From OT to OLE with Almost-Linear Communication

Geoffroy Couteau
Université Paris Cité, CNRS, IRIF
couteau@irif.fr

Naman Kumar
Université Paris Cité, CNRS, IRIF
naman.kumar@irif.fr

Abstract

Following the recent work of (Doerner et al., CCS 2025), we introduce an improved reduction of OLE to OT. We prove the following: there is a perfectly-secure OLE-to-OT reduction over any n-bit field $\mathbb{F}$ using almost-linear communication $n \cdot 2^{O(\log^* n)}$ and almost-constant rounds $O((\log^* n)^2)$. In the course of proving our result, we also introduce new perfectly-secure protocols for computing shares of the equality and greater-than predicate on n-bit strings in the OT-hybrid model using $O(n)$ communication and $\log^* n$ rounds. These results are of independent interest.

Contents

1 Introduction **1**
- 1.1 Our contribution . . . 2

2 Preliminaries **3**
- 2.1 OT and OLE . . . 3
- 2.2 COT_n from ROT_n . . . 5
- 2.3 Secure multiplication . . . 5

3 Technical Overview **6**
- 3.1 Improved Communication Complexity via CRT . . . 7
- 3.2 Going Beyond $O(\ell \log \ell)$ Through Recursion . . . 7

4 OLE with Almost-Constant Overhead **9**

5 Constant-Overhead Secure Comparison in $\log^\star$ Rounds **13**
- 5.1 Equality test with almost-linear communication in $\log^*$ rounds . . . 13
- 5.2 Equality test with linear communication in $\log^*$ rounds . . . 18
- 5.3 Secure comparison with linear communication in $\log^*$ rounds . . . 20

References **24**

1 Introduction

Oblivious transfers (OT), introduced in the seminal works of Rabin [Rab05] and Even, Goldreich, and Lampel [EGL82], enable one party, the receiver, to securely learn one out of two messages held by a sender. Oblivious transfer is the cornerstone of secure computation, as any secure computation task can be realized given access to an OT protocol [Kil88]. Given its fundamental importance, a vast research effort has been dedicated to the design of protocols that generate a large number of oblivious transfers, starting with the seminal works of [Bea96, IKNP03] on OT extensions, and several works on generating OT with low communication [KK13, Roy22, CDD+24], including the recent lines of work on *silent* OT extension [BCGI18, BCG+19b, BCG+19a] that include silent OT extension given only a public key setup [OSY21, BCM+24]. Due to this widespread research effort, modern protocols can now generate millions of OTs on commodity hardware using only a few bits of communication per OT.

Yet securely computing a function using only oblivious transfers comes with a downside: in all known protocols, the communication cost of (computationally-efficient) OT-based secure computation scales linearly with the size of the Boolean circuit representing the function [GMW87]. As Boolean circuits are a low-level representation, this often represents a major overhead. As a consequence, many modern protocols sidestep this limitation by moving away from making black-box use of oblivious transfer, and leveraging instead other forms of setup or cryptographic assumptions.

1.0.1 Beyond the Boolean Circuit Size Barrier

One approach, initiated by Gentry [Gen09], is to design protocols whose communication scales only sublinearly with the size of the Boolean circuit. Following Gentry's breakthrough, the existence of secure computation protocols with sublinear communication has been shown from a wide variety of cryptographic assumptions [BGI16, BCG+19b, CM21, OSY21, RS21, ADOS22, BCM22, BCM23, DIJL23, CK24, ARS24, CKY25], that includes most traditional public key assumptions. Another approach escapes the limitations of Boolean circuits by designing protocols that directly handle other computational models, such as arithmetic circuits [DPSZ12], RAM [Gol87], or specific functionalities [KO97, FNP04]. Perhaps the most influential result in this vein is the work of [DPSZ12] that showed how to leverage somewhat homomorphic encryption and zero-knowledge proofs to generate *oblivious linear evaluation* over large fields, enabling much faster protocols for evaluating arithmetic circuits over large fields.

1.0.2 Oblivious linear evaluation

(OLE) is a natural extension of OT over large fields $\mathbb{F}$ (or rings), in which the sender inputs an affine function $(a, b) \in \mathbb{F}^2$ and the receiver learns the evaluation $a \cdot x + b$ at a point of its choice. OLEs enable secure evaluation of field multiplications over shared value in a similar fashion as OT over $\mathbb{F}_2$. This makes generating OLEs a natural target to securely evaluate functions that involve ring arithmetic, and OLE over large fields are at the heart of many modern protocols, including, for example, threshold signatures [Lin17, DKLs24, ABC+24]. Unfortunately, protocols for generating OLEs typically lag behind those for generating OTs: while one can generate thousands of OLEs per seconds using lattice-based [DPSZ12, KPR18] or code-based techniques [BCG+20b, BCCD23, LXYY25], this is several orders of magnitude slower than the rate at which modern protocols can produce OTs. In fact, OT extension protocols are so efficient that despite the fact that OLEs ought to be

a *replacement* for OTs better suited for large fields, the best way to implement an OLE protocol is often by reduction to OT. Standard examples include the general protocol MASCOT [KOS16] or the state-of-the-art threshold ECDSA signature from [DKLs24], that employs the seminal OT-to-OLE protocol of Gilboa [Gil99]. Building upon this observation, the recent work of [DHIM25] initiated the study of the efficiency of OT-based OLE constructions, and showed how OLEs over an n-bit field can be efficiently implemented from OTs using $O(n \cdot \log n)$ communication.

1.1 Our contribution

In this work, we continue the line of work initiated in [DHIM25]. As advocated in [DHIM25], we position our study in a setting where the participants are given access to a source of *random* oblivious transfers (ROTs), where the sender receives two random strings (s_0, s_1) and the receiver receives (b, s_b) for a random bit b. A good motivation for this model is that recent works on silent OT extension [BCGI18, BCG+19b, BCG+19a] allow the participants, after a one-time interaction, to locally generate an arbitrary amount of (pseudo)random oblivious transfers without any communication. In fact, stronger primitives such as pseudorandom correlation functions (PCFs) [BCG+20a, BCG+22, BCM+25] let the parties, after an initial setup, generate on-demand an unbounded supply of ROTs with zero communication, and *public key* PCFs [OSY21, BCM+24] even enable replacing this one-time setup by a standard PKI.

Our original motivation stems from the following observation: to evaluate an arithmetic circuit over an n-bit field using OTs, the naive approach is to construct a Boolean circuit that computes the same function. However, even using the recent breakthrough result of [HVDH21] to compute n-bit multiplication using $O(n \log n)$ Boolean operations, this still results in an $O(\log n)$ overhead (with galactic constants) over the cost of evaluating the arithmetic circuit given random OLEs. The recent work of [DHIM25] shows that one can do better than this naive approach: they construct OLEs over n-bit fields (recall that an OLE can afterwards be used to perform secure multiplications over the field) from OTs using $O(n \cdot \log n)$ bits of communication without going through the Boolean circuit for multiplication. As a result, they get a far better result, and a concretely efficient protocol, but still only improve over the size of the best known Boolean circuit for the task by a constant. In this work, we ask:

Is it possible to design an OT-based OLE protocol with communication sublinear in the size of the best-known Boolean circuit for multiplication?

A positive answer would have a surprising consequence: it would show that there exists a large class of functions, namely the class of all (polysize) arithmetic circuits over superconstant fields or rings, which can be securely computed using solely oblivious transfers with a communication *sublinear* in the size of the best-known Boolean circuit for computing these functions.[1] Perhaps surprisingly we provide a positive answer to this question, showing that OLEs can be obtained from ROTs using *almost* linear communication:

Theorem 1. *Let $\mathbb{F}$ be a field of size $|\mathbb{F}| \approx 2^n$. There exists a perfectly secure protocol in the* ROT*-hybrid model that securely instantiates the* OLE *functionality over $\mathbb{F}$ with semi-honest security, using:*

[1] Of course, in this class, many functions can admit small Boolean circuits. But there will be a vast majority of circuits for which the best known conversion to Boolean circuit replaces each arithmetic gate by the best-known Boolean circuit for this gate.

- *Almost-linear communication* $O(n \cdot 2^{\log^* n})$
- *Almost-constant rounds* $O((\log^* n)^2)$.

We recall that $\log^* n$ denotes the iterated logarithm, *i.e.*, the number of times log must be applied to n before it becomes smaller than 1. This is an extremely slow-growing function: $\log^* n = 5$ happens only for integers n that are longer than 65536 bits, and $\log^* n > 5$ happens only for galactic integers n, longer than 2^{65536} bits. We note that using known results from the literature [IPS08, DGN+17], our result extends to perfectly secure protocols with malicious security in the ROT-hybrid model with the same communication and round complexity (up to a constant). Our result represents the first instance of beating the "circuit size barrier" using oblivious transfer for a relatively large and important class of functions.

In the course of proving our main result, we obtain several other results of independent interest: we construct perfectly-secure protocols in the ROT-hybrid model for testing equality between strings, and secure comparison (with shared outputs in both cases).

Theorem 2. *Let* ET_n *and* SC_n *denote the functionalities that, on inputs* a, b *from two parties, return additive shares of a bit equal to* 1 *iff* $a = b$ *(for* ET_n*) or* $a > b$ *(for* SC_n*). There exists a perfectly secure protocol in the* ROT*-hybrid model that securely instantiates* ET_n *or* SC_n *with semi-honest security, using:*

- *Linear communication* $O(n)$
- *Almost-constant rounds* $O(\log^* n)$.

Previously, linear-communication protocols in the ROT-hybrid model for ET and SC were known, but only with $O(\log n)$ rounds [GSV07]. Alternatively, [Cou18] gives a protocol for ET with $O(n \log n)$ communication and $O(\log^* n)$ rounds, and a protocols for SC with $O(n \log n)$ communication and $O(\log \log n)$ rounds. To our knowledge, our protocols are the first to achieve strictly linear communication in an almost-constant number of rounds.

2 Preliminaries

Notation. We write $a \leftarrow\!\$\ S$ for sampling uniformly from a set. Given a bit $a \in \{0, 1\}$, we write $\bar{a}$ for $1 - a$. We let $\mathsf{EQ}(a, b)$ denote the equality predicate: $\mathsf{EQ}(a, b) = 1$ iff $a = b$. We let $\mathsf{GT}(a, b)$ denote the greater-than predicate: $\mathsf{GT}(a, b) = 1$ iff $a > b$. We let $\oplus$ denote the bitwise-XOR.

2.1 OT and OLE

We start by introducing the ROT_n and COT_n functionalities, in particular their corruptible variants, which allows corrupted parties to choose their outputs.

Functionality ROT_n

Parties. A sender S and a receiver R

Parameters. length parameter n

Description.

- If both parties are honest, sample $(s_0, s_1) \leftarrow\$ (\{0,1\}^n)^2$ and $b \leftarrow\$ \{0,1\}$. Return (s_0, s_1) to S and (b, s_b) to R.
- If S is corrupted, receive $(s_0, s_1) \in \{0,1\}^n \times \{0,1\}^n$ from the adversary $\mathcal{A}$. Sample $b \leftarrow\$ \{0,1\}$ and return (b, s_b) to R.
- If R is corrupted, receive $(b, s_b) \in \{0,1\} \times \{0,1\}^n$ from the adversary $\mathcal{A}$. Sample $s_{1-b} \leftarrow\$ \{0,1\}^n$ and return (s_0, s_1) to S.

Functionality 1: Random oblivious transfer of n-bit strings

Functionality COT_n

Parties. A sender S and a receiver R

Parameters. Sender input length n.

Description. Upon receiving $\delta \in \{0,1\}^n$ from S, and $b \in \{0,1\}$ from R,

- If both parties are honest, sample $s_0 \leftarrow\$ \{0,1\}^n$. Return s_0 to S and $s_0 \oplus \delta \cdot b$ to R.
- If S is corrupted, receive $s_0 \in \{0,1\}^n$ from the adversary $\mathcal{A}$. Return $s_0 \oplus \delta \cdot b$ to R.
- If R is corrupted, receive $s_b \in \{0,1\}^n$ from the adversary $\mathcal{A}$. Set $s_{1-b} \leftarrow s_b \oplus \delta \cdot b$ and return s_0 to S.

Functionality 2: Correlated oblivious transfer of n-bit strings

Remark 1. We can also similarly define $\mathsf{ROT}_{\mathbb{Z}_n}$ and $\mathsf{COT}_{\mathbb{Z}_n}$ to be the random and correlated OT functionalities over the group/ring $\mathbb{Z}_n$. In this setting, the parameters are a ring $\mathbb{Z}_n$ of size n and S's input to $\mathsf{COT}_{\mathbb{Z}_n}$ consists of a ring element $\delta \in \mathbb{Z}_n$, and the outputs are elements $(s_0, s_0 + \delta)$ separated by a δ offset, ie. arithmetic shares of δ.

We also recall the $\mathsf{OLE}_{\mathbb{Z}_n}$ functionality as below.

Functionality $\mathsf{OLE}_{\mathbb{Z}_n}$

Parties. A sender S and a receiver R

Parameters. Integer ring $n \in \mathbb{Z}$.

Description. On input $(a, b) \in \mathbb{Z}_n^2$ from S and $x \in \mathbb{Z}_n$ from R, output $ax + b \pmod{n}$ to receiver.

Functionality 3: Oblivious Linear Evaluation of n-bit strings

2.2 COT_n from ROT_n

We now recall how COT_n can be implemented using black-box access to ROT_n using $\log n+1$ bits of communication. As stated, our protocols are in the COT_n-hybrid model and our analysis proceeds in the ROT_n-hybrid model, which allows us to assume the existence of efficient pseudorandom OTs via preprocessing. The following treatment has been taken from [DHIM25].

Protocol $\Pi_{\mathsf{COT}}(n)$

Parties. A sender S and a receiver R.

Parameters. $n \in \mathbb{N}$.

Oracles. ROT_n.

Inputs. S has input $\delta \in \{0,1\}_n$, and R has input $i \in \{0,1\}$.

Operations.
1. The parties jointly call ROT_n. Let (a_0, a_1) denote S's output and (i', z) denote R's output.
2. R sends $f \leftarrow i \oplus i'$ to S.
3. If $f = 1$, S swaps (a_0, a_1). Then it sends $b \leftarrow a_1 - a_0 - \delta$.

Outputs. S outputs a_0, R outputs z if $i = 0$ and b otherwise.

Protocol 1: COT_n from ROT_n.

The following theorem is taken from [DHIM25].

Theorem 3. *Protocol 1 perfectly UC-realizes* COT_n *in the* ROT_n*-hybrid model.*

Remark 2. The same protocol extends trivially to arbitrary groups, in particular $\mathbb{Z}_n$, and we write $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$ for the protocol implementing $\mathsf{COT}_{\mathbb{Z}_n}$ in the $\mathsf{ROT}_{\mathbb{Z}_n}$-hybrid model with one $\mathbb{Z}_n$-element of communication.

2.3 Secure multiplication

We recall another standard functionality from the literature.

Functionality SM

Parties. A sender S and a receiver R

Description. Upon receiving bits $a^{\mathsf{S}}, b^{\mathsf{S}}$ from S and $a^{\mathsf{R}}, b^{\mathsf{R}}$ from R,

- Compute $c = (a^{\mathsf{S}} \oplus a^{\mathsf{R}}) \cdot (b^{\mathsf{S}} \oplus b^{\mathsf{R}})$.
- If both parties are honest, sample $r \leftarrow\!\!\$\ \{0,1\}$. Return $c \oplus r$ to S and r to R.
- If S is corrupted, receive $s \in \{0,1\}$ from the adversary $\mathcal{A}$. Return $c \oplus s$ to R.
- If R is corrupted, receive $r \in \{0,1\}$ from the adversary $\mathcal{A}$.Return $c \oplus r$ to S.

Functionality 4: Secure bit multiplication

It is well-known that SM can be realized using two parallel calls to COT_1, and we omit the straightforward proof. In particular, this yields a protocol for SM in the ROT_1-hybrid model using 2 bits of communication in two rounds.

Protocol Π_{SM}

Parties. A sender S and a receiver R.

Oracles. COT_1.

Inputs. Bits $a^{\mathsf{S}}, b^{\mathsf{S}}$ from S and $a^{\mathsf{R}}, b^{\mathsf{R}}$ from R

Operations.
1. S and R call COT_1 on inputs $a^{\mathsf{S}}, b^{\mathsf{R}}$ and receive outputs s_0, r_0
2. S and R call COT_1 on inputs $b^{\mathsf{S}}, a^{\mathsf{R}}$ and receive outputs s_1, r_1

Outputs. S outputs $s_0 \oplus s_1 \oplus a^{\mathsf{S}} \cdot b^{\mathsf{S}}$, R outputs $r_0 \oplus r_1 \oplus a^{\mathsf{R}} \cdot b^{\mathsf{R}}$.

Protocol 2: SM from COT_1.

3 Technical Overview

Let us start by recalling the definition of OT and OLE for any modulus $n \in \mathbb{N}$. Both of these functionalities are 2-party functionalities between a sender S and a receiver R.

1. $\mathsf{OT}_{\mathbb{Z}_n}$: Takes as input two strings $(a_0, a_1) \in \mathbb{Z}_n^2$ from the sender and a bit $b \in \{0,1\}$ from the receiver, and sends the output a_b to the receiver.
2. $\mathsf{OLE}_{\mathbb{Z}_n}$: Takes as input two strings $(a, b) \in \mathbb{Z}_n^2$ from the sender and a string $x \in \mathbb{Z}_n$ from the receiver, and sends the output $ax + b \mod n$ to the receiver.

The starting point of our work is a simple protocol of Gilboa that constructs $\mathsf{OLE}_{\mathbb{Z}_n}$ given access to an $\mathsf{OT}_{\mathbb{Z}_n}$-hybrid. We review Gilboa's protocol below. While Gilboa's protocol provides full security against a malicious receiver, in this paper we are interested in only the semi-honest setting.

Protocol Gilboa's Protocol, $\Pi_{\mathsf{GOLE}}(n)$

Parties. A sender S and receiver R.
Parameters. $n \in \mathbb{N}$. Set $\ell := \lceil \log n \rceil$.
Oracles. $\mathsf{OT}_{\mathbb{Z}_n}$.
Inputs. S holds $(a, b) \in \mathbb{Z}_n^2$ and receiver holds $x \in \mathbb{Z}_n$.
Description.

1. R decomposes its input $x = x_0 \| x_1 \| \cdots \| x_{\ell-1}$, where

$$x = \sum_{i=0}^{\ell-1} 2^i x_i.$$

2. S samples $\{b_i\}_{i=0}^{\ell-1}$ to be an arithmetic secret sharing of b over $\mathbb{Z}_n$.

3. The parties call $\ell - 1$ instances of $\mathsf{OT}_{\mathbb{Z}_n}$ in parallel. To the ith OT, S uses as input $(b_i, b_i + 2^i a)$, while R uses as input x_i. Let $\{z_i\}_{i=0}^{\ell-1}$ be the output received by R.

Output. R outputs $y = \sum_{i=0}^{\ell-1} z_i$. S outputs nothing.

It is easy to see that Gilboa's protocol makes use of ℓ calls to $\mathsf{OT}_{\mathbb{Z}_n}$, for a total communication of $O(\ell^2)$ bits (provided that each $\mathsf{OT}_{\mathbb{Z}_n}$ has a communication of $O(\ell)$ bits).

3.1 Improved Communication Complexity via CRT

We now recall the work of [DHIM25], which makes the observation that if n is a smooth modulus then invoking Gilboa's protocol over the CRT rings can result in improved communication.

Suppose that we want to perform $\mathsf{OLE}_{\mathbb{Z}_q}$ for some prime q. Set n to be a smooth integer such that $n > q^2 2^\kappa$ where κ is a statistical security parameter. Further, let n factor into distinct primes $\{n_i\}_{i \in [\ell \log \ell]}$ (we know from the prime number theorem that if n, is, say, an $O(\ell)$ bit integer that is the product of the first k primes, then $k \approx \ell / \log \ell$). Suppose we now decompose n via CRT into the n_i (which are of size at most $\log \ell$) and carry out Gilboa's protocol over $\mathbb{Z}_{n_i}$. It follows that the total communication now is

$$\frac{\ell}{\log \ell} \times (\log \ell)^2 = O(\ell \log \ell).$$

This is a near-quadratic improvement. However, note that we have only carried out $\mathsf{OLE}_{\mathbb{Z}_n}$, whereas we originally wanted to carry out $\mathsf{OLE}_{\mathbb{Z}_q}$. This can be fixed using the following modification.

1. S chooses (a, b') as input, where $b' \leftarrow [n]$ such that $b' \equiv b \pmod q$, and b' has been selected from $[0, 2^{\kappa-1} \cdot q^2)$. Note that $ax \leq q^2$, and hence this ensures that b' is not in the final zone $[2^{\kappa-1} \cdot q^2, 2^\kappa \cdot q^2)$, which means that $ax + b'$ does not 'wraparound' n, ensuring that the OLE can be taken over the integers.

2. Once R has obtained y over $\mathsf{OLE}_{\mathbb{Z}_n}$, it can output $y \pmod q$ as the answer.

It is easy to wonder why we need the parameter κ at all: can the same process not be repeated with (a, b) as input and simply choosing $n > q^2 + q$, which ensures that $ax + b < n$? The problem occurs because we wish to perform $\mathsf{OLE}_{\mathbb{Z}_q}$ and not $\mathsf{OLE}_{\mathbb{Z}_n}$. In particular, while reconstructing, the receiver learns $ax + b$ over the integers: which is a lot more than $ax + b \pmod q$!

3.2 Going Beyond $O(\ell \log \ell)$ Through Recursion

Ideally, this process can be recursed to obtain further communication gains. However, we quickly hit an obstacle: for performing $\mathsf{OLE}_{\mathbb{Z}_q}$, we need to choose a smooth modulus n such that $n > 2^\kappa q^2$.

3.2.1 Naive Recursion

What we want to do is recursively apply CRT, so that we can break down our primes of size $\log \ell$ into primes of size $\log(\log \ell)$, for instance, and then recurse. This gives us the following recurrence relation:

$$T(\ell) = \frac{\ell}{\log \ell} T(\log \ell)$$

In general, we repeat this $\log^* \ell$ times, until the cost of share conversion becomes constant.

The problem is that suppose that we are now performing OLE over successively smaller fields; recall that to get OLE over n, we performed OLE over $n_1, n_2, \ldots$ and then applied CRT. If we want to apply CRT on the n_i, then note that the n_i are primes, and hence we need to pick a suitable smooth integer m_i, let's say, which has the property that $m_i > n_i^2 \cdot 2^\kappa$. But this is a problem! While choosing m_i, we were making an additional cost of κ bits. Now, we are making an additional cost of κ bits *per OLE*, which leads to a total of $\ell / \log \ell * \kappa$ additional bits. This means that any improvements we get while recursing *should* lead to a protocol of communication

$$\frac{\ell}{\log \ell} \times \frac{\log \ell}{\log \log \ell} * (\log \log \ell)^2 = \ell \log \log \ell$$

but doesn't, because of the κ factor. Instead, we get a protocol of communication

$$\frac{\ell}{\log \ell} \times \frac{\kappa \log \ell}{\log \log \ell} * (\log \log \ell)^2 = \ell \kappa \log \log \ell$$

which is a κ factor scale up. However, if we wish to achieve negligible correctness error, then $\kappa = O(\lambda) = O(\ell)$, which destroys any gains that we have gotten.

3.2.2 Fixing via Modular Reduction

To bypass this obstacle, we first make the following modification to the protocol: instead of using b as its input, the sender picks a uniform $r \leftarrow \mathbb{Z}_n$ and uses (a, r) as its input to $\mathsf{OLE}_{\mathbb{Z}_n}$. This ensures that after a single invocation of $\mathsf{OLE}_{\mathbb{Z}_n}$, the parties have shares of $ax + b \pmod n$: S sets $b - r$ to be its share, and R sets its output $ax + r$ to be its share. Note that since r is selected uniformly at random, $ax + r \pmod n$ reveals no information about (a, b), and hence can be safely held by R.

Note now that if the shares are interpreted over $\mathbb{Z}$, then the parties could either be holding shares of $ax + b$ or $ax + b + n$. Suppose that the parties knew that they were holding shares of $ax + b$ over the integers; then they could simply reduce their individual shares mod q and obtain shares of $ax + b$ over $\mathbb{Z}_q$. Similarly, if they knew that they were holding shares of $ax + b + n$ over $\mathbb{Z}$, then S could modify its own share to be $b - r - n$, and peform the same reduction mod q.

We describe a step-by-step procedure to ensure that the parties can perform modular reduction:

1. The parties perform a single secure comparison for ℓ-bit strings that checks whether $\langle y \rangle_n^{\mathsf{S}} + \langle y \rangle_n^{\mathsf{R}} > n$, receiving the output as $\langle s \rangle$ (which is a share of 0 if no and 1 otherwise).

2. S then sets

   $$(\langle y \rangle_n^{\mathsf{S}} - n\langle s \rangle^{\mathsf{S}} \mod q, \langle y \rangle_n^{\mathsf{S}} - n(1 - \langle s \rangle^{\mathsf{S}}) \mod q)$$

   as its inputs to an $\mathsf{OT}_{\lceil \log q \rceil}$, and R sets $\langle s \rangle^{\mathsf{R}}$ as its choice bit. Let y' be the output obtained by R.

3. R outputs $y' + \langle y \rangle_n^{\mathsf{R}} \mod q$.

Thus, assuming access to a secure comparison protocol for ℓ-bit strings, the parties can perform modular reduction of their shares from $\mathbb{Z}_n$ to $\mathbb{Z}_q$.

3.2.3 Putting it Together

The previous discussion motivates the following protocol for $\mathsf{OLE}_{\mathbb{Z}_q}$ in the OT-hybrid model:

1. The parties CRT find the smallest primorial $n > q$ and decompose their inputs into elements of smaller prime rings.
2. The parties run Gilboa's protocol over the small primes and recurse over the larger primes, obtaining shares of $\mathsf{OLE}_{\mathbb{Z}_n}$.
3. The parties then perform a secure comparison and one OT to reduce their shares to $\mathsf{OLE}_{\mathbb{Z}_q}$.

While this protocol is essentially correct, any efficiency gains depend strongly on the efficiency of the secure comparison. We dedicate Section 5 to introducing new, highly efficient secure comparison protocols. Our new protocols are of a similar recursive nature. We build our protocols progressively, first constructing an equality test protocol with $n \log^* n$ communication in $\log^* n$ rounds, then showing how to reduce the communication to $O(n)$, and finally building an $O(n)$-communication $\log^* n$-round secure comparison protocol on top of our equality test protocol. All our intermediate protocols are simple in hindsight and elegantly use a recursive approach.

4 OLE with Almost-Constant Overhead

In this section, we introduce our complete protocol for $\mathsf{OLE}_{\mathbb{Z}_n}$. Our protocol is defined in the $\{\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \le t}, \mathsf{SC}_{\lceil \log n \rceil}, \mathsf{COT}_{\mathbb{Z}_q}\}$-hybrid model, and is perfectly secure. Our protocol achieves a total communication on the order of $O(n \cdot 2^{\log^* n})$ bits and a round complexity of $O((\log^* n)^2)$ for primes q of size $O(n)$, where n is taken to be the first primorial greater than q. The full description of our protocol is provided in Protocol 3.

We now recall the following oracles, introduced in more detail in the next section:

1. $\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \le t}$ is a set of oracles that perform OLE over the smaller fields of the CRT domain.
2. SC_k is an oracle for the corruptible secure comparison functionality over k-bit strings, defined in Functionality 6.
3. $\mathsf{COT}_{\mathbb{Z}_q}$ is an oracle for the corruptible OT functionality over $\mathbb{Z}_q$, defined in Functionality 2.

Protocol $\Pi_{\mathsf{OLE}}(q)$

Parties. A sender S and a receiver R.

Parameters. Integer ring size q. Absolute constant c. Let n denote the first primorial larger than q^2, and t denote its number of factors (hence, $n = \prod_{i=1}^{t} p_i$ where $p_1, p_2, \cdots$ are the first distinct primes).

Oracles. $\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i\le t}$, $\mathsf{SC}_{\lceil \log n\rceil}$, $\mathsf{COT}_{\mathbb{Z}_q}$.

Inputs. S has inputs $(a,b)\in\mathbb{Z}_q^2$, and R has input $x\in\mathbb{Z}_q$.

Operations. Let $\{a_i\}_{i\in[t]}$, $\{b_i\}_{i\in[t]}$ and $\{x_i\}_{i\in[t]}$ be the CRT decompositions of a, b and x respectively.

1. S uniformly samples $r \leftarrow\!\$\, \mathbb{Z}_n$. Let $\{r_i\}_{i\in[t]}$ be the CRT decomposition of r.
2. For $0 < i \le c$, the parties run $\Pi_{\mathsf{GOLE}}(p_i)$ with S's input as (a_i, r_i) and the receiver's input as x_i. For each $c < i \le t$, the parties call $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$ with S's inputs being (a_i, r_i) and R's input being x_i. These can be run in parallel.
3. Let z_i be the receiver's output of $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$. R sets $z' \leftarrow \mathsf{CRT}(n, \{z_i\}_{i\in[t]})$.
4. S sets $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} \leftarrow r - b$ and R sets $\langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} \leftarrow z'$, which are interpreted as shares over the integers.
5. S and R call $\mathsf{SC}_{\lceil \log n\rceil}$ on inputs $(\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}},\, n - \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}})$ and receive output $\langle s\rangle_{\mathbb{Z}_2}$.
6. S and R call $\mathsf{COT}_{\mathbb{Z}_q}$ on $(2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q, \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2})$ and receive outputs $(r', r' + (2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q)\cdot\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2})$. S then sends $w = \langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} - n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} \pmod q - r'$ along with $b - r$ to R.

Outputs. R outputs $y \leftarrow r' + (2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q)\cdot\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} + w + b - r$, S outputs nothing.

Protocol 3: OLE over $\mathbb{Z}_q$.

Remark 3. While the above protocol is given in the $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$ hybrid model, we note that when instantiated we will implement it as follows: $0 < i \le c$, the parties run $\Pi_{\mathsf{GOLE}}(p_i)$ as stated, while for $c < i \le t$, the parties run $\Pi_{\mathsf{OLE}}(p_i)$. In both cases, S's input as (a_i, r_i) and the receiver's input as x_i.

The correctness of the protocol is easy to analyze. Note that after Step 4, the parties possess shares of $z = ax + b$ over $\mathbb{Z}$, which are $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} := b - r$ and $\langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} := ax + r$. Since $r \in \mathbb{Z}_n$, we have that $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} + \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}}$ is equal to either $ax + b$ or $ax + b + n$ depending on whether there has been a wraparound. Depending on the result of $\mathsf{SC}_{\lceil \log n\rceil}$, we have

- If $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} < n - \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}}$, then $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} + \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} < n$ and $ax + b$ is already over the integers. Then $\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}} = \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}}$ and

$$\begin{aligned} y &= r' + (2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q)\cdot\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} + w + b - r \\ &= r' + (2n\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} - n \pmod q)\cdot\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} \\ &\qquad + \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} - n\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} \pmod q - r' + b - r \pmod q \\ &= z' + b - r \pmod q \\ &= ax + r + b - r \pmod q \\ &= ax + b \pmod q \end{aligned}$$

- If $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} > n - \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}}$, then $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} + \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} > n$ and $ax + b + n$ is over the integers. Then $\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}} = 1 - \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}}$

and

$$\begin{aligned}
y &= r' + (2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q) \cdot \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} + w + b - r \\
&= r' + (2n(1 - \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2}) - n \pmod q) \cdot \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2} \\
&\qquad + \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}} - n(1 - \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2}) \pmod q - r' + b - r \pmod q \\
&= z' + b - r \pmod q \\
&= ax + r + b - r \pmod q \\
&= ax + b \pmod q
\end{aligned}$$

In both cases R receives $ax + b \pmod q$ as desired. Thus correctness follows from the correctness of $\mathsf{SC}_{\lceil \log n \rceil}$ and that of $\mathsf{COT}_{\mathbb{Z}_q}$.

Communication Complexity. We consider the communication complexity of $\Pi_{\mathsf{OLE}}(q)$ in the ROT-hybrid model when COT and SC are instantiated with $\Pi_{\mathsf{COT}}(\mathbb{Z}_q)$ and $\Pi_{\mathsf{SC}}(\lceil \log n \rceil)$ respectively, and the $\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \le t}$ are instantiated as per Remark 3. Note that $\log n = O(\log q)$.

1. Let $k(x) = t$ denote the number of prime factors of the first primorial bigger than x. By the prime number theorem, it follows that $k(x) = \Theta(|x|/|\log x|)$.
2. Let $|p_t|$ denote the length of the largest prime factor of said primorial. From the prime number thoerem, again, we have that and $|p_t| = O(|\log x|)$.

Let $x = \log q$, and let $\mathsf{C}_{\mathsf{OLE}}(x)$ denote the communcation complexity of $\Pi_{\mathsf{OLE}}(q)$ (performed with strings of length x).

Claim. $\mathsf{C}_{\mathsf{OLE}}(x) = O(x \cdot 2^{\log^*(x)})$.

Proof. We start by substituting the asymptotic complexities to obtain:

$$\begin{aligned}
\mathsf{C}_{\mathsf{OLE}}(x) &= \sum_{i=0}^{c} O((\log p_i)^2) + \sum_{i=c+1}^{k(n)} \mathsf{C}_{\mathsf{OLE}}(p_i) + \mathsf{C}_{\mathsf{SC}}(O(x)) + \mathsf{C}_{\mathsf{COT}}(\mathbb{Z}_q) + O(x) \\
&\le O\left(\frac{x}{\log x}\right) \cdot (\mathsf{C}_{\mathsf{OLE}}(\log x)) + O(x) + O(\log x) + O(x) + O(1) \\
&\le O\left(\frac{x}{\log x}\right) \cdot (\mathsf{C}_{\mathsf{OLE}}(\log x)) + O(x)
\end{aligned}$$

We can set concrete constants C_1 and C_2 to get

$$\mathsf{C}_{\mathsf{OLE}}(x) = \frac{C_1 x}{\log x} \cdot (\mathsf{C}_{\mathsf{OLE}}(\log x)) + C_2 x$$

Expanding for k steps, we have

$$\begin{aligned}
\mathsf{C}_{\mathsf{OLE}}(x) &= \frac{C_1 x}{\log x} \cdot (\mathsf{C}_{\mathsf{OLE}}(\log x)) + C_2 x \\
&= \frac{C_1 x}{\log x} \left(\frac{C_1 \log x}{\log \log x} (\dots) + C_2 \log \log x \right) + C_2 \log(x) \\
&\le \sum_{i=0}^{k-1} C_2 \mathsf{itlog}_i(x) \prod_{j=0}^{i-1} \frac{C_1 \mathsf{itlog}_j(x)}{\mathsf{itlog}_{j+1}(x)} + \mathsf{C}_{\mathsf{OLE}}(\mathsf{itlog}_k(x)) \cdot \prod_{j=0}^{k-1} \frac{C_1 \mathsf{itlog}_j(x)}{\mathsf{itlog}_{j+1}(x)}
\end{aligned}$$

Note that the product here is telescoping: for each product, we get

$$\prod_{j=0}^{i} \frac{C_1 \mathsf{itlog}_j(x)}{\mathsf{itlog}_{j+1}(x)} = \frac{C_1 x}{\log x} \times \frac{C_1 \log x}{\log \log x} \times \ldots \times \frac{C_1 \mathsf{itlog}_i(x)}{\mathsf{itlog}_{i+1}(x)} = \frac{C_1^i x}{\mathsf{itlog}_{i+1}(x)}$$

We repeat this process until $\mathsf{C}_{\mathsf{OLE}}(\cdot)$ reduces to constant, which happens after $\log^*(x)$ steps; in particular, $\mathsf{itlog}_{\log^*(x)}(x) = O(1)$. This gives us

$$\mathsf{C}_{\mathsf{OLE}}(x) \leq O\left(x \cdot \sum_{i=0}^{\log^* x} C_1^i\right) = O(x \cdot C_1^{\log^* x}),$$

which is almost-linear, and we complete the claim. ■

Round Complexity. We proceed with the round complexity in a similar manner, in the ROT-hybrid model when COT and SC are instantiated with $\Pi_{\mathsf{COT}}(\mathbb{Z}_q)$ and $\Pi_{\mathsf{SC}}(\lceil \log n \rceil)$ respectively, and the $\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \leq t}$ are instantiated as per Remark 3. Note, however, that in this setting Step 2 can be done entirely in parallel. This gives us

$$\begin{aligned}
\mathsf{R}_{\mathsf{OLE}}(x) &= \mathsf{R}_{\mathsf{OLE}}(p_i) + \mathsf{R}_{\mathsf{SC}}(O(x)) + \mathsf{R}_{\mathsf{COT}}(\mathbb{Z}_q) + O(1) \\
&\leq \mathsf{R}_{\mathsf{OLE}}(\log x) + O(\log^*(x)) + O(1) \\
&= \mathsf{R}_{\mathsf{OLE}}(\log x) + O(\log^*(x))
\end{aligned}$$

The recurrence is quite simple and unfolds after $\log^*(x)$ steps to

$$\begin{aligned}
\mathsf{R}_{\mathsf{OLE}}(x) &= \mathsf{R}_{\mathsf{OLE}}(\log x) + O(\log^*(x)) \\
&\leq \mathsf{R}_{\mathsf{OLE}}(\mathsf{itlog}_{\log^*}(x))) + \sum_{i=0}^{\log^*(x)} O(\log^*(x)) \\
&= O\left((\log^*(x))^2\right).
\end{aligned}$$

Security Analysis. We now show that $\Pi_{\mathsf{OLE}}(q)$ is perfectly secure in the $(\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \leq t}, \mathsf{SC}_{\lceil \log n \rceil}, \mathsf{COT}_{\mathbb{Z}_q})$-hybrid model. The composition theorem implies that $\Pi_{\mathsf{OLE}}(q)$ remains secure even when the functionalities are replaced by $\Pi_{\mathsf{OLE}}(p_i)$, $\Pi_{\mathsf{SC}}(\lceil \log n \rceil)$ and $\Pi_{\mathsf{COT}}(\mathbb{Z}_q)$ respectively, assuming that the protocols securely implement the functionalities in the semi-honest model. In particular, for $\Pi_{\mathsf{OLE}}(q)$, this involves revoking the following proof of security recursively until we reach p_i such that $i < c$, where c is a fixed absolute constant. Once this has been reached, the $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$ is implemented by $\Pi_{\mathsf{GOLE}}(p_i)$, and henceforth the proof below does not require any recursion.

Assume that the parties are honest. In this case, we have verified correctness in the previous section, and the outputs are correct provided that $\Pi_{\mathsf{GOLE}}(p_i)$, $\{\mathsf{OLE}_{\mathbb{Z}_{p_i}}\}_{i \leq t}$, $\mathsf{SC}_{\lceil \log n \rceil}$, and $\mathsf{COT}_{\mathbb{Z}_q}$ are correct.

Now assume that S is corrupted. We show a straight-line simulator that perfectly emulates R as follows. For all invocations of $\Pi_{\mathsf{GOLE}}(p_i)$, Sim calls the simulator for $\Pi_{\mathsf{GOLE}}(p_i)$, otherwise it emulates $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$. In this case it receives (a_i, r_i) from S and returns nothing. In step 5 it then receives $\langle z \rangle_{\mathbb{Z}}^{\mathsf{S}}$ from the sender and $\langle s \rangle_{\mathbb{Z}_2}^{\mathsf{S}}$ from the adversary, and sends $\langle s \rangle_{\mathbb{Z}_2}^{\mathsf{S}}$ to the sender. Finally

in step 6, Sim takes as input $2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q$ from S and receives r' from the adversary, and outputs r' to the sender. This completes the simulation perfectly.

We now assume that R is corrupted. We show a similar straight-line simulator that perfectly emulates S. In step 1, the simulator uniformly samples $r \leftarrow\!\!\$\ \mathbb{Z}_n$. In Step 2, Sim either invokes the simulator for $\Pi_{\mathsf{GOLE}}(p_i)$ or it emulates $\mathsf{OLE}_{\mathbb{Z}_{p_i}}$ by receiving x_i from the receiver and returning $a_i x_i + b_i$. Sim sets $r = \mathsf{CRT}(n, \{r_i\}_{i\le t})$ and $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} = r - b$. In step 5 it gets $n - \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}}$ from R and $\langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2}$ from the adversary, which it then outputs to R. If $\langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} < n - \langle z\rangle^{\mathsf{R}}_{\mathbb{Z}}$, it sets $\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} := \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2}$, and $\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} := 1 - \langle s\rangle^{\mathsf{R}}_{\mathbb{Z}_2}$ otherwise. In step 6, Sim gets input $\langle s'\rangle^{\mathsf{R}}_{Z_2}$ from the receiver, and sends the adversary's response c' to R. It then sets

$$w := \langle z\rangle^{\mathsf{S}}_{\mathbb{Z}} - n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} \pmod q + (2n\langle s\rangle^{\mathsf{S}}_{\mathbb{Z}_2} - n \pmod q)\cdot \langle s'\rangle^{\mathsf{R}}_{\mathbb{Z}_2} - c'$$

and sends w and $b - r$ to the receiver, completing the simulation.

5 Constant-Overhead Secure Comparison in $\log^\star$ Rounds

In this section, we introduce several new perfectly secure and perfectly correct protocols for securely computing (shares of) the equality predicate and the greater-than predicate in the ROT-hybrid model. Our protocols improve on the round/communication tradeoff of previous protocols from the literature. In particular, our protocols $\Pi_{\mathsf{ET}}(n)$ and $\Pi_{\mathsf{SC}}(n)$ are the first, to our knowledge, to achieve simultaneously $O(n)$ communication in $O(\log^* n)$ rounds in the ROT-hybrid model. We start by introducing the ET functionality below.

Functionality ET_n

Parties. A sender S and a receiver R

Parameters. Input bit length n

Description. On input $a \in \{0,1\}^n$ from S and $b \in \{0,1\}^n$ from R,

- If both parties are honest, sample $r \leftarrow\!\!\$\ \{0,1\}$. Return r to S and $r \oplus \mathsf{EQ}(a,b)$ to R.
- If S (resp. R) is corrupted, receive $r \in \{0,1\}$ from the adversary $\mathcal{A}$. Return r to S and $r \oplus \mathsf{EQ}(a,b)$ to R (resp. r to R and $r \oplus \mathsf{EQ}(a,b)$ to S).

Functionality 5: Equality test of n-bit string

5.1 Equality test with almost-linear communication in $\log^*$ rounds

We start by recalling a straightforward 2-round ET protocol with exponential communication (see for example [Cou18], but a similar protocol is implicit in other works), denoted $\Pi^e(n)$.

Protocol $\Pi^e_{\mathsf{ET}}(n)$

Parties. A sender S and a receiver R

Parameters. Input bit-length n.

Oracles. COT_2.

Inputs. S holds $a \in \{0,1\}_n$ and R holds $b \in \{0,1\}_n$.

Operations. For every subset $T \subseteq [n]$,

1. S computes $u_T = \prod_{i \in T}(1 - a_i)$ and R computes $v_T = \prod_{i \notin T} b_i$.
2. S and receiver call COT_2 on inputs (u_T, v_T), and receive outputs $s_T, r_T \in \{0,1\}$.

Output. S outputs $\bigoplus_{T \subseteq [n]} s_T$ and R outputs $\bigoplus_{T \subseteq [n]} r_T$.

Protocol 4: Exponential-communication, two-round protocol for equality tests

For perfect correctness, observe that COT_2 distributes shares of the product of the parties inputs: $s_T \oplus r_T = u_T \cdot v_T = \prod_{i \in T}(1 - a_i) \cdot \prod_{i \notin T} b_i$. Therefore,

$$\bigoplus_{T \subseteq [n]} (s_T \oplus r_T) = \bigoplus_{T \subseteq [n]} \prod_{i \in T} (1 - a_i) \cdot \prod_{i \notin T} b_i = \prod_{i=1}^{n} (1 \oplus a_i \oplus b_i) = \mathsf{EQ}(a, b).$$

We omit the proof of perfect security, which is standard and straightforward. Using the protocol $\Pi_{\mathsf{COT}}(2)$ to instantiate the calls to COT using two bits of communication in two rounds, the protocol $\Pi^e_{\mathsf{ET}}(n)$ communicates 2^{n+1} bits in two rounds; this can be reduced to $\mathsf{C}_{\mathsf{ET}^e}(n) = 2^{n+1} - 4$ by iterating only over strict nonempty subsets T (for $T = [n]$ or $T = \varnothing$, one party knows the full monomial and can XOR it to its output directly). We will use this naive protocol as a subroutine of our communication-efficient protocols.

Protocol $\Pi^*_{\mathsf{ET}}(n)$

Parties. A sender S and a receiver R

Parameters. Input bit-length n. Let $k = k(n) := \lceil \log n \rceil$ and $m = m(n) := \lceil n/k(n) \rceil$.

Oracles. $\mathsf{ET}_{k(n)}$, $\mathsf{COT}_{\mathbb{Z}_n}$, COT_2.

Inputs. S holds $a \in \{0,1\}^n$ and R holds $b \in \{0,1\}^n$.

Operations. If $n \leq 2$, S and R run $\Pi^e_{\mathsf{ET}}(n)$, receive outputs (s, r), and skip the next steps. Otherwise, S and R decompose their inputs as $a = (a_0 || \cdots || a_{m-1})$ and $b = (b_0 || \cdots || b_{m-1})$ respectively, where $a_i, b_i \in \{0,1\}^{k(n)}$. For $i = 0$ to $m - 1$,

1. S and R call ET_k on input (a_i, b_i) and receive respective outputs $u_i, v_i \in \{0,1\}$.
2. S and R call $\mathsf{COT}_{\mathbb{Z}_n}$ on inputs $(1 - 2u_i, 1 - v_i)$, and receive outputs $s_i, r_i \in \mathbb{Z}_n$.

Eventually, S sets $a' = \sum_{i=0}^{m-1}(s_i - u_i) \bmod n$ and R sets $b' = \sum_{i=0}^{m-1} r_i \bmod n$, both viewed as k-bit strings. S and R call ET_k on inputs (a', b') and receive outputs (s, r).

Output. S outputs s and R outputs r.

Protocol 5: Almost-linear communication equality tests in $\log^*$ rounds

5.1.1 Communication complexity.

We analyze the communication complexity of $\Pi^*_{\mathsf{ET}}(n)$ in the ROT-hybrid model when

- All calls to ET_k in step 1 are instantiated with a recursive invocation of $\Pi^*_{\mathsf{ET}}(k)$.
- All calls to $\mathsf{COT}_{\mathbb{Z}_n}$ are instantiated with $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$.
- The last ET_k call is instantiated either with a recursive invocation of $\Pi^*_{\mathsf{ET}}(k)$ (case 1), or with an invocation of $\Pi^e_{\mathsf{ET}}(k)$ (case 2).

Let $\mathsf{C}_{\mathsf{ET}^*}(n)$ denote the case-1 communication complexity of $\Pi^*_{\mathsf{ET}}(n)$ in the ROT-hybrid model, and $\mathsf{C}'_{\mathsf{ET}^*}(n)$ denote its case-2 communication complexity.

Claim. $\mathsf{C}_{\mathsf{ET}^*}(n) \le (1 + o(1)) \cdot n \log^* n$.

Proof. We have

$$\begin{aligned}
\mathsf{C}_{\mathsf{ET}^*}(n) &= m(n) \cdot (\mathsf{C}_{\mathsf{ET}^*}(k(n)) + \mathsf{C}_{\mathsf{COT}}(\mathbb{Z}_n)) + \mathsf{C}_{\mathsf{ET}^*}(k(n)) \\
&= (\lceil n/\lceil \log n \rceil \rceil + 1) \cdot \mathsf{C}_{\mathsf{ET}^*}(\lceil \log n \rceil) + \lceil n/\lceil \log n \rceil \rceil \cdot \log n \\
&\le (n/\log n + 2) \cdot \mathsf{C}_{\mathsf{ET}^*}(\lceil \log n \rceil) + n + \log n.
\end{aligned}$$

After t steps of the recursion, the above becomes

$$\begin{aligned}
\mathsf{C}_{\mathsf{ET}^*}(n) &\le \prod_{i=0}^{t-1} \left(\frac{\log^{(i)} n}{\log^{(i+1)} n} + 2 \right) \cdot \mathsf{C}_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) \\
&\qquad + \sum_{i=0}^{t-1} \prod_{j=0}^{i-1} \left(\frac{\log^{(j)} n}{\log^{(j+1)} n} + 2 \right) \cdot (\mathsf{clog}^{(i)} n + \log \mathsf{clog}^{(i)} n) \\
&= (1 + o(1)) \cdot \frac{n}{\log^{(t)} n} \cdot \mathsf{C}_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) + (1 + o(1)) \cdot \sum_{i=0}^{t-1} \frac{n \cdot \mathsf{clog}^{(i)} n}{\log^{(i)} n} \\
&= (1 + o(1)) \cdot \left(\frac{n}{\log^{(t)} n} \cdot \mathsf{C}_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) + t \cdot n \right),
\end{aligned}$$

where clog denotes the function $\lceil \log(\cdot) \rceil$ and for a function f, $f^{(i)}$ denotes f composed with itself i times ($f^{(0)}$ is the identity). Now, for any constant c, let $\mathsf{itlog}_c(n)$ and $\mathsf{itclog}_c(n)$ denote the integers such that $\log c < \log^{(\mathsf{itlog}_c(n))} n \le c$ and $\log c < \mathsf{clog}^{(\mathsf{itclog}_c(n))} n \le c$ respectively.

Claim. *For every integer $c \geq 1$, $\mathsf{itlog}_c(n) = \mathsf{itclog}_c(n)$.*

Proof. Define the iterated exponential $\mathsf{IE}_t(x)$ as follows: $\mathsf{IE}_0(x) = x$, and $\mathsf{IE}_{t+1}(x) = 2^{\mathsf{IE}_t(x)}$. It is clear that $\mathsf{itlog}_c(n) = \min\{t \ : \ \mathsf{IE}_t(c) \geq n\}$. Then, if after t recursive steps of clog we have $n_t \leq c$, using the fact that for any integer $m \in \mathbb{Z}$, $\lceil x \rceil \leq m \iff x \leq m$, we get $\log n_{t-1} \leq c$, hence $n_{t-1} \leq 2^c = \mathsf{IE}_1(c)$. Iterating t times this procedure, we get $n_0 = n \leq \mathsf{IE}_t(c)$. Conversely, if $n \leq \mathsf{IE}_t(c)$, using the same identity $\lceil x \rceil \leq m \iff x \leq m$ yields $\mathsf{clog}(n) \leq \mathsf{IE}_{t-1}(c)$, and eventually $\mathsf{itclog}_c(n) \leq \mathsf{IE}_0(c) = c$. Therefore, $\mathsf{itclog}_c(n) = \min\{t \ : \ \mathsf{IE}_t(c) \geq n\}$, and the claim follows. ■

Fix an integer constant $c \geq 1$. Setting $t = \mathsf{itlog}_c(n)$ in the previous formula and using the fact that $\mathsf{C}_{\mathsf{ET}^*}$ is monotonically increasing, we get

$$\begin{aligned}\mathsf{C}_{\mathsf{ET}^*}(n) &\leq (1+o(1)) \cdot \left(\frac{n}{\log c} \cdot \mathsf{C}_{\mathsf{ET}^*}(c) + \mathsf{itlog}_c(n) \cdot n \right) \\ &= (1+o(1)) \cdot \mathsf{itlog}_c(n) \cdot n + \Theta(n).\end{aligned}$$

In particular, using $\mathsf{C}_{\mathsf{ET}^*}(2) = 2^{2+1} - 4 = 4$ and $\mathsf{itlog}_2(n) = \log^*(n) - 1$ yields $\mathsf{C}_{\mathsf{ET}^*}(n) \leq (1+o(1)) \cdot n \cdot (\log^* n + 3)$, which concludes the proof. ■

We now turn to the communication complexity of $\mathsf{C}'_{\mathsf{ET}^*}(n)$.

Claim. $\mathsf{C}'_{\mathsf{ET}^*}(n) \leq (1+o(1)) \cdot n \cdot (5 \log^* n - 1)$

Proof. The analysis is very similar to that of $\mathsf{C}_{\mathsf{ET}^*}(n)$. We have

$$\begin{aligned}\mathsf{C}'_{\mathsf{ET}^*}(n) &= m(n) \cdot (\mathsf{C}'_{\mathsf{ET}^*}(k(n)) + \mathsf{C}_{\mathsf{COT}}(\mathbb{Z}_n)) + \mathsf{C}_{\mathsf{ET}^e}(k(n)) \\ &= \left\lceil \frac{n}{\lceil \log n \rceil} \right\rceil \cdot \left(\mathsf{C}'_{\mathsf{ET}^*}(\lceil \log n \rceil) + \log n \right) + 2^{\lceil \log n \rceil + 1} - 4 \\ &\leq \left(\frac{n}{\log n} + 1 \right) \cdot \mathsf{C}'_{\mathsf{ET}^*}(\lceil \log n \rceil) + 5n + \log n.\end{aligned}$$

After t steps of the recursion, the above becomes

$$\begin{aligned}\mathsf{C}'_{\mathsf{ET}^*}(n) &\leq \prod_{i=0}^{t-1} \left(\frac{\log^{(i)} n}{\log^{(i+1)} n} + 1 \right) \cdot \mathsf{C}'_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) \\ &\qquad + \sum_{i=0}^{t-1} \prod_{j=0}^{i-1} \left(\frac{\log^{(j)} n}{\log^{(j+1)} n} + 1 \right) \cdot (5\mathsf{clog}^{(i)} n + \log \mathsf{clog}^{(i)} n) \\ &= (1+o(1)) \cdot \frac{n}{\log^{(t)} n} \cdot \mathsf{C}'_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) + (5+o(1)) \cdot \sum_{i=0}^{t-1} \frac{n \cdot \mathsf{clog}^{(i)} n}{\log^{(i)} n} \\ &= (1+o(1)) \cdot \left(\frac{n}{\log^{(t)} n} \cdot \mathsf{C}_{\mathsf{ET}^*}(\mathsf{clog}^{(t)} n) + 5t \cdot n \right).\end{aligned}$$

Using $t = \mathsf{itlog}_c(n)$,

$$\mathsf{C}'_{\mathsf{ET}^*}(n) \leq (1+o(1)) \cdot \left(\frac{n}{\log c} \cdot \mathsf{C}'_{\mathsf{ET}^*}(c) + 5\mathsf{itlog}_c(n) \cdot n \right).$$

In particular, using $\mathsf{C}'_{\mathsf{ET}^*}(2) = 2^{2+1} - 4 = 4$ and $\mathsf{itlog}_2(n) = \log^*(n) - 1$ yields $\mathsf{C}'_{\mathsf{ET}^*}(n) \leq (1+o(1)) \cdot n \cdot (5 \log^* n - 1)$, which concludes the proof. ■

5.1.2 Round complexity.

We first analyze the communication complexity $\mathsf{R}_{\mathsf{ET}^*}(n)$ of the first variant. Using the same implementations of the oracles, we have:

Claim. $\mathsf{R}_{\mathsf{ET}^*}(n) = 2 \cdot \mathsf{R}_{\mathsf{ET}^*}(\mathsf{clog}(n))$

Proof. First, we show that the protocol $\Pi^*_{\mathsf{ET}}(n)$ can be assumed to have the following structure: S speaks first, R speaks last. The proof is an immediate induction: $\Pi^*_{\mathsf{ET}}(2) = \Pi^e_{\mathsf{ET}}(2)$ amounts to one call to COT_2. The protocol $\Pi_{\mathsf{COT}}(2)$ has two rounds and can be instantiated with S speaking first and R speaking last, by observing that COT_2 is a symmetric functionality. Then, $\Pi^*_{\mathsf{ET}}(n)$ starts with parallel invocations of $\Pi^*_{\mathsf{ET}}(k)$ (where R speaks first), followed by $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$, followed by $\Pi^*_{\mathsf{ET}}(k)$ where R speaks last.

Now, since the middle call to $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$ involves two rounds, with R speaking first and S speaking last. This incurs no round overhead, as the first $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$ can be sent with the last round of the parallel calls to $\Pi^*_{\mathsf{ET}}(k)$, and the second $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$ can be sent with the first round of the final call to $\Pi^*_{\mathsf{ET}}(k)$. ■

Using $\mathsf{R}_{\mathsf{ET}^*}(2) = 2$ and $\mathsf{itlog}_2(n) = \log^* n - 1$, we get as an immediate corollary:

Corollary 4. $\mathsf{R}_{\mathsf{ET}^*}(n) = 2^{\log^* n}$.

We now turn to the round complexity of the second variant, denoted $\mathsf{R}'_{\mathsf{ET}^*}(n)$:

Claim. $\mathsf{R}'_{\mathsf{ET}^*}(n) = \mathsf{R}'_{\mathsf{ET}^*}(\mathsf{clog}(n)) + 2$

The proof uses the same induction as before to show that, w.l.o.g., the variant can be assumed to have S speaking first and R speaking last, which allows merging the two rounds of the middle call to $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$ with the calls to $\Pi^*_{\mathsf{ET}}(k)$ and $\Pi^e_{\mathsf{ET}}(k)$ (the latter having two rounds). Using $\mathsf{R}'_{\mathsf{ET}^*}(2) = 2$ and $\mathsf{itlog}_2(n) = \log^* n - 1$, we get as an immediate corollary:

Corollary 5. $\mathsf{R}'_{\mathsf{ET}^*}(n) = 2 \cdot \log^* n$.

5.1.3 Security analysis.

we prove that the protocol $\Pi^*_{\mathsf{ET}}(n)$ satisfies perfect security in the $\{\mathsf{ET}_k, \mathsf{COT}_{\mathbb{Z}_n}, \mathsf{COT}_2\}$-hybrid model. By a standard composition theorem [Gol01, Section 7.3.1], this implies that $\Pi^*_{\mathsf{ET}}(n)$ remains perfectly secure in the semi-honest model when the functionalities are replaced with $\Pi^*_{\mathsf{ET}}(k)$, $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$, and $\Pi_{\mathsf{COT}}(2)$, assuming that these protocols securely implement the corresponding functionalities in the semi-honest model. In the case of $\Pi^*_{\mathsf{ET}}(k)$, this requires recursively invoking the proof of security below until we reach $k \leq 2$, in which case $\Pi^*_{\mathsf{ET}}(k)$ collapses to $\Pi^e_{\mathsf{ET}}(k)$.

Assume that both parties are honest. If $n \leq 2$, correctness follows from the correctness of $\Pi^e_{\mathsf{ET}}(n)$. Otherwise, observe that for every $n > 2$, it holds that $k(n) \geq 2$, and therefore $m(n) \leq n/2 + 1 < n$. We will use this property to guarantee, when we sum m bits over $\mathbb{Z}_n$, that there is no wraparound modulo n. Going over each step:

1. After calling ET_k, S and R hold bits u_i, v_i such that $u_i \oplus v_i = \mathsf{EQ}(a_i, b_i)$.
2. After calling $\mathsf{COT}_{\mathbb{Z}_n}$ on input $(1 - 2u_i, 1 - v_i)$, S and R hold s_i, r_i such that $r_i = s_i + (1 - 2u_i) \cdot (1 - v_i) \bmod n$. Observe that equivalently,

$$r_i = (s_i - u_i) + u_i + (1 - v_i) - 2u_i(1 - v_i) = (s_i - u_i) + u_i \oplus \bar{v}_i \bmod n.$$

Therefore, $a' = \sum_{i=0}^{m-1} r_i = \sum_{i=0}^{m-1}(s_i - u_i) + \sum_{i=0}^{m-1} u_i \oplus \bar{v}_i \bmod n$, which translates to $a' - b' = \sum_{i=0}^{m-1} \mathsf{NEQ}(a_i, b_i) \bmod n$, which is zero if an only if $\mathsf{NEQ}(a_i, b_i) = 0$ for all i (as $m < n$, there is no wraparound), if and only if $a = b$. Therefore, $\mathsf{EQ}(a, b) = \mathsf{EQ}(a', b')$, and correctness follows from the correctness of ET_k on input (a', b').

Assume now that S is corrupted. We describe a straight-line simulator that perfectly emulates R. Sim operates as follows: if $n \leq 2$, Sim invokes the simulator for $\Pi^e_{\mathsf{ET}}(n)$. Otherwise, it emulates the functionalities $\mathsf{ET}_k, \mathsf{COT}_{\mathbb{Z}_n}, \mathsf{COT}_2$. In steps 1 and 2, there is nothing to emulate: Sim receives a_i from S and u_i from the adversary in step 1, and Sim receives $1 - 2u_i$ from the sender and s_i from the adversary in step 2. Eventually, Sim receives a' from S and s from the adversary. It instructs S to input a to ET_n in the ideal world, and send s to ET_n on behalf of the ideal adversary. This completes the simulation; perfect security follows immediately. We omit the converse direction, where R is corrupted, as it is essentially identical.

5.2 Equality test with linear communication in $\log^*$ rounds

Using the protocol $\Pi^*_{\mathsf{ET}}(n)$ as a subroutine, we explain how to further reduce the communication overhead to $O(n)$ while keeping the number of rounds $O(\log^* n)$. The protocol makes use of the standard (corruptible) secure multiplication functionality SM represented on Figure 4.

Protocol $\Pi_{\mathsf{ET}}(n)$

Parties. A sender S and a receiver R

Parameters. Input bit-length n.

Oracles. SM.

Inputs. S holds $a \in \{0,1\}_n$ and R holds $b \in \{0,1\}_n$.

Operations. If $n = 1$, S and R set $s = a$, $r = b$ respectively and skip the next steps. Otherwise, write $a = (a_0, \cdots, a_{n-1}) \in \{0,1\}^n$ and $b = (b_0, \cdots, b_{n-1}) \in \{0,1\}^n$. Set $T = n/\omega(\log^* n)$ and $m = n$.

1. For $i = 0$ to $\lfloor m/2 \rfloor - 1$, S and R call SM on inputs $((1 - a_{2i}, 1 - a_{2i+1}), (b_{2i}, b_{2i+1}))$ and receive respective outputs $u_i, v_i \in \{0,1\}$.
2. If m is even, redefine $m = m/2$, $a = (1 - u_0, \cdots, 1 - u_{m-1})$, $b = (v_0, \cdots, v_{m-1})$. If $m \leq T$, break the loop. Otherwise, go back to step 1.
3. If m is odd, set $(u_{(m-1)/2}, v_{(m-1)/2}) = (a_{m-1}, b_{m-1})$, redefine $m = (m+1)/2$, $a = (1 - u_0, \cdots, 1 - u_m)$, $b = (v_0, \cdots, v_m)$. If $m \leq T$, break the loop. Otherwise, go back to step 1.

Eventually, S and R run $\Pi^*_{\mathsf{ET}}(m)$ on inputs (a, b) and receive (s, r).

Output. S outputs s and R outputs r.

Protocol 6: Linear-communication protocol for equality tests in $\log^*$ rounds

5.2.1 Communication complexity.

We analyze the communication complexity of $\Pi_{\mathsf{ET}}(n)$ in the ROT-hybrid model when SM is instantiated with Π_{SM} from Figure 2. In the protocol $\Pi^*_{\mathsf{ET}}(m)$, we use the following instantiation of the oracles:

- ET_k in step 1 is instantiated with a recursive invocation of $\Pi^*_{\mathsf{ET}}(k)$.
- $\mathsf{COT}_{\mathbb{Z}_n}$ is instantiated with $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$.
- The last ET_k call is instantiated with an invocation of $\Pi^e_{\mathsf{ET}}(k)$.

Let $\mathsf{C}_{\mathsf{ET}}(n)$ denote this quantity.

Claim. $\mathsf{C}_{\mathsf{ET}}(n) \leq (4 + o(1)) \cdot n$.

Proof. Let $m_0 = n$ and for $i \geq 1$, let m_i, denote the value of m after the i-th reduction step. Observe that in each step i, the parties call SM exactly $m_{i-1} - m_i$ times. Therefore, the total number of calls to SM is given by the telescopic sum $\sum_{i=1}^{\ell} m_{i-1} - m_i = n - m_\ell$, where ℓ denotes the final reduction round. By definition, m_ℓ is the first value below T, and $m_\ell \geq T/2$, hence the number of calls to SM is at most $n - T/2$. Since SM uses 4 bits of communication, we get

$$\begin{aligned}
\mathsf{C}_{\mathsf{ET}}(n) &\leq 4 \cdot (n - T/2) + \mathsf{C}_{\mathsf{ET}^*}(T) \\
&\leq 4 \cdot n + (5 + o(1)) \cdot T \log^* T \\
&\leq 4 \cdot n + (5 + o(1)) \cdot \log^* n \cdot \frac{n}{\omega(\log^* n)} \\
&= 4 \cdot n + o(n),
\end{aligned}$$

which concludes the proof. ■

5.2.2 Round complexity.

We turn to the round complexity of $\Pi_{\mathsf{ET}}(n)$, denoted $\mathsf{R}_{\mathsf{ET}}(n)$, under the same instantiations of the oracles.

Claim. $\mathsf{R}_{\mathsf{ET}}(n) = 2 \cdot \log^* n + (1 + o(1)) \cdot \log \log^* n = (2 + o(1)) \cdot \log^* n$.

Proof. Each step of the iterations reduce the input size from m to $\lceil m/2 \rceil$, and the iterations stop as soon as $m \leq T = n/\omega(\log^* n)$. After t iterations, an easy induction shows that we have $m = \lceil n/2^t \rceil$. Then, a sufficient condition for $\lceil n/2^t \rceil \leq T$ is to have $n/2^t \leq T - 1$, hence $t \geq \log(n/(T-1)) = \log \omega(\log^* n) = (1 + o(1)) \log \log^* n$. Each iteration requires one parallel call to SM, which takes two rounds. Because SM is symmetric (S and R can swap roles), the rounds can be interleaved, making the total number of rounds $(1 + o(1)) \cdot \log \log^* n$. Eventually, the final call to $\Pi^*_{\mathsf{ET}}(m)$ requires $\mathsf{R}'_{\mathsf{ET}^*}(m) = 2 \cdot \log^* m \leq 2 \cdot \log^* n$ rounds. This concludes the proof. ■

5.2.3 Security analysis.

We first consider the case where both parties are honest. Going through the operations,

1. After calling SM on inputs $(\bar{a}_{2i}, \bar{a}_{2i+1})$ and (b_{2i}, b_{2i+1}), the parties receive u_i, v_i satisfying $u_i \oplus v_i = (\bar{a}_{2i} \oplus b_{2i}) \cdot (\bar{a}_{2i+1} \oplus b_{2i+1})$. Observe that $1 - u_i = v_i$ iff $u_i \oplus v_i = 1$ iff $(\bar{a}_{2i} \oplus b_{2i} = \bar{a}_{2i+1} \oplus b_{2i+1} = 1$ iff $a_{2i} = b_{2i}$ and $a_{2i+1} = b_{2i+1}$. Therefore, redefining a to $(1 - u_0, \cdots, 1 - u_{\lceil m/2 \rceil - 1})$ and b to $(v_0, \cdots, v_{\lceil m/2 \rceil - 1})$ preserves the value of $\mathsf{EQ}(a, b)$.

Eventually, S and R run Π^*_{ET} on input (a, b), and correctness follows from that of Π^*_{ET}. We now consider the case where S is corrupted. The simulator Sim is straightforward: it emulates SM (as SM is corruptible, this amounts only to receiving messages $(\bar{a}_{2i}, \bar{a}_{2i+1})$ from S and u_i from the adversary), and keeps track of a by locally redefining $a = (1 - u_0, \cdots, 1 - u_{\lceil m/2 \rceil - 1})$. When the iterations terminate, Sim invokes the simulator of $\Pi^*_{\mathsf{ET}}(m)$ on sender input a. The converse case, where R is corrupted, is essentially identical and we omit it.

5.3 Secure comparison with linear communication in $\log^*$ rounds

Equipped with our new linear-communication, $\log^*$-round equality test protocol, we turn to building a perfectly-secure secure comparison protocol in the ROT-hybrid model with similar efficiency features. We introduce on Figure 6 the functionality SC_n. For convenience, we introduce on Figure 7 a second functionality, SSC_n, where the inputs to be compared are bitwise-shared among the parties.

Functionality SC_n

Parties. A sender S and a receiver R

Parameters. Input bit length n

Description. On input $a \in \{0,1\}^n$ from S and $b \in \{0,1\}^n n$ from R,

- If both parties are honest, sample $r \leftarrow\$ \{0,1\}$. Return r to S and $r \oplus \mathsf{GT}(a, b)$ to R.
- If S (resp. R) is corrupted, receive $r \in \{0,1\}$ from the adversary $\mathcal{A}$. Return r to S and $r \oplus \mathsf{GT}(a, b)$ to R (resp. r to R and $r \oplus \mathsf{GT}(a, b)$ to S).

Functionality 6: Corruptible functionality for secure comparison of n-bit strings

Functionality SSC_n

Parties. A sender S and a receiver R

Parameters. Input bit length n

Description. On input $a^{\mathsf{S}}, b^{\mathsf{S}} \in \{0,1\}^n$ from S and $a^{\mathsf{R}}, b^{\mathsf{R}} \in \{0,1\}^n$ from R,

- Define $a = a^{\mathsf{S}} \oplus a^{\mathsf{R}}$ and $b = b^{\mathsf{S}} \oplus b^{\mathsf{R}}$.
- If both parties are honest, sample $r \leftarrow\$ \{0,1\}$. Return r to S and $r \oplus \mathsf{GT}(a, b)$ to R.
- If S (resp. R) is corrupted, receive $r \in \{0,1\}$ from the adversary $\mathcal{A}$. Return r to S and $r \oplus \mathsf{GT}(a, b)$ to R (resp. r to R and $r \oplus \mathsf{GT}(a, b)$ to S).

Functionality 7: Corruptible functionality for secure comparison of shared n-bit strings

5.3.1 An exponential communication protocol for SSC.

We start with the following observation: there exists a protocol that securely instantiates SSC_n using a single round of $3 \cdot 2^n - 4$ parallel calls to the secure multiplication functionality SM. While the protocol is fairly simple, it has not been pointed out to our knowledge, and naively writing $\mathsf{GT}(a, b)$ in algebraic normal form and invoking a secure multiplication for each monomial results in a protocol with a much larger number of calls to SM.

Lemma 6. *Let $a^{\mathsf{S}}, b^{\mathsf{S}} \in \{0,1\}^n$ be the inputs of S, and $a^{\mathsf{R}}, b^{\mathsf{R}} \in \{0,1\}^n$ be the inputs of R. Denote $a = a^{\mathsf{S}} \oplus a^{\mathsf{R}}$ and $b = b^{\mathsf{S}} \oplus b^{\mathsf{R}}$. Let $N(n) = 3 \cdot 2^n - 4$. Then there exists $2n$-variate polynomials $(Q^0_{\mathsf{S}}, Q^N_{\mathsf{S}})$ and $(Q^0_{\mathsf{R}}, Q^N_{\mathsf{R}})$ such that*

$$\mathsf{GT}(a, b) = \bigoplus_{i=0}^{N-1} Q^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \cdot Q^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) \oplus Q^N_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \oplus Q^N_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}).$$

Proof. We prove Lemma 6 by induction. When $n = 1$, $\mathsf{GT}(a, b) = a \cdot \bar{b} = (a^{\mathsf{S}} \oplus a^{\mathsf{R}}) \cdot (b^{\mathsf{S}} \oplus \bar{b}^{\mathsf{R}}) = a^{\mathsf{S}}\bar{b}^{\mathsf{R}} \oplus a^{\mathsf{R}}b^{\mathsf{S}} \oplus a^{\mathsf{S}}b^{\mathsf{S}} \oplus a^{\mathsf{R}}\bar{b}^{\mathsf{R}}$ is a XOR of $N(1) = 3 \cdot 2^1 - 4 = 2$ cross-terms and two polynomials in $(a^{\mathsf{S}}, b^{\mathsf{S}})$ and $(a^{\mathsf{R}}, b^{\mathsf{R}})$ respectively. Then, let us denote $\underline{a}, \underline{b}$ the values a, b respectively, stripped of their last bit. Observe that

$$\mathsf{GT}(a, b) = a_{n-1}\bar{b}_{n-1} \oplus (a_{n-1} \oplus \bar{b}_{n-1}) \cdot \mathsf{GT}(\underline{a}, \underline{b}),$$

since $a_{n-1}\bar{b}_{n-1} = 1$ iff $a_{n-1} > b_{n-1}$, and $a_{n-1} \oplus \bar{b}_{n-1} = 1$ iff $a_{n-1} = b_{n-1}$, hence the above equation captures the fact that $a > b$ iff $a_{n-1} > b_{n-1}$ or $a_{n-1} = b_{n-1}$ and $\underline{a} > \underline{b}$. Now, write $a_{n-1} \oplus \bar{b}_{n-1} = u^{\mathsf{S}} \oplus u^{\mathsf{R}}$ with $u^{\mathsf{S}} = a^{\mathsf{S}}_{n-1} \oplus b^{\mathsf{S}}_{n-1}$ and $u^{\mathsf{R}} = a^{\mathsf{R}}_{n-1} \oplus \bar{b}^{\mathsf{R}}_{n-1}$. By the induction hypothesis,

$$\begin{aligned}
&(a_{n-1} \oplus \bar{b}_{n-1}) \cdot \mathsf{GT}(\underline{a}, \underline{b}) \\
&= (u^{\mathsf{S}} \oplus u^{\mathsf{R}}) \cdot \left(\bigoplus_{i=0}^{N-1} Q^i_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}}) \cdot Q^i_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}}) \oplus Q^N_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}}) \oplus Q^N_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}}) \right) \\
&= \bigoplus_{i=0}^{2N+1} P^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \cdot P^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) \oplus P^{2N+4}_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \oplus P^{2N+4}_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}})
\end{aligned}$$

where we define

- For $i = 0$ to $N - 1$ $P^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) := u^{\mathsf{S}} \cdot Q^i_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}})$, $P^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) := Q^i_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}})$.
- For $i = N$ to $2N - 1$ $P^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) := Q^i_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}})$, $P^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) := u^{\mathsf{R}} \cdot Q^i_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}})$.
- $P^{2N}_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) := u_{\mathsf{S}}$ and $P^{2N}_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) := Q^N_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}})$.
- $P^{2N+1}_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) := Q^N_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}})$ and $P^{2N+1}_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) := u_{\mathsf{R}}$.
- $P^{2N+4}_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) := u_{\mathsf{S}} \cdot Q^N_{\mathsf{S}}(\underline{a}^{\mathsf{S}}, \underline{b}^{\mathsf{S}})$ and $P^{2N+4}_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) := u_{\mathsf{R}} \cdot Q^N_{\mathsf{R}}(\underline{a}^{\mathsf{R}}, \underline{b}^{\mathsf{R}})$.

Then, using

$$\begin{aligned} a_{n-1}\bar{b}_{n-1} &= (a^{\mathsf{S}}_{n-1} \oplus a^{\mathsf{R}}_{n-1}) \cdot (b^{\mathsf{S}}_{n-1} \oplus \bar{b}^{\mathsf{R}}_{n-1}) \\ &= a^{\mathsf{S}}_{n-1}\bar{b}^{\mathsf{R}}_{n-1} \oplus a^{\mathsf{R}}_{n-1}b^{\mathsf{S}}_{n-1} \oplus a^{\mathsf{S}}_{n-1}b^{\mathsf{S}}_{n-1} \oplus a^{\mathsf{R}}_{n-1}\bar{b}^{\mathsf{R}}_{n-1}, \end{aligned}$$

this adds two more cross terms defining the polynomials $P^{2N+2}_{\mathsf{S}}, P^{2N+2}_{\mathsf{R}}$ and $P^{2N+3}_{\mathsf{S}}, P^{2N+3}_{\mathsf{R}}$, and two constant terms that can be added directly to the polynomials $P^{2N+4}_{\mathsf{S}}, P^{2N+4}_{\mathsf{R}}$ respectively. We conclude by observing that $N(n+1) = 3 \cdot 2^{n+1} - 4 = 2 \cdot (3 \cdot 2^n - 4) + 4 = 2N(n) + 4$. ■

We now turn to the protocol $\Pi_{\mathsf{SSC}}(n)$, represented on Figure 7.

Protocol $\Pi_{\mathsf{SSC}}(n)$

Parties. A sender S and a receiver R

Parameters. Input bit-length n.

Oracles. SM.

Inputs. S holds $a^{\mathsf{S}}, b^{\mathsf{S}} \in \{0,1\}^n$ and R holds $a^{\mathsf{R}}, b^{\mathsf{R}} \in \{0,1\}^n$. Let $N = 3 \cdot 2^n - 4$.

Operations. Let $(Q^0_{\mathsf{S}}, Q^N_{\mathsf{S}})$ and $(Q^0_{\mathsf{R}}, Q^N_{\mathsf{R}})$ be the $2n$-variate polynomials guaranteed by Lemma 6 such that

$$\mathsf{GT}(a,b) = \bigoplus_{i=0}^{N-1} Q^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \cdot Q^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) \oplus Q^N_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \oplus Q^N_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}).$$

- For $i = 0$ to $N-1$, S and R call SM on respective inputs $Q^i_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}})$ and $Q^i_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}})$, and receive outputs s_i, r_i.
- S sets $s = Q^N_{\mathsf{S}}(a^{\mathsf{S}}, b^{\mathsf{S}}) \oplus \bigoplus_{i=0}^{N-1} s_i$ and R sets $r = Q^N_{\mathsf{R}}(a^{\mathsf{R}}, b^{\mathsf{R}}) \oplus \bigoplus_{i=0}^{N-1} r_i$.

Output. S outputs s and R outputs r.

Protocol 7: Exponential-communication protocol for secure comparison on shared strings in two rounds

Correctness follows immediately from Lemma 6, and perfect security in the SM-hybrid model is also immediate. We now turn to $\Pi_{\mathsf{SC}}(n)$, our linear-communication protocol, which uses Π_{SSC} as a subroutine.

Protocol $\Pi_{\mathsf{SC}}(n)$

Parties. A sender S and a receiver R

Parameters. Input bit-length n. Let $k = k(n) := \lceil \log n \rceil$ and $m = m(n) := \lceil n/k(n) \rceil$.
▷ For $n \geq 3$, $m + 1 \leq n$

Oracles. $\mathsf{ET}_{k(n)}$, $\mathsf{COT}_{\mathbb{Z}_n}$, COT_k, SSC_k.

Inputs. S holds $a \in \{0,1\}_n$ and R holds $b \in \{0,1\}_n$.

Operations. If $n \leq 2$, S and R run $\Pi^e_{\mathsf{SC}}(n)$, receive outputs (s, r), and skip the next steps. Otherwise, S and R decompose their inputs as $a = (a_0||\cdots||a_{m-1})$ and $b = (b_0||\cdots||b_{m-1})$ respectively, where $a_i, b_i \in \{0,1\}^{k(n)}$. Let $s_0 := 0$ and $r_0 := 0$. Let i^* denote the smallest i such that $a_i \neq b_i$ (or $i^* = m$ if $a = b$). For $i = 0$ to $m-1$,

1. S and R call ET_k on input (a_i, b_i) and receive respective outputs $u_i, v_i \in \{0,1\}$.
   $\triangleright\ u_i \oplus v_i = \mathsf{EQ}(a_i, b_i)$
2. S and R call $\mathsf{COT}_{\mathbb{Z}_n}$ on inputs $(1 - 2u_i, 1 - v_i)$, and receive outputs $s_i, r_i \in \mathbb{Z}_n$.
   $\triangleright\ r_i - (s_i - u_i) = u_i \oplus \bar{v}_i = \mathsf{NEQ}(a_i, b_i) \bmod n$
3. S sets $s'_i \leftarrow \sum_{j=0}^{i}(s_j - u_j) \bmod n$ and R sets $r'_i \leftarrow \sum_{j=0}^{i} r_j \bmod n$.
   $\triangleright\ r'_i - s'_i = \sum_{j=0}^{i}(u_j \oplus \bar{v}_j) = \sum_{j=0}^{i} \mathsf{NEQ}(a_j, b_j) \bmod n$
4. S and R call ET_k on input (s'_i, r'_i) and receive respective outputs $u'_i, v'_i \in \{0,1\}$.
   $\triangleright\ u'_i \oplus v'_i = 1$ iff $i < i^*$
5. S and R set $u''_i \leftarrow u'_i - u'_{i-1}$ and $v''_i \leftarrow v'_i - v'_{i-1}$, defining u''_0, v''_0 to be 1.
   $\triangleright\ u''_i \oplus v''_i = 1$ iff i is the first block where $a_i \neq b_i$
6. S and R call COT_k on inputs (u''_i, b_i) (S playing the role of the receiver and conversely) and receive respective outputs $b^{\mathsf{S}}_i, b^{\mathsf{R}}_i$. R sets $b^{\mathsf{R}}_i \leftarrow b^{\mathsf{R}}_i \oplus v''_i \cdot b_i$
   $\triangleright\ b^{\mathsf{S}}_i \oplus b^{\mathsf{R}}_i = u''_i \cdot b_i \oplus v''_i \cdot b_i = \mathsf{EQ}(i, i^*) \cdot b_i$
7. S and R call COT_k on inputs (a_i, v''_i) and receive respective outputs $a^{\mathsf{S}}_i, a^{\mathsf{R}}_i$. S sets $a^{\mathsf{S}}_i \leftarrow a^{\mathsf{S}}_i \oplus u''_i \cdot a_i$ $\quad\triangleright\ a^{\mathsf{S}}_i \oplus a^{\mathsf{R}}_i = v''_i \cdot a_i \oplus u''_i \cdot a_i = \mathsf{EQ}(i, i^*) \cdot a_i$

Eventually, S and R call SSC_k on respective inputs $(\bigoplus_{i=0}^{m-1} a^{\mathsf{S}}_i, \bigoplus_{i=0}^{m-1} b^{\mathsf{S}}_i)$ and $(\bigoplus_{i=0}^{m-1} a^{\mathsf{R}}, \bigoplus_{i=0}^{m-1} b^{\mathsf{R}})$ and receive outputs (s, r).
$\triangleright\ \bigoplus_{i=0}^{m-1}(a^{\mathsf{S}}_i \oplus a^{\mathsf{R}}_i) = a_{i^*}$ and $\bigoplus_{i=0}^{m-1}(b^{\mathsf{S}}_i \oplus b^{\mathsf{R}}_i) = b_{i^*}$

Output. S outputs s and R outputs r.

Protocol 8: Secure comparison with linear communication and $\log^*$ rounds

5.3.2 Communication complexity.

We analyze the communication complexity of $\Pi_{\mathsf{SC}}(n)$ when the oracles ET_k, $\mathsf{COT}_{\mathbb{Z}_n}$, COT_k, and SSC_k are instantiated with the protocols $\Pi_{\mathsf{ET}}(k)$, $\Pi_{\mathsf{COT}}(\mathbb{Z}_n)$, and $\Pi_{\mathsf{SSC}}(k)$ respectively. We have

$$\begin{aligned}
\mathsf{C}_{\mathsf{SC}}(n) &= \left\lceil \frac{n}{\lceil \log n \rceil} \right\rceil \cdot (2 \cdot \mathsf{C}_{\mathsf{ET}}(\lceil \log n \rceil) + \log n + 1 + 2 \cdot \lceil \log n \rceil + 2) + \mathsf{C}_{\mathsf{SSC}}(\lceil \log n \rceil) \\
&\leq \left(\frac{n}{\lceil \log n \rceil} + 1 \right) \cdot (11 \lceil \log n \rceil + o(\log n)) + 3 \cdot 2^{\lceil \log n \rceil} \\
&\leq 17n + 11 \log n + o(\log n).
\end{aligned}$$

5.3.3 Round complexity.

We now turn to the round complexity of $\Pi_{\mathsf{SC}}(n)$. Among the subroutines, all involve 2 rounds except for $\Pi_{\mathsf{ET}}(k)$, which has round complexity $(2 + o(1)) \log^* n$. It follows immediately that $\Pi_{\mathsf{SC}}(n)$ has round complexity

$$\mathsf{R}_{\mathsf{SC}}(n) = (4 + o(1)) \cdot \log^* n.$$

5.3.4 Security analysis.

We first consider the setting where both parties are honest. Correctness follows by tracking the comments marked in blue on Protocol 8:

1. After calling ET_k, the parties have shares u_i, v_i of $\mathsf{EQ}(a_i, b_i)$.
2. After calling $\mathsf{COT}_{\mathbb{Z}_n}$, the parties receive s_i, r_i such that $r_i = s_i + (1 - 2u_i) \cdot (1 - v_i)$, which translates to $r_i - (s_i - u_i) = u_i \oplus \bar{v}_i = \mathsf{NEQ}(a_i, b_i)$.
3. After summing, the parties get substractive shares of $\sum_{j=0}^{i} \mathsf{NEQ}(a_j, b_j) \bmod n$. Note that since $m + 1 \leq n$ when $n \geq 3$, the sum does not cause a wraparound. Therefore, $r'_i = s'_i$ iff $\mathsf{EQ}(a_j, b_j) = 1$ for $j = 0$ to i, iff $i < i^*$, since i^* is the first block where a, b differ.
4. After calling ET_k, the parties therefore get XOR shares u_i, v_i of 1 for all $i < i^*$, and of 0 for all $i \geq i^*$.
5. The differences u''_i, v''_i are XOR shares of 0 for every i except i^*.
6. After calling COT_k, the parties get shares of $u''_i \cdot b_i$. By adding $v''_i \cdot b_i$ to its share, R turns the shares into shares of $(u''_i \oplus v''_i) \cdot b_i = \mathsf{EQ}(i, i^*) \cdot b_i$.
7. This step proceeds symmetrically and lets the party obtain shares of $\mathsf{EQ}(i, i^*) \cdot a_i$.

Eventually, by XORing their shares, the parties obtain shares of a_{i^*} and b_{i^*}. When $i^* < m$, since i^* is the first block where a, b differ, it holds that $\mathsf{GT}(a, b) = \mathsf{GT}(a_{i^*}, b_{i^*})$, and the parties obtain the correct output by running Π_{SSC} on shares of (a_{i^*}, b_{i^*}). When $i^* = m$, we have $a = b$, and the $a_i^{\mathsf{S}}, a_i^{\mathsf{R}}$ (resp. $b_i^{\mathsf{S}}, b_i^{\mathsf{R}}$) are all shares of 0 (since $\mathsf{EQ}(i, i^*) = 0$ for $i = 0$ to $m - 1$). Since $\mathsf{GT}(a, b) = \mathsf{GT}(0, 0)$, the parties obtain the correct output by running Π_{SSC} on shares of $(0, 0)$.

We now turn to perfect security against a corrupted sender. We let Sim operate as follows: if $n \leq 2$, Sim runs the simulator for $\Pi^{\mathsf{e}}_{\mathsf{SC}}(n)$ with sender input a. Else, Sim emulates calls to ET_k, $\mathsf{COT}_{\mathbb{Z}_n}$, and COT_k by receiving values from S and the adversary (and returning nothing). In the last two calls to COT_k, Sim receives u''_i, a_i from S and $b_i^{\mathsf{S}}, a_i^{\mathsf{S}}$ from the adversary. Sim updates $a_i^{\mathsf{S}} \leftarrow a_i^{\mathsf{S}} \oplus u''_i \cdot a_i$. Eventually, Sim calls the simulator of $\Pi_{\mathsf{SSC}}(k)$, forwarding $(\bigoplus_{i=0}^{m-1} a_i^{\mathsf{S}}, \bigoplus_{i=0}^{m-1} b_i^{\mathsf{S}})$ as inputs for S. Upon receiving the output s, Sim sends a to SC_n on behalf of S, and s to SC_n on behalf of the ideal adversary. Security against a corrupted receiver proceeds identically, and we omit the proof.

References

[ABC+24] Michael Adjedj, Constantin Blokh, Geoffroy Couteau, Arik Galansky, Antoine Joux, and Nikolaos Makriyannis. Two-round 2PC ECDSA at the cost of 1 OLE. Cryptology ePrint Archive, Report 2024/1950, 2024.

[ADOS22] Damiano Abram, Ivan Damgård, Claudio Orlandi, and Peter Scholl. An algebraic framework for silent preprocessing with trustless setup and active security. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 421–452, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

[ARS24] Damiano Abram, Lawrence Roy, and Peter Scholl. Succinct homomorphic secret sharing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pages 301–330, Zurich, Switzerland, May 26–30, 2024. Springer, Cham, Switzerland.

[BCCD23] Maxime Bombar, Geoffroy Couteau, Alain Couvreur, and Clément Ducros. Correlated pseudorandomness from the hardness of quasi-abelian decoding. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 567–601, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[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, London, UK, November 11–15, 2019. ACM Press.

[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, Cham, Switzerland.

[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, Durham, NC, USA, November 16–19, 2020. IEEE Computer Society Press.

[BCG+20b] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators from ring-LPN. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 387–416, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.

[BCG+22] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Nicolas Resch, and Peter Scholl. Correlated pseudorandomness from expand-accumulate codes. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 603–633, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

[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, *TCC 2022: 20th Theory of Cryptography Conference, Part II*, volume 13748 of *Lecture Notes in Computer Science*, pages 121–150, Chicago, IL, USA, November 7–10, 2022. Springer, Cham, Switzerland.

[BCM23] Elette Boyle, Geoffroy Couteau, and Pierre Meyer. Sublinear-communication secure multiparty computation does not require FHE. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 159–189, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[BCM+24] Dung Bui, Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Fast public-key silent OT and more from constrained Naor-Reingold. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pages 88–118, Zurich, Switzerland, May 26–30, 2024. Springer, Cham, Switzerland.

[BCM+25] Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, and Elahe Sadeghi. Fast pseudorandom correlation functions from sparse lpn. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 439–473. Springer, 2025.

[Bea96] Donald Beaver. Correlated pseudorandomness and the complexity of private computations. In *28th Annual ACM Symposium on Theory of Computing*, pages 479–488, Philadephia, PA, USA, May 22–24, 1996. ACM Press.

[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 Berlin Heidelberg, Germany.

[CDD+24] Geoffroy Couteau, Lalita Devadas, Srinivas Devadas, Alexander Koch, and Sacha Servan-Schreiber. QuietOT: Lightweight oblivious transfer with a public-key setup. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part II*, volume 15485 of *Lecture Notes in Computer Science*, pages 197–231, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

[CK24] Geoffroy Couteau and Naman Kumar. 10-party sublinear secure computation from standard assumptions. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part IX*, volume 14928 of *Lecture Notes in Computer Science*, pages 39–73, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

[CKY25] Geoffroy Couteau, Naman Kumar, and Xiaxi Ye. Multiparty homomorphic secret sharing and more from LPN and MQ. In Benny Applebaum and Huijia (Rachel) Lin, editors, *TCC 2025: 23rd Theory of Cryptography Conference, Part II*, volume 16269 of *Lecture Notes in Computer Science*, pages 498–530, Aarhus, Denmark, December 1–5, 2025. Springer, Cham, Switzerland.

[CM21] Geoffroy Couteau and Pierre Meyer. Breaking the circuit size barrier for secure computation under quasi-polynomial LPN. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 842–870, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland.

[Cou18] Geoffroy Couteau. New protocols for secure equality test and comparison. In Bart Preneel and Frederik Vercauteren, editors, *ACNS 2018: 16th International Conference on Applied Cryptography and Network Security*, volume 10892 of *Lecture Notes in Computer Science*, pages 303–320, Leuven, Belgium, July 2–4, 2018. Springer, Cham, Switzerland.

[DGN+17] Nico Döttling, Satrajit Ghosh, Jesper Buus Nielsen, Tobias Nilges, and Roberto Trifiletti. TinyOLE: Efficient actively secure two-party computation from oblivious linear function evaluation. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 2263–2276, Dallas, TX, USA, October 31 – November 2, 2017. ACM Press.

[DHIM25] Jack Doerner, Iftach Haitner, Yuval Ishai, and Nikolaos Makriyannis. From OT to OLE with subquadratic communication. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *ACM CCS 2025: 32nd Conference on Computer and Communications Security*, pages 2249–2263, Taipei, Taiwan, October 13–17, 2025. ACM Press.

[DIJL23] Quang Dao, Yuval Ishai, Aayush Jain, and Huijia Lin. Multi-party homomorphic secret sharing and sublinear MPC from sparse LPN. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part II*, volume 14082 of *Lecture Notes in Computer Science*, pages 315–348, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[DKLs24] Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. Threshold ECDSA in three rounds. In *2024 IEEE Symposium on Security and Privacy*, pages 3053–3071, San Francisco, CA, USA, May 19–23, 2024. IEEE Computer Society Press.

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 643–662, Santa Barbara, CA, USA, August 19–23, 2012. Springer Berlin Heidelberg, Germany.

[EGL82] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors,

Advances in Cryptology – CRYPTO'82, pages 205–210, Santa Barbara, CA, USA, 1982. Plenum Press, New York, USA.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19, Interlaken, Switzerland, May 2–6, 2004. Springer Berlin Heidelberg, Germany.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press.

[Gil99] Niv Gilboa. Two party RSA key generation. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 116–129, Santa Barbara, CA, USA, August 15–19, 1999. Springer Berlin Heidelberg, Germany.

[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.

[Gol87] Oded Goldreich. Towards a theory of software protection and simulation by oblivious RAMs. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 182–194, New York City, NY, USA, May 25–27, 1987. ACM Press.

[Gol01] Oded Goldreich. *Foundations of cryptography: volume 2, basic applications*, volume 2. Cambridge university press, 2001.

[GSV07] Juan A. Garay, Berry Schoenmakers, and José Villegas. Practical and secure solutions for integer comparison. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007: 10th International Conference on Theory and Practice of Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 330–342, Beijing, China, April 16–20, 2007. Springer Berlin Heidelberg, Germany.

[HVDH21] David Harvey and Joris Van Der Hoeven. Integer multiplication in time o(nlog\,n). *Annals of Mathematics*, 193(2):563–617, 2021.

[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 Berlin Heidelberg, Germany.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591, Santa Barbara, CA, USA, August 17–21, 2008. Springer Berlin Heidelberg, Germany.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *20th Annual ACM Symposium on Theory of Computing*, pages 20–31, Chicago, IL, USA, May 2–4, 1988. ACM Press.

[KK13] Vladimir Kolesnikov and Ranjit Kumaresan. Improved OT extension for transferring short secrets. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 54–70, Santa Barbara, CA, USA, August 18–22, 2013. Springer Berlin Heidelberg, Germany.

[KO97] Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science*, pages 364–373, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press.

[KOS16] Marcel Keller, Emmanuela Orsini, and Peter Scholl. MASCOT: Faster malicious arithmetic secure computation with oblivious transfer. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 830–842, Vienna, Austria, October 24–28, 2016. ACM Press.

[KPR18] Marcel Keller, Valerio Pastro, and Dragos Rotaru. Overdrive: Making SPDZ great again. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 158–189, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Cham, Switzerland.

[Lin17] Yehuda Lindell. Fast secure two-party ECDSA signing. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 613–644, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Cham, Switzerland.

[LXYY25] Zhe Li, Chaoping Xing, Yizhou Yao, and Chen Yuan. Efficient pseudorandom correlation generators for any finite field. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025, Part V*, volume 15605 of *Lecture Notes in Computer Science*, pages 145–175, Madrid, Spain, May 4–8, 2025. Springer, Cham, Switzerland.

[OSY21] Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 678–708, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland.

[Rab05] Michael O. Rabin. How to exchange secrets with oblivious transfer. Cryptology ePrint Archive, Report 2005/187, 2005.

[Roy22] Lawrence Roy. SoftSpokenOT: Quieter OT extension from small-field silent VOLE in the minicrypt model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in*

Cryptology – CRYPTO 2022, Part I, volume 13507 of *Lecture Notes in Computer Science*, pages 657–687, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.

[RS21] Lawrence Roy and Jaspal Singh. Large message homomorphic secret sharing from DCR and applications. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 687–717, Virtual Event, August 16–20, 2021. Springer, Cham, Switzerland.