Two-Round MPC without Round Collapsing Revisited – Towards Efficient Malicious Protocols

Huijia Lin[1] and Tianren Liu[2⋆]

[1] University of Washington, Seattle, US, rachel@cs.washington.edu
[2] CFCS, Peking University, Beijing, China, trl@pku.edu.cn

Abstract. Recent works have made exciting progress on the construction of round optimal, *two-round*, Multi-Party Computation (MPC) protocols. However, most proposals so far are still complex and inefficient.
In this work, we improve the simplicity and efficiency of two-round MPC in the setting with dishonest majority and malicious security. Our protocols make use of the Random Oracle (RO) and a generalization of the Oblivious Linear Evaluation (OLE) correlated randomness, called tensor OLE, over a finite field $\mathbb{F}$, and achieve the following:

- *MPC for Boolean Circuits:* Our two-round, maliciously secure MPC protocols for computing Boolean circuits, has overall (asymptotic) computational cost $O(S \cdot n^3 \cdot \log|\mathbb{F}|)$, where S is the size of the circuit computed, n the number of parties, and $\mathbb{F}$ a field of characteristic two. The protocols also make black-box calls to a Pseudo-Random Function (PRF).
- *MPC for Arithmetic Branching Programs (ABPs):* Our two-round, information theoretically and maliciously secure protocols for computing ABPs over a general field $\mathbb{F}$ has overall computational cost $O(S^{1.5} \cdot n^3 \cdot \log|\mathbb{F}|)$, where S is the size of ABP computed.

Both protocols achieve security levels inverse proportional to the size of the field $|\mathbb{F}|$.
Our construction is built upon the simple two-round MPC protocols of [Lin-Liu-Wee TCC'20], which are only semi-honest secure. Our main technical contribution lies in ensuring malicious security using simple and lightweight checks, which incur only a constant overhead over the complexity of the protocols by Lin, Liu, and Wee. In particular, in the case of computing Boolean circuits, our malicious MPC protocols have the same complexity (up to a constant overhead) as (insecurely) computing Yao's garbled circuits in a distributed fashion.
Finally, as an additional contribution, we show how to efficiently generate tensor OLE correlation in fields of characteristic two using OT.

1 Introduction

Improving efficiency is a central theme in the design of cryptographic protocols. Two important aspects are *computational efficiency* and *round efficiency*. In the

⋆ The work was partially done when Liu was a postdoctoral researcher at University of Washington.

context of secure Multi-Party Computation (MPC) protocols, since the seminal works in the 80s [Yao82,GMW87,BGW88,CCD88], remarkable improvements have been made on both fronts.

- In the past decade, innovative design and implementation improvements have drastically reduced the computational cost of MPC, leading to efficient protocols more and more applicable to practical situations (e.g., the SPDZ protocols [DPSZ12] and its followup works).
- Another long line of researches on minimizing the round complexity of MPC recently culminated at the construction of *two-round* MPC protocols based on the (minimal) assumption of two-round Oblivious Transfer (OT) in the Common Reference String (CRS) model [BL18,GS18]. Two rounds are optimal even for achieving only semi-honest security and with trusted setups [FKN94,IK97].

However, so far, most two-round MPC protocols are complex and inefficient, especially those achieving malicious security (even in correlated randomness and/or trusted setup model). Encouraged by the efficiency improvement in the realm of many-round MPC in the past decade, this work strives to improve the simplicity and efficiency of two-round MPC in the malicious setting with dishonest majority. Existing techniques can be broadly classified as follows:

- *Round Collapsing:* Introduced by [GGHR14] and initially relying on strong primitives such as indistinguishability obfuscation (iO) or witness encryption [GP15,CGP15,DKR15,GLS15], the round collapsing approach was improved in [GS17,BL18,GS18,GIS18,BLPV18] to rely on just malicious 2-round OT. The complexity of this approach stems from applying the *garbling technique* (e.g., [Yao82,AIK04]) to the next step function of a many-round MPC protocol to collapse the number of rounds to two. The non-black-box use of the underlying MPC protocol hurts both asymptotic and concrete efficiency.
- *Using Generic Non-Interactive Zero Knowledge (NIZK):* This approach starts with designing two-round MPC that are semi-maliciously secure[3], and then transform them to maliciously secure ones by applying generic NIZK to detect deviation from the protocol specification. Two round semi-malicious protocols can be built either via the above round collapsing approach or using primitives supporting homomorphic computation, such as, multi-key fully homomorphic encryption [AJL+12,MW16,CM15,BP16,PS16,AJJM20] or homomorphic secret sharing [BGI16,BGI17,BGI+18,BGMM20]. Using NIZK to prove about the execution of the semi-malicious MPC protocols is inefficient and leads to non-black-box use of underlying assumptions.
- *MPC-in-the-Head [IKOS07,IPS08,IKSS21]:* Another generic method for strengthening weak security to strong security is the "MPC-in-the-head" transformations [IKOS07,IPS08]. The recent work by [IKSS21] showed how to perform such transformations in just two rounds. To obtain a two-round maliciously secure protocol, the transformation uses a two-round protocol with

[3] These are protocols secure against corrupted parties who follow the protocol specification but may choose its input and randomness arbitrarily.

(enhanced) semi-honest security [GIS18,LLW20], to *emulate* the execution of another two-round protocol that is maliciously secure in the honest majority setting [IKP10,Pas12]. The overall complexity is the (multiplicative) compound complexity of both protocols.

We observe that existing designs of two-round malicious MPC all apply generic transformations – using garbling or NIZK or MPC – to some underlying MPC, which often leads to non-black-box constructions (with exceptions [GIS18,IKSS21]) and inefficient protocols. To improve the state-of-affairs, we consider protocols that use the Random Oracle (RO) and simple correlated randomness that can be efficiently generated in an offline phase, and aim for either information theoretic security or computational security with black-box use of simple cryptographic tools like Pseudo-Random Functions (PRFs). As seen in the literature, both the random oracle and the online-offline model are extremely successful settings for designing efficient cryptographic protocols.

Our Results: We present a ligitweight construction of 2-round malicious MPC protocols, using RO and an enhanced version of the Oblivious Linear Evaluation (OLE) correlation, called *tensor* OLE. The OLE correlation is the arithmetic generalization of the OT correlation over a finite field $\mathbb{F}$. It distributes to one party (a_1, b_1) and another (a_2, b_2) which are random elements in $\mathbb{F}$ subject to satisfying the equation $a_1 a_2 = b_1 + b_2$[4]. The tensor OLE correlation further generalizes the OLE correlation to higher dimension: For dimension $k_1 \times k_2$,

$$P_1 \text{ holds: } \mathbf{a}_1 \in \mathbb{F}^{k_1},\ \mathbf{B}_1 \in \mathbb{F}^{k_1 \times k_2}, \qquad P_2 \text{ holds: } \mathbf{a}_2 \in \mathbb{F}^{k_2},\ \mathbf{B}_2 \in \mathbb{F}^{k_1 \times k_2}$$
$$\text{where } \mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2 \text{ are random, subject to } \mathbf{a}_1 \mathbf{a}_2^{\intercal} = \mathbf{B}_1 + \mathbf{B}_2 \ .$$

In our protocols, we will use pairwise tensor OLE correlation, with only small constant dimension, concretely 4×4 and 1×11. Such correlation can be generated efficiently in an offline phase using off-the-shelf OLE protocols [BCGI18,CDI+19]. We also show how to efficiently generate tensor OLE correlation for fields of characteristic two using OT, which in turn can be generated with good concrete efficiency [IKNP03,BCG+19]. We can further rely on pseudorandom correlation, which can be efficiently generated in using techniques described in [BCG+20].

Using tensor OLE correlation and RO, we obtain the following protocols, in the setting of static corruption and security with abort.

MPC for Boolean Circuits: Our first result is a construction of efficient two-round maliciously secure MPC protocols for general Boolean circuits. The protocols make use of RO and tensor OLE over a finite field $\mathbb{F}$ of characteristic two, as well as black-box calls to a PRF. (Note that we choose to not instantiate the PRF with RO, because the latter is used for a different purpose. To obtain standard-model protocols, we will employ the random oracle heuristic to replace RO with a real-life hash function. By separating PRF from RO, we reduce the use of heuristic.) When computing an n-ary circuit C, the overall computational costs of all parties is $O(|C| \cdot n^3 \cdot \log \mathbb{F})$. The security level of the protocol is inverse

[4] When the field is GF(2), OLE correlation coincides with the OT correlation.

proportional to the field size $|\mathbb{F}|^{-1}$; thus, $\log|\mathbb{F}|$ can be viewed as the effective security parameter. More formally,

Theorem 1 (MPC for Boolean Circuits, informal). *Let $\mathbb{F}$ be a finite field of characteristic two. Let C an n-ary Boolean circuit $C : \{0,1\}^{\ell_1} \times \cdots \times \{0,1\}^{\ell_n} \to \{0,1\}^{\ell}$. Assume the existence of a PRF F with security level $2^{-\kappa(\lambda)}$ where λ is the seed length.*

There exists a two-round MPC protocol Π that securely computes C, using RO and tensor OLE correlated randomness, making black-box calls to the PRF, and achieving

- *overall complexity $O(\Gamma \log \mathbb{F})$ for $\Gamma := n^3 \cdot |C|$, and*
- *ϵ-computational, malicious security with selective abort, against up to $n-1$ corruption, where the security level $\epsilon = \frac{O(\Gamma + n^2 \cdot Q_{\mathrm{RO}})}{|\mathbb{F}|} + \frac{O(n \cdot |C|)}{2^{\kappa(\log|\mathbb{F}|)}}$, and Q_{RO} is the number of random oracle queries that the adversary makes.*

More specifically, parties in our protocols communicate in total $O(\Gamma)$ elements in $\mathbb{F}$, perform in total $O(\Gamma)$ arithmetic operations in $\mathbb{F}$, use in total $O(\Gamma)$ pairs of tensor OLE correlated randomness of constant dimensions, and make in total $O(\Gamma)$ random oracle calls and $O(|C| \cdot n)$ PRF calls. In addition, we can enhance the protocol to have security with unanimous abort at the cost of increasing the complexity by an additive $\mathrm{poly}(n, \lambda)$ overhead.

Our construction follows the technique in [BMR90,DI05,LPSY15] developed in the context of constructing constant-round MPC. They showed that to securely computing a Boolean circuit C, it suffices to securely compute Yao's garbling of the circuit [Yao82]. Furthermore, the latter can be computed by a *degree three* polynomial f over a finite filed $\mathbb{F}$ of characteristic 2 – we call them the **distributed-Yao** polynomials. Therefore, designing 2-round protocols for general circuits boils down to designing 2-round protocols for computing the degree three distributed-Yao polynomials. This is indeed the approach taken by [LLW20]; however, they achieve only semi-honest security. In this work, we further achieve malicious security (see Lemma 1 below), and like [LLW20], our protocols incur only *constant overheads* – their overall asymptotically complexity is the same as the complexity of distributed-Yao polynomials.

We give slightly more detail on distributed-Yao polynomials. They compute Yao's garbled circuits in a special way: First, labels for a wire u has form $\boldsymbol{\ell}_{u,b} = s^{(1)}_{u,b} \| \ldots \| s^{(n)}_{u,b}$, where $s^{(i)}_{u,b} \in \mathbb{F}$ is a PRF key sampled by party P_i. Next, the garbled table for a gate g with input wire u, v and output wire o contains entries of the form $\boldsymbol{\ell}_{o,g(a,b)} \oplus (\bigoplus_i Y^{(i)}_{u,a}) \oplus (\bigoplus_i Y^{(i)}_{v,b})$, where $Y^{(i)}_{u,a}$ and $Y^{(i)}_{v,b}$ are pseudorandom one-time-pads generated via PRF using party P_i's keys $s^{(i)}_{u,a}$ and $s^{(i)}_{v,b}$ respectively (evaluating on different inputs). Hence, the output label is hidden as long as one of the PRF keys is hidden. These entries are additionally permuted using mask bits k_u, k_v which are additively shared among all parties. The important point made by [BMR90,DI05,LPSY15] is that the PRF evaluations can be done locally by each party, and given the PRF outputs as inputs to f, such a garbled circuit can be computed in just degree 3 in $\mathbb{F}$. Analyzing the distributed-Yao

polynomial for a circuit C reveals that it contains $O(\Gamma) = O(|C| \cdot n^3)$ monomials over $\mathbb{F}$. In comparison, our MPC protocol implementing C has overall complexity $O(|C| \cdot n^3 \cdot \log |\mathbb{F}|)$ incurring only a constant-overhead over distributed-Yao.

MPC for Arithmetic Branching Programs (ABPs): Using similar approach, we obtain efficient, two-round, MPC protocols for ABPs over field $\mathbb{F}$. Here, we compute instead the distributed version of the degree three randomized encoding of Applebaum, Ishai, and Kushilevitz (AIK) [AIK04] for ABPs. More precisely, for an ABP g, we shall compute the n-ary polynomial $f((\mathbf{x}_1, \mathbf{r}_1), \cdots, (\mathbf{x}_n, \mathbf{r}_n)) = \mathsf{AIK}_g((\mathbf{x}_1, \cdots \mathbf{x}_n);\ \Sigma_{i\in[n]}\mathbf{r}_i)$, where the randomness used for computing the AIK encoding is additively shared among all n parties. We refer to this polynomial the **distributed-AIK** polynomial. The complexity of the resulting MPC protocol is determined by the number of monomials in this polynomial, which is $O(\Gamma) = O(|g|^{1.5}n^2)$. However, different from the case for circuits, our two-round protocols now incur a factor $O(n)$ overhead. Constant-overhead can be retained by adding one more round. More formally,

Theorem 2 (MPC for ABPs, informal). *Let $\mathbb{F}$ be a finite field. Let g be an n-ary arithmetic branching program over $\mathbb{F}$, $g : \mathbb{F}^{l_1} \times \cdots \times \mathbb{F}^{l_n} \to \mathbb{F}$. Denote by $|g|$ the size of the matrix $M_g(\cdot)$ describing g s.t. $\det(M_g(x)) = g(x)$ for any x.*

There exists a two-round MPC protocol Π that securely computes f, using RO *and tensor* OLE *correlated randomness and achieving*

- *overall complexity $O(\Gamma \cdot n \cdot \log |\mathbb{F}|)$ for $\Gamma = |g|^{1.5}n^2$ and*
- *ϵ-statistical, malicious security with abort, against up to $n-1$ corruption where the statistical simulation error is $\epsilon = \frac{O(\Gamma \cdot n + n^2 Q_{\mathrm{RO}})}{|\mathbb{F}|}$ and Q_{RO} is the number of random oracle queries that the adversary makes.*

Furthermore, there is a three-round protocol achieving the same as above, but with overall complexity $O(\Gamma \cdot \log |\mathbb{F}|)$.

More specifically, parties of the 3-round protocols communicate in total $O(\Gamma)$ elements in $\mathbb{F}$, perform in total $O(\Gamma)$ arithmetic operations in $\mathbb{F}$, and use in total $O(\Gamma)$ pairs of tensor OLE correlated randomness of constant dimensions.

MPC for degree three polynomials: The key that enables above theorems is our construction of, two-round, MPC protocols for computing degree *three* polynomials over an (arbitrary) sufficiently large finite field $\mathbb{F}$. Importantly, the protocol has **constant overhead** – when computing polynomials with Γ monomials over $\mathbb{F}$, our protocols have **overall complexity** $O(\Gamma \cdot \log |\mathbb{F}|)$. Furthermore, the protocol makes only black-box use to the underlying field $\mathbb{F}$.

In order to achieve constant overhead, we only require these protocols to achieve a weaker malicious security, called *security with output substitution*. Intuitively, the protocol ensures the usual privacy guarantee of honest parties inputs – that nothing about honest parties' inputs are revealed beyond the output $\mathbf{y} = f(\mathbf{x}_1, \cdots, \mathbf{x}_n)$. But the honest party may (unanimously) receive incorrect output – the adversary always learn $\mathbf{y}$, and can replace it with another output $\mathbf{y}'$ of its choice without honest parties noticing.

Lemma 1 (MPC for degree 3 polynomials, Informal). *Let $\mathbb{F}$ be a finite field. Let f be an n-ary degree three polynomial over $\mathbb{F}$, $f : \mathbb{F}^{\ell_1} \times \cdots \times \mathbb{F}^{\ell_n} \to \mathbb{F}^{\ell}$; denote by $|f|$ the number of monomials in f.*

There exists a two-round MPC protocol Π that securely computes f, using RO *and tensor* OLE *correlated randomness and achieving the following:*

- *overall complexity $O(|f|)$, and*
- *ϵ-statistical, malicious security with output substitution, against up to $n-1$ corruption, where the statistical simulation error is $\epsilon = \frac{O(|f|+n^2 Q_{\mathrm{RO}})}{|\mathbb{F}|}$ and Q_{RO} is the number of random oracle queries that the adversary makes.*

Our construction easily generalizes to computing constant degree polynomials with constant overhead, which might be of independent interests.

Using the above protocols to compute the distributed-Yao or distributed-AIK polynomials gives 2-round protocols for circuits or ABPs respectively, but achieving only security with output substitution. We complement this by a generic transformation that enhances security with output substitution to security with abort. Essentially, the transformation computes a related circuit (or ABP resp.) that computes not only the output, but also authenticates of the output using partys' private keys (supplied as part of the input). Since security with output substitution ensures the privacy of honest parties' keys, the adversary can no longer substitute the output without being detected. This transformation incurs only a small additive overhead in the case of circuits, but a multiplicative overhead $O(n)$ in the case of ABPs. That's why our two-round ABP protocols do not achieve constant overhead over the complexity of distributed-AIK. We show a different transformation that uses one more round to recover constant overhead.

Tensor OLE over $\mathsf{GF}(2^\lambda)$ from OT: As a final contribution, we construct an efficient 4-round protocol for generating the tensor OLE correlation over $\mathsf{GF}(2^\lambda)$.

Theorem 3 (Tensor OLE over $\mathsf{GF}(2^\lambda)$ from OT, Informal). *There is a 4-round, statistically and maliciously secure two party computation protocol for sampling tensor OLE correlations, in the OT hybrid model.*

Our protocol is simple and efficient; in particular, it does not use any generic 2PC techniques such as garbling and zero-knowledge protocols. Thus, parties can run this efficient protocol to generate tensor OLE correlations in an offline stage using OT, which in turn can be generated with concrete efficiency [IKNP03,BCG+19].

Comparison with Prior Two-Round MPC As discussed before, prior 2-round MPC constructions can be categorized into three types depending on their main technique: 1) round collapsing, 2) using NIZK, and 3) MPC-in-the-head. Almost all protocols using round collapsing and all protocols using NIZK make non-black-box use of underlying cryptographic primitive (e.g., MPC and MKFHE etc.), and many of them have poor asymptotic efficiency (e.g., [BL18,GS18]). The only black-box constructions are [GIS18,IKSS21], which as we discuss below are less efficient than our protocols.

The construction of [GIS18] is in the OT correlation model and uses the round collapsing technique. To compute a Boolean circuit f, parties need to garble the

next step functions of an information theoretically and maliciously secure MPC protocol Π for f making black-box calls to OT (e.g. [IPS08]). Let C_Π be the circuit induced by Π with depth d_Π and size $|C_\Pi|$. The overall communication complexity is at least $(d_\Pi|C_\Pi|n^2\lambda^2)$, where λ is the security parameter. Since d_Π is at least the depth d of f, and $|C_\Pi|$ at least the size of f. This leads to a dependency on $d \cdot |f|$, which is worse than our complexity.

The construction of [IKSS21] following the MPC-in-the-head approach uses a two-round protocol with (enhanced) semi-honest security such as [GIS18,LLW20], to emulate the execution of another two-round protocol that is maliciously secure in the honest majority setting [IKP10,Pas12]. Consider for instance, the complexity of [LLW20] is already $\Omega(|f|n^3\lambda)$ and a loose lower bound of the complexity of [IKP10] is $\Omega(Sn^5\lambda)$. The overall complexity is at least $\Omega(Sn^8\lambda^2)$.

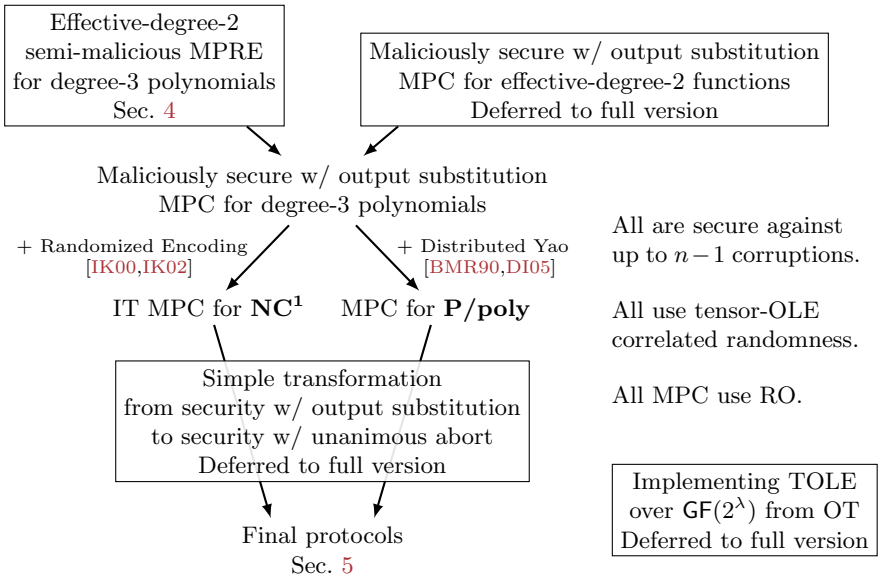

Fig. 1. Our roadmap

2 Technical Overview

Our construction follows the overall structure of the semi-honest 2-round MPC protocols of [LLW20], which uses OLE correlated randomness. However, to make LLW maliciously secure, we face two challenges:

Challenge 1: Design *simple* and *efficient* checks to detect malicious behaviours. To beat previous works, we avoid any generic transformation, such as, using generic NIZK, or even cryptographic operations. Our core protocol will be information-theoretic secure, only use black-box operations over the field. Thus the detection of malicious moves can only rely on arithmetic methods.

Challenge 2: An adversary may lie about the *correlated randomness* it received. Such malicious behavior can hardly be caught even if we allow generic NIZK proof. This is because no party can write "using proper correlated randomness" as a NP-statement, and such statement naturally involves at least 2 parties who jointly hold the correlated randomness.

To deal with these challenges, we use tensor OLE correlated randomness instead of the scalar version, so that we have more room to play with. We also use random oracle to generate challenges in the Fiat-Shamir style, so that our arithmetic proofs become non-interactive. We start with *security with output substitution*, and later transform to security with (unanimous) abort.

In the rest of the overview, we will walk through our constructions. We follow the successful paradigm of [IK00,IK02,ABT18,ABT19,LLW20]: reduce the task of securely computing a function f to the task of securely computing a simpler function $\hat{f}$. Such reduction is captured by MPRE. Sec. 2.1 revisits the definition of MPRE. Sec. 2.2 briefly presents our new MPRE, which reduces the task of computing general functions to the task of computing so-called "effective-degree-2" functions. Sec. 2.3 outlines 2-round malicious MPC protocols for computing effective-degree-2 functions. Composing them yields a 2-round MPC for general functions, but it only satisfies a weak notion of security. Sec. 2.4 outlines how to lift the security by a simple transformation. Our new constructions of MPRE and MPC are based on tensor OLE correlated randomness. In Sec. 2.5, we show how to generate such correlated randomness from OT. See Fig. 1 for a summary of the technical components and their sections in the technical body.

2.1 Multi-Party Randomized Encoding

For a n-party function f, an MPRE of f consists of n preprocessing functions $h_1, \ldots, h_n$, an encoding function $\hat{f}$, and a decoding function Dec, such that,

$$\mathsf{Decode}(\hat{f}(h_1(\mathbf{x}_1, \mathbf{r}_1, \mathbf{r}'_1), \cdots, h_n(\mathbf{x}_n, \mathbf{r}_n, \mathbf{r}'_n))) = f(\mathbf{x}_1, \cdots, \mathbf{x}_n) \ .$$

where the preprocessing function h_i is computed locally by party P_i on its input $\mathbf{x}_i$, randomness $\mathbf{r}_i$, and correlated randomness $\mathbf{r}'_i$. A semi-honest or malicious MPRE guarantees that to securely compute f, it suffices to securely compute $\hat{f}$ against semi-honest or malicious parties. That is, the following canonical protocol computing f in the $\hat{f}$-hybrid world is semi-honestly (resp. maliciously) secure. [5]

The canonical protocol for MPRE Party P_i has input $\mathbf{x}_i$ and correlated randomness $\mathbf{r}'_i$, samples local randomness $\mathbf{r}_i$, computes $\hat{\mathbf{x}}_i = h_i(\mathbf{x}_i, \mathbf{r}_i, \mathbf{r}'_i)$ and feeds $\hat{\mathbf{x}}_i$ to the funtionality $F_{\hat{f}}$ computing $\hat{f}$, so that every party learns $\hat{y} := \hat{f}(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n)$. Then every party outputs $y = \mathsf{Dec}(\hat{y})$.

[5] An equivalent definition of semi-honest MPRE can be found in [ABT18], in which it is just called "MPRE". In [ABT19], malicious MPRE is called "non-interactive reduction" and the canonical protocol of a MPRE is called "$\hat{f}$-oracle-aided protocol". Both [ABT18] and [ABT19] consider the honest majority setting, so they only require the canonical protocol to be secure against a bounded number of corruptions.

In other words, MPRE is a non-interactive reduction between MPC tasks. Thanks to the composition of MPC protocols, MPRE schemes naturally composes: Given an MPRE for f as described above, and another MPRE scheme for $\hat{f}$ with preprocessing functions $h'_1, \ldots, h'_n$, the encoding function $\hat{f}'$, their composition gives an MPRE for f with encoding function $\hat{f}'$ and preprocessing functions $h'_1 \circ h_1, \cdots, h'_n \circ h_n$. As such, as demonstrated in [ABT18,ABT19,LLW20], MPRE enables a modular approach for designing round-optimal MPC: To construct a round-optimal MPC protocol Π_f for computing f, the construction of [LLW20] proceeds in three steps:

- *Step 1: Degree 3 MPRE for circuits.* First, obtain a malicious MPRE for circuits, whose encoding function g has degree 3. It turns out that the classical degree 3 (centralized) randomized encoding, given by Yao's garbled circuits, is such a MPRE, where no local preprocessing is needed (i.e., h_i is the identity function). This has been implicitly observed and leveraged in many prior works, e.g., [BMR90,IK00,IK02,DI05,LLW20].
- *Step 2: Effective degree 2 MPRE for degree 3 Polynomials.* Then, design a MPRE of g whose encoding function $\hat{g}$ has degree 2. In this step, the preprocessing functions are non-trivial and such MPRE is said to have effective degree 2.
- *Step 3: 2-round MPC for degree 2 polynomials.* Finally, design a round-optimal MPC protocol $\Pi_{\hat{g}}$ for computing $\hat{g}$.

Composing the MPRE schemes from the first two steps gives an MPRE for circuit f with encoding function $\hat{g}$. The desired protocol Π_f is then obtained by instantiating the $\hat{g}$-oracle in the canonical protocol with $\Pi_{\hat{g}}$. Note that Π_f has the same communication complexity and round complexity as $\Pi_{\hat{g}}$. The contribution of [LLW20] lies in giving efficient instantiation of Step 2 and 3 in the OLE correlated randomness model over a field $\mathbb{F}$ of characteristic 2, and their final protocol Π_f has complexity $O(|f|n^3 \log |\mathbb{F}|)$. The main drawback is that their protocols are only semi-honest secure.

Semi-Malicious MPRE Our construction improves upon [LLW20] to achieve malicious security. Towards this, we introduce *semi-malicious* MPRE. As the name suggested, a MPRE of f is semi-maliciously secure if its canonical protocol is semi-maliciously secure, i.e., against adversaries who may choose arbitrary local randomness $\mathbf{r}_i$ and correlated randomness $\mathbf{r}'_i$ of corrupted parties, but computes the preprocessing functions correctly. Equivalently, semi-malicious MPRE means the following protocol is maliciously secure in the $\hat{f} \circ h$-hybrid world.

The canonical protocol for semi-malicious MPRE Party P_i has input $\mathbf{x}_i$, samples local randomness $\mathbf{r}_i$, and receives $\hat{\mathbf{r}}_i$, where $(\hat{\mathbf{r}}_1, \ldots, \hat{\mathbf{r}}_n)$ is the correlated randomness. Every P_i feeds $(\mathbf{x}_i, \mathbf{r}_i, \hat{\mathbf{r}}_i)$ to an oracle computing $\hat{f} \circ h$, so that every party learns

$$\hat{y} = (\hat{f} \circ h)(\mathbf{x}_1, \mathbf{r}_1, \hat{\mathbf{r}}_1, \ldots, \mathbf{x}_n, \mathbf{r}_n, \hat{\mathbf{r}}_n) := \hat{f}(h_1(\mathbf{x}_1, \mathbf{r}_1, \hat{\mathbf{r}}_1), \ldots, h_n(\mathbf{x}_n, \mathbf{r}_n, \hat{\mathbf{r}}_n)) \,.$$

Then every party outputs $y = \mathsf{Dec}(\hat{y})$.

Now, in order to construct 2-round malicious MPC protocol for general circuits, we modify the second and third steps above to the following

- *Step 2:* **Semi-malicious** *Effective degree 2 MPRE for degree 3 Polynomials.* Design a *semi-malicious* MPRE for any degree-3 function f, whose encoding function $\hat{f}$ has degree 2;
- *Step 3: 2-round MPC for* **effective** *degree 2 polynomials.* Construct 2-round *malicious* MPC for computing $\hat{f} \circ h$, which is an effective-degree-2 function.

Composing the above two steps gives a round-optimal maliciously secure MPC protocol for degree 3 functions (Lemma 1); using it to compute the degree 3 maliciously secure MPRE for circuit f from Step 1 gives a round-optimal maliciously secure MPC protocol for f.

Next, we outline our instantiation for Step 2 in Sec. 2.2 and that for Step 3 in Sec. 2.3. The entire roadmap is illustrated in Fig. 1.

2.2 Semi-Malicious Effective-Degree-2 MPRE

This section will outline the construction of semi-malicious effective-degree-2 MPRE for any degree-3 function. We start by "canonicalizing" the degree-3 function. A degree-3 polynomial can always be written as the sum of monomials $\sum_t c_t x_{t,1} x_{t,2} x_{t,3}$, where c_t is the constant coefficient of the t-th monomial. Let the party holding $x_{t,j}$ also sample random $z_{t,j}$, then

$$\Big(x_{t,1}x_{t,2}x_{t,3} + z_{t,1} + z_{t,2} + z_{z,3} \text{ for each } t,\ \sum\nolimits_t c_t(z_{t,1} + z_{t,2} + z_{z,3})\Big), \tag{1}$$

is (the encoding function of) a malicious MPRE for $\sum_t c_t x_{t,1} x_{t,2} x_{t,3}$, as shown in [BGI+18,GIS18,LLW20]. So it suffices to construct semi-malicious effective-degree-2 MPRE for (1). Here (1) is in what we call *canonical form*: Every coordinate of $\hat{f}$ either linear, or looks like $x_1x_2x_3 + z_1 + z_2 + z_3$.

As we are constructing *semi-malicious* MPRE, it is fine to construct MPRE for each coordinate of (1) separately then simply concatenate them together. That is, it suffices to consider the complete 3-party functionality

$$\mathsf{3MultPlus}((x_1, z_1), (x_2, z_2), (x_3, z_3)) := x_1x_2x_3 + z_1 + z_2 + z_3\,.$$

3MultPlus has a semi-honest effective-degree-2 MPRE (Fig. 2), which will be recalled in Sec. 4.1. For the overview, what matters is that

- This MPRE uses scalar OLE correlated randomness. That is, party P_1 receives $a_1, b_1 \in \mathbb{F}$, party P_2 receives a_2, b_2 such that a_1, a_2, b_1, b_2 are random subject to $a_1a_2 = b_1 + b_2$.
- This MPRE has perfect semi-honest security.

Our roadmap requires a semi-malicious effective-degree-2 MPRE for 3MultPlus. Semi-malicious security means the corrupted parties may arbitrarily choose their local randomness or modify the correlated randomness they received.

	Party P_1	Party P_2	Party P_3
Input:	x_1, z_1	x_2, z_2	x_3, z_3
Local Randomness:	$a_{4,1}, a_{5,1}$	$a_{4,2}, a_{5,2}$	a_3
Correlated Randomness:	a_1, b_1	a_2, b_2	
Output:	$\begin{bmatrix} x_1 - a_1 & \binom{a_3 x_1 + a_1 x_3}{-a_1 a_3 - a_4} & \binom{b_1 x_3 + b_2 x_3 + a_5 x_1 - a_1 a_5 +}{a_4 x_2 - a_2 a_4 + z_1 + z_2 + z_3} \\ -1 & x_3 - a_3 & a_2 x_3 - a_5 \\ & -1 & x_2 - a_2 \end{bmatrix}$ where $a_4 := a_{4,1} + a_{4,3}$ and $a_5 := a_{5,2} + a_{5,3}$.		

Fig. 2. Semi-honest MPRE for 3MultPlus [LLW20]

In standard model, semi-malicious security is implied by perfect semi-honest security, because conditional on any choice of corrupted parties' local randomness, the adversary has no advantage. But in the correlated randomness model, the semi-malicious adversary may lie about correlated randomness.

- For example, say P_1, P_2 are corrupted. If they feed a_1, b_1, a_2, b_2 as correlated randomness such that $a_1 a_2 = b_1 + b_2$, then the privacy still follows from the perfect semi-honest security. But if P_1, P_2 choose a_1, b_1, a_2, b_2 such that $a_1 a_2 \neq b_1 + b_2$, the privacy is lost (as we will show in Sec. 4.2).
- For another example, say P_1 is corrupted. If corrupted P_1 does not modify the portion of correlated randomness (a_1, b_2) it received, then the privacy still follows from the perfect semi-honest security. But if P_1 lies about (a_1, b_2), then with overwhelming probability $a_1 a_2 \neq b_1 + b_2$, and the privacy is lost (as we will show in Sec. 4.2).

In either case, if honest P_3 want to protect its privacy, it needs (and suffices) to ensure that $a_1 a_2 \neq b_1 + b_2$.

Our solution against this privacy threat is called *conditional disclosure of secrets (CDS) encoding*. Let party P_3 locally sample a random mask s. CDS encoding is a sub-module (of the final MPRE) that reveals s if and only if $a_1 a_2 \neq b_1 + b_2$. Then a candidate MPRE consists of two parts:

i) the semi-honest MPRE for 2MultPlus one-time padded by s;
ii) CDS encoding, which reveals s if and only if $a_1 a_2 = b_1 + b_2$.

But the CDS encoding resolves P_3's privacy concern at a cost: The adversary will learn a linear function in (a_1, b_1, a_2, b_2) if P_3 is corrupted. To overcome it, we have to replace the scalar OLE correlated randomness in the semi-honest MPRE by *tensor OLE correlated randomness*. That is, party P_1 receives vector $\mathbf{a}_1$, matrix $\mathbf{B}_1$; party P_2 receives vector $\mathbf{a}_2$, matrix $\mathbf{B}_2$; such that $\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2$ are random subject to $\mathbf{a}_1 \mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$. We abuse the notation and let a_1, b_1, a_2, b_2 denote the first coordinate of $\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2$ respectively. Then a_1, b_1, a_2, b_2 are random subject to $a_1 a_2 = b_1 + b_2$, i.e., their distribution is scalar OLE correlated randomness.

The next candidate MPRE is made up of:

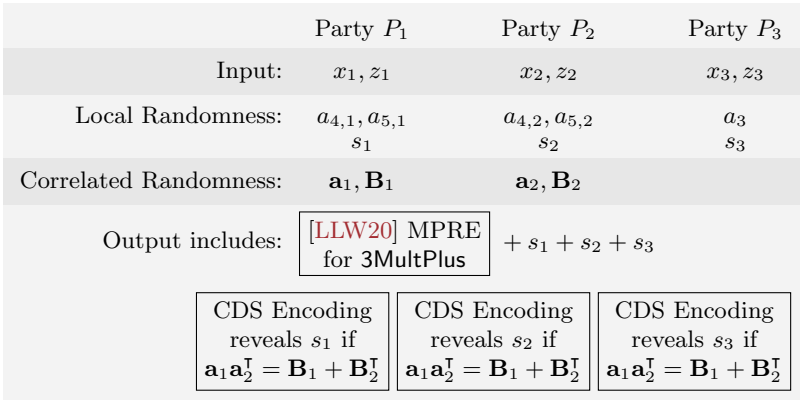

Fig. 3. Semi-Malicious MPRE for 3MultPlus

i) the semi-honest MPRE for 2MultPlus one-time padded by s;
ii) CDS encoding, which reveals s if and only if $\mathbf{a}_1\mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$.

Additionally, CDS encoding will let the adversary learn a linear leakage function in $(\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2)$ if P_3 is corrupted. But due to our careful design of CDS encoding, the leakage is one-time padded by the remaining coordinates of $\mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2$, so that no information about a_1, b_1, a_2, b_2 is revealed.

So far we focus on the security concern of P_3. Party P_1, P_2 have similar concern. So in the actual semi-malicious MPRE for 3MultPlus (shown in Fig. 3), every party P_i locally sample random mask s_i, The final MPRE consists of:

i) the semi-honest MPRE for 2MultPlus one-time padded by $s_1 + s_2 + s_3$;
ii) (for each $i \in \{1, 2, 3\}$) CDS encoding that reveals s_i iff $\mathbf{a}_1\mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$.

Of course, the three instances of CDS encoding are carefully designed, so that their leakages jointly reveal no information about a_1, b_1, a_2, b_2.

In the rest of the section, we explain how CDS encoding works. W.l.o.g., we assume the secret mask is sampled by P_3.

By our discussion so far, it seems that P_3 has to sample a random matrix s_3 in order to one-time pad the [LLW20] MPRE. But as we will discover in the technical body, it is sufficient to one-time pad only the top right coordinate of the [LLW20] MPRE. So s_3 is a random scalar sampled by P_3.

We start with a simpler task, P_3 only want to verify if P_1, P_2 use legit tensor OLE correlation. Here "legit" means in the support of the distribution. A simple encoding is to let P_3 additionally sample random vectors $\mathbf{q}_1, \mathbf{q}_2$, and the encoding outputs

$$p_1 = \langle \mathbf{a}_1, \mathbf{q}_1 \rangle, \quad p_2 = \langle \mathbf{a}_2, \mathbf{q}_2 \rangle, \quad p_3 = \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \mathbf{q}_1\mathbf{q}_2^\intercal \rangle \,. \tag{2}$$

Then P_3 can check whether $p_1p_2 = p_3$. If $\mathbf{a}_1\mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$, then $p_1p_2 = p_3$ always holds. Otherwise, $p_1p_2 \neq p_3$ with overwhelming probability. Note that the encoding in (2) is of effective degree 2, because P_3 can locally compute $\mathbf{q}_1\mathbf{q}_2^\intercal$.

In the CDS encoding, party P_3 additionally samples random r_1, r_2. The CDS encoding outputs

$$p_1, \quad p_2, \quad p_3, \quad \text{and} \quad \mathbf{c} := \begin{bmatrix} 1 & p_2 \\ p_1 & p_3 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} + \begin{bmatrix} 0 \\ s_3 \end{bmatrix}.$$

If $\mathbf{a}_1\mathbf{a}_2^\intercal \neq \mathbf{B}_1 + \mathbf{B}_2^\intercal$ then $\left[\begin{smallmatrix} 1 & p_2 \\ p_1 & p_3 \end{smallmatrix}\right]$ has full-rank with overwhelming probability, and $\mathbf{c}$ is one-time padded by (r_1, r_2), thus no information about s is revealed. Otherwise when $\mathbf{a}_1\mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$, it is easy to verify that $\langle(-p_1, 1), \mathbf{c}\rangle = s$. Note that CDS encoding is of effective degree 2, because it is a linear function in $\mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2$, whose coefficient can be locally computed by P_3.

If P_3 is corrupted, the adversary chooses $\mathbf{q}_1, \mathbf{q}_2$ and learns p_1, p_2, p_3 (defined by (2)) as leakage. By enforcing some constraints on $\mathbf{q}_1, \mathbf{q}_2$ (will be discussed in the main body), the leakage will not reveal any information about a_1, b_1, a_2, b_2.

2.3 MPC for Effective-Degree-2 Functions

Given an effective-degree-2 function $g = \hat{f} \circ h$

$$g(\mathbf{x}_1, \ldots, \mathbf{x}_n) := \hat{f}(h_1(\mathbf{x}_1), \ldots, h_n(\mathbf{x}_n)),$$

we can assume w.l.o.g. that each coordinate of $\hat{f}$ has the canonical form[6]

$$\hat{f}_t(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n) = \mathsf{2MultPlus}(\underbrace{(x_1, z_1)}_{\text{owned by } P_{i_t}}, \underbrace{(x_2, z_2)}_{\text{owned by } P_{j_t}}) = x_1x_2 + z_1 + z_2$$

where x_1, z_1 are two coordinates of $\hat{\mathbf{x}}_{i_t}$ and x_2, z_2 are two coordinates of $\hat{\mathbf{x}}_{j_t}$.

As presented by [LLW20], there is a 2-round semi-honest MPC protocol for $\mathsf{2MultPlus}$ that uses scalar OLE correlated randomness (Fig. 4). Via parallel repetition, it implies 2-round semi-honest MPC for any effective-degree-2 functions that uses scalar OLE correlated randomness.

Towards malicious security, the starting point is the observation that the protocol for $\mathsf{2MultPlus}$ in Fig. 4 is maliciously secure *with output substitution*. Security with output substitution means the adversary, after learning the output y, may adaptively choose y' so that all honest parties (unanimously) output y'.

A natural idea is to compute all $\hat{f}_t$ using parallel sessions of the protocol in Fig. 4. Each session computes one coordinate. But parallelization does not meet the security requirement, because of the following two issues:

Consistency. Say two coordinates of $\hat{f}$ equal $x_1x_2 + z_1 + z_2$ and $x_1x_3 + z_1' + z_3$ respectively, where party P_1 owns x_1, z_1, z_1', party P_2 owns x_2, z_2, party P_3 owns x_3, z_3. We have to check whether P_1 feeds the *same* x_1 to the two corresponding sessions.

[6] Sec. 2.2 outlines how to canonicalize $\hat{f}$. Formally, canonical form allows some coordinates to be linear instead of $\mathsf{2MultPlus}$. The linear coordinates are easier to handle. We ignore them in the overview.

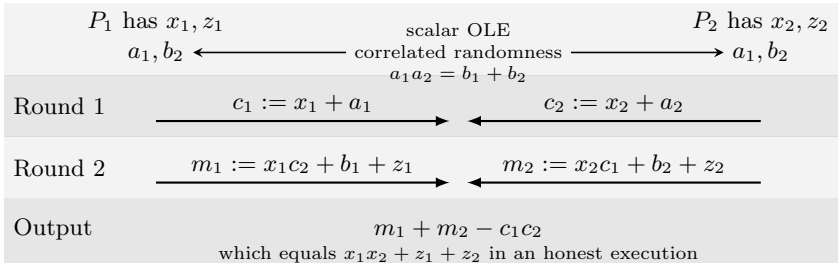

Fig. 4. 2-round semi-honest MPC for 2MultPlus

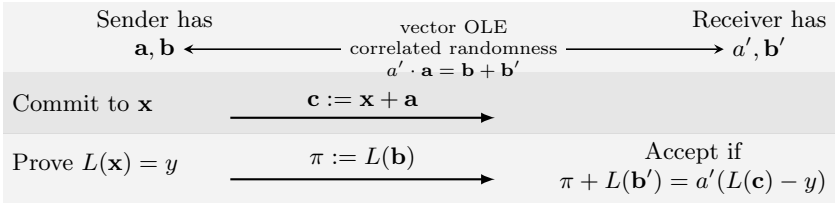

Fig. 5. commit-and-prove-linear

Well-Formedness. We have to check whether P_i computes $\hat{\mathbf{x}}_i = h_i(\mathbf{x}_i)$ correctly. Note that we can also assume the preprocessing functions h_i is a small arithmetic circuit that only contains multiplication gates, because such property is satisfied by our MPRE.

To resolve the well formedness issue, we introduce a commit-and-prove-linear scheme which uses vector OLE correlated randomness. It enables the sender to commit to a vector $\mathbf{x}$, then later prove $L(\mathbf{x}) = y$ for any linear function L.

As shown in Fig. 5, the scheme uses vector OLE correlated randomness between the sender and the receiver. That is, the sender receives random vectors $\mathbf{a}, \mathbf{b}$, the receiver recovers random $a', \mathbf{b}'$ subject to $a'\mathbf{a} = \mathbf{b} + \mathbf{b}'$. To commit to a vector $\mathbf{x}$, the sender simply sends $\mathbf{c} := \mathbf{x} + \mathbf{a}$ to the receiver. Later, for any linear function L, the sender can prove $L(\mathbf{x}) = y$ by sending $\pi := L(\mathbf{b})$ to the receiver; and the receiver accepts the proof iff $\pi + L(\mathbf{b}') = a'(L(\mathbf{c}) - y)$. Such scheme has

Completeness An honest proof $\pi := L(\mathbf{b})$ will always be accepted because $\pi + L(\mathbf{b}') = L(\mathbf{b}) + L(\mathbf{b}') = a' \cdot L(\mathbf{a}) = a'(L(\mathbf{c}) - L(\mathbf{x})) = a'(L(\mathbf{c}) - y)$.

Statistical Soundness If $L(\mathbf{x}) \neq y$, any proof will be reject with overwhelming probability due to the randomness of a'.

Zero-knowledge No information about $\mathbf{x}$, other than $L(\mathbf{x})$, will be revealed to receiver, since the receiver can predict π from $\mathbf{c}, L, y$.

Reusability Given a commitment $\mathbf{c}$, the sender can prove multiple adaptively chosen linear statements $L_1(\mathbf{x}) = y_1, \ldots, L_t(\mathbf{x}) = y_t$ about the underlying message $\mathbf{x}$. The statistical soundness error is $O(t/|\mathbb{F}|)$. No information about $\mathbf{x}$ other than $L_1(\mathbf{x}), \ldots, L_t(\mathbf{x})$ will be revealed.

Then, inspired by a linear PCP scheme from [BCI+13], the sender can generate zero-knowledge proofs of more complex arithmetic statements. A particular interesting statement for ours is multiplication. Say the sender will commit to message $\mathbf{x} = (x_1, x_2, x_3, \dots)$, and then prove $x_1 x_2 = x_3$ to the receiver. We design **ProveProd** sub-protocol for such demand:

1. In order to prove $x_1 x_2 = x_3$, the sender samples random g_1, g_2 and extends its message into
$$\mathbf{x} = (x_1, g_1, x_2, g_2, x_3, x_1 g_2, x_2 g_1, g_1 g_2, \dots) .$$
The dimension of the correlated randomness should be extended correspondingly. The sender commits to the extended $\mathbf{x}$ instead.
2. Random challenges q_1, q_2 are sampled.
3. The sender proves to the receiver that
$$\langle (q_1, 1), (\underset{\uparrow}{\mathbf{x}[1]}, \underset{(x_1, g_1)}{\mathbf{x}[2]}) \rangle = p_1, \quad \langle (q_2, 1), (\underset{\uparrow}{\mathbf{x}[3]}, \underset{(x_2, g_2)}{\mathbf{x}[4]}) \rangle = p_2,$$
$$\langle (q_1 q_2, q_1, q_2, 1), (\mathbf{x}[5], \mathbf{x}[6], \underset{\uparrow}{\mathbf{x}[7]}, \underset{(x_3, x_1 g_2, x_2 g_1, g_1 g_2)}{\mathbf{x}[8]}) \rangle = p_3 .$$
The receiver accepts if $p_1 p_2 = p_3$.

Similar to the discussion in our CDS encoding, if $\overset{x_1}{\underset{\downarrow}{\mathbf{x}[1]}} \cdot \overset{x_2}{\underset{\downarrow}{\mathbf{x}[3]}} \neq \overset{x_3}{\underset{\downarrow}{\mathbf{x}[5]}}$, then $p_1 p_2 \neq p_3$ with overwhelming probability due to the randomness of $q_1 q_2$. No information about x_1, x_2, x_3 are leaked if the sender is proving a true statement, as the leakage p_1, p_2 are one-time padded by g_1, g_2, and p_3 is determined by $p_3 = p_1 p_2$.

If the random challenges q_1, q_2 are sampled by the receiver, the proof will have $1/|\mathbb{F}|$ soundness error. To make the proof non-interactive, the challenges can be sampled by the random oracle, so that the soundness error becomes the number of adversary's query to random oracle divided by $|\mathbb{F}|$.

So far we can prove multiplication relation for 3 values behind a commitment. This will resolve the concern of well-formedness. To resolve the concern of consistency, we need to prove that values behind different commitments are the same. That is, say the sender will commit to messages $\mathbf{x}_1 = (x_1, \dots)$, $\mathbf{x}_2 = (x_2, \dots)$, and want to convince the receiver that $x_1 = x_2$. We design **ProveSame** sub-protocol for such proof:

1. In order to prove $x_1 = x_2$, the sender samples random g and extends its message into
$$\mathbf{x}_1 = (x_1, g, \dots), \quad \mathbf{x}_2 = (x_2, g, \dots) .$$
The sender commits to the extended $\mathbf{x}_1, \mathbf{x}_2$ instead.
2. A random challenge q is sampled.
3. The sender proves to the receiver that
$$\langle (q, 1), (\underset{\uparrow}{\mathbf{x}_1[1]}, \underset{(x_1, g)}{\mathbf{x}_1[2]}) \rangle = p_1, \quad \langle (q, 1), (\underset{\uparrow}{\mathbf{x}_2[1]}, \underset{(x_2, g)}{\mathbf{x}_2[2]}) \rangle = p_2 .$$
The receiver accepts if $p_1 = p_2$.

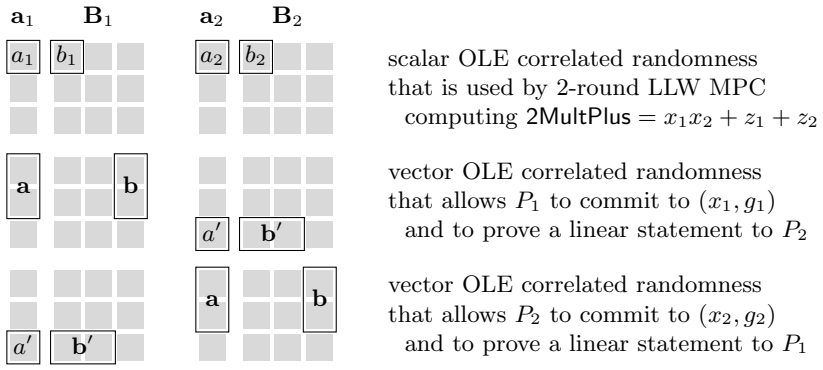

Fig. 6. Multiple roles of $\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2$

Similarly, when $\mathbf{x}_1\overset{x_1}{\downarrow}[1] \neq \mathbf{x}_2\overset{x_2}{\downarrow}[1]$, the probability $p_1 = p_2$ is no more than $1/|\mathbb{F}|$, due to the randomness of challenge q. No information about x_1, x_2 are leaked if the sender is proving a true statement, as the leakage p_1 is one-time padded by g, and p_2 is determined by $p_2 = p_1$.

Putting pieces together. We develop the natural idea of using parallel sessions of the protocol for 2MultPlus. Each coordinate of $\hat{f}$ will be computed by a modified version of the 2-round LLW protocol for 2MultPlus (Fig. 4).

In the first round of the LLW protocol for 2MultPlus, party P_1 sends $c_1 := x_1 + a_1$, party P_2 sends $c_2 := x_2 + a_2$, where x_1, x_2 are their inputs, and a_1, a_2 are parts of the scalar OLE correlated randomness. Here c_i is essentially a commitment to x_i.

In the modified LLW protocol for 2MultPlus, the two parties use 3×3 tensor OLE correlated randomness. That is, party P_1 receives $\mathbf{a}_1 \in \mathbb{F}^3, \mathbf{B}_1 \in \mathbb{F}^{3\times 3}$, party P_2 receives $\mathbf{a}_2 \in \mathbb{F}^3, \mathbf{B}_2 \in \mathbb{F}^{3\times 3}$, where $\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2$ are random subject to $\mathbf{a}_1\mathbf{a}_2^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$.[7] In the first round, P_1 broadcasts

$$\mathbf{c}_1 := (x_1, g_1) + \mathbf{a}_1[1{:}2],$$

where $\mathbf{a}_1[1{:}2]$ denotes the first two coordinates of $\mathbf{a}_1$. The choice of g_1 will be discussed later. Symmetrically, party P_2 broadcasts $\mathbf{c}_2 := (x_2, g_2) + \mathbf{a}_2[1{:}2]$. The two commitments $\mathbf{c}_1, \mathbf{c}_2$ will have multiple roles:

Compute 2MultPlus**.** Their first coordinates $\mathbf{c}_1[1], \mathbf{c}_2[1]$ are of the same form as the first message in the original LLW protocol. In the second round, P_1, P_2 will proceed with the original LLW protocol by taking $\mathbf{c}_1[1], \mathbf{c}_2[1]$ as the first round messages.

P_1 **proves consistency.** Party P_1 may need to prove that the same x_1 is used across different sessions. Note that $\mathbf{c}$ can be viewed as a commitment in

[7] Note the transpose of $\mathbf{B}_2$. This makes the equation remains unchanged upon exchanging subscripts.

the commit-and-prove-linear scheme. P_1 takes $\mathbf{a}_1[1{:}2], \mathbf{B}_1[1{:}2, 3]$ and P_2 takes $\mathbf{a}_2[3], \mathbf{B}_2[3, 1{:}2]$ as the vector OLE correlated randomness (as shown in Fig. 6). Then the ProveSame sub-protocol can prove P_1 is using a consistent value. P_1 sets g_1 according to the sub-protocol.

P_2 **proves consistency.** Symmetric to the above.

To resolve the well-formedness concern, every party has to prove that it honestly evaluates the preprocessing function. Since the preprocessing function only has multiplication gates, a party can prove well-formedness by using the ProveProd sub-protocol. (Together with ProveSame sub-protocol. Because once a party proves $x_1 x_2 = x_3$ using ProveProd, it also need to prove that the "x_1" in ProveProd is the same the "x_1" it uses in $\mathsf{2MultPlus}$ sessions.)

2.4 Lift Security with Output Substitution

Security with output substitution is a weak notion of security. The adversary, after learning the function output y, may adaptively choose y' so that all honest parties (unanimously) output y'. Once we constructed MPC for $\mathbf{P/poly}$ that is secure with output substitution against malicious corruptions, we can lift its security to the standard notion of security with (unanimous) abort, with the help of *consensus MAC*, which will be introduced in the full version. We claim that the following protocol is secure with abort.

1. Party P_i has input x_i, and samples MAC key k_i.
2. Using a protocol that is secure with output substitution to compute
$$(y, \sigma) := f(x_1, \ldots, x_n), \mathsf{Sign}_{k_1,\ldots,k_n}(f(x_1, \ldots, x_n)) .$$
3. Party P_i outputs y if π is a valid signature on y w.r.t. key k_i; aborts otherwise.

As the protocol suggested, consensus MAC has i) a signing algorithm Sign, which takes a message, n keys, generates a signature; and ii) a verification algorithm Verify, which takes a message, a signature and a key, outputs "accept" or "reject".

In the protocol, the adversary, after learning (y, σ), may adaptively replace the output by $(y', \sigma') \neq (y, \sigma)$. In order to achieve security with unanimous abort, all honest parties should reject (y', σ'). This hints how to define consensus MAC: the adversary wins the following game with negligible probability.

1. The adversary chooses the set of corrupted parties $C \subseteq [n]$, and chooses key k_i for each corrupted party $P_i \in C$. Every honest party $P_i \notin C$ samples k_i.
2. The adversary chooses message y and learns $\pi = \mathsf{Sign}_{k_1,\ldots,k_n}(y)$.
3. The adversary adaptively chooses $(y', \pi') \neq (y, \pi)$.
4. The adversary wins if $\mathsf{Verify}_{k_i}(y', \pi') \to$ accept for some honest party $P_i \notin C$.

Here is a simple consensus MAC scheme, whose security will be proven in the full version. Party P_i samples two random number a_i, b_i and let key $k_i := (a_i, b_i)$. The signature $\mathsf{Sign}_{k_1,\ldots,k_n}(y)$ is the degree-n polynomial π such that
$$\pi(0) = y, \quad \pi(a_1) = b_1, \quad \ldots, \quad \pi(a_n) = b_n .$$

The verification accepts a message-signature pair (y, π) w.r.t. key $k_i = (a_i, b_i)$ if and only if $\pi(0) = y$ and $\pi(a_i) = b_i$.

2.5 Tensor OLE Correlated Randomness Generation from OT

We require tensor OLE correlated randomness over a large boolean extension field $\mathbb{F} = \mathsf{GF}(2^\lambda)$, where λ is the security parameter. To generate OLE correlated randomness, it suffices to implement a protocol computing tensor OLE

$$\mathsf{TOLE}((\mathbf{a}, \mathbf{B}), \mathbf{x}) := \mathbf{a}\mathbf{x}^\intercal + \mathbf{B}$$

or random tensor OLE, and later randomize the input-output tuple.

The starting point is a semi-honest protocol [Gil99]. Say the sender has vector $\mathbf{a}$, matrix $\mathbf{B}$; the receiver has vector $\mathbf{x}$ and should learn $\mathbf{Y} = \mathbf{a}\mathbf{x}^\intercal + \mathbf{B}$. Note that $\mathbf{a}\mathbf{x}^\intercal + \mathbf{B}$, as a function in $\mathbf{x}$, is affine over $\mathsf{GF}(2)$. That is, if we let subscript $_{(2)}$ denote bit representations, there exists a binary matrix $M_{\mathbf{a}}$ such that $(\mathbf{Y})_{(2)} = (\mathbf{a}\mathbf{x}^\intercal + \mathbf{B})_{(2)} = M_{\mathbf{a}} \cdot (\mathbf{x})_{(2)} + (\mathbf{B})_{(2)}$. Thus the receiver can learn $\mathbf{Y}$ from OT.

Such protocol is not secure against a malicious sender, who can choose an arbitrary M_1 and let the receiver learn $(\mathbf{Y})_{(2)} = M_1 \cdot (\mathbf{x})_{(2)} + (\mathbf{B})_{(2)}$. To detect malicious behaviour, the receiver samples a random matrix $\mathbf{Y}^{\text{shadow}}$ and let the sender learn $\mathbf{B}^{\text{shadow}} = \mathbf{a}\mathbf{x}^\intercal + \mathbf{Y}^{\text{shadow}}$. Formally, the sender learns

$$(\mathbf{B}^{\text{shadow}})_{(2)} = M_2 \cdot (\mathbf{a})_{(2)} + (\mathbf{Y}^{\text{shadow}})_{(2)}$$

from OT, where $M_2 = M_{\mathbf{x}}$ if the receiver is honest. The receiver samples a random matrix H over $\mathsf{GF}(2)$, sends H to the sender as a challenge, and asks the sender to guess $H \cdot (\mathbf{Y} + \mathbf{Y}^{\text{shadow}})_{(2)}$. Note that, if both parties are honest, $H \cdot (\mathbf{Y} + \mathbf{Y}^{\text{shadow}})_{(2)} = H \cdot (\mathbf{B} + \mathbf{B}^{\text{shadow}})_{(2)}$. If the sender is corrupted,

$$\begin{aligned} H \cdot (\mathbf{Y} + \mathbf{Y}^{\text{shadow}})_{(2)} &= H \cdot \left(M_1 \cdot (\mathbf{x})_{(2)} + (\mathbf{B})_{(2)} + M_{\mathbf{a}} \cdot (\mathbf{x})_{(2)} + (\mathbf{B}^{\text{shadow}})_{(2)} \right) \\ &= H \cdot (M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)} + H \cdot (\mathbf{B} + \mathbf{B}^{\text{shadow}})_{(2)} \,. \end{aligned}$$

Thus the sender will not be caught if and only if he can guess $H \cdot (M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)}$ correctly. Let H has λ rows and assume w.l.o.g. that the receiver samples $\mathbf{x}$ at random. Then the sender will be caught will overwhelming probability if $\text{rank}(M_1 + M_{\mathbf{a}}) \geq \lambda$. Because in such case, $H \cdot (M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)}$ has large entropy conditioning on the sender's knowledge.

If $\text{rank}(M_1 + M_{\mathbf{a}}) < \lambda$, say we give $(M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)}$ to the corrupted sender, this may only help the adversary. The corrupted sender can compute $(\mathbf{B}')_{(2)} = (M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)} + (\mathbf{B})_{(2)}$. (Honest sender simply let $\mathbf{B}' = \mathbf{B}$.) Note that $\mathbf{Y} = \mathbf{a}\mathbf{x}^\intercal + \mathbf{B}'$. The sender and receiver output $(\mathbf{a}, \mathbf{B}'), (\mathbf{x}, \mathbf{Y})$ respectively.

Such a protocol is insecure, for two reasons.

- If the sender is corrupted, he additionally knows $(M_1 + M_{\mathbf{a}}) \cdot (\mathbf{x})_{(2)}$, which is an at most λ-bit leakage of $\mathbf{x}$. If the receiver is corrupted, she additionally knows
$$H \cdot (\mathbf{B} + \mathbf{B}^{\text{shadow}})_{(2)} := H \cdot (M_2 + M_{\mathbf{b}}) \cdot (\mathbf{a})_{(2)},$$

which is an at most λ-bit leakage of $\mathbf{a}$. The leakage can be removed by using randomness extractor and left-over hash lemma.
- The corrupted sender (resp. receiver) can arbitrarily choose $\mathbf{a}, \mathbf{B}$ (resp. $\mathbf{x}$). To ensure randomness, an additional re-randomization step is needed.

3 Definition of Multi-Party Randomized Encoding

Definition 1 (Multi-Party Randomized Encoding). *Let $f : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}$ be some n-party function. A* multi-party randomized encoding *(MPRE) of f is specified by*

- *Local randomness space $\mathcal{R}_i$ for $i \in [n]$. Correlated randomness space $\mathcal{R}'_1 \times \cdots \times \mathcal{R}'_n$ together with a distribution $\mathcal{D}$ over it.*
- *Local preprocessing function $h_i : \mathcal{X}_i \times \mathcal{R}_i \times \mathcal{R}'_i \to \hat{\mathcal{X}}_i$.*
- *Encoding function $\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \to \hat{\mathcal{Y}}$.*
- *Decoding function $\mathsf{Dec} : \hat{\mathcal{Y}} \to \mathcal{Y}$.*

Such that for any input $(x_1, \ldots, x_n)$, the encoding

$$\hat{y} := \hat{f}\big(h_1(x_1, r_1, r'_1), \ldots, h_n(x_n, r_n, r'_n)\big) \tag{3}$$

represents $y = f(x_1, \ldots, x_n)$ in the following sense:

Correctness *For any input $(x_1, \ldots, x_n) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, randomness $(r_1, \ldots, r_n) \in \mathcal{R}_1 \times \cdots \times \mathcal{R}_n$ and correlated randomness $(r'_1, \ldots, r'_n)$ in the support of $\mathcal{D}$, the corresponding encoding $\hat{y}$ defined by (3) satisfies that $f(x_1, \ldots, x_n) = \mathsf{Dec}(\hat{y})$.*

Semi-Malicious Security *We say MPRE is computationally (resp. statistically) semi-malicious secure with output substitution, if the following canonical protocol computationally (resp. statistically) securely implements $\mathcal{F}^{\mathsf{os}}_f$:*

- *The protocol assumes ideal functionality $\mathcal{F}^{\mathsf{os}}_{\hat{f} \circ h}$.*
- *On input x_i, party P_i samples $r_i \leftarrow \mathcal{R}_i$ locally, receives r'_i where $(r'_1, \ldots, r'_n)$ is sampled from $\mathcal{D}$, then inputs (x_i, r_i, r'_i) to the ideal functionality $\mathcal{F}^{\mathsf{os}}_{\hat{f} \circ h}$. Note that a corrupted party may adaptively modify its input of $\mathcal{F}^{\mathsf{os}}_{\hat{f} \circ h}$.*
- *Upon receiving $\hat{y}$ from $\mathcal{F}^{\mathsf{os}}_{\hat{f} \circ h}$, party decodes $y = \mathsf{Dec}(\hat{y})$, and outputs y.*

Malicious Security *We say MPRE is computationally (resp. statistically) malicious secure with output substitution, if the following canonical protocol computationally (resp. statistically) securely implements $\mathcal{F}^{\mathsf{os}}_f$:*

- *The protocol assumes ideal functionality $\mathcal{F}^{\mathsf{os}}_{\hat{f}}$.*
- *On input x_i, party P_i samples $r_i \leftarrow \mathcal{R}_i$ locally, receives r'_i where $(r'_1, \ldots, r'_n)$ is sampled from $\mathcal{D}$, and locally computes $\hat{x}_i = h_i(x_i, r_i, r'_i)$ then inputs $\hat{x}_i$ to the ideal functionality $\mathcal{F}^{\mathsf{os}}_{\hat{f}}$.*
- *Upon receiving $\hat{y}$ from $\mathcal{F}^{\mathsf{os}}_{\hat{f}}$, party decodes $y = \mathsf{Dec}(\hat{y})$, and outputs y.*

The Real Execution $\mathsf{Exec}_{C,\mathcal{A},\mathcal{Z}}(1^\lambda, \Pi)$	The Ideal Execution $\mathsf{Ideal}_{C,\mathcal{A},\mathcal{S},\mathcal{Z}}(1^\lambda, \mathcal{F}_f^{\mathsf{os}})$
1. The adversary $\mathcal{A}$ receives the corrupted parties' portions of correlated randomness $\{r'_i\}_{i\in C}$.	1. The simulator $\mathcal{S}$ receives the corrupted parties' portions of correlated randomness $\{r'_i\}_{i\in C}$, and forwards them to $\mathcal{A}$.
2. The environment $\mathcal{Z}$ chooses the input x_i for each honest party $P_i \notin C$. Every honest P_i samples local randomness r_i, receives his portion of correlated randomness r'_i and sends (x_i, r_i, r'_i) to $\mathcal{F}_{\hat{f}\circ h}^{\mathsf{os}}$.	2. The environment $\mathcal{Z}$ chooses the input x_i for each honest party $P_i \notin C$. Every honest dummy P_i directly inputs x_i to $\mathcal{F}_f^{\mathsf{os}}$.
3. A corrupted party $P_i \in C$ may input any $(\bar{x}_i, \bar{r}_i, \bar{r}'_i)$ to $\mathcal{F}_{\hat{f}\circ h}^{\mathsf{os}}$.	3. Any corrupted party's input to $\mathcal{F}_{\hat{f}\circ h}^{\mathsf{os}}$ is hijacked by $\mathcal{S}$. The simulator $\mathcal{S}$ extracts input x_i for each $i \in C$, and sends x_i to $\mathcal{F}_f^{\mathsf{os}}$ on behalf of dummy P_i.
4. $\mathcal{F}_{\hat{f}\circ h}^{\mathsf{os}}$ sends the output $\hat{y}$ to $\mathcal{A}$. The adversary $\mathcal{A}$ chooses and sends $\hat{y}'$ back.	4. $\mathcal{F}_f^{\mathsf{os}}$ sends the output y to $\mathcal{S}$, who generates and sends $\hat{y}$ to $\mathcal{A}$. The adversary $\mathcal{A}$ chooses and sends $\hat{y}'$ back to $\mathcal{S}$, who sends $y' = \mathsf{Dec}(\hat{y}')$ to $\mathcal{F}_f^{\mathsf{os}}$.
5. Every honest party receives $\hat{y}'$ from $\mathcal{F}_{\hat{f}\circ h}^{\mathsf{os}}$, and output $y' = \mathsf{Dec}(\hat{y}')$ to $\mathcal{Z}$.	5. Every honest dummy party receives y' from $\mathcal{F}_f^{\mathsf{os}}$, and output y' to $\mathcal{Z}$.

Fig. 7. The Security Game of Semi-Malicious MPRE.

We expand the definition of semi-malicious MPRE in more detail by describing the ideal world and real world of the security game of the canonical protocol (Fig. 7). Apparently, the adversary and the environment learn no information during the last two steps in the security game.

Effective Degree. By definition, the task of computing f against malicious corruptions is reduced to the task of computing $\hat{f}$, if MPRE is maliciously secure. To minimize the round complexity for computing $\hat{f}$, the classical approach is to reduce the arithmetic degree of $\hat{f}$. The degree of $\hat{f}$ is called the *effective degree* of this MPRE.

Formally, let $\mathbb{F}$ be a finite field. Let

$$\left(\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \to \hat{\mathcal{Y}}, h_1, \ldots, h_n\right)$$

be a MPRE for f, such that $\hat{\mathcal{X}}_1, \ldots, \hat{\mathcal{X}}_n, \hat{\mathcal{Y}}$ are vector spaces over field $\mathbb{F}$. The arithmetic degree of $\hat{f}$ over $\mathbb{F}$ is called the effective degree of the MPRE.

Arithmetic Preprocessing. By definition, the task of computing f against malicious corruptions is reduced to computing $\hat{f} \circ h$ if MPRE is semi-maliciously secure. To minimize the round complexity for computing $\hat{f} \circ h$, besides reducing the degree of $\hat{f}$, we also need the preprocessing functions $h_1, \ldots, h_n$ to be computable by poly-size arithmetic circuits.

Formally, let $(\hat{f}, h_1, \ldots, h_n)$ be a MPRE for f, who has low effective degree over a field $\mathbb{F}$. The MPRE has arithmetic preprocessing if its input spaces, local

randomness spaces and correlated randomness spaces are vector spaces over $\mathbb{F}$, and the local preprocessing functions $h_1, \ldots, h_n$ are computed by poly-size arithmetic circuits over $\mathbb{F}$.

If a MPRE has effective degree 2 and arithmetic preprocessing over $\mathbb{F}$, we say $\hat{f} \circ g$ is an *effective-degree-2 function* over $\mathbb{F}$.

Definition 2 (Effective-Degree-2 Function). *A function $g : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}$ is of* effective degree 2 *over a field $\mathbb{F}$, if $\mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{Y}$ are vector spaces over $\mathbb{F}$ and there exist*

- *$h_i : \mathcal{X}_i \to \hat{\mathcal{X}}_i$, an arithmetic circuit over $\mathbb{F}$, for each $i \le n$.*
- *$\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \to \mathcal{Y}$, a degree-2 arithmetic function over $\mathbb{F}$.*

such that for all $(x_1, \ldots, x_n) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$,

$$\hat{f}(h_1(x_1), \ldots, h_n(x_n)) = g(x_1, \ldots, x_n).$$

4 MPRE for Degree-3 Functions

Let λ be the security parameter. All objects implicitly depend on λ. Most objects in this section is arithmetic over a finite field $\mathbb{F} = \mathbb{F}(\lambda)$, such that $|\mathbb{F}| = \Omega(2^\lambda)$.

Canonical Form Polynomials. Before we construct MPRE for degree-3 functions, we observe that it is w.l.o.g. to assume the degree-3 function f is of the following canonical form.

For any **canonical degree-3 function** $f : \mathbb{F}^{\ell_1} \times \cdots \times \mathbb{F}^{\ell_n} \to \mathbb{F}^\ell$, there is an index set $\mathcal{I} \subseteq [\ell]$, such that for each $t \in [\ell]$,

- If $t \notin \mathcal{I}$, the t-th coordinate of f, denoted by f_t, is of degree at most 2.
- If $t \in \mathcal{I}$, then $f_t = x_1 x_2 x_3 + z_1 + z_2 + z_3$ where x_i, z_i are from the same party. More formally, let $\mathbf{x}_i \in \mathbb{F}^{\ell_i}$ denote the i-th input of f, then

  $$f_t(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \mathbf{x}_{i_{t,1}}[j_{t,1}] \cdot \mathbf{x}_{i_{t,2}}[j_{t,2}] \cdot \mathbf{x}_{i_{t,3}}[j_{t,3}] + \mathbf{x}_{i_{t,1}}[j'_{t,1}] + \mathbf{x}_{i_{t,2}}[j'_{t,2}] + \mathbf{x}_{i_{t,3}}[j'_{t,3}]$$

  for some $i_{t,1}, i_{t,2}, i_{t,3} \in [n]$ and $j_{t,1}, j'_{t,1} \in [\ell_{i_{t,1}}]$, $\ldots$, $j_{t,3}, j'_{t,3} \in [\ell_{i_{t,3}}]$.

We assume w.l.o.g. that f is a canonical degree-3 function. It is known in the literature that (e.g. shown by [BGI+18,GIS18,LLW20]) every degree-3 function f has a semi-honest MPRE whose encoding function is canonical. The MPRE does not use correlated randomness, and is perfectly secure, thus it is also semi-maliciously secure. The MPRE does not has preprocessing (preprocessing functions are identity functions), thus semi-malicious security implies malicious security.

Moreover, such canonicalization does not increase complexity. Remind that the complexity measure we care about is its total number of monomials $\mathsf{mc}(f)$. It is not difficult to show that $\mathsf{mc}(\hat{f}) = O(\mathsf{mc}(f))$, where $\hat{f}$ is the encoding function of the MPRE for f we just discussed.

4.1 Background: Semi-honest MPRE for Degree-3 Functions

Due to canonicalization, it is sufficient to consider the minimal complete function

$$\mathsf{3MultPlus}\Big((x_1,z_1),(x_2,z_2),(x_3,z_3)\Big) = x_1x_2x_3 + z_1 + z_2 + z_3.$$

It only involves three parties P_1, P_2, P_3. Party P_i has input $x_i, z_i \in \mathbb{F}$. The output can also be presented as a $\mathbb{F}$-modular branching program:

$$x_1x_2x_3 + z_1 + z_2 + z_3 = \det \begin{bmatrix} x_1 & z_1+z_2+z_3 \\ -1 & x_3 & \\ & -1 & x_2 \end{bmatrix}$$

As shown by AIK, it has a degree-3 random encoding

$$\begin{bmatrix} 1 & a_1 & a_4 \\ & 1 & \\ & & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 & z_1+z_2+z_3 \\ -1 & x_3 & \\ & -1 & x_2 \end{bmatrix} \cdot \begin{bmatrix} 1 & a_3 & a_5 \\ & 1 & a_2 \\ & & 1 \end{bmatrix}$$
$$= \begin{bmatrix} x_1 - a_1 & \begin{pmatrix} a_3x_1 + a_1x_3 \\ -a_1a_3 - a_4 \end{pmatrix} & \begin{pmatrix} \boxed{a_1a_2x_3} + a_5x_1 - a_1a_5 + \\ a_4x_2 - a_2a_4 + z_1 + z_2 + z_3 \end{pmatrix} \\ -1 & x_3 - a_3 & a_2x_3 - a_5 \\ & -1 & x_2 - a_2 \end{bmatrix},$$

where $a_1, \ldots, a_5 \in \mathbb{F}$ are the randomness of the encoding.

Notice that the randomized encoding only has one degree-3 monomial term $a_1a_2x_3$ (highlighted by a box). Assume a_1, a_2 are sampled from scalar OLE correlated randomness, that is, $a_1, b_1, a_2, b_2 \in \mathbb{F}$ are randomly sampled conditioning on $a_1a_2 = b_1 + b_2$, then $a_1a_2x_3 = (b_1 + b_2)x_3$ becomes degree-2. This observation is formalized by [LLW20], there is an effective-degree-2 semi-honest MPRE for $\mathsf{3MultPlus}$ using OLE correlated randomness, as presented in Fig. 8.

4.2 CDS Encoding

We observe that the MPRE presented in Fig. 8 is not semi-maliciously secure. Semi-malicious security does not follow automatically from perfect semi-honest security because of the correlated randomness. Party P_1 or P_2 can locally modify its portion of the correlated randomness. For example, if corrupted P_1 replaces b_1 by $b_1 + 1$, the decoding will output $x_1x_2x_3 + x_3 + z_1 + z_2 + z_3$. Similarly, if P_1 replaces a_1 by $a_1 + 1$, the decoding will output $x_1x_2x_3 - a_2x_3 + z_1 + z_2 + z_3$. In either case, privacy is lost.

To prevent such attacks, we will invent a tool called *conditional disclosure of secret (CDS) encoding*, which reveals a secret only if a_1, a_2, b_1, b_2 satisfy the OLE relation $a_1a_2 = b_1 + b_2$.

We start with protecting P_3's privacy, the cases of protecting P_1 or P_2 are similar. Party P_1 has $a_1, b_1 \in \mathbb{F}$. Party P_2 has $a_2, b_2 \in \mathbb{F}$. Party P_3 has a secret $s \in \mathbb{F}$. We would like to construct a CDS encoding that achieves three goals:

Fig. 8. The Effective-Degree-2 Semi-Honest MPRE for $\mathsf{3MultPlus}$

Input: P_i has $x_i, z_i \in \mathbb{F}$.
Local Randomness: P_1 samples $a_{4,1} \in \mathbb{F}$; P_2 samples $a_{5,2} \in \mathbb{F}$; P_3 samples $a_3, a_{4,3}, a_{5,3} \in \mathbb{F}$.
Correlated Randomness: $a_1, b_1, a_2, b_2 \in \mathbb{F}$ are randomly sampled under constraint $a_1 a_2 = b_1 + b_2$. P_1 receives (a_1, b_1). P_2 receives (a_2, b_2). P_3 has no correlated randomness.
Preprocessing: None. I.e., the preprocessing functions are identity functions.
Encoding Function: Outputs the following matrix

$$\widehat{\mathsf{3MultPlus}}\Big(\big((x_1,z_1),a_{4,1},(a_1,b_1)\big),\big((x_2,z_2),a_{5,2},(a_2,b_2)\big),\big((x_3,z_3),(a_3,a_{4,3},a_{5,3}),\perp\big)\Big) = \begin{bmatrix} x_1 - a_1 & \begin{pmatrix} a_3x_1 + a_1x_3 \\ -a_1a_3 - a_4 \end{pmatrix} & \begin{pmatrix} b_1x_3 + b_2x_3 + a_5x_1 - a_1a_5 + \\ a_4x_2 - a_2a_4 + z_1 + z_2 + z_3 \end{pmatrix} \\ -1 & x_3 - a_3 & a_2x_3 - a_5 \\ & -1 & x_2 - a_2 \end{bmatrix}, \quad (4)$$

where $a_4 := a_{4,1} + a_{4,3}$ and $a_5 := a_{5,2} + a_{5,3}$.
Decoding Function: Outputs the determinant of the encoding.

- To disclose s if $a_1 a_2 = b_1 + b_2$.
- To hide s if $a_1 a_2 \neq b_1 + b_2$.
- To reveal nothing about a_1, a_2, b_1, b_2 except whether $a_1 a_2 = b_1 + b_2$.
- The encoding function is quadratic.

We stress that the CDS encoding we are going to build *is not* a MPRE for any function, since it does not follow the same syntax. For example, the security of CDS encoding will rely on the fact that (a_1, a_2, b_1, b_2) is sampled from OLE correlated randomness. The CDS encoding will be used as a sub-module of the semi-maliciously secure MPRE in Sec. 4.3, where CDS encoding is carefully aligned with the rest of the MPRE.

It turns out that, towards building CDS encoding, we need to replace scalar OLE correlated randomness with tensor OLE correlated randomness. Let P_1 receive random $\mathbf{a}_1 \in \mathbb{F}^2, \mathbf{B}_1 \in \mathbb{F}^{2\times 2}$ and let P_2 receive random $\mathbf{a}_2 \in \mathbb{F}^2, \mathbf{B}_2 \in \mathbb{F}^{2\times 2}$, such that $\mathbf{a}_1\mathbf{a}_1^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$. Party P_1, P_2 use the first coordinates of the tensor-OLE correlated randomness as the original scalar OLE correlated randomness. That is, let $(a_1, b_1, a_2, b_2) := (\mathbf{a}_1[1], \mathbf{B}_1[1], \mathbf{a}_2[1], \mathbf{B}_2[1])$. The remaining coordinates will be used to hide a_1, b_1, a_2, b_2.

The CDS encoding is formally described in Fig. 9. As we emphasized, CDS encoding is not a MPRE. For correctness, the secret s can be decoded if $\mathbf{a}_1\mathbf{a}_1^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$. For privacy, the encoding can be simulated given the corrupted parties' input and the following information:

- The secret s if $\mathbf{a}_1\mathbf{a}_1^\intercal = \mathbf{B}_1 + \mathbf{B}_2^\intercal$.
- $p_1 = \langle \mathbf{a}_1, \mathbf{q}_1 \rangle$, $p_2 = \langle \mathbf{a}_2, \mathbf{q}_2 \rangle$, $p_3 = \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \mathbf{q}_1\mathbf{q}_2^\intercal \rangle$, where $\mathbf{q}_1 = (q_1, 1), \mathbf{q}_2 = (q_2, 1)$ are sampled by party P_3. (If the adversary corrupts P_3, it can adaptively choose q_1, q_2.)

Fig. 9. CDS Encoding

The outer MPRE specifies 3 parties, denoted by P_1, P_2, P_3

Input: Party P_3 receives as input $s \in \mathbb{F}$. Party P_1 receives random $\mathbf{a}_1 \in \mathbb{F}^2, \mathbf{B}_1 \in \mathbb{F}^{2\times 2}$. Party P_2 receives random $\mathbf{a}_2 \in \mathbb{F}^2, \mathbf{B}_2 \in \mathbb{F}^{2\times 2}$. $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2)$ is supposed to be sampled from tensor-OLE correlated randomness.
Randomness: In addition, P_3 samples $q_1, q_2, r_1, r_2 \in \mathbb{F}$.
Preprocessing: P_3 locally computes $r_1q_1, r_2q_2, q_1q_2, r_2q_1, r_2q_1q_2$.
Encoding Function: Define $\mathbf{q}_1 = (q_1, 1), \mathbf{q}_2 = (q_2, 1)$. The functionality outputs $p_1 = \langle \mathbf{a}_1, \mathbf{q}_1 \rangle$, $p_2 = \langle \mathbf{a}_2, \mathbf{q}_2 \rangle$, $p_3 = \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \mathbf{q}_1\mathbf{q}_2^\intercal \rangle$ and

$$\mathbf{c} = \begin{bmatrix} 1 & \langle \mathbf{a}_2, \mathbf{q}_2 \rangle \\ \langle \mathbf{a}_1, \mathbf{q}_1 \rangle & \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \mathbf{q}_1\mathbf{q}_2^\intercal \rangle \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} + \begin{bmatrix} 0 \\ s \end{bmatrix}$$

Decoding Function: Check if $p_1p_2 = p_3$. If so, output $\langle \mathbf{c}, (-p_1, 1) \rangle$. Otherwise, output $\perp$.

Let us convey some intuitions why the three goals of CDS encoding in Fig. 9 are achieved.

- For disclosing s: If $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2)$ is in the support of tensor-OLE correlated randomness, then $p_1p_2 = p_3$, thus the matrix $\left[\begin{smallmatrix} 1 & p_2 \\ p_1 & p_3 \end{smallmatrix}\right]$ is not full-rank. The secret s can still be recovered from $\mathbf{c}$.
- For hiding s: If $a_1a_2 \neq b_1 + b_2$, it means $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{B}_1, \mathbf{B}_2)$ is not tensor-OLE correlation. Then $p_1p_2 \neq p_3$ with high probability due to the randomness of q_1, q_2. Since matrix $\left[\begin{smallmatrix} 1 & p_2 \\ p_1 & p_3 \end{smallmatrix}\right]$ is full-rank, the secret s is perfectly masked by (r_1, r_2).
- The encoding also leaks information about $\mathbf{a}_1, \mathbf{B}_1, \mathbf{a}_2, \mathbf{B}_2$, but as we will show, a_1, b_1, a_2, b_2 remain hidden. p_1, p_2 are one-time padded by $\mathbf{a}_1[2], \mathbf{a}_2[2]$. We leave the analysis of p_3 to the next section.
- The encoding function outputs $p_1 = \langle \mathbf{a}_1, \mathbf{q}_1 \rangle$, $p_2 = \langle \mathbf{a}_2, \mathbf{q}_2 \rangle$,

$$p_3 = \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \left[\begin{smallmatrix} q_1q_2 & q_1 \\ q_2 & 1 \end{smallmatrix}\right] \rangle, \quad \mathbf{c} = \begin{bmatrix} r_1 + \langle \mathbf{a}_2, \left[\begin{smallmatrix} r_2q_2 \\ r_2 \end{smallmatrix}\right] \rangle \\ \langle \mathbf{a}_1, \left[\begin{smallmatrix} r_1q_1 \\ r_1 \end{smallmatrix}\right] \rangle + \langle \mathbf{B}_1 + \mathbf{B}_2^\intercal, \left[\begin{smallmatrix} r_2q_1q_2 & r_2q_1 \\ r_2q_2 & r_2 \end{smallmatrix}\right] \rangle \end{bmatrix}$$

  has degree 2 after $r_1q_1, r_2q_2, q_1q_2, r_2q_1, r_2q_1q_2$ are locally computed by P_3.

4.3 Semi-Malicious MPRE for Degree-3 Functions

Based on the CDS encoding (Fig. 9) we discussed in Sec. 4.2, we construct a semi-maliciously secure MPRE (Fig. 10) for canonical degree-3 functions. The MPRE has effective degree 2.

The idea is simple: Three parties P_1', P_2', P_3' need to compute $x_1x_2x_3 + z_1 + z_2 + z_3$. Every party P_i' samples a random mask s_i, one-time pads the output by s_i, and reveals s_i if and only if P_1' and P_2' use legit OLE correlated randomness. The last operation is allowed by our CDS encoding.

Fig. 10. The Semi-Malicious MPRE For Degree-3 Functions

Function Input: P_i has $\mathbf{x}_i \in \mathbb{F}^{\ell_i}$

Function: A canonical degree-3 function f. By definition, there exists an index set $\mathcal{I}$. For each $t \notin \mathcal{I}$, the degree of f_t is at most 2. For each $t \in \mathcal{I}$, f_t equals $\mathbf{x}_{i_{t,1}}[j_{t,1}] \cdot \mathbf{x}_{i_{t,2}}[j_{t,2}] \cdot \mathbf{x}_{i_{t,3}}[j_{t,3}] + \mathbf{x}_{i_{t,1}}[j'_{t,1}] + \mathbf{x}_{i_{t,2}}[j'_{t,2}] + \mathbf{x}_{i_{t,3}}[j'_{t,3}]$ for some $i_{t,1}, i_{t,2}, i_{t,3} \in [n]$ and $j_{t,i}, j'_{t,i} \in [\ell_i]$.
As long as t is clear in the context, we refer to $P_{i_{t,1}}, P_{i_{t,2}}, P_{i_{t,3}}$ as P'_1, P'_2, P'_3 resp. and refer to $\mathbf{x}_{i_{t,1}}[j_{t,1}], \mathbf{x}_{i_{t,2}}[j_{t,2}], \mathbf{x}_{i_{t,3}}[j_{t,3}], \mathbf{x}_{i_{t,1}}[j'_{t,1}], \mathbf{x}_{i_{t,2}}[j'_{t,2}], \mathbf{x}_{i_{t,3}}[j'_{t,3}]$ as $x_1, x_2, x_3, z_1, z_2, z_3$ resp.

Randomness: For each $t \in \mathcal{I}$, P'_1 samples $s_{t,1}, a_{t,4,1} \in \mathbb{F}$, P'_2 samples $s_{t,2}$, $a_{t,5,2} \in \mathbb{F}$, P'_3 samples $s_{t,3}, a_{t,3}, a_{t,4,3}, a_{t,5,3} \in \mathbb{F}$. They also sample additional randomness according to the sub-module of CDS encoding (Fig. 9).
We will omit t in subscript, if there is no confusion.

Correlated Randomness: For each $t \in \mathcal{I}$, P'_1 receives $\mathbf{a}_{t,1} \in \mathbb{F}^4, \mathbf{B}_{t,1} \in \mathbb{F}^{4\times 4}$, P'_2 receives $\mathbf{a}_{t,2} \in \mathbb{F}^4, \mathbf{B}_{t,2} \in \mathbb{F}^{4\times 4}$ where $(\mathbf{a}_{t,1}, \mathbf{a}_{t,2}, \mathbf{B}_{t,1}, \mathbf{B}_{t,2})$ is sampled from 4×4 tensor OLE correlated randomness.
We will omit t from the subscript, if there is no confusion.

Preprocessing: Required by the sub-module of CDS encoding (Fig. 9).

Encoding Function: For each $t \notin \mathcal{I}$, output f_t.
For each $t \in \mathcal{I}$, output

$$M_t = \widehat{\mathsf{3MultPlus}}\big(((x_1, z_1 + s_1), a_{4,1}, (a_1, b_1)),\\ ((x_2, z_2 + s_2), a_{5,2}, (a_2, b_2)),\quad ((x_3, z_3 + s_3), (a_3, a_{4,3}, a_{5,3}), \bot)\big)$$

where $a_1 := \mathbf{a}_1[1], a_2 := \mathbf{a}_2[1], b_1 := \mathbf{B}_1[1,1], b_2 := \mathbf{B}_2[1,1]$.
P'_1, P'_2, P'_1 use CDS encoding to disclose s_1, conditioning on

$$\begin{bmatrix}\mathbf{a}_1[1]\\ \mathbf{a}_1[3]\end{bmatrix}\begin{bmatrix}\mathbf{a}_2[1]\\ \mathbf{a}_2[3]\end{bmatrix}^\intercal = \begin{bmatrix}\mathbf{B}_1[1,1] & \mathbf{B}_1[1,3]\\ \mathbf{B}_1[3,1] & \mathbf{B}_1[3,3]\end{bmatrix} + \begin{bmatrix}\mathbf{B}_2[1,1] & \mathbf{B}_2[1,3]\\ \mathbf{B}_2[3,1] & \mathbf{B}_2[3,3]\end{bmatrix}^\intercal$$

P'_1, P'_2, P'_2 use CDS encoding to disclose s_2, conditioning on

$$\begin{bmatrix}\mathbf{a}_1[1]\\ \mathbf{a}_1[4]\end{bmatrix}\begin{bmatrix}\mathbf{a}_2[1]\\ \mathbf{a}_2[4]\end{bmatrix}^\intercal = \begin{bmatrix}\mathbf{B}_1[1,1] & \mathbf{B}_1[1,4]\\ \mathbf{B}_1[4,1] & \mathbf{B}_1[4,4]\end{bmatrix} + \begin{bmatrix}\mathbf{B}_2[1,1] & \mathbf{B}_2[1,4]\\ \mathbf{B}_2[4,1] & \mathbf{B}_2[4,4]\end{bmatrix}^\intercal$$

P'_1, P'_2, P'_3 use CDS encoding to disclose s_3, conditioning on

$$\begin{bmatrix}\mathbf{a}_1[1]\\ \mathbf{a}_1[2]\end{bmatrix}\begin{bmatrix}\mathbf{a}_2[1]\\ \mathbf{a}_2[2]\end{bmatrix}^\intercal = \begin{bmatrix}\mathbf{B}_1[1,1] & \mathbf{B}_1[1,2]\\ \mathbf{B}_1[2,1] & \mathbf{B}_1[2,2]\end{bmatrix} + \begin{bmatrix}\mathbf{B}_2[1,1] & \mathbf{B}_2[1,2]\\ \mathbf{B}_2[2,1] & \mathbf{B}_2[2,2]\end{bmatrix}^\intercal$$

Decoding Function: For each $t \notin \mathcal{I}$, output f_t.
For each $t \in \mathcal{I}$, decode the CDS encoding to recover s_1, s_2, s_3. If s_1, s_2, s_3 are recovered, output $\det M_t - s_1 - s_2 - s_3$. Otherwise, output $\bot$.

Lemma 2. *Fig. 10 presents a semi-maliciously secure MPRE for degree-3 function f, whose effective degree is 2.*

The proof and the efficiency analysis are deferred to the full version.

	ℓ-size ABP		ℓ-gate circuit
	complexity	statistical security	complexity
2-round, security w/ output substitution	$O(n^2\ell^{1.5})$	$O(\frac{n^2\ell^{1.5}+n^2\cdot Q_{\mathsf{RO}}}{\lvert\mathbb{F}\rvert})$	$O(n^3\ell)$
3-round, security w/ unanimous abort			
2-round, security w/ selective abort	$O(n^3\ell^{1.5})$	$O(\frac{n^3\ell^{1.5}+n^2\cdot Q_{\mathsf{RO}}}{\lvert\mathbb{F}\rvert})$	
2-round, security w/ unanimous abort	$\mathrm{poly}(n,\ell)$	$O(\frac{\mathrm{poly}(n,\ell)+n^2\cdot Q_{\mathsf{RO}}}{\lvert\mathbb{F}\rvert})$	$O(n^3\ell)+\mathrm{poly}(n,\lambda)$

Complexity is measure by the number of field elements communicated.

Fig. 11. The final MPC protocols

5 Putting Pieces Together

MPC protocols for degree-3 functions. Let f be a degree-3 function. By Lemma 2, there is an effective-degree-2 semi-malicious MPRE that has arithmetic preprocessing and uses tensor-OLE correlated randomness. In the full version, we present 2-round maliciously secure MPC protocol for computing any effective-degree-2 function using tensor-OLE correlated randomness (outlined in Sec. 2.3). Their composition is a 2-round statistically maliciously secure MPC protocol for computing f.

MPC protocols for circuits. Let f be a function that is computed by a Boolean circuit of ℓ gates. In the full version, we recall the reduction from **P/poly** to degree-3. So the tasking of computing f can be reduced to computing a degree-3 function $\hat{f}$. Therefore, there is a 2-round MPC protocol that is maliciously secure with output substitution. The protocol makes black-bow calls to PRF. The security of the protocol can be lifted by the technique presented in the full version (outlined in Sec. 2.4). So, there are, as shown in Fig. 11,

- A 3-round MPC protocol that is maliciously secure with unanimous abort and has the same complexity.
- A 2-round MPC protocol that is maliciously secure with selective abort and has the same complexity.
- A 2-round MPC protocol that is maliciously secure with unanimous abort and has an additive polynomial growth on the complexity.

MPC for arithmetic branching programs. Let f be an arithmetic $\mathbf{NC^1}$ function. As we recall in the full version, there is a reduction from $\mathbf{NC^1}$ to degree-3. So the task of computing f can be reduced to computing a degree-3 function $\hat{f}$. Therefore, there is a 2-round MPC protocol that is maliciously secure with output substitution.

Similar to the case of Boolean circuits, the security can be lifted to security with selective/unanimous abort, at various costs, as shown in Fig. 11.

Acknowledgement We thank Antigoni Polychroniadou for being our shepherd and her help, and the anonymous Crypto reviewers for their helpful comments and suggestions. We thank Hoeteck Wee for being part of the initial discussion and his suggestion of directions. Finally, we thank Stefano Tessaro for his comments.

Huijia Lin and Tianren Liu were supported by NSF grants CNS-1936825 (CAREER), CNS-2026774, a JP Morgan AI research Award, a Cisco research award, and a Simons Collaboration on the Theory of Algorithmic Fairness. In addition, Tianren Liu was supported by NSFC excellent young scientists fund program.

References

ABT18. Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Perfect secure computation in two rounds. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 152–174. Springer, Heidelberg, November 2018.

ABT19. Benny Applebaum, Zvika Brakerski, and Rotem Tsabary. Degree 2 is complete for the round-complexity of malicious MPC. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part II*, volume 11477 of *LNCS*, pages 504–531. Springer, Heidelberg, May 2019.

AIK04. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. In *45th FOCS*, pages 166–175. IEEE Computer Society Press, October 2004.

AJJM20. Prabhanjan Ananth, Abhishek Jain, Zhengzhong Jin, and Giulio Malavolta. Multi-key fully-homomorphic encryption in the plain model. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 28–57. Springer, Heidelberg, November 2020.

AJL+12. Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 483–501. Springer, Heidelberg, April 2012.

BCG+19. 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, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 489–518. Springer, Heidelberg, August 2019.

BCG+20. 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, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 387–416. Springer, Heidelberg, August 2020.

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*, pages 896–912. ACM Press, October 2018.

BCI+13. Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 315–333. Springer, Heidelberg, March 2013.

BGI16. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 1292–1303. ACM Press, October 2016.

BGI17. 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, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 163–193. Springer, Heidelberg, April / May 2017.

BGI+18. Elette Boyle, Niv Gilboa, Yuval Ishai, Huijia Lin, and Stefano Tessaro. Foundations of homomorphic secret sharing. In Anna R. Karlin, editor, *ITCS 2018*, volume 94, pages 21:1–21:21. LIPIcs, January 2018.

BGMM20. James Bartusek, Sanjam Garg, Daniel Masny, and Pratyay Mukherjee. Reusable two-round MPC from DDH. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 320–348. Springer, Heidelberg, November 2020.

BGW88. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

BL18. Fabrice Benhamouda and Huijia Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 500–532. Springer, Heidelberg, April / May 2018.

BLPV18. Fabrice Benhamouda, Huijia Lin, Antigoni Polychroniadou, and Muthuramakrishnan Venkitasubramaniam. Two-round adaptively secure multiparty computation from standard assumptions. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 175–205. Springer, Heidelberg, November 2018.

BMR90. Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *22nd ACM STOC*, pages 503–513. ACM Press, May 1990.

BP16. Zvika Brakerski and Renen Perlman. Lattice-based fully dynamic multikey FHE with short ciphertexts. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 190–213. Springer, Heidelberg, August 2016.

CCD88. David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th ACM STOC*, pages 11–19. ACM Press, May 1988.

CDI+19. Melissa Chase, Yevgeniy Dodis, Yuval Ishai, Daniel Kraschewski, Tianren Liu, Rafail Ostrovsky, and Vinod Vaikuntanathan. Reusable non-interactive secure computation. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 462–488. Springer, Heidelberg, August 2019.

CGP15. Ran Canetti, Shafi Goldwasser, and Oxana Poburinnaya. Adaptively secure two-party computation from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 557–585. Springer, Heidelberg, March 2015.

CM15. Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 630–656. Springer, Heidelberg, August 2015.

DI05. Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 378–394. Springer, Heidelberg, August 2005.

DKR15. Dana Dachman-Soled, Jonathan Katz, and Vanishree Rao. Adaptively secure, universally composable, multiparty computation in constant rounds. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 586–613. Springer, Heidelberg, March 2015.

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, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, Heidelberg, August 2012.

FKN94. Uriel Feige, Joe Kilian, and Moni Naor. A minimal model for secure computation (extended abstract). In *26th ACM STOC*, pages 554–563. ACM Press, May 1994.

GGHR14. Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure MPC from indistinguishability obfuscation. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 74–94. Springer, Heidelberg, February 2014.

Gil99. Niv Gilboa. Two party RSA key generation. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 116–129. Springer, Heidelberg, August 1999.

GIS18. Sanjam Garg, Yuval Ishai, and Akshayaram Srinivasan. Two-round MPC: Information-theoretic and black-box. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part I*, volume 11239 of *LNCS*, pages 123–151. Springer, Heidelberg, November 2018.

GLS15. S. Dov Gordon, Feng-Hao Liu, and Elaine Shi. Constant-round MPC with fairness and guarantee of output delivery. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 63–82. Springer, Heidelberg, August 2015.

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 171–185. Springer, Heidelberg, August 1987.

GP15. Sanjam Garg and Antigoni Polychroniadou. Two-round adaptively secure MPC from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 614–637. Springer, Heidelberg, March 2015.

GS17. Sanjam Garg and Akshayaram Srinivasan. Garbled protocols and two-round MPC from bilinear maps. In Chris Umans, editor, *58th FOCS*, pages 588–599. IEEE Computer Society Press, October 2017.

GS18. Sanjam Garg and Akshayaram Srinivasan. Two-round multiparty secure computation from minimal assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

IK97. Yuval Ishai and Eyal Kushilevitz. Private simultaneous messages protocols with applications. In *Fifth Israel Symposium on Theory of Computing and Systems, ISTCS 1997, Ramat-Gan, Israel, June 17-19, 1997, Proceedings*, pages 174–184. IEEE Computer Society, 1997.

IK00. Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *41st FOCS*, pages 294–304. IEEE Computer Society Press, November 2000.

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*, volume 2380 of *LNCS*, pages 244–256. Springer, Heidelberg, July 2002.

IKNP03. Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 145–161. Springer, Heidelberg, August 2003.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

IKP10. Yuval Ishai, Eyal Kushilevitz, and Anat Paskin. Secure multiparty computation with minimal interaction. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 577–594. Springer, Heidelberg, August 2010.

IKSS21. Yuval Ishai, Dakshita Khurana, Amit Sahai, and Akshayaram Srinivasan. On the round complexity of black-box secure MPC. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 214–243, Virtual Event, August 2021. Springer, Heidelberg.

IPS08. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.

LLW20. Huijia Lin, Tianren Liu, and Hoeteck Wee. Information-theoretic 2-round MPC without round collapsing: Adaptive security, and more. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 502–531. Springer, Heidelberg, November 2020.

LPSY15. Yehuda Lindell, Benny Pinkas, Nigel P. Smart, and Avishay Yanai. Efficient constant round multi-party computation combining BMR and SPDZ. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 319–338. Springer, Heidelberg, August 2015.

MW16. Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 735–763. Springer, Heidelberg, May 2016.

Pas12. Anat Paskin-Cherniavsky. *Secure computation with minimal interaction*. PhD thesis, Computer Science Department, Technion, Haifa, Israel, 2012. advised by Yuval Ishai and Eyal Kushilevitz.

PS16. Chris Peikert and Sina Shiehian. Multi-key FHE from LWE, revisited. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 217–238. Springer, Heidelberg, October / November 2016.

Yao82. Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.