{cds}_1, \mathsf{st}') \leftarrow \mathsf{CDS.Enc}(1^\lambda, (S_2, r_2))$ for some random coins r_2.

Indistinguishability of hybrids follows from the receiver security of the underlying CDS. Note that this hybrid corresponds to $\mathsf{copir}_1 \leftarrow \mathsf{Query}(1^\lambda, S_2)$. □

Lemma 18 (Statistical sender security)**.** *The scheme presented in Construction 5 is statistical sender secure given that* CoPIR *is self-reducible and semi-honest (statistical) sender secure and* CDS *is statistical sender secure.*

Proof. We first define the extractor CoPIR.Ext that takes as input a first message copir_1 and outputs a set $S = \{a_1, \ldots, a_t\}$, where $|S| \leq t$. Let CDS.Ext be the extractor of the CDS scheme as defined in Definition 13.

$\mathsf{CoPIR.Ext}(\mathsf{copir}_1)$**:**

- Parse copir_1 as $(\mathsf{copir}'_1, \mathsf{cds}_1)$.
- Run $w \leftarrow \mathsf{CDS.Ext}(\mathsf{cds}_1)$ and parse $w = (S, r)$ (here S is a set of size at most t).
- If $(\mathsf{copir}'_1, w) \in \mathcal{L}_{\mathsf{CoPIR}}$ then output S.
- Else, set $S = \{1, \ldots, t\}$ and output S.

We now show that for any pair of databases $(\mathbf{D}, \mathbf{D}')$

$$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D}) \approx_s \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}')$$

where $S \leftarrow \mathsf{CoPIR.Ext}(\mathsf{copir}_1)$ and $\mathbf{D}'_j = \mathbf{D}_j$ for $j \notin S$. We divide the proof in two cases.

Case 1. In the first case, we assume that $(\mathsf{copir}'_1, w) \notin \mathcal{L}_{\mathsf{CoPIR}}$. Now consider the following sequence of hybrids.

Hybrid $\mathcal{H}_0$. This is the real experiment.

Hybrid $\mathcal{H}_1$. This hybrid is identical to the previous one except that we compute $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, \mathcal{L}_{\mathsf{CoPIR}}, (\gamma_1, \ldots, \gamma_t))$ for uniformly chosen $\gamma_i \leftarrow\!\!\$\, \mathbb{H}$.

Statistical indistinguishability of hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ follows from the statistical sender security of the underlying CDS since $(\mathsf{copir}'_1, w) \notin \mathcal{L}_{\mathsf{CoPIR}}$.

Hybrid $\mathcal{H}_{2,i}$. This hybrid is identical to the previous one except that we replace the i-th position of DB^* by $\delta_i \leftarrow\!\!\$\, \mathbb{H}$. This hybrid is defined for $i = 1, \ldots, t$.

The hybrids $\mathcal{H}_{2,i-1}$ and $\mathcal{H}_{2,i}$ are identically distributed since $\mathsf{DB}^*_i + \beta_i \approx_s \delta_i$ for all $i = 1, \ldots, t$ where $(\mathcal{H}_{2,0} = \mathcal{H}_1)$.

Hybrid $\mathcal{H}_{3,i}$. This hybrid is identical to the previous one except that we replace $\mathbf{D}_i$ by $\mathbf{D}'_i$. This hybrid is defined for $i = 1, \ldots, t$.

Statistical indistinguishability of $\mathcal{H}_{3,i-1}$ and $\mathcal{H}_{3,i}$ follows from the self-reducibility of the underlying CoPIR scheme.

Note that the hybrid $\mathcal{H}_{3,t}$ corresponds to the game where $\mathsf{copir}_2 \leftarrow \mathsf{Send}(\mathsf{copir}_1, \mathbf{D}')$ where $\mathbf{D}'_i = \mathbf{D}$ for all $i \notin S$.

Case 2. In this case, we assume that $(\mathsf{copir}'_1, w) \in \mathcal{L}_{\mathsf{CoPIR}}$. In this case copir'_1 is well-formed. Now consider the following sequence of hybrids.

Hybrid $\mathcal{H}_0$. This is the real experiment.

Hybrid $\mathcal{H}_1$. This hybrid is identical to the previous one except that we replace $\mathbf{D}$ by $\mathbf{D}'$ where $\mathbf{D}'$ is such that $\mathbf{D}'_i = \mathbf{D}_i$ for all $i \notin S$.

Statistical indistinguishability of hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ follows from the semi-honest statistical security of the underlying CoPIR. □

Communication complexity. We now analyze the communication complexity of our scheme. In the following, let $1 + \rho$ be the rate of the underlying semi-honest CoPIR scheme. Moreover, let F be the verification circuit of $\mathcal{L}_{\mathsf{CoPIR}}$ which is of size $\mathsf{poly}(t, \lambda)$

- **Size of copir_1.** The message copir_1 is composed by $(\mathsf{copir}'_1, \mathsf{cds}_1)$. We have that
  - $|\mathsf{copir}'_1| = \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)$
  - $|\mathsf{cds}_1| = \mathsf{poly}(\lambda, |F|) = \mathsf{poly}(t, \lambda)$.

  Thus, the total size of $|\mathsf{copir}_1|$ is $\mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda, t)$ as required.
- **Download rate.** The message copir_1 is composed by $(\mathsf{copir}'_2, \mathsf{cds}_2)$. We have that
  - $|\mathsf{copir}'_2| = |\mathbf{D}| \cdot (1 + \rho)$.
  - $|\mathsf{cds}_2| = \mathsf{poly}(\lambda, t)$.

  Hence, the download rate of the scheme in Construction 5 is

$$\rho_{\mathsf{down}} = \frac{|\mathbf{D}| \cdot (1 + \rho) + \mathsf{poly}(\lambda, t)}{|\mathbf{D}|} = 1 + \rho + \frac{\mathsf{poly}(\lambda, t)}{|\mathbf{D}|}$$

  which tends to 1 if $t < |\mathbf{D}|^{1-\varepsilon}$ for some $\varepsilon > 0$.

PIR-compatibility. Assume that the underlying CoPIR scheme is PIR-compatible. Then the scheme presented in construction 5 is also PIR compatible. To see this, note that the first message copir_1 is composed by $(\mathsf{copir}'_1, \mathsf{cds}_1)$. Hence, the PIR scheme can just ignore cds_1 and interpret copir'_1 as a first message.

Computational complexity. Since the computational complexity of the CDS scheme is independent of m, the scheme inherits the computational complexity of the underlying CoPIR scheme.

Instantiation. We now explain how we can instantiate the building blocks of our construction to obtain a black-box construction for SSP co-PIR. The underlying semi-honest co-PIR scheme CoPIR is intantiated using the scheme in Construction 4 which is semi-honest secure, self-reducible and PIR-compatible for some PIR scheme (see Section 8).

Note that the receiver's message is composed by t first messages of the underlying CoPIR scheme. Hence, if we use the scheme from Construction 4 (which is itself instantiated using the Constructions 2 and 3) then the receiver's message is composed by $\{\mathsf{copir}_{1,i}\}_{i\in[t]} = \{\mathsf{pk}_i, \{\mathsf{ct}_{i,j}\}_{j\in[\mu]}, \mathsf{q}_i\}_{i\in[t]}$, where each $\mathsf{ct}_{i,j}$ encrypts a matrix $a_{i,j} \cdot \mathbf{I}$ where $a_{i,j} \in \{0,1\}$ and $\mathbf{I}$ is the identity matrix, and q_i is a PIR message parametrized by the block size k (which is used in the block-to-bit co-PIR transformation of Section 7).

If we use a SSP PIR scheme then we are guaranteed that the receiver learns nothing additional from the answers to q_i (note that the input of the receiver to the PIR can be arbitrary). Hence, to guarantee SSP for our overall protocol we just need to guarantee that $\mathsf{copir}_{1,i}$ are well-formed. This is done via a CDS (as in the construction above) and note that we can instantiate the CDS using the scheme of Construction 1 from Section 4. That is, the language of well-formed $\mathsf{copir}_{1,i}$ is exactly the language $\mathcal{L}_{\mathsf{pk}}$ defined in Section 4.

This gives us a black-box construction for SSP co-PIR.

As a side note, we remark that the CDS scheme in the construction above can also be instantiated with any generic CDS scheme with the required properties at the cost of making non black-box use of cryptographic primitives.

10 Statistical Sender Private Oblivious Transfer with Optimal Rate

As an application for our statistical sender secure co-PIR scheme, we build an OT scheme. This OT scheme has overall rate 1 and achieves statistical sender privacy.

Before presenting our construction, we present some notation that we will use throughout this section.

- $\mathsf{RowMatrix}(\ell, n, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$: Takes row-vectors $\mathbf{v}_1, \ldots, \mathbf{v}_\ell \in \{0,1\}^n$ and outputs a matrix
$$\mathbf{V} = \begin{pmatrix} — & \mathbf{v}_1 & — \\ & \vdots & \\ — & \mathbf{v}_\ell & — \end{pmatrix},$$
i.e. for every $i \in [\ell]$ the i-th row of $\mathbf{V}$ is the row-vector $\mathbf{v}_i$.

Ingredients. We will need the following ingredients for our protocol:

- A PIR scheme $\mathsf{PIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Retrieve})$.
- A (bit) co-PIR scheme $\mathsf{CoPIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ that is PIR-compatible parametrized by m.
- A rate-1 circuit-private LHE scheme $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Shrink}, \mathsf{DecShrink})$ with plaintext space $\{0,1\}^\ell$ and for which shrinked ciphertexts have the form $\mathsf{ct} = (g, d_1, \ldots, d_\ell)$ where $g \in \mathbb{G}$ is a group element (for some large enough group $\mathbb{G}$, namely a DDH group) and $d_i \in \{0,1\}$.
- A download rate-1 CDS scheme $\mathsf{CDS} = (\mathsf{Enc}, \mathsf{Send}, \mathsf{Release})$ for the language
$$\mathcal{L} = \left\{ \mathsf{pk}, \{\mathsf{ct}_i\}_{i\in[\ell]} : \exists (r, \mathbf{s}_i, r_i) \text{ s.t. } \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, \ell; r) \\ \mathbf{S}_i = \mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{s}_i) \\ \mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{S}_i; r_i) \end{array} \right\}$$
for some $\mathbf{s}_i \in \{0,1\}^n$.
- The binary $\mathsf{LPN}(n, m, \rho)$ problem with dimension $n = \mathsf{poly}(\lambda)$, $m = n \cdot \ell \cdot \mathsf{poly}(\lambda)$ samples and slightly sub-constant noise-rate $\rho = m^{1-\epsilon}$.

Construction 6 (Optimal-rate SSP OT)**.** *We now describe the scheme in full detail.*

$\mathsf{OTR}(1^\lambda, \mathbf{b} \in \{0,1\}^{m\ell})$:

- *Parse* $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_\ell)$*, where the* $\mathbf{b}_i \in \{0,1\}^m$ *are blocks of size* m*.*
- *Choose* $\mathbf{A} \leftarrow\!\!\$\, \{0,1\}^{n\times m}$ *uniformly at random and compute a pair of public and secret key* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, \ell; r)$ *using random coins* $r \in \{0,1\}^\lambda$*.*
- *For all* $i \in [\ell]$*, choose* $\mathbf{s}_i \leftarrow\!\!\$\, \{0,1\}^n$*, and* $\mathbf{e}_i \leftarrow\!\!\$\, \chi_{m,t}$*, compute* $\mathbf{c}_i \leftarrow \mathbf{s}_i\mathbf{A} + \mathbf{e}_i + \mathbf{b}_i$*, and set* $\mathbf{S}_i \leftarrow \mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{s}_i)$*. Compute a matrix-ciphertext* $\mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{S}_i; r_i)$ *using random coins* $r_i \in \{0,1\}^\lambda$*.*
- *Compute* $(\mathsf{cds}_1, \tilde{\mathsf{st}}) \leftarrow \mathsf{CDS.Enc}(1^\lambda, w)$ *where* $w = (r, \{\mathbf{S}_i, r_i\}_{i\in[\ell]})$*.*
- *For all* $i \in [\ell]$ *set* $J_i = \mathsf{Supp}(\mathbf{e}_i)$ *to be the support of* $\mathbf{e}_i$*. Compute* $(\mathsf{copir}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{CoPIR.Query}(J_i)$*.*[19]
- *Output* $\mathsf{ot}_1 = \big(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}\}_{i\in[\ell]}, \mathsf{cds}_1\big)$ *and* $\mathsf{st} = (\mathsf{sk}, \{\mathsf{st}_i, J_i\}_{i\in[\ell]}, \tilde{\mathsf{st}})$*.*

$\mathsf{OTS}(\mathsf{ot}_1, (\mathbf{m}_0, \mathbf{m}_1) \in (\{0,1\}^{m\ell})^2)$:

- *Parse* $\mathbf{m}_0 = (\mathbf{m}_{0,1}, \ldots, \mathbf{m}_{0,\ell})$ *and* $\mathbf{m}_1 = (\mathbf{m}_{1,1}, \ldots, \mathbf{m}_{1,\ell})$*, where each* $\mathbf{m}_{b,i} = (m_{b,i,1}, \ldots, m_{b,i,m}) \in \{0,1\}^m$*. Parse* $\mathsf{ot}_1 = \big(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}\}_{i\in[\ell]}, \mathsf{cds}_1\big)$*.*
- *For* $i \in [\ell]$ *set* $\mathbf{z}_i = \mathbf{m}_{0,i}$*.*
- *Set* $\mathbf{Z} = \mathsf{RowMatrix}(\ell, m, \mathbf{z}_1, \ldots, \mathbf{z}_\ell)$*.*
- *For all* $i \in [\ell]$ *set* $\mathbf{C}_i = \mathsf{SingleRowMatrix}(\ell, m, i, \mathbf{c}_i)$ *and* $\mathbf{D}_i = \mathsf{Diag}(m, \mathbf{m}_{1,i} - \mathbf{m}_{0,i})$*.*
- *Define the* $\mathbb{Z}_2$*-linear function* $f : (\{0,1\}^{\ell\times n})^\ell \to \{0,1\}^{\ell\times m}$ *via*

$$f(\mathbf{X}_1, \ldots, \mathbf{X}_\ell) = \left(\sum_{i=1}^{\ell} (-\mathbf{X}_i\mathbf{A} + \mathbf{C}_i) \cdot \mathbf{D}_i\right) + \mathbf{Z}.$$

  Additionally, define the $\mathbb{Z}_2$*-linear function* $g : (\{0,1\}^{\ell\times n})^\ell \to \{0,1\}^{\ell\times m}$ *via*

$$g(\mathbf{X}_1, \ldots, \mathbf{X}_\ell) = \left(\sum_{i=1}^{\ell} (-\mathbf{X}_i\mathbf{A} + \mathbf{C}_i + \mathbf{U}_i) \cdot \mathbf{D}_i\right) + \mathbf{Z}.$$

  where $\mathbf{U}_i \leftarrow \mathsf{SingleRowMatrix}(\ell, m, i, \mathbf{1})$ *and* $\mathbf{1} = (1, \ldots, 1)$ *is the vector of length* m *which is* 1 *everywhere.*
- *Compute* $\mathsf{CT}_1 \leftarrow \mathsf{LHE.Eval\&Shrink}(\mathsf{pk}, f, \mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ *and* $\mathsf{CT}_2 \leftarrow \mathsf{LHE.Eval\&Shrink}(\mathsf{pk}, g, \mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$*.*
- *Parse* CT_1 *as* $\{g_i, d_{i,1}, \ldots, d_{i,\ell}\}_{i\in[m]}$ *where each* $g_i \in \mathbb{G}$ *and* $d_{i,j} \in \{0,1\}$*. Similarly parse* CT_2 *as* $\{h_i, f_{i,1}, \ldots, f_{i,\ell}\}_{i\in[m]}$ *each* $h_i \in \mathbb{G}$ *and* $f_{i,j} \in \{0,1\}$*.*
- *For all* $i \in [\ell]$ *set* $\mathbf{D}_i = (d_{1,i}, \ldots, d_{m,i})$ *and* $\mathbf{F}_i = (f_{1,i}, \ldots, f_{m,i})$*. Compute* $\mathsf{copir}_{2,i} \leftarrow \mathsf{CoPIR.Send}(\mathsf{copir}_{1,i}, \mathbf{D}_i)$ *and* $\mathsf{r}_i \leftarrow \mathsf{PIR.Send}(\mathsf{q}_i, \mathbf{F}_i)$ *where* $\mathsf{copir}_{1,i}$ *is parsed as the PIR message* q_i*.*
- *Set* $\mathsf{ot}_2' = \{g_i, \mathsf{copir}_{2,i}, h_i, \mathsf{r}_i\}_{i\in[\ell]}$*.*
- *Compute* $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, \mathcal{L}, \mathsf{ot}_2')$*.*
- *Output* $\mathsf{ot}_2 = \mathsf{cds}_2$*.*

[19] Recall that, since the CoPIR scheme is PIR-compatible then $\mathsf{copir}_{1,i}$ also corresponds to a first message PIR with input J_i.

$\mathsf{OTD}(\mathsf{ot}_2, \mathsf{st})$:

- *Parse* ot_2 *as* cds_2 *and* $\mathsf{st} = (\mathsf{sk}, \{\mathsf{st}_i, J_i\}_{i\in[\ell]}, \tilde{\mathsf{st}})$.
- *Compute* $\mathsf{ot}_2' \leftarrow \mathsf{CDS.Release}(\mathsf{cds}_2, \tilde{st})$. *Parse* ot_2' *as* $\{g_i, \mathsf{copir}_{2,i}, h_i, \mathsf{r}_i\}_{i\in[\ell]}$.
- *For all* $i \in [\ell]$ *compute* $\tilde{\mathbf{D}}_i = (\tilde{d}_{1,i}, \ldots, \tilde{d}_{m,i}) \leftarrow \mathsf{CoPIR.Retrieve}(\mathsf{copir}_{2,i}, \mathsf{st}_i)$.
- *Set* $\tilde{\mathsf{CT}}_1$ *to be* $\{g_i, \tilde{d}_{i,1}, \ldots, \tilde{d}_{i,\ell}\}_{i\in[m]}$. *Compute* $\mathbf{W} \leftarrow \mathsf{LHE.DecShrink}(\mathsf{sk}, \tilde{\mathsf{CT}}_1)$ *where* $\mathbf{W} \in \{0,1\}^{\ell\times m}$. *Parse* $\mathbf{W} = (w_{i,j})_{i\in[\ell],j\in[m]}$ *where* $w_{i,j} \in \{0,1\}$
- *For all* $i \in [\ell]$ *compute* $(v_{i,J_i[1]}, \ldots, v_{i,J_i[t]}) \leftarrow \mathsf{PIR.Retrieve}(\mathsf{r}_i, \mathsf{st}_i)$. *Additionally for all* $j \in [t]$, *compute* $y_{i,J_i[j]} \leftarrow \mathsf{LHE.Dec}(h_{J_i[j]}, v_{i,J_i[j]})$.
- *Set* $\mathbf{M} = (m_{i,j})_{i\in[\ell],j\in[m]} \in \{0,1\}^{\ell\times m}$ *where*

$$m_{i,j} = \begin{cases} y_{i,J_i[l]} \text{ if } l = J_i[l] \\ w_{i,j} \text{ otherwise} \end{cases}.$$

- *Output* $\mathbf{M}$.

We now analyze our scheme. We first show that it is correct and secure.

Theorem 4 (Correctness)**.** *The scheme presented in Construction 6 is correct given that* LHE, CoPIR *and* PIR *are correct.*

Proof. First, by the correctness of the underlying CDS scheme we have that $\mathsf{ot}_2' = \mathsf{CDS.Release}(\mathsf{cds}_2, \tilde{\mathsf{st}})$ where $\mathsf{ot}_2' = \{g_i, \mathsf{copir}_{2,i}, h_i, \mathsf{r}_i\}_{i\in[\ell]}$.

By the correctness of the underlying CoPIR we have that for all $i \in [\ell]$ $d_{i,j} = \tilde{d}_{i,j}$ for $j \notin J_i$.

By the linear homomorphism correctness of LHE, the ciphertext CT_1 encrypts

$$\begin{aligned}\tilde{\mathbf{W}} &= f(\mathbf{S}_1, \ldots, \mathbf{S}_\ell) \\ &= \left(\sum_{i=1}^{k} (-\mathbf{S}_i\mathbf{A} + \mathbf{C}_i) \cdot \mathbf{D}_i\right) + \mathbf{Z}\end{aligned}$$

where each row $\tilde{\mathbf{w}}_i$ of $\tilde{\mathbf{W}}$ is equal to

$$\begin{aligned}\tilde{\mathbf{w}}_i &= (-\mathbf{s}_i\mathbf{A} + \mathbf{c}_i)\mathbf{D}_i + \mathbf{z}_i \\ &= (-\mathbf{s}_i\mathbf{A} + \mathbf{s}_i\mathbf{A}_i + \mathbf{e}_i + \mathbf{b}_i)\mathbf{D}_i + \mathbf{m}_{0,i} \\ &= \mathbf{b}_i \odot (\mathbf{m}_{1,i} - \mathbf{m}_{0,i}) + \mathbf{m}_{0,i} + \mathbf{e}_i \odot (\mathbf{m}_{1,i} - \mathbf{m}_{0,i}).\end{aligned}$$

For all positions $j \notin J_i$

$$\tilde{w}_{i,j} = b_{i,j}(m_{1,i,j} - m_{0,i,j}) + m_{0,i,j}.$$

This means that applying $\mathsf{LHE.DecShrink}$ on $\tilde{\mathsf{CT}}_1$ we obtain $\mathbf{W}$ such that $w_{i,j} = \tilde{w}_{i,j}$ since $\mathsf{CT}_1 = \mathsf{CT}_1$ for all coordinates $i \in [\ell]$ and $j \notin J_i$.

Similarly, CT_2 encrypts

$$\begin{aligned}\tilde{\mathbf{X}} &= f(\mathbf{S}_1, \ldots, \mathbf{S}_\ell) \\ &= \left(\sum_{i=1}^{k} (-\mathbf{S}_i\mathbf{A} + \mathbf{C}_i + \mathbf{U}_i) \cdot \mathbf{D}_i\right) + \mathbf{Z}\end{aligned}$$

where each row $\tilde{\mathbf{x}}_i$ of $\tilde{\mathbf{X}}$ is equal to

$$\begin{aligned}\tilde{\mathbf{x}}_i &= (-\mathbf{s}_i\mathbf{A} + \mathbf{c}_i + \mathbf{1})\mathbf{D}_i + \mathbf{z}_i \\ &= (-\mathbf{s}_i\mathbf{A} + \mathbf{s}_i\mathbf{A}_i + \mathbf{e}_i + \mathbf{b}_i + \mathbf{1})\mathbf{D}_i + \mathbf{m}_{0,i} \\ &= \mathbf{b}_i \odot (\mathbf{m}_{1,i} - \mathbf{m}_{0,i}) + \mathbf{m}_{0,i} + \mathbf{e}_i \odot (\mathbf{m}_{1,i} - \mathbf{m}_{0,i}) + \mathbf{1} \odot (\mathbf{m}_{1,i} - \mathbf{m}_{0,i}).\end{aligned}$$

For all positions $j \in J_i$ we have that

$$\tilde{x}_{i,j} = (b_{i,j} + e_{i,j} + 1)(m_{1,i,j} - m_{0,i,j}) + m_{0,i,j} \tag{1}$$
$$= b_{i,j}(m_{1,i,j} - m_{0,i,j}) + m_{0,i,j} \tag{2}$$

since the coordinates of the error vector $\mathbf{e}_i$ cancel out with $\mathbf{1}$.

By the correctness of the PIR, $v_{i,J_i[l]} = f_{i,J_i[l]}$ for all $l \in [t]$. This means that applying $\mathsf{LHE.DecShrink}$ on $(h_{J_i[l]}, f_{i,J_i[l]})$ yields $\tilde{x}_{i,J_i[l]}$ for all $l \in [t]$. □

Theorem 5 (Receiver security)**.** *The scheme presented in Construction 6 is receiver secure assuming that* LHE *is IND-CPA,* CDS*,* CoPIR *and* PIR *are receiver secure and that the* $\mathsf{LPN}(n, m, \rho)$ *assumption holds for slightly sub-constant noise-rate* $\rho = m^{1-\epsilon}$ *where* $\epsilon > 0$*.*

Proof. The proof follows the following sequence of hybrids.

Hybrid $\mathcal{H}_0$. This hybrid is the real game.

Hybrid $\mathcal{H}_1$. This hybrid is identical to the previous one except that the receiver computes $\mathsf{cds}_1 \leftarrow \mathsf{CDS.Enc}(1^\lambda, w)$ where $w = 0$.

Indistinguishability of hybrids follows from the receiver security of the underlying CDS.

Hybrid $\mathcal{H}_{2,i}$. This hybrid is identical to the previous one except that the receiver computes $\mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{0})$. The hybrid is defined for $i = 1, \ldots, \ell$.

Indistinguishability of hybrids $\mathcal{H}_{2,i-1}$ and $\mathcal{H}_{2,i}$ for $i = 1, \ldots, \ell$ (where $\mathcal{H}_{2,0} = \mathcal{H}_1$) follows from the IND-CPA security of LHE.

Hybrid $\mathcal{H}_{3,i}$. This hybrid is identical to the previous one except that the receiver computes $(\mathsf{copir}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{CoPIR.Query}(J_i')$ where J_i' is a uniformly chosen subset of $[m]$ of size t. This hybrid is defined for $i = 1, \ldots, \ell$.

Indistinguishability of hybrids $\mathcal{H}_{3,i-1}$ and $\mathcal{H}_{3,i}$ for $i = 1, \ldots, \ell$ (where $\mathcal{H}_{3,0} = \mathcal{H}_{2,\ell}$) follows from the receiver security of CoPIR.

Hybrid $\mathcal{H}_{4,i}$. This hybrid is identical to the previous one except that the receiver samples $\mathbf{c}_i \leftarrow\!\!\$\, \{0,1\}^m$. This hybrid is defined for $i = 1, \ldots, \ell$.

Indistinguishability of hybrids $\mathcal{H}_{4,i-1}$ and $\mathcal{H}_{4,i}$ for $i = 1, \ldots, \ell$ (where $\mathcal{H}_{4,0} = \mathcal{H}_{3,\ell}$) follows from the LPN assumption.

Note that the last hybrid is independent of $\mathbf{b}$ and we conclude the proof. □

Theorem 6 (Statistical sender security)**.** *The scheme presented in Construction 6 is statistical sender secure assuming that* LHE *is statistically circuit private and that* CDS*,* CoPIR *and* PIR *are statistical sender secure.*

Proof. Let $\mathsf{CoPIR.Ext}$ and $\mathsf{CDS.Ext}$ be the extractors for the CoPIR and CDS schemes respectively. Let $\mathcal{R}$ be the relation for the language $\mathcal{L}$ described above.

$\mathsf{OT.Ext}(\mathsf{ot}_1)$**:**

- Parse ot_1 as $\left(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}\}_{i\in[\ell]}, \mathsf{cds}_1\right)$.
- Run $w \leftarrow \mathsf{CDS.Ext}(\mathsf{cds}_1)$ where $w = (r, \mathbf{S}_i, r_i)$. Set $x = (\mathsf{pk}, \{\mathsf{ct}_i\}_{i\in[\ell]})$.
- If $\mathcal{R}(x, w) = 1$ then do the following:
  - For all $i \in [\ell]$ recover $J_i \leftarrow \mathsf{CoPIR.Ext}(\mathsf{copir}_{1,i})$ where J_i is a set of size at most t. Set $\mathbf{e}_i \in \mathbb{Z}_2^m$ to be the vector such that $\mathsf{Supp}(\mathbf{e}_i) = J_i$.
  - For all $i \in [\ell]$ set $\mathbf{b}_i = \mathbf{s_i A} + \mathbf{e}_i + \mathbf{c}_i \mod 2$.
  - Output $\mathbf{b} = (\mathbf{b_1}, \ldots, \mathbf{b}_\ell)$.
- Else if $\mathcal{R}(x, w) = 0$ then output $\mathbf{b} = 0$.

We now prove that for any two messages $\mathbf{m}_0, \mathbf{m}_1 \in \{0,1\}^{m\ell}$ and any ot_1 we have that

$$\mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_0, \mathbf{m}_1) \approx_s \mathsf{OTS}(\mathsf{ot}_1, \mathbf{m}_{\mathbf{b}'}, \mathbf{m}_{\mathbf{b}'})$$

where $\mathbf{b}' \leftarrow \mathsf{OT.Ext}(\mathsf{ot}_1)$. The proof divides into two cases:

Case 1. We first analyze the case where $\mathcal{R}(x, w) = 1$. That is, ct_i are valid encryptions of $\mathbf{S}_i$ where $\mathbf{S}_i = \mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{s}_i)$

Hybrid $\mathcal{H}_0$. This hybrid is the real experiment.

Hybrid $\mathcal{H}_1$. This hybrid is identical to the previous one except that we set $\mathbf{W}^* \leftarrow f(\tilde{\mathbf{S}}_1, \ldots, \tilde{\mathbf{S}}_\ell)$ and compute $\mathsf{CT}_1 \leftarrow \mathsf{LHE.Sim}(\mathsf{aux}, \mathsf{pk}, \mathbf{W}^*)$. Statistical indistinguishability of hybrids $\mathcal{H}_1$ and $\mathcal{H}_0$ follows from the statistical circuit-privacy of LHE.

Hybrid $\mathcal{H}_2$. This hybrid is identical to the previous one except that we set $\mathbf{X}^* \leftarrow g(\tilde{\mathbf{S}}_1, \ldots, \tilde{\mathbf{S}}_\ell)$ and compute $\mathsf{CT}_1 \leftarrow \mathsf{LHE.Sim}(\mathsf{aux}, \mathsf{pk}, \mathbf{X}^*)$. Statistical indistinguishability of hybrids $\mathcal{H}_2$ and $\mathcal{H}_1$ follows from the statistical circuit-privacy of LHE.

Hybrid $\mathcal{H}_{3,i}$. This hybrid is identical to the previous one except that we set $\mathbf{D}_i$ to be $(d_{1,i}, \ldots, d_{m,i})$ where for $j \notin J_i$, $d_{j,i}$ is computed as in the previous hybrid and for $j \in J_i$ we set $d_{j,i} = 0$, where $J_i \leftarrow \mathsf{CoPIR.Ext}(\mathsf{copir}_{1,i})$. The hybrid is defined for $i = 1, \ldots, \ell$.

Statistical indistinguishability of hybrids $\mathcal{H}_{3,i-1}$ and $\mathcal{H}_{3,i}$ for $i = 1, \ldots, \ell$ (where $\mathcal{H}_{3,0} = \mathcal{H}_2$) follows from the statistical sender security of the underlying CoPIR.

Hybrid $\mathcal{H}_{4,i}$. This hybrid is identical to the previous one except that we set $\mathbf{F}_i$ to be $(\mathbf{f}_{1,i}, \ldots, \mathbf{f}_{m,i})$ where for $j \notin J_i$ we set $\mathbf{f}_{j,i} = 0$ and for $j \in J_i$ we set $\mathbf{f}_{j,i} = 0$ as in the previous hybrid, where $J_i \leftarrow \mathsf{CoPIR.Ext}(\mathsf{copir}_{1,i})$.[20] The hybrid is defined for $i = 1, \ldots, \ell$.

Statistical indistinguishability of hybrids $\mathcal{H}_{4,i-1}$ and $\mathcal{H}_{4,i}$ for $i = 1, \ldots, \ell$ (where $\mathcal{H}_{4,0} = \mathcal{H}_{3,\ell}$) follows from the statistical sender security of the underlying PIR.

Hybrid $\mathcal{H}_5$. This hybrid is identical to the previous one except that $\mathbf{W}^*$ is computed in the following way.: Let $\mathbf{w}_i = (w_{i,1}, \ldots, w_{i,m})$ be the rows of $\mathbf{W}^*$. For all positions $j \notin J_i$, $w_{i,j}$ is computed as in the previous hybrid, whereas for all positions $j \in J_i$, we set $w_{i,j} = 0$.

Note that $d_{j,i} = 0$ is the part of the ciphertext encoding $w_{i,j}$. Hence, the hybrids are identically distributed.

[20] Recall that due to the PIR compatibility, we have that $\mathsf{PIR.Ext}(\mathsf{q}_i) = \mathsf{CoPIR.Ext}(\mathsf{copir}_{1,i})$ where $\mathsf{q}_i = \mathsf{copir}_{1,i}$.

Hybrid $\mathcal{H}_6$. This hybrid is identical to the previous one except that $\mathbf{X}^*$ is computed in the following way.: Let $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m})$ be the rows of $\mathbf{X}^*$. For all positions $j \in J_i$, $x_{i,j}$ is computed as in the previous hybrid, whereas for all positions $j \notin J_i$, we set $x_{i,j} = 0$.

Note that $f_{j,i} = 0$ is the part of the ciphertext encoding $x_{i,j}$. Hence, the hybrids are identically distributed.

Note that the last hybrid does not depend on $\mathbf{m_{1-b}}$.

Case 2. In the second case we assume that $\mathcal{R}(x, w) = 0$. Since in this case the statement is false we can replace ot'_2 by 0 and compute $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, \mathcal{L}, 0)$. Statistical indistinguishability follows from the statistical sender security of the underlying CDS scheme. □

Communication complexity. We now analyze the communication complexity of our scheme. We start by analyzing the size of the receiver's message.

- **Size of ot_1.** The message ot_1 is composed by $\big(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}\}_{i\in[\ell]}, \mathsf{cds}_1\big)$. We have that
  - $|\mathsf{pk}| = \ell \cdot \mathsf{poly}(\lambda)$
  - $|\mathbf{A}| = n \times m$
  - $|\{\mathsf{ct}_i\}_{i\in[\ell]}| = \ell^2 \cdot n \cdot \mathsf{poly}(\lambda)$
  - $|\{\mathsf{copir}_{1,i}\}_{i\in[\ell]}| = t \cdot \mathsf{poly}(\lambda) \cdot m^{1-\delta}$.
  - $|\mathsf{cds}_1| = \ell^2 \cdot n \cdot \mathsf{poly}(\lambda)$.

  Thus the upload rate ρ_{up} can be upper-bounded by

$$\begin{aligned}\rho_{\mathsf{up}} &= \frac{\ell \cdot \mathsf{poly}(\lambda) + n \cdot m + \ell^2 \cdot n \cdot \mathsf{poly}(\lambda) + m^{1-\delta} \cdot t \cdot \mathsf{poly}(\lambda)}{m \cdot \ell} \\ &\leq 1 + \frac{\mathsf{poly}(\lambda)}{m} + \frac{n}{\ell} + \frac{\ell \cdot n \cdot \mathsf{poly}(\lambda)}{m} + \frac{t \cdot \mathsf{poly}(\lambda)}{m \cdot \ell}.\end{aligned}$$

  Setting $\ell = \lambda n$, $m = n^2\mathsf{poly}(\lambda)$ (for a sufficiently large $\mathsf{poly}(\lambda)$, then the upload-rate can be upper bounded by $1 + \mathcal{O}(1/\lambda)$.

- **Size of ot_2.** The message ot_2 is composed by cds_2. We have that $|\mathsf{cds}_2| = |\mathsf{ot}'_2| + \mathsf{poly}(x, w, \lambda) = |\mathsf{ot}'_2| + \mathsf{poly}(\ell, n, \lambda)$. The message ot'_2 is composed by $\big(\{g_i, \mathsf{copir}_{2,i}, h_i, \mathsf{r}_i\}_{i\in[\ell]}\big)$. Let $1 + \rho$ be the rate of the underlying CoPIR scheme. We have that
  - $|\{g_i\}_{i\in[\ell]}| = \ell \cdot \mathsf{poly}(\lambda)$
  - $|\{\mathsf{copir}_{2,i}\}_{i\in[\ell]}| = \ell \cdot |\mathbf{D}_i| \cdot (1 + \rho)$ where $|\mathbf{D}| = m$.
  - $|\{h_i\}_{i\in[\ell]}| = \ell \cdot \mathsf{poly}(\lambda)$
  - $|\{\mathsf{r}_i\}_{i\in[\ell]}| = \ell \cdot \mathsf{polylog}(m)\mathsf{poly}(\lambda)$.

  Thus the dowload rate ρ_{down} can be upper bounded by

$$\begin{aligned}\rho_{\mathsf{down}} &= \frac{2 \cdot \ell \cdot \mathsf{poly}(\lambda) + \ell \cdot m \cdot (1 + \rho) + \ell \cdot \mathsf{polylog}(m)\mathsf{poly}(\lambda) + \mathsf{poly}(\ell, n, \lambda)}{\ell \cdot m} \\ &\leq 1 + \rho + \frac{2\mathsf{poly}(\lambda)}{m} + \frac{\mathsf{polylog}(m) \cdot \mathsf{poly}(\lambda)}{m} + \frac{\mathsf{poly}(\ell, n, \lambda)}{\ell \cdot m}.\end{aligned}$$

  Setting $t = (\ell)^{1-\varepsilon}$ for some $\varepsilon > 0$ and m to be sufficiently large such that $\mathsf{poly}(\ell, n, \lambda) = m^{1-\varepsilon'}$, then we obtain a download rate that tends to 1 for large enough m.

Computational complexity. We briefly analyze the computational complexity of our protocol. Let $M = m\ell$ be the total number of OTs performed. Moreover let $T = t\ell$.

- **Receiver's work.** Let $\mathsf{C}^{\mathsf{R}}_{\mathsf{CoPIR}}$, $\mathsf{C}^{\mathsf{R}}_{\mathsf{PIR}}$, $\mathsf{C}^{\mathsf{R}}_{\mathsf{CDS}}$ be the receiver's computational complexity in the underlying CoPIR, PIR and CDS schemes. The receiver needs to encrypt ℓ matrices of size $\ell \times n$ (which takes work sublinear in M for a proper choice of m) and prepare ℓ LPN samples which takes work proportional to M. Moreover the receiver needs to run once CoPIR, PIR and CDS all of which take work at most $\mathcal{O}(M^{2-\gamma}T)$ (for some $1 > \gamma > 0$ if we use the scheme from Section 9). For a proper choice of t (i.e. sublinear in m) then the total computational complexity grows with $M^{1+\varepsilon}\mathsf{poly}(\lambda)$ for $1 > \varepsilon > 0$.
- **Sender's work.** Similarly, the sender's computational complexity is dominated by the CoPIR which takes work $\mathcal{O}(M^{2-\gamma}T)$. A similar analysis as above gives us that the total computational complexity grows with $M^{1+\varepsilon}\mathsf{poly}(\lambda)$ for $1 > \varepsilon > 0$.

11 Statistical Sender Private Oblivious Linear Evaluation with Overall Rate 1

Similarly to [DGI+19, BBDP22], our SSP OT protocol presented in Construction 6 can be extended to an OLE over finite fields of polynomial order.

We first describe the OLE primitive.

Definition 32. *Let $\mathbb{F}$ be a finite field. A two-round statistical sender private oblivious linear evaluation (SSP OLE) scheme OLE over F is composed by the following algorithms:*

- $\mathsf{OLER}(1^\lambda, \mathbf{b})$ *takes as input the security parameter λ and a vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{F}^n$. It outputs a message ole_1 and a private state st.*
- $\mathsf{OLES}(\mathsf{ole}_1, \mathbf{a}, \mathbf{b})$ *takes as input a first OLE message ole_1 and two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$. It outputs a second OLE message ole_2.*
- $\mathsf{OLED}(\mathsf{ole}_2, \mathsf{st})$ *takes as input a second OLE message ole_2 and a private state st. It outputs a vector $\mathbf{y}$.*

An OLE should be correct, sender secure and receiver secure. Here, we consider a special sender security definition called statistical sender security.

Definition 33 (Correctness)**.** *An OLE scheme OLE is said to be correct if for any $n \in \mathsf{poly}(\lambda)$, any pair $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$ and any $\mathbf{x} \in \mathbb{F}^n$, we have that*

$$\Pr\left[y_i = x_i a_i + b_i \forall i \in [n] : \begin{array}{c}(\mathsf{ole}_1, \mathsf{st}) \leftarrow \mathsf{OLER}(1^\lambda, \mathbf{x}) \\ \mathsf{ole}_2 \leftarrow \mathsf{OLES}(\mathsf{ole}_1, \mathbf{a}, \mathbf{b}) \\ \mathbf{y} = \mathsf{OLED}(\mathsf{ole}_2, \mathsf{st})\end{array}\right] = 1.$$

Definition 34 (Statistical sender privacy)**.** *An OLE scheme OLE is said to be statistical sender private if for any $\lambda \in \mathbb{N}$, any $n = \mathsf{poly}(\lambda)$, there exists an extractor $\mathsf{OLE.Ext}$ and a simulator $\mathsf{OLE.Sim}$ such that for any pair $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$ and any ole_1 we have that*

$$\mathsf{OLES}(\mathsf{ole}_1, \mathbf{a}, \mathbf{b}) \approx_s \mathsf{Sim}(\mathsf{ole}_1, \mathbf{y})$$

where $\mathbf{x} \leftarrow \mathsf{OT.Ext}(\mathsf{ole}_1)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ such that $y_i = x_i a_i + b_i$ for all $i \in [n]$.

Definition 35 (Receiver security)**.** *An OLE scheme OLE is said to be receiver secure if for any PPT adversary $\mathcal{A}$, λ and any $\mathbf{x}, \mathbf{x}' \in \mathbb{F}^n$ we have that*

$$\left|\begin{array}{c}\Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathsf{ole}_1) : (\mathsf{ole}_1, \mathsf{st}) \leftarrow \mathsf{OLER}(1^\lambda, \mathbf{x})] - \\ \Pr[1 \leftarrow \mathcal{A}(1^\lambda, \mathsf{ole}_1) : (\mathsf{ole}_1, \mathsf{st}) \leftarrow \mathsf{OLER}(1^\lambda, \mathbf{x}')]\end{array}\right| \leq \mathsf{negl}(\lambda).$$

A non-compact scheme. We remark that it is easy to build a SSP OLE over $\mathbb{Z}_p$ for some $p = \mathsf{poly}(\lambda)$ from any statistically malicious circuit-private LHE over $\mathbb{Z}_p$ such as the scheme presented in Appendix A. The resulting scheme is non-compact meaning that i) the size of the receiver's rate is $\mathsf{poly}(\lambda, |x|) = \mathsf{poly}(\lambda, \log p)$ where $x \in \mathbb{Z}_p$ is the receiver's input; and ii) the size of the sender's message is $\mathsf{poly}(\lambda, |a|, |b|) = \mathsf{poly}(\lambda, \log p)$ where (a, b) is the input of the sender.

11.1 Extension to Co-PIR over $\mathbb{Z}_q$.

We first sketch how our SSP co-PIR scheme (in which the database $\mathbf{D}$ is over $\mathbb{Z}_2$) can be extended into an SSP co-PIR over $\mathbb{Z}_q$ for a prime q such that $q = \mathsf{poly}(\lambda)$. To show this, we will show that each of the individual steps can be extended to support values over $\mathbb{Z}_q$.

Semi-honest one-query co-PIR. The scheme from Section 6 can easily be extended to support databases of the form $\mathbf{D} \in (\mathbb{Z}_q^k)^m$, by considering the following modifications: We use as building blocks codes $\mathcal{C}_1, \mathcal{C}_2$ and a LHE LHE over $\mathbb{Z}_q$ (instead of schemes over $\mathbb{Z}_q$) and a function decomp_q which decomposes an integer into base q (instead of the function $\{0,1\}$).

The arguments for correctness, receiver security and statistical (semi-honest) sender security can be straightforwardly adapted to the $\mathbb{Z}_q$ case, where $q = \mathsf{poly}(\lambda)$. It is also easy to see that the scheme also fulfills local correctness and self-reducibility using the same arguments.

SSP Co-PIR over $\mathbb{Z}_q$. The transformations from sections 7 (here we consider *position* co-PIR where the receiver obtains a single component of the block), 8 and 9 work for any modulus q. Since the semi-honest one-query co-PIR has all the required properties, we can first apply the transformation of Section 8 followed by the one of Section 9. The resulting scheme is a SSP co-PIR over $\mathbb{Z}_q$. Finally, turning a block co-PIR into a single element co-PIR is easy as some of the erased positions can be sent to the receiver via a PIR just as in Section 7.

CDS for DDH-based encryption. It remains to show how the CDS scheme from Section 4 can be adapted for the language

$$\mathcal{L}_{\mathsf{pk}} = \left\{ \{\mathsf{ct}_i\}_{i\in[\mu]} : \mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, a_i \cdot \mathbf{I}_\ell) \wedge a_i \in \mathbb{Z}_q \right\}$$

for $q = \mathsf{poly}(\lambda)$. The main difference is that the sender samples $\mathbf{r}_{j,i}$ for all $j \in [q]$ and sets

$$\mathbf{y}_{1,i} = \begin{pmatrix} \mathbf{r}_{1,i} \\ \vdots \\ \mathbf{r}_{1,i} \end{pmatrix} \quad \text{and} \quad \mathbf{y}_{2,i} = \begin{pmatrix} \mathbf{d}_i \\ \mathbf{d}_i - \mathbf{r}_{2,i} \\ \vdots \\ \mathbf{d}_i - (q-1)\mathbf{r}_{q,i} \end{pmatrix}.$$

For correctness note that if $a_i \in \mathbb{Z}_q$ then the i-th row of $\mathbf{w}_i$ reveals $\mathbf{d}_i$. The argument for statistical sender security can also be straightforwardly adapted to the $\mathbb{Z}_q$ case.

11.2 Statistical Sender Private OLE with Optimal Rate

We now present the protocol for SSP OLE with optimal rate. The construction follows the same outline as in Construction 6. The main difference is that, for the positions where there is an LPN error, the receiver is going to send a first message of a non-compact OLE. The sender computes the second message of the non-compact OLE scheme for all possible values in its database and then sends the results over a PIR to the receiver.

This construction can also be seen as a compiler that takes a non-compact OLE and turns it into an optimal-rate OLE by additionally assuming LPN and DDH.

Ingredients. Let p be a prime number. We will describe an OLE over $\mathbb{Z}_p$ for $p = \mathsf{poly}(\lambda)$. We will need the following ingredients for our protocol:

- A PIR scheme $\mathsf{PIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Retrieve})$.
- A (single position) co-PIR scheme $\mathsf{CoPIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Rec})$ that is PIR-compatible parametrized by m.
- A rate-1 circuit-private LHE scheme $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval}, \mathsf{Shrink}, \mathsf{DecShrink})$ with plaintext space $\mathbb{Z}_p^\ell$ and for which shrinked ciphertexts have the form $\mathsf{ct} = (g, d_1, \ldots, d_\ell)$ where $g \in \mathbb{G}$ (for some large enough group $\mathbb{G}$) and $d_i \in \mathbb{Z}_p$.
- An (low-rate) OLE scheme $\mathsf{LROLE} = (\mathsf{Rec}, \mathsf{Send}, \mathsf{Dec})$ over $\mathbb{Z}_p$.
- A download rate-1 CDS scheme $\mathsf{CDS} = (\mathsf{Enc}, \mathsf{Send}, \mathsf{Release})$ for the language

$$\mathcal{L} = \left\{ \mathsf{pk}, \{\mathsf{ct}_i\}_{i\in[\ell]} : \exists (r, \mathbf{s}_i, r_i) \text{ s.t. } \begin{array}{c} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, \ell; r) \\ \mathbf{S}_i = \mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{s}_i) \\ \mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{S}_i; r_i) \end{array} \right\}$$

  for some $\mathbf{s}_i \in \mathbb{Z}_p^n$.
- The binary $\mathsf{LPN}(n, m, \rho, p)$ problem with dimension $n = \mathsf{poly}(\lambda)$, $m = n \cdot \ell \cdot \mathsf{poly}(\lambda)$ samples and slightly sub-constant noise-rate $\rho = m^{1-\epsilon}$.

Construction 7 (Optimal-rate SSP OLE)**.** *We now describe the scheme in full detail.*

$\mathsf{OLER}(1^\lambda, \mathbf{b} \in \mathbb{Z}_p^{m\ell})$:

- *Parse* $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_\ell)$*, where the* $\mathbf{b}_i \in \mathbb{Z}_p^m$ *are blocks of size* m*.*
- *Choose* $\mathbf{A} \leftarrow\!\!\$\, \mathbb{Z}_p^{n\times m}$ *uniformly at random and compute a pair of public and secret key* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{LHE.KeyGen}(1^\lambda, \ell)$*.*
- *For all* $i \in [\ell]$*, choose* $\mathbf{s}_i \leftarrow\!\!\$\, \mathbb{Z}_p^n$*, and* $\mathbf{e}_i \leftarrow\!\!\$\, \chi_{p,m,t}$*, compute* $\mathbf{c}_i \leftarrow \mathbf{s}_i\mathbf{A} + \mathbf{e}_i + \mathbf{b}_i$*, and set* $\mathbf{S}_i \leftarrow \mathsf{SingleRowMatrix}(\ell, n, i, \mathbf{s}_i)$*. Compute a matrix-ciphertext* $\mathsf{ct}_i \leftarrow \mathsf{LHE.Enc}(\mathsf{pk}, \mathbf{S}_i)$*.*
- *Compute* $(\mathsf{cds}_1, \tilde{\mathsf{st}}) \leftarrow \mathsf{CDS.Enc}(1^\lambda, w)$ *where* $w = (r, \{\mathbf{S}_i, r_i\}_{i\in[\ell]})$*.*
- *For all* $i \in [\ell]$ *set* $J_i = \mathsf{Supp}(\mathbf{e}_i)$ *to be the support of* $\mathbf{e}_i$*. Compute* $(\mathsf{copir}_{1,i}, \mathsf{st}_i) \leftarrow \mathsf{CoPIR.Query}(J_i)$*.*[21]
- *For all* $i \in [\ell]$ *and all* $j \in [t]$*, compute* $(\mathsf{lrole}_{1,i,j}, \mathsf{st}'_{i,j}) \leftarrow \mathsf{LROLE.Rec}(1^\lambda, b_{i,J[j]})$*.*
- *Output* $\mathsf{ole}_1 = \big(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}, \{\mathsf{lrole}_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}, \mathsf{cds}_1\big)$ *and* $\mathsf{st} = (\mathsf{sk}, \{\mathsf{st}_i, J_i, \{\mathsf{st}'_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}, \tilde{\mathsf{st}})$*.*

$\mathsf{OLES}(\mathsf{ole}_1, (\mathbf{v}_0, \mathbf{v}_1) \in (\mathbb{Z}_p^{m\ell})^2)$:

- *Parse* $\mathbf{v}_0 = (\mathbf{v}_{0,1}, \ldots, \mathbf{v}_{0,\ell})$ *and* $\mathbf{v}_1 = (\mathbf{v}_{1,1}, \ldots, \mathbf{v}_{1,\ell})$*, where each* $\mathbf{v}_{b,i} = (v_{b,i,1}, \ldots, v_{b,i,m}) \in \mathbb{Z}_p^m$ *and* $v_{b,i,j} \in \mathbb{Z}_p$*. Parse* ole_1 *as* $\big(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}, \{\mathsf{lrole}_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}, \mathsf{cds}_1\big)$*.*
- *For* $i \in [\ell]$ *set* $\mathbf{z}_i = \mathbf{v}_{1,i}$*.*
- *Set* $\mathbf{Z} = \mathsf{RowMatrix}(\ell, m, \mathbf{z}_1, \ldots, \mathbf{z}_\ell)$*.*
- *For all* $i \in [\ell]$ *set* $\mathbf{C}_i = \mathsf{SingleRowMatrix}(\ell, m, i, \mathbf{c}_i)$ *and* $\mathbf{D}_i = \mathsf{Diag}(m, \mathbf{v}_{0,i})$*.*

[21] Recall that, since the CoPIR scheme is PIR-compatible then $\mathsf{copir}_{1,i}$ also corresponds to a first message PIR with input J_i.

- *Define the $\mathbb{Z}_p$-linear function $f : (\mathbb{Z}_p^{\ell\times n})^\ell \to \mathbb{Z}_p^{\ell\times m}$ via*

$$f(\mathbf{X}_1,\ldots,\mathbf{X}_\ell) = \left(\sum_{i=1}^{\ell}(-\mathbf{X}_i\mathbf{A}+\mathbf{C}_i)\cdot\mathbf{D}_i\right)+\mathbf{Z}.$$

- *For all $i\in[\ell]$, $j\in[t]$ and $o\in[m]$ compute* $\mathsf{lrole}_{2,i,j,o} \leftarrow \mathsf{LROLE}(\mathsf{lrole}_{i,j}, v_{0,i,o}, v_{1,i,o})$.
- *Compute* $\mathsf{CT}_1 \leftarrow \mathsf{LHE.Eval\&Shrink}(\mathsf{pk}, f, \mathsf{ct}_1,\ldots,\mathsf{ct}_\ell)$.
- *Parse* CT_1 *as* $\{g_i, d_{i,1},\ldots,d_{i,\ell}\}_{i\in[m]}$ *where each* $g_i\in\mathbb{G}$ *and* $d_{i,j}\in\mathbb{Z}_p$.
- *For all $i\in[\ell]$ set* $\mathbf{D}_i = (d_{1,i},\ldots,d_{m,i})$. *Compute* $\mathsf{copir}_{2,i} \leftarrow \mathsf{CoPIR.Send}(\mathsf{copir}_{1,i}, \mathbf{D}_i)$
- *For all $i\in[\ell]$, parse* $\mathsf{copir}_{1,i}$ *as* $\mathsf{q}_i = (\mathsf{q}_{i,1},\ldots,\mathsf{q}_{i,t})$.
- *For all $i\in[\ell]$ and $j\in[t]$ set* $\mathbf{F}_{i,j} = (\mathsf{lrole}_{2,i,j,1},\ldots,\mathsf{lrole}_{2,i,j,m})$. *Compute* $\mathsf{r}_{i,j} \leftarrow \mathsf{PIR.Send}(\mathsf{q}_{i,j}, \mathbf{F}_{i,j})$.
- *Set* $\mathsf{ole}_2' = \{g_i, \mathsf{copir}_{2,i}, \{\mathsf{r}_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}$.
- *Compute* $\mathsf{cds}_2 \leftarrow \mathsf{CDS.Send}(\mathsf{cds}_1, \mathcal{L}, \mathsf{ole}_2')$.
- *Output* $\mathsf{ole}_2 = \mathsf{cds}_2$.

$\mathsf{OLED}(\mathsf{ole}_2, \mathsf{st})$:

- *Parse* ole_2 *as* cds_2 *and* $\mathsf{st} = (\mathsf{sk}, \{\mathsf{st}_i, J_i, \{\mathsf{st}'_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}\tilde{\mathsf{st}})$.
- *Run* $\mathsf{ole}_2' \leftarrow \mathsf{CDS.Release}(\mathsf{cds}_2, \tilde{\mathsf{st}})$. *Parse* ole_2' *as* $\{g_i, \mathsf{copir}_{2,i}, \{\mathsf{r}_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}$.
- *For all $i\in[\ell]$ compute* $\tilde{\mathbf{D}}_i = (\tilde{d}_{1,i},\ldots,\tilde{d}_{m,i}) \leftarrow \mathsf{CoPIR.Retrieve}(\mathsf{copir}_{2,i}, \mathsf{st}_i)$.
- *Set* $\tilde{\mathsf{CT}}_1$ *to be* $\{g_i, \tilde{d}_{i,1},\ldots,\tilde{d}_{i,\ell}\}_{i\in[m]}$. *Compute* $\mathbf{W} \leftarrow \mathsf{LHE.DecShrink}(\mathsf{sk}, \tilde{\mathsf{CT}}_1)$ *where* $\mathbf{W}\in\mathbb{Z}_p^{\ell\times m}$. *Parse* $\mathbf{W} = (w_{i,j})_{i\in[\ell],j\in[m]}$ *where* $w_{i,j}\in\mathbb{Z}_p$
- *For all $i\in[\ell]$ and all $j\in[t]$ compute* $\mathsf{lrole}_{2,i,J_i[t]} \leftarrow \mathsf{PIR.Retrieve}(\mathsf{r}_{i,j}, \mathsf{st}_i)$. *Additionally, compute* $y_{i,J_i[j]} \leftarrow \mathsf{LROLE.Dec}(\mathsf{lrole}_{2,i,J_i[t]}, \mathsf{st}'_{i,j})$.
- *Set* $\mathbf{M} = (m_{i,j})_{i\in[\ell],j\in[m]} \in \mathbb{Z}_p^{\ell\times mk}$ *where*

$$m_{i,j} = \begin{cases} y_{i,J_i[l]} \text{ if } l = J_i[l] \\ w_{i,j} \text{ otherwise} \end{cases}.$$

- *Output* $\mathbf{M}$.

The proofs of the following theorems follow the same reasoning as the proofs of theorems 4, 5 and 6 respectively.

Theorem 7 (Correctness)**.** *Assume that $p = \mathsf{poly}(\lambda)$. The scheme presented in Construction 7 is correct assuming that* LHE*,* CDS*,* CoPIR*,* PIR *and* LROLE *are correct.*

Theorem 8 (Receiver security)**.** *The scheme presented in Construction 7 is receiver secure assuming that* LHE *is IND-CPA,* CoPIR*,* CDS*,* PIR *and* LROLE *are receiver secure and that the* $\mathsf{LPN}(n, m, \rho)$ *assumption holds for slightly sub-constant noise-rate $\rho = m^{1-\epsilon}$ where $\epsilon > 0$.*

Theorem 9 (Statistical sender security)**.** *The scheme presented in Construction 7 is statistical sender secure assuming that* LHE *is statistically circuit private and that* CDS*,* CoPIR*,* PIR *and* LROLE *are statistical sender secure.*

Communication complexity. We now analyze the communication complexity of the scheme. Let $p = \mathsf{poly}(\lambda)$ (which is a necessary condition for correctness).

- **Size of ot_1.** The message ot_1 is composed by $\left(\mathsf{pk}, \mathbf{A}, \{\mathsf{ct}_i, \mathbf{c}_i, \mathsf{copir}_{1,i}, \{\mathsf{lrole}_{i,j}\}_{j\in[t]}\}_{i\in[\ell]}\right)$. We have that
  - $|\mathsf{pk}| = \ell \cdot \mathsf{poly}(\lambda)$
  - $|\mathbf{A}| = n \cdot m \cdot \log p$
  - $|\{\mathsf{ct}_i\}_{i\in[\ell]}| = \ell^2 \cdot n \cdot \mathsf{poly}(\lambda)$
  - $|\{\mathsf{copir}_{1,i}\}_{i\in[\ell]}| = t \cdot \mathsf{poly}(\lambda) \cdot m^{1-\delta}$.
  - $|\{\mathsf{lrole}_{i,j}\}_{i\in[\ell],j\in[t]}| = \ell \cdot t \cdot \mathsf{poly}(\lambda, \log q)$.
  - $|\mathsf{cds}_1| = \ell n \cdot \mathsf{poly}(\lambda)$.

  Thus the upload rate ρ_{up} can be upper-bounded by

$$\rho_{\mathsf{up}} \leq 1 + \frac{\mathsf{poly}(\lambda)}{m} + \frac{n}{\ell} + \frac{\ell \cdot n \cdot \mathsf{poly}(\lambda)}{m} + \frac{t \cdot \mathsf{poly}(\lambda)}{m \cdot \ell} + \frac{t \cdot \mathsf{poly}(\lambda)}{m}.$$

  Setting ℓ, n, m and t as in Section 10, the upload-rate can be upper bounded by $1 + \mathcal{O}(1/\lambda)$.

- **Size of ot_2.** Following a very similar argument as in Section 10, we can upper-bound the download rate ρ_{down} by

$$\rho_{\mathsf{down}} \leq 1 + \rho + \frac{\mathsf{poly}(\lambda)}{m} + \frac{t \cdot \mathsf{polylog}(m) \cdot \mathsf{poly}(\lambda)}{m}$$

  where $1 + \rho$ is the rate of the underlying CoPIR scheme. Again, choosing t, m as in Section 10 yields overall rate 1.

12 Two-Party Secure Computation with Overall Communication of $\mathcal{O}(|\mathcal{C}|) + \mathsf{poly}(\lambda)$

We now show an application of our optimal overall rate SSP OT. This application is in constructing a 2-Party secure computation (2PC) scheme with overall communication of $\mathcal{O}(|\mathcal{C}|) + \mathsf{poly}(\lambda)$, where $\mathcal{C}$ is the circuit to be computed and provides statistical semi-honest security against one of the parties and computational semi-honest security against the other party.

The protocol is just the classical GMW protocol [GMW87] in the OT correlations model where the OT correlations are generated using our SSP OT.

Specifically, to compute a secret sharing of the AND of two wires whose values are themselves secret shared as (a_1, a_2) and (b_1, b_2) respectively, the parties first compute locally $a_1 \cdot b_1$ and $a_2 \cdot b_2$. One of the parties acts as the sender and the other acts as the receiver in two instances of 1-out-of-2 OT. Assume w.l.o.g. that P_1 acts as the sender and P_2 acts as the receiver. P_2 uses a_2 and b_2 as its choice bits and P_1 uses $(r_1, b_1 + r_1)$ and $(r_2, a_1 + r_2)$ respectively as the sender messages. P_2 obtains $a_2 b_1 + r_1$ and $a_1 b_2 + r_2$ as the outputs of the two OT executions. P_1 sets the share of the AND to be $a_1 b_1 + r_1 + r_2$ and P_2 sets its share to be $a_2 b_2 + a_1 b_2 + r_1 + a_2 b_1 + r_2$. Note that instantiating the GMW protocol with an OT scheme that does not have overall rate-1 incurs an communication complexity of $\mathsf{poly}(|\mathcal{C}|, \lambda)$.

Lemma 19 ([GMW87]). *Given a circuit $\mathcal{C} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$, there exists a two-party $O(|\mathcal{C}|)$-round protocol in the OT-correlation model such that*

- *The protocol provides semi-honest statistical security.*
- *The communication complexity is upper-bounded by $6|\mathcal{C}| + n + m + \mathsf{poly}(\lambda)$.*

- *Both parties share* $2|\mathcal{C}|$ *OT correlations.*

For each gate, the parties need to perform 2 chosen input OTs. If they share 2 random OT correlations, these can be derandomized using the standard transformation from random OT to chosen input OT which takes 3 bits of communication per OT. Thus, the total communication is 6 bits per gate. If we setup the OT correlations using our SSP OT scheme, we obtain the following corollary.

Corollary 2. *Given a circuit* $\mathcal{C} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$*, there exists a two-party* α*-round protocol in the standard model such that*

- *The protocol is semi-honest secure against one of the parties and statistically semi-honest secure against the other one.*
- *The communication complexity is upperbounded by* $10|\mathcal{C}| + n + m + \mathsf{poly}(\lambda)$*.*

The $2 \cdot |\mathcal{C}|$ OT correlations can be shared using the OT scheme from Section 10 incurring in total communication approaching $4|\mathcal{C}| + \mathsf{poly}(\lambda)$ for large enough $|\mathcal{C}|$. Plugging this with the lemma above, we obtain a scheme with total communication $10|\mathcal{C}| + n + m + \mathsf{poly}(\lambda)$. Moreover, statistical security for one of the parties follow from the SSP property of the underlying OT.

Acknowledgements

Pedro Branco was partially funded by the German Federal Ministry of Education and Research (BMBF) in the course of the 6GEM research hub under grant number 16KISK038. Nico Döttling: Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Akshayaram Srinivasan was supported in part by a SERB startup grant and Google India Research Award.

References

[ADD+22] Divesh Aggarwal, Nico Döttling, Jesko Dujmovic, Mohammad Hajiabadi, Giulio Malavolta, and Maciej Obremski. Algebraic Restriction Codes and Their Applications. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:15, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[AIK04] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. In *45th Annual Symposium on Foundations of Computer Science*, pages 166–175, Rome, Italy, October 17–19, 2004. IEEE Computer Society Press.

[AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135, Innsbruck, Austria, May 6–10, 2001. Springer, Heidelberg, Germany.

[App17] Benny Applebaum. *Garbled Circuits as Randomized Encodings of Functions: a Primer*, pages 1–44. Springer International Publishing, Cham, 2017.

[BBD+20] Zvika Brakerski, Pedro Branco, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Constant ciphertext-rate non-committing encryption from standard assumptions. In *TCC 2020: 18th Theory of Cryptography Conference, Part I*, Lecture Notes in Computer Science, pages 58–87. Springer, Heidelberg, Germany, March 2020.

[BBDP22] Zvika Brakerski, Pedro Branco, Nico Döttling, and Sihang Pu. Batch-OT with optimal rate. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022*, pages 157–186, Cham, 2022. Springer International Publishing.

[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. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 291–308. ACM Press, November 11–15, 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. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 489–518, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[BCG+20a] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. In *61st Annual Symposium on Foundations of Computer Science*, pages 1069–1080. IEEE Computer Society Press, 2020.

[BCG+20b] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators from ring-LPN. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2020, Part II*, Lecture Notes in Computer Science, pages 387–416, Santa Barbara, CA, USA, August 16–20, 2020. Springer, Heidelberg, Germany.

[BCGI18] Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 896–912, Toronto, ON, Canada, October 15–19, 2018. ACM Press.

[BCM22] Elette Boyle, Geoffroy Couteau, and Pierre Meyer. Sublinear secure computation from new assumptions. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography*, pages 121–150, Cham, 2022. Springer Nature Switzerland.

[BD18] Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 370–390, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany.

[BDGM19] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 407–437, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany.

[BF22] Nir Bitansky and Sapir Freizeit. Statistically sender-private OT from LPN and derandomization. Cryptology ePrint Archive, Paper 2022/185, 2022. `https://eprint.iacr.org/2022/185`.

[BFJ+20] Saikrishna Badrinarayanan, Rex Fernando, Aayush Jain, Dakshita Khurana, and Amit Sahai. Statistical ZAP arguments. In Vincent Rijmen and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, Lecture Notes in Computer Science, pages 642–667. Springer, Heidelberg, Germany, May 2020.

[BGI16] Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 509–539, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[BGI+17a] Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany.

[BGI17b] Elette Boyle, Niv Gilboa, and Yuval Ishai. Group-based secure computation: Optimizing rounds, communication, and computation. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017, Part II*, volume 10211 of *Lecture Notes in Computer Science*, pages 163–193, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[BKM20] Zvika Brakerski, Venkata Koppula, and Tamer Mour. NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2020, Part III*, Lecture Notes in Computer Science, pages 738–767, Santa Barbara, CA, USA, August 16–20, 2020. Springer, Heidelberg, Germany.

[BPS22] Saikrishna Badrinarayanan, Sikhar Patranabis, and Pratik Sarkar. Statistical security in two-party computation revisited. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography*, pages 181–210, Cham, 2022. Springer Nature Switzerland.

[CGH+21] Melissa Chase, Sanjam Garg, Mohammad Hajiabadi, Jialin Li, and Peihan Miao. Amortizing rate-1 OT and applications to PIR and PSI. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography*, pages 126–156, Cham, 2021. Springer International Publishing.

[CGKS95] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science*, pages 41–50, Milwaukee, Wisconsin, October 23–25, 1995. IEEE Computer Society Press.

[DGI+19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[DMO00] Giovanni Di Crescenzo, Tal Malkin, and Rafail Ostrovsky. Single database private information retrieval implies oblivious transfer. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 122–138, Bruges, Belgium, May 14–18, 2000. Springer, Heidelberg, Germany.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GH19] Craig Gentry and Shai Halevi. Compressible FHE with applications to PIR. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019: 17th Theory of Cryptography Conference, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 438–464, Nuremberg, Germany, December 1–5, 2019. Springer, Heidelberg, Germany.

[GHO20] Sanjam Garg, Mohammad Hajiabadi, and Rafail Ostrovsky. Efficient range-trapdoor functions and applications: Rate-1 OT and more. In *TCC 2020: 18th Theory of Cryptography Conference, Part I*, Lecture Notes in Computer Science, pages 88–116. Springer, Heidelberg, Germany, March 2020.

[GJJM20] Vipul Goyal, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Statistical zaps and new oblivious transfer protocols. In Vincent Rijmen and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, Lecture Notes in Computer Science, pages 668–699. Springer, Heidelberg, Germany, May 2020.

[GMPW20] Nicholas Genise, Daniele Micciancio, Chris Peikert, and Michael Walter. Improved discrete gaussian and subgaussian analysis for lattice cryptography. In *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I*, Lecture Notes in Computer Science, pages 623–651. Springer, Heidelberg, Germany, 2020.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, NY, USA, May 25–27, 1987. ACM Press.

[HK12] Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *Journal of Cryptology*, 25(1):158–193, January 2012.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st Annual Symposium on Foundations of Computer Science*, pages 294–304, Redondo Beach, CA, USA, November 12–14, 2000. IEEE Computer Society Press.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *ICALP 2002: 29th International Colloquium on Automata, Languages and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pages 244–256, Malaga, Spain, July 8–13, 2002. Springer, Heidelberg, Germany.

[IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 145–161, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.

[IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 433–442, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

[IP07] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany.

[JJ21] Abhishek Jain and Zhengzhong Jin. Non-interactive zero knowledge from sub-exponential ddh. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 3–32, Cham, 2021. Springer International Publishing.

[JKKR17] Abhishek Jain, Yael Tauman Kalai, Dakshita Khurana, and Ron Rothblum. Distinguisher-dependent simulation in two rounds and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 158–189, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 60–73, New York, NY, USA, 2021. Association for Computing Machinery.

[KKS18] Yael Tauman Kalai, Dakshita Khurana, and Amit Sahai. Statistical witness indistinguishability (and more) in two messages. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 34–65, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany.

[KLVW22] Yael Tauman Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and ram delegation. Cryptology ePrint Archive, Paper 2022/1320, 2022. `https://eprint.iacr.org/2022/1320`.

[KM20] Dakshita Khurana and Muhammad Haris Mughees. On statistical security in two-party computation. In *TCC 2020: 18th Theory of Cryptography Conference, Part II*, Lecture Notes in Computer Science, pages 532–561. Springer, Heidelberg, Germany, March 2020.

[KS17] Dakshita Khurana and Amit Sahai. How to achieve non-malleability in one or two rounds. In Chris Umans, editor, *58th Annual Symposium on Foundations of Computer Science*, pages 564–575, Berkeley, CA, USA, October 15–17, 2017. IEEE Computer Society Press.

[MS20] Daniele Micciancio and Jessica Sorrell. Simpler statistically sender private oblivious transfer from ideals of cyclotomic integers. In *Advances in Cryptology – ASIACRYPT 2020, Part II*, Lecture Notes in Computer Science, pages 381–407. Springer, Heidelberg, Germany, December 2020.

[NP01] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In S. Rao Kosaraju, editor, *12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 448–457, Washington, DC, USA, January 7–9, 2001. ACM-SIAM.

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 536–553, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[Pei10] Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 187–196, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

[Rab05] Michael O Rabin. How to exchange secrets with oblivious transfer. *Cryptology ePrint Archive*, 2005.

A Rate-1 Linearly-Homomorphic Encryption from DDH

In this section we present a rate-1 LHE from DDH that fulfills malicious circuit-privacy.

In the following, let $\mathcal{G}$ be a (prime-order) *group generator*, that is, $\mathcal{G}$ is an algorithm that takes as an input a security parameter 1^λ and outputs $(\mathbb{G}, p, g)$, where $\mathbb{G}$ is the description of a multiplicative cyclic group, p is the order of the group which is always a prime number unless differently specified, and g is a generator of the group. In the following we state the decisional version of the Diffie-Hellman (DDH) assumption.

Definition 36 (Decisional Diffie-Hellman Assumption)**.** *Let $(\mathbb{G}, p, g) \leftarrow\!\$\, \mathcal{G}(1^\lambda)$. We say that the DDH assumption holds (with respect to $\mathcal{G}$) if for any PPT adversary $\mathcal{A}$*

$$\left|\Pr[1 \leftarrow \mathcal{A}((\mathbb{G}, p, g), (g^a, g^b, g^{ab}))] - \Pr[1 \leftarrow \mathcal{A}((\mathbb{G}, p, g), (g^a, g^b, g^c))]\right| \leq \mathsf{negl}(\lambda)$$

where $a, b, c \leftarrow\!\$\, \mathbb{Z}_p$.

Randomized rounding. Let $\rho_s(\mathbf{x})$ be probability distribution of the Gaussian distribution over $\mathbb{R}^n$ with parameter s and centered in 0. We define the discrete Gaussian distribution $D_{S,s}$ over S and with parameter s by the probability distribution $\rho_s(\mathbf{x})/\rho(S)$ for all $\mathbf{x} \in S$ (where $\rho_s(S) = \sum_{\mathbf{x}\in S} \rho_s(\mathbf{x})$).

Definition 37 ([Pei10])**.** *Let $\sigma > 0$. For any $x \in \mathbb{R}$, the gaussian rounding $\lceil x \rfloor_\sigma$ is a random variable supported on $\mathbb{Z}$ defined by*

$$\lceil x \rfloor_\sigma = x + D_{\mathbb{Z}-x,\sigma}.$$

In other words, $\lceil x \rfloor_\sigma$ is a discrete gaussian centered on $x \in \mathbb{R}$ but supported on $\mathbb{Z}$.

We will use the following convolution lemma which provides a *simulation property* for gaussian rounding. This lemma follows from a lemma presented in [GMPW20].

Lemma 20 ([BBDP22])**.** *Let $\epsilon > 0$ be bounded by a sufficiently small constant and let $\sigma_1, \sigma_2 \geq \eta_\epsilon(\mathbb{Z})$. Then it holds for all $x, y \in \mathbb{R}$ that*

$$\lceil x \rfloor_{\sigma_1} + \lceil y \rfloor_{\sigma_2} \approx_s \lceil x + y \rfloor_{\sqrt{\sigma_1^2+\sigma_2^2}}.$$

It immediately follows from Lemma 20 that it holds for every integer $p \geq 2$ that

$$\lceil x \rfloor_{\sigma_1} + \lceil y \rfloor_{\sigma_2} \mod p \approx_s \lceil x + y \rfloor_{\sqrt{\sigma_1^2+\sigma_2^2}} \mod p.$$

Shrinking ciphertexts. We present lemmas from previous works that guarantee the existence of shrinking algorithms. That is, given a specific (packed) El Gamal encryption, there are algorithms that allow us to shrink the ciphertext to a rate-1 ciphertext. Both algorithms are inspired by previous works [BGI16, DGI+19, BBD+20].

In the following let $(\mathbb{G}, p, g) \leftarrow\!\$\, \mathcal{G}(1^\lambda)$ and $k \in \mathbb{Z}$. Consider an El Gamal public key of the form $\mathsf{pk} = (g, (h_1, \ldots, h_k) = (g, (g^{x_1}, \ldots, g^{x_k})) \in \mathbb{G}^{k+1}$ for $x_1, \ldots, x_k \leftarrow\!\$\, \mathbb{Z}_p$ (here, $\mathbf{x} = (x_1, \ldots, x_k)$ is the secret key). Consider the modified El Gamal encryption algorithm where a ciphertext for $\mathbf{m} = (m_1, \ldots, m_k) \in \{0,1\}^k$ is of the form $\mathsf{ct} = (c_1, (c_{2,1}, \ldots, c_{2,k})) \in \mathbb{G}^{k+1}$ where $c_1 = g^r$ and $c_{2,i} = h_i^r g^{\lceil m_i(p/2) \rfloor_\sigma}$.

Lemma 21 ([BBDP22])**.** *There exists a pair of (expected) PPT algorithms $(\mathsf{Shrink}_{\mathsf{DDH}}, \mathsf{DecShrink}_{\mathsf{DDH}})$ such that if $\mathsf{ct} = (g, h_1, \ldots, h_n)$ is a modified El Gamal ciphertext encrypting a message $\mathbf{m} \in \{0,1\}^n$ (as described above) the we have that*

- $(c, K, b_1, \ldots, b_n) \leftarrow \mathsf{Shrink}_{\mathsf{DDH}}(\mathsf{ct})$.
- $\mathbf{m} \leftarrow \mathsf{DecShrink}_{\mathsf{DDH}}(\mathsf{sk}, \mathsf{Shrink}_{\mathsf{DDH}}(\mathsf{ct}))$.

A.1 Construction

We now sketch the construction which is heavily based on the constructions from [ADD+22, BBDP22].

Construction 8. *Let* $(\mathsf{GShrink}_{\mathsf{DDH}}, \mathsf{GDecShrink}_{\mathsf{DDH}})$ *be the pair of algorithms described in Lemma 21.*

$\mathsf{KeyGen}(1^\lambda, k)$:

- $(\mathbb{G}, p, g) \leftarrow\!\$\, \mathcal{G}(1^\lambda)$
- *Sample* $x_1, \ldots, x_k \leftarrow\!\$\, \mathbb{Z}_p$. *Compute* $h_i = g^{x_i}$.
- *Output* $\mathsf{pk} = (\mathbb{G}, p, g, h_1, \ldots, h_k)$ *and* $\mathsf{sk} = \mathbf{x} = (x_1, \ldots, x_k)$.

$\mathsf{Enc}(\mathsf{pk}, \mathbf{m} = (m_1, \ldots, m_k))$:

- *Parse* pk *as* $(\mathbb{G}, p, g, h_1, \ldots, h_k)$.
- *Sample* $r \leftarrow\!\$\, \mathbb{Z}_p$. *Compute* $c_1 = g^r$ *and* $c_{2,i} = h_i^r g^{m_i}$ *for* $i \in [k]$.
- *Output* $\mathsf{ct} = (c_1, (c_{2,1}, \ldots, c_{2,k}))$.

$\mathsf{Eval}(\mathsf{pk}, f, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell))$

- *Parse* pk *as* $(\mathbb{G}, p, g, h_1, \ldots, h_k)$, f *as* $f(\mathbf{x}_1, \ldots, \mathbf{x}_\ell) = \sum_{i=1}^{\ell} a_i \mathbf{x}_i + \mathbf{b}$ *for* $\mathbf{a} = (a_1, \ldots, a_\ell) \in \mathbb{Z}_2^\ell$ *and* $\mathbf{b} \in \mathbb{Z}_2^k$ *and* ct_i *as* $(c_{1,i}, \mathbf{c}_{2,i})$ *where* $\mathbf{c}_{2,i} = (c_{2,1,i}, \ldots, c_{2,k,i}))$ *for* $i \in [\ell]$.
- *Compute* $\bar{\mathsf{ct}} = (\bar{c}_1, (\bar{c}_{2,1}, \ldots, \bar{c}_{2,1}))$ *where*

$$\bar{c}_1 = \prod_{i=1}^{\ell} \left(c_{1,i}^{\lceil a_i \frac{p}{2} \rfloor_\sigma} \cdot (g \cdot c_{1,i}^{-1})^{\lceil 0 \rfloor_\sigma} \right) \cdot g^t$$

and

$$\bar{\mathbf{c}}_2 = \bigodot_{i=1}^{\ell} \left(\mathbf{c}_{2,i}^{\lceil a_i \frac{p}{2} \rfloor_\sigma} \odot (g \cdot \mathbf{c}_{2,i}^{-1})^{\lceil 0 \rfloor_\sigma} \right) \odot \left(g^{\lceil b_1 \frac{p}{2} \rfloor_\sigma}, \ldots, g^{\lceil b_k \frac{p}{2} \rfloor_\sigma} \right) \odot (h_1^t, \ldots, h_k^t)$$

for $t \leftarrow\!\$\, \mathbb{Z}_p$ *and where* $\odot$ *denotes the component-wise multiplication.*

- *Output* $\bar{\mathsf{ct}}$.

$\mathsf{Shrink}(\mathsf{pk}, \mathsf{ct})$: *Output* $\bar{\mathsf{ct}} \leftarrow \mathsf{Shrink}_{\mathsf{DDH}}(\mathsf{pk}, \mathsf{ct})$.

$\mathsf{DecShrink}(\mathsf{sk}, \mathsf{ct})$: *Output* $\mathbf{m} \leftarrow \mathsf{DecShrink}_{\mathsf{DDH}}(\mathsf{sk}, \bar{\mathsf{ct}})$.

Proofs for correctness, IND-CPA, and semi-honest statistical circuit privacy are presented in [BBDP22].

A.2 Alternative Evaluation

We now show how to implement the alternative evaluation algorithms that we need in Section 4. These alternative evaluation algorithms are presented in [ADD+22]. We refer the reader for the proofs of correctness.

Construction 9 (Alternative evaluation algorithms)**.** *Let* $\mathsf{LHE} = (\mathsf{KeyGen}, \mathsf{Enc})$ *be the algorithms described in Construction 8.*

$\mathsf{AltEval}_1(\mathsf{pk}, (\mathbf{a}, \mathbf{b}), \mathsf{CT})$:

- *Parse* pk *as* $(\mathbb{G}, p, g, h_1, \ldots, h_k)$, $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{Z}_p^m$, $\mathbf{b} \in \mathbb{Z}_p^k$ *and* $\mathsf{CT} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_m)$ *where* $\mathsf{ct}_i = (g_i, d_{i,1}, \ldots, d_{i,k}) = (g_i, \mathbf{d}_i)$.
- *Sample* $r \leftarrow\!\!\$\, \mathbb{Z}_p$.
- *Compute* $c = g^r \prod_{i=1}^m g_i^{a_i}$ *and* $\mathbf{d} = \mathbf{h}\left(\bigodot_{i=1}^m \mathbf{d}_i^{a_i}\right) \cdot g^{\mathbf{b}}$.
- *Output* $\mathsf{ct} = (c, \mathbf{d})$.

$\mathsf{AltEval}_2(\mathsf{pk}, \mathbf{R} \in \mathbb{Z}_p^{m \times n}, \mathsf{ct})$:

- *Parse* pk *as* $(\mathbb{G}, p, g, h_1, \ldots, h_k)$, $\mathbf{R} = (r_{i,j}) \in \mathbb{Z}_p^{m \times k}$ *and* ct *as* $(f, d_1, \ldots, d_k)$.
- *Compute the new public key* $\mathsf{pk}^{\mathsf{R}} = (h'_1, \ldots, h_k)$.
- *For all* $j \in [m]$ *compute* $e_j = h_j'^{t_j} \cdot \prod_{i=0}^k d_i^{r_{i,j}}$ *where* t_i *is a random element. Set* $\mathbf{e} = (e_1, \ldots, e_m)$.
- *Output* $\mathsf{ct} = (f, \mathbf{e})$.

$\mathsf{AltDec}(\mathsf{sk}, \mathbf{R}, \tilde{\mathsf{ct}})$:

- *Parse* $\mathsf{sk} = (x_1, \ldots, x_k)$, $\mathbf{R} = (r_{i,j}) \in \mathbb{Z}_p^{m \times k}$ *and* $\mathsf{ct} = (c, \mathbf{e})$.
- *Compute* $\mathbf{s}' = \mathbf{R}\mathbf{s}^T$ *and* $\mathbf{m} = \mathsf{dlog}_g(\mathbf{e}/c^{\mathbf{s}'})$.
- *Output* $\mathbf{m}$.

The scheme fulfills a form of malicious statistical circuit privacy that we now describe. There exists an extractor and a simulator such that for all messages $\mathbf{M}$, all functions $f(\mathbf{X}) = \mathbf{R}(\mathbf{X}\mathbf{a}^T + \mathbf{b}^T)$ where $\mathbf{R} \leftarrow\!\!\$\, \mathbb{Z}_p^{m \times n}$ is a uniform matrix where $m > n$, all public keys pk and all ciphertexts CT we have that

$$\left\{(\mathbf{R}, \tilde{\mathsf{ct}})\right\} \approx_s \left\{(\mathbf{R}, \hat{\mathsf{ct}}\right\}$$

where $\tilde{ct}$ is obtained by applying $\mathsf{AltEval}_1$ and $\mathsf{AltEval}_2$ (as in Definition 22) to CT and $\hat{\mathsf{ct}} \leftarrow \mathsf{LHE.Sim}(\mathsf{pk}, \mathsf{aux}, f(\mathbf{M}))$ where $\mathbf{M} \leftarrow \mathsf{LHE.Ext}(\mathsf{pk}, \mathsf{CT})$.

Lemma 22. *The scheme presented above is statistically malicious circuit private.*

Proof. We first describe the extractor for the LHE. The extractor $\mathsf{LHE.Ext}(\mathsf{pk}, \mathsf{ct})$ first extracts the secret key from pk. That is, it finds x_i such that $g^{x_i} = h_i$. Let $\mathbf{x}$ be the secret key.

The simulator now samples a random $\mathbf{R}$, computes the new secret key $\mathbf{R}\mathbf{x}^T$. Given $f(\mathbf{M})$ the simulator computes a fresh encryption of $f(\mathbf{M})$ under the new public key. □

B Statistical Sender Secure Co-PIR from PIR

Here we present an alternative construction for SSP co-PIR from PIR. First, we show how to construction a SSP one-query co-PIR from PIR. Then, we can recurse it using the transformation from Construction 4 to obtain a co-PIR. Statistical sender security of the final construction follows from the security of the underlying PIR.

B.1 One-Query Co-PIR from PIR.

A one-query co-PIR can also be built from a PIR scheme. The resulting one-query co-PIR inherits the security of the underlying PIR. Thus, using the scheme of [ADD+22], which is a PIR that is statistical sender secure, we obtain a statistical sender secure one-query co-PIR.

Construction 10. *Let* $\mathsf{PIR} = (\mathsf{Query}, \mathsf{Send}, \mathsf{Retrieve})$ *be a (download rate-1) PIR with statistical security for the sender.*

$\mathsf{Query}(1^\lambda, i^* \in [m])$**:**

- *Compute* $(\mathsf{q}, \mathsf{st}_1) \leftarrow \mathsf{PIR.Query}(1^\lambda, m, i^*)$.
- *Output* $\mathsf{copir}_1 = \mathsf{q}$ *and* $\mathsf{st} = \mathsf{st}_1$.

$\mathsf{Send}(\mathsf{copir}_1, \mathbf{D} \in (\{0,1\}^m)$**:**

- *Parse* $\mathsf{copir}_1 = \mathsf{q}$. *Additionally, parse* $\mathbf{D} = (D_1, \ldots, D_m)$.
- *For all* $i \in [m]$ *set*
$$\mathbf{C}_i = (D_1, \ldots, D_{i-1}, 0, D_{i+1}, \ldots, D_m).$$
That is, $\mathbf{C}_i$ *is equal to* $\mathbf{D}$ *except that the* i*-th coordinate is set to* 0.
- *Compute* $\mathsf{r} \leftarrow \mathsf{PIR.Send}(\mathsf{q}, (\mathbf{C}_1, \ldots, \mathbf{C}_m))$.
- *Output* $\mathsf{copir}_2 = \mathsf{r}$.

$\mathsf{Rec}(\mathsf{copir}_2, \mathsf{st})$**:**

- *Parse* $\mathsf{copir}_2 = \mathsf{r}$ *and* st *as* st_1.
- *Compute* $\tilde{\mathbf{C}} \leftarrow \mathsf{PIR.Retrieve}(\mathsf{r}_j, \mathsf{st}_1)$.
- *Output* $\tilde{\mathbf{D}} = \tilde{\mathbf{C}}$.

Lemma 23 (Correctness)**.** *The scheme presented in Construction 10 is correct given that the underlying PIR is correct.*

Proof. By the correctness of the underlying PIR scheme we have that

$$\tilde{\mathbf{C}} = (D_1, \ldots, D_{i^*-1}, 0, D_{i^*+1}, \ldots, D_m).$$

Hence $\tilde{\mathbf{D}}_i = \mathbf{D}_i$ for all $i \neq i^*$. □

Lemma 24 (Receiver security)**.** *The scheme presented in Construction 10 is receiver secure given that* PIR *is receiver secure.*

Proof. The receiver's message contains only a PIR message. Thus, if we have an adversary that breaks the receiver security of the scheme then we can build an adversary that breaks the receiver security of the underlying PIR scheme. □

Lemma 25 (Statistical sender security)**.** *The scheme presented in Construction 10 is statistically sender secure given that the underlying PIR scheme is statistically sender secure.*

Using the extractor of the underlying PIR we can extract i^* from a maliciously chosen q. Then, we can replace all $\mathbf{C}_i$ by $\mathbf{0}$ for all $i \neq i^*$. This change goes unnoticed by the statistical security of the underlying PIR scheme. Finally, we can replace all ocurrences of 0 by $\mathbf{C}'_i$ where $\mathbf{C}'_i$ is equal to $\mathbf{D}'$ except for the i-th position which is equal to 0 and where $\mathbf{D}'$ is equal to $\mathbf{D}$ everywhere except for the i^* position. This last hybrid corresponds to $\mathsf{Send}(\mathsf{copir}, \mathbf{D}')$.

Communication complexity. We now analyze the communication complexity of the scheme. Suppose that PIR has a first message of size $|\mathsf{q}| = B \cdot \mathsf{poly}(\lambda)$ and a second message of size $|\mathsf{r}| = B + \mathsf{poly}(\lambda) \cdot \mathsf{polylog}(m)$ and that allows the receiver to retrieve blocks of size B. By reusing the receiver's message, we can make $|\mathsf{r}| = B\mathsf{poly}(\lambda) + \mathsf{poly}(\lambda) \cdot \mathsf{polylog}(m)$. Recall that th $\mathsf{ADD+}$ scheme fulfills these requirements.

- **Size of $|\mathsf{copir}_1|$.** The bitsize of the message $\mathsf{copir}_1 = \mathsf{q}$ is bounded by $\mathsf{poly}(\lambda)\mathsf{polylog}(m)$.
- **Download rate.** The bitsize of the sender's message $\mathsf{copir}_2 = \mathsf{r}$ can be bounded by $|m| + \mathsf{poly}(\lambda) \cdot \mathsf{polylog}(m)$. Hence, it achieves download rate-1.

Computational complexity. We now analyze the computational complexity of the scheme in terms of the size of $\mathbf{D}$.

- **Receiver.** The Query algorithm is independent of m. In the Rec algorithm, one needs to perform a decryption for each element in $[m]$. Hence, the receiver's computational complexity can be bounded by $|\mathbf{D}| \cdot \mathsf{poly}(\lambda)$.
- **Sender.** The Send algorithm needs to a database of size m for each entry of the underlying PIR (which itself has m entries). Hence, the sender's computational complexity can be bounded by $|\mathbf{D}|^2 \cdot \mathsf{poly}(\lambda)$.

PIR-compatibility. The message copir_1 is composed by a first message q of a PIR scheme. Hence, we can also use it as a first message of PIR scheme. This means that the scheme from Construction 10 is PIR-compatible.

Self-reducibility. If we instantiate the PIR scheme with the scheme of [ADD+22] then we have self-reducibility as the sender's message in [ADD+22] is a ciphertext where each component encrypts a single of of the database. This follows from the following property of [ADD+22]: the output r of $\mathsf{ADD+}.\mathsf{Send}(\mathsf{q}, \mathbf{D})$ can be decomposed into $\mathsf{r} = (\mathsf{head}, \alpha_1, \ldots, \alpha_m)$ where each α_i encodes a bit of $\mathbf{D} = (D_1, \ldots, D_m)$ and $|\mathsf{head}| = \mathsf{polylog}(|\mathbf{D}|) \cdot \mathsf{poly}(\lambda)$.

B.2 Bootstrapping into a SSP Co-PIR.

First, note that the construction described above already yields a bit co-PIR. We can use the transformation of Section 8 to obtain a multiple query co-PIR. The scheme described above already provides (malicious) statistical sender security if the underlying PIR has statistical sender security. Hence, we do not need to apply the transformation of Section 9.

C Yet Another Co-PIR Construction

In this section we sketch an alternative construction for semi-honest co-PIR which can then be bootstrapped into a full-fledge co-PIR using the transformations of Sections 8, 9 and 7.

The receiver message is constructed as the first message of Construction 2. The sender has as input a database $\mathbf{D} = (\mathbf{D}_1, \ldots, \mathbf{D}_m)$ where each $\mathbf{D}_i \in \mathbb{Z}_q^k$. It then computes the function $(i - i^*).\mathbf{D}_i$ over $\mathbb{Z}_q$ and compresses the resulting evaluated ciphertext into a value over $\mathbb{Z}_q$. The receiver decrypts the ciphertext.

It can be easily shown that this scheme is correct and secure following a similar rational as the ones in the proofs of Section 6. Moreover, the communication complexity analysis can be done in a similar way.

However, in terms of communication, this scheme achieves complexity scaling with m^2. To see this first notice that $q > m$ otherwise the equation $i - i^* = 0 \mod p$ may have multiple solutions for $i \in [m]$ and we want it to have a single solution. Additionally, the computational complexity of the compression mechanism of the underlying LHE scales with q as it needs to compute all q *breakpoints* before making a decision on which to sent/decrypt. This happens for both the sender and the receiver.