

# Information-Theoretic 2-Round MPC without Round Collapsing: Adaptive Security, and More

Huijia Lin<sup>1</sup>, Tianren Liu<sup>1</sup>, and Hoeteck Wee<sup>2</sup>

<sup>1</sup>University of Washington, Seattle, US  
{rachel,tianrenl}@cs.washington.edu

<sup>2</sup>NTT Research and ENS, Paris, France  
wee@di.ens.fr

## Abstract

We present simpler and improved constructions of 2-round protocols for secure multi-party computation (MPC) in the semi-honest setting. Our main results are new information-theoretically secure protocols for arithmetic NC1 in two settings:

- (i) the plain model tolerating up to  $t < n/2$  corruptions; and
- (ii) in the OLE-correlation model tolerating any number of corruptions.

Our protocols achieve adaptive security and require only black-box access to the underlying field, whereas previous results only achieve static security and require non-black-box field access. Moreover, both results extend to polynomial-size circuits with computational and adaptive security, while relying on black-box access to a pseudorandom generator. In the OLE correlation model, the extended protocols for circuits tolerate up to  $n - 1$  corruptions.

Along the way, we introduce a conceptually novel framework for 2-round MPC that does not rely on the round collapsing framework underlying all of the recent advances in 2-round MPC.

## 1 Introduction

Secure multi-party computation (MPC) [Yao86, GMW87, BGW88, CCD88] allows a group of  $n$  mutually distrusting parties to jointly evaluate a function over their private inputs in a manner that reveals nothing beyond the output of the function. In this work, we focus on semi-honest two-round MPC protocols. The state of the art, following the recent breakthroughs in [BL18, GS18] may be broadly classified as follows:

- protocols for NC<sup>1</sup> achieving information-theoretic security tolerating  $t < n/2$  adversarial parties [ABT18];
- protocols for polynomial-size circuits **P/poly** achieving computational security tolerating  $t < n/2$  adversarial parties, assuming the existence of one-way functions [ABT18, ACGJ18];
- protocols for polynomial-size circuits **P/poly** achieving computational security tolerating  $t < n$  adversarial parties, assuming the existence of oblivious transfer [BL18, GS18].

All of these constructions follow the same high-level “round collapsing” strategy introduced in [GGHR14]. In particular, they apply garbled circuits to the circuits of parties’ algorithms of a multi-round MPC protocol, where the garbling is used to collapse the multi-round MPC protocol to a 2-round protocol.

## 1.1 Our Results

We present simpler and improved constructions of 2-round protocols for secure multi-party computation (MPC) in the semi-honest setting. Our main results are new information-theoretically secure protocols for arithmetic  $\text{NC}^1$  in two settings:

- (i) the plain model tolerating up to  $t < n/2$  corruptions; and
- (ii) in the OLE-correlation model tolerating any number of corruptions.

Two parties with an Oblivious Linear Evaluation (OLE) correlation hold respectively random elements  $(a^{(1)}, b^{(1)})$  and  $(a^{(2)}, b^{(2)})$  such that  $a^{(1)}a^{(2)} = b^{(1)} + b^{(2)}$  over a field.

Our protocols achieve adaptive security [CFGN96, Can00] and require only black-box access to the underlying field, whereas previous results only achieve static security and require non-black-box field access. Moreover, both results extend to polynomial-size circuits with computational and adaptive security, while relying on black-box access to a pseudorandom generator. In the OLE correlation model, the extended protocols for circuits tolerate up to  $n - 1$  corruptions. While the honest majority setting is a natural and well-established model, we believe that the OLE-correlation model is also very natural to study, especially for arithmetic computation: OLE correlations enable very efficient online computation, and the correlations themselves can be generated efficiently in the pre-processing phase [BCGI18, BCG<sup>+</sup>20]. We provide a comparison of our results with the state of the art in Figure 1 and Figure 2.

Along the way, we introduce a conceptually novel framework for 2-round MPC that does not rely on the round collapsing framework underlying all of the recent advances in 2-round MPC starting from [GGHR14].

*Our Techniques.* The crux of our protocols, following [ABT18, IK00, IK02], is a way to “encode” degree-3 polynomials into randomized polynomials that have degree 2 after pre-processing of local inputs and randomness – known as multi-party randomized encodings (MPREs). Following the round-collapsing framework of 2-round MPC, prior MPRE schemes garble the next-step circuits of a multi-round MPC protocol, to reduce the degree from 3 to 2.

We construct MPRE directly without using “inner” multi-round MPC. We observe that the [IK02] randomizing polynomials give a way to replace the multiplication between two input elements with multiplication between two random elements. With an OLE-correlation, the product of two random elements are additively shared between two parties, immediately reducing the degree to 2. In the honest majority setting, we exploit a delicate interplay between the IK02 randomized polynomials and the 2-round BGW [BGW88] protocol for computing degree-2 polynomials (or essentially Shamir’s secret sharing scheme) to turn multiplication between two input elements into multiplication between two local random elements, again reducing the degree to 2. Our MPRE schemes and 2-round MPC protocols based on them enjoy simplicity and better efficiency.

*Information-Theoretic Security vs Adaptive Security.* The folklore belief is that any information theoretically secure protocol is also adaptively secure with an *inefficient* simulator. Therefore, to formally prove adaptive security, the technical issue is presenting an explicit efficient simulator. We

Reference	Class	IT/Comp	Corruption	Black-box field	Adaptive
[IK00, IK02, BGW88]	arith-NC <sup>1</sup>	IT	$t < n/3$	yes	yes
[DI05, BMR90, Yao86]	<b>P/poly</b>	Comp	$t < n/3$	–	yes
[GIS18]	NC <sup>1</sup>	IT	$t < n/2$	–	yes*
[ABT18]	NC <sup>1</sup>	IT	$t < n/2$	no	no
[ABT18, ACGJ18, GIS18]	<b>P/poly</b>	Comp	$t < n/2$	–	no
this work	arith-NC <sup>1</sup>	IT	$t < n/2$	yes	yes
this work	<b>P/poly</b>	Comp	$t < n/2$	–	yes

Figure 1: Summary of semi-honest 2-round MPC protocols with a honest majority. All of the constructions for **P/poly** (starting with [DI05]) make black-box use of a PRG. The protocol by [GIS18] handles only a constant number of parties. \* They did not fully specify the adaptive simulator.

Reference	Class	Model	Assumptions	Adaptive
[BL18, GS18]	<b>P/poly</b>	standard	OT	no
[BLPV18]	<b>P/poly</b>	standard	adaptive OT	yes
[GIS18]	<b>P/poly</b>	standard	NIOT	yes*
[IMO18]	<b>P/poly</b>	$n$ -ary correlations	OWF	yes
this work	arith-NC1	2-ary correlated randomness	IT	yes
this work	<b>P/poly</b>	2-ary correlated randomness	OWF	yes

Figure 2: Summary of semi-honest 2-round MPC protocols with a honest minority (that is, any  $t < n$ ). \* They did not fully specify the adaptive simulator.

systematically present and analyze efficient adaptive simulators for our protocols, taking into account different corruption schedules. The analysis benefits greatly from our simpler and modular approach.

## 2 Technical Overview

We present an overview of our constructions, focusing on the honest-majority 2-round MPC for arith-NC1, followed by a more detailed comparison with prior approaches.

Following [IK00, IK02], to construct 2-round MPC for arith-NC1, it suffices to construct a 2-round protocol for the 3-party functionality  $(x_1, x_2, x_3) \mapsto x_1x_2x_3$ . More precisely, we need the functionality  $((x_1, s_1), (x_2, s_2), (x_3, s_3)) \mapsto x_1x_2x_3 + s_1 + s_2 + s_3$ ; for simplicity, we ignore the additive terms in this overview, as they are easy to handle. As with [ABT18], the starting point of our construction is the BGW protocol for computing  $x_1x_2x_3$ . In BGW and also in ABT, the parties (i) multiply Shamir shares of  $x_2, x_3$  for threshold  $t$ , (ii) perform degree reduction to obtain Shamir shares of  $x_2x_3$  for threshold  $t$ , (iii) multiply the ensuing shares by that of  $x_1$  to obtain Shamir shares of  $x_1x_2x_3$  for threshold  $2t$ , (iv) interpolate the shares to recover  $x_1x_2x_3$ . Our construction replaces steps (ii) and (iii) with a completely different gadget.

**MPRE.** A  $(n, t)$ -MPRE [ABT18] for a  $n$ -party functionality  $f(\mathbf{x}_1, \dots, \mathbf{x}_n)$  is a randomized function  $\hat{f}(\mathbf{x}_1, \dots, \mathbf{x}_n; \mathbf{r}_1, \dots, \mathbf{r}_n)$  with the following properties:

- (correctness) There exists an efficient decoder Dec such that for all  $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n), \mathbf{r} = (\mathbf{r}_1, \dots, \mathbf{r}_n)$ ,

$$\text{Dec}(\hat{f}(\mathbf{x}; \mathbf{r})) = f(\mathbf{x})$$

- (security) We say that the MPRE is (selectively) secure against up to  $t$  corruptions if there exists a simulator  $\text{Sim}$  such that for any  $\mathbf{x}_1, \dots, \mathbf{x}_n$  and any subset  $T \subseteq [N], |T| \leq t$ ,

$$\text{Sim}(f(\mathbf{x}_1, \dots, \mathbf{x}_n), \mathbf{x}_T) \approx \left( \hat{f}(\mathbf{x}_1, \dots, \mathbf{x}_n; \mathbf{r}_1, \dots, \mathbf{r}_n), \mathbf{r}_T \right)$$

by distribution, where  $\mathbf{r}_1, \dots, \mathbf{r}_n$  on the right side are random, and  $\mathbf{x}_T := (\mathbf{x}_i : i \in T)$ ,  $\mathbf{r}_T := (\mathbf{r}_i : i \in T)$ .

- (effective degree) We say that a MPRE has effective degree  $d$  if there exists functions  $h_1, \dots, h_n$  such that  $\hat{f}$  can be expressed as a degree  $d$  function of  $h_1(\mathbf{x}_1, \mathbf{r}_1), \dots, h_n(\mathbf{x}_n, \mathbf{r}_n)$ . The functions  $h_i$  capture pre-computation on the local input  $\mathbf{x}_i$  and randomness  $\mathbf{r}_i$  of party  $P_i$ .

In this work, we think of  $\mathbf{x}_1, \dots, \mathbf{x}_n, \mathbf{r}_1, \dots, \mathbf{r}_n$  as vectors over some field  $\mathbb{F}$ . In addition, we define the following new properties:

- We say that an MPRE is adaptively secure if the adversaries can adaptively decide which party to corrupt next, based on the encoding and/or local input and randomness of previously corrupted parties. Correspondingly, simulation is done in an “online” fashion using the output and/or inputs of already corrupted parties.
- We extend MPRE security with leakage: Each party  $P_i$  is associated with a leakage function  $L_i$ . If  $P_i$  is corrupted, the simulator will get  $L_i(\mathbf{x}_1, \dots, \mathbf{x}_n)$  in addition to  $\mathbf{x}_i$ . Unless otherwise specified, the leakage function  $L_i$  simply outputs  $\perp$ .

MPRE with leakage is the key notion that captures our main gadget which uses preprocessing to reduce the degree of IK randomized polynomials from 3 to 2. This notion is also new to this work.

## 2.1 Our Basic Construction

**Main gadget.** Our main gadget is MPRE for the 4-party functionality

$$((x, \mu), a, b, \perp) \mapsto xab + \mu$$

with the following properties:

- (I) it has effective degree 2;
- (II) tolerates any number of corruptions with leakage  $L_4((x, \mu), a, b, \perp) = (a, b)$ .

To build this gadget, we start with the IK02 randomized encoding for  $xab + \mu$  where

$$(x, a, b, \mu; w_1, w_2, w_3, w_4, w_5) \mapsto \begin{bmatrix} a - w_1 & aw_3 + xw_1 - w_3w_1 - w_2 & w_2b - w_2w_5 - w_1w_4 + \boxed{w_1w_5x} + w_4a + \mu \\ -1 & x - w_3 & -w_4 + w_5x \\ & -1 & b - w_5 \end{bmatrix} \quad (1)$$

As a quick warm-up, observe that we can have  $P_4$  sample all of the randomness  $w_1, w_2, w_3, w_4, w_5$ . This achieves effective degree 2, but with leakage  $L_4((x, \mu), a, b, \perp) = (x, a, b, \mu)$ . We show that by distributing the randomness more cleverly, we can reduce the leakage upon corruption of  $P_4$  to just  $a, b$  while preserving effective degree 2.

In particular, we will crucially rely on the fact that the randomized encoding contains exactly one monomial  $w_1w_5x$  of degree 3. In our MPRE,

- $w_2, w_3, w_4$  are shared additively,  $w_i = w_i^{(1)} + w_i^{(4)}$ , between  $P_1$  and  $P_4$  (if both  $P_1$  and  $P_4$  are corrupted, then the adversary already learns all inputs  $x, a, b, \mu$ );
- $P_4$  samples  $w_1, w_5$  and pre-computes  $w_1 w_5$  so that the encoding has effective degree two.

In summary, the MPRE computes the following in effective degree 2:

$$\hat{f}((x, \mu), a, b, \perp ; \mathbf{w}) = g\left((x, \mu, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}), a, b, (w_1 w_5, w_2^{(4)}, w_3^{(4)}, w_4^{(4)})\right) = (1)$$

To handle corruption of  $P_4$  in the analysis of the MPRE, we crucially rely on the fact that we can simulate the randomized encoding together  $w_1, w_5$  given  $(xab + \mu, a, b)$ . To see this, observe that given a simulated encoding  $\Pi$  and  $a, b$ , one can compute matching  $w_1 = \Pi[1, 1] + a$  and  $w_5 = \Pi[3, 3] + b$ .

**MPRE for  $x_1 x_2 x_3$  with OLE correlations.** A two-party OLE correlation over  $\mathbb{F}$  is a pair

$$(w^{(1)}, b^{(1)}), (w^{(2)}, b^{(2)}) : b^{(1)} + b^{(2)} = w^{(1)} \cdot w^{(2)}$$

Observe that in the IK02 randomized encoding Equation (1), multiplication of input elements  $a$  and  $b$  is replaced with multiplication of random elements  $w_1$  and  $w_5$ . If assuming OLE correlation between  $P_2, P_3$ , the IK02 encoding can be computed in degree 2, without any leakage to  $P_4$  (in fact there is no need for  $P_4$  at all). This gives an effective degree 2 MPRE for computing the 3-party functionality

$$x_1, x_2, x_3 \mapsto x_1 x_2 x_3$$

- $P_2$  and  $P_3$  hold  $(w_1, b^{(1)}), (w_5, b^{(5)})$  such that  $b^{(1)} + b^{(5)} = w_1 w_5$ ;
- $w_2, w_3, w_4$  are shared additively between  $P_1, P_2, P_3$ .

Then the encoding computes the following in effective degree 2:

$$\begin{aligned} & \hat{f}(x_1, x_2, x_3 ; \mathbf{w}, \mathbf{b}) \\ & = g\left((x_1, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}), (x_2, w_1, b^{(1)}, w_2^{(2)}, w_3^{(2)}, w_4^{(2)}), (x_3, w_5, b^{(5)}, w_2^{(3)}, w_3^{(3)}, w_4^{(3)})\right) = (1)|_{\mu=0} \end{aligned}$$

Since every degree-3 polynomial can be expanded into a sum of degree-3 monomials, we immediately obtain a degree-2 MPRE for computing general degree-3 polynomials, by computing independent MPRE for each degree-3 monomial.

**Lemma 2.1** (MPRE for degree-3, honest minority). *There exists an adaptively secure MPRE for degree-3 polynomials with effective degree 2 in the OLE-correlation model, for  $t \leq n$ .*

**MPRE for  $x_1 x_2 x_3$  for honest majority.** Next, we build a  $n$ -party MPRE with effective degree 2 for

$$x_1, x_2, x_3, \underbrace{\perp, \dots, \perp}_{n-3} \mapsto x_1 x_2 x_3$$

tolerating  $t < n/2$  corruptions, as long as  $|\mathbb{F}| > n$  (without any leakage). For simplicity, we consider the setting where  $P_1$  is never corrupted. Following the overview,

- $P_2$  samples a random degree- $t$  polynomial  $Q_2$  such that  $Q_2(0) = x_2$ .

- Similarly,  $P_3$  samples  $Q_3$  with  $Q_3(0) = x_3$ .
- $P_1$  samples a random degree- $(n - 1)$  polynomial  $Z$  such that  $Z(0) = 0$ .

Now, consider the polynomial

$$Y := x_1 Q_2 Q_3 + Z$$

Observe that  $Y$  has degree at most  $n - 1$ , and satisfies  $Y(0) = x_1 x_2 x_3$ . Then, for each  $i = 1, 2, \dots, n$ , parties  $P_1, P_2, P_3, P_i$  run the gadget MPRE to compute

$$((x_1, Z(i)), Q_2(i), Q_3(i), \perp) \mapsto Y(i) = x_1 Q_2(i) Q_3(i) + Z(i)$$

The output party can recover  $Y(0) = x_1 x_2 x_3$  given  $Y(1), \dots, Y(n)$  via polynomial interpolation. In summary, the MPRE is the parallel composition of  $n$  gadget MPRE and hence have effective degree 2.

$$\hat{F}(x_1, x_2, x_3, \underbrace{\perp, \dots, \perp}_{P_4 \text{ to } P_n}; \mathbf{r}) = \left( \hat{f} \left( \underbrace{(x_1, Z(i))}_{P_1}, \underbrace{Q_2(i)}_{P_2}, \underbrace{Q_3(i)}_{P_3}, \underbrace{\perp}_{P_i} \right) \right)_{i \in [n]}$$

We can in fact prove security of this MPRE for up to  $t < n/2$  corruptions, as long as  $P_1$  is not corrupted. We sketch the security proof for the setting where the last  $t$  parties  $P_{n-t+1}, \dots, P_n$  are corrupted:

- We can simulate the encoding by sampling a random degree  $n - 1$  polynomial  $Y$  whose constant term is  $x_1 x_2 x_3$ , thanks to the randomization via  $Z$ ;
- To simulate the view of the last  $t$  parties, security of the gadget MPRE tells us that it suffices to simulate  $Q_2(i), Q_3(i), i = n - t + 1, \dots, n$ . By the security of Shamir's secret sharing, these are just a collection of uniformly random field elements, and leaks no additional information to the adversary.

More generally,  $P_1$  may be corrupted, at which point  $x_1$  and the polynomial  $Z$  are revealed. To ensure privacy of  $x_2, x_3$  in this case, we need to modify the polynomial to  $Y := x_1 Q_2 Q_3 + Z + S$ , with an additional random degree- $(n - 1)$  polynomial  $S$  jointly sampled by all parties, with  $P_i$  sampling  $S(i)$  at random. To recover the output  $x_1 x_2 x_3$ , the parties additionally compute  $S(0)$ , which is a linear function over local inputs.

Since MPRE for computing degree-3 monomials gives MPRE for general degree-3 polynomials, we obtain

**Lemma 2.2** (MPRE for degree-3, honest majority). *There exists an adaptively secure MPRE for degree-3 polynomials with effective degree 2 in the plain model, for  $t < n/2$ .*

**Handling adaptive corruptions.** All our MPRE schemes introduced so far have perfect information theoretic security. In later sections, we construct an efficient and stateful simulator for simulating the view of adaptive adversaries. In particular, the simulator Sim can be decomposed into a stateful two-subroutine simulator (SimO, SimI) in which SimO( $f(\mathbf{x}_1, \dots, \mathbf{x}_n)$ ) simulates the encoding  $\hat{f}(\mathbf{x}_1, \dots, \mathbf{x}_n; \mathbf{r}_1, \dots, \mathbf{r}_n)$ , and SimI( $i, \mathbf{x}_i$ ) simulates  $\mathbf{r}_i$ , in the order that the adaptive adversary corrupts parties.

**Putting Pieces Together for NC<sup>1</sup>** Given an MPRE for computing degree-3 polynomials in a model (the OLE correlation model or in the plain model with honest majority), we can “lift” it to handle arithmetic NC<sup>1</sup> computation in the same model, while preserving the effective degree. The IK02 randomized encoding [IK02] for arith-NC1 allows for transforming a function  $g$  in NC<sup>1</sup> by a degree-3 polynomial  $\hat{g}$ , such that,  $\hat{g}(\mathbf{x}_1, \dots, \mathbf{x}_n; \mathbf{r})$  reveals only  $g(\mathbf{x}_1, \dots, \mathbf{x}_n)$  and nothing else. This means it suffices to compute the following  $n$ -party degree-3 functionality where randomness  $\mathbf{r}$  is *additively* shared among all parties.

$$(\mathbf{x}_1, \mathbf{r}^{(1)}) \cdots (\mathbf{x}_n, \mathbf{r}^{(n)}) \mapsto \hat{g}(\mathbf{x}_1, \dots, \mathbf{x}_n; \mathbf{r} = \sum_i \mathbf{r}^{(i)}). \quad (2)$$

The above is an effective-degree-3 MPRE for arithmetic NC<sup>1</sup>. We further reduce the effective degree to 2, by computing the effective-degree-3 MPRE using the effective-degree-2 MPREs for degree 3 polynomials.

**Lemma 2.3** (MPRE for arith-NC1). *There exist adaptively secure MPRE for arith-NC1 with effective degree 2 in the OLE-correlation model for any number  $t \leq n$  of corruptions, and in the plain model for  $t < n/2$ .*

Finally, to obtain 2-round MPC for arith-NC1, we compute the effective-degree-2 MPRE using 2-round MPC for degree-2 polynomials. In the honest majority model, the BGW protocol has only 2 rounds when computing degree-2 polynomials. In the OLE correlation model, we design a very simple 2-round protocol for computing degree-2 polynomials.

**Extension to circuits.** Starting from Yao’s garbled circuits, we can get a  $(n - 1)$ -private MPRE for **P/poly** with effective degree 3 that makes black-box use of a PRG  $G$ , using the techniques introduced in [DI05, BMR90]. For simplicity, consider garbling a single gate  $g$  with input wire  $u, v$  and output wire  $o$ . For each input/output wire  $j$ , each party  $P_i$  samples a pair of PRG seeds  $s_{j,0}^{(i)}, s_{j,1}^{(i)}$  corresponding the wire having value 0 or 1; the two labels for wire  $j$  is then set to  $\ell_{j,b} = s_{j,b}^{(1)} \parallel \dots \parallel s_{j,b}^{(n)}$ . To hide the labels of the output wire  $o$ , each party locally expands their seeds through  $G$ , and hide label  $\ell_{o,g(a,b)}$  using the XOR of PRG outputs from all parties. For instance,

$$\ell_{o,g(a,b)} \oplus \left( \bigoplus_i G_d(s_{u,a}^{(i)}) \right) \oplus \left( \bigoplus_i G_{d'}(s_{v,b}^{(i)}) \right)$$

where  $G_d$  for  $d = 0$  or  $1$  outputs the first or second half of the PRG output bits respectively, and  $d, d'$  are set so that the same output bit is never reused. These table entries are further randomly permuted using mask bits  $k_u, k_v$  which are additively shared among all parties. The computed encoding is secure as long as one party remains uncorrupted. The computation makes black-box use of the PRG and has effective degree 3 after pre-processing of form:

$$h(\mathbf{x}_i; \mathbf{k}^{(i)}, \mathbf{s}^{(i)}) = (x_i, (k_j^{(i)}, s_{j,0}^{(i)}, s_{j,1}^{(i)}, G(s_{j,0}^{(i)}), G(s_{j,1}^{(i)}))_j)$$

We can then combine this with our MPRE for degree-3 polynomials with effective degree 2 (over a sufficiently large field extension of  $\mathbb{F}_2$ ).

**Lemma 2.4** (MPRE for **P/poly**). *There exist adaptively secure MPRE for **P/poly** with effective degree 2 in the OLE-correlation model for any number  $t \leq n - 1$  of corruptions, and in the plain model for  $t < n/2$ . The scheme makes black-box use of a PRG.*

2-round MPC protocols for **P/poly** in the same models then follow.

### 3 Preliminaries and Definitions

For any positive integer  $n$ , define  $[n] := \{1, 2, \dots, n\}$ . For any set  $S \subseteq [n]$  and vector  $\mathbf{x} = (x_1, \dots, x_n)$ , where  $x_i$  itself can be a vector, let  $\mathbf{x}[S]$  denote the indexed set  $(x_i)_{i \in S}$ . Let  $\mathbb{F}$  denote a finite field, and  $\otimes$  tensor product.

#### 3.1 MPC Protocols

**Definition 1** (Functionality). An  $n$ -party functionality is a function  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$ , where  $\mathcal{X}_i$  is the  $i$ -th party's input domain and  $\mathcal{Y}$  is the output space.

**Definition 2** (MPC Protocol). An  $r$ -rounds MPC protocol  $\Pi$  for a  $n$ -party functionality  $f$  consists of  $n$  algorithms  $(C_i)_{i \in [n]}$ . An execution of  $\Pi$  with inputs  $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X}_1 \times \dots \times \mathcal{X}_n$  and security parameter  $1^\lambda$  proceeds as follows:

**Randomness** Each party  $P_i$  samples local randomness  $\mathbf{r}_i \leftarrow \mathcal{R}_i$ , where  $\mathcal{R}_i$  is the local randomness space of the  $P_i$ . It initializes its state as  $\text{st}_i^{(0)} = (\mathbf{x}_i, \mathbf{r}_i)$ .

**Round**  $1 \leq j \leq r$ : Every party  $P_i$  computes  $(m_{i \rightarrow 1}^{(j)}, \dots, m_{i \rightarrow n}^{(j)}) \leftarrow C_i(1^\lambda, \text{st}_i^{(j-1)})$ . For every  $i' \in [n] \setminus \{i\}$ ,  $P_i$  sends message  $m_{i \rightarrow i'}^{(j)}$  to party  $P_{i'}$ , and receives message  $m_{i' \rightarrow i}^{(j)}$  from party  $P_{i'}$ . It updates its state  $\text{st}_i^{(j)} = (\text{st}_i^{(j-1)}, (m_{i' \rightarrow i}^{(j)})_{i' \in [n] \setminus \{i\}})$ .

**Output:** After  $r$  rounds, every party  $P_i$  computes  $\mathbf{y}_i \leftarrow C_i(1^\lambda, \text{st}_i^{(r)})$ , and outputs  $\mathbf{y}_i$ .

Define the view of party  $P_i$  in the above execution to be  $\text{VIEW}_\Pi(1^\lambda, \mathbf{x})[i] = \text{st}_i^{(r)} = (\mathbf{x}_i, \mathbf{r}_i, (m_{i' \rightarrow i}^{(j)})_{i' \in [n] \setminus \{i\}, j \in [r]})$ . Let  $\text{VIEW}_\Pi(1^\lambda, \mathbf{x})$  denote the array of views of all parties.

We also consider MPC protocol that relies on *correlated randomness*. If the MPC protocol relies on correlated randomness, which is a distribution  $\mathcal{D}$  over  $\mathcal{R}'_1 \times \dots \times \mathcal{R}'_n$ , then in each execution of the protocol,  $(\mathbf{r}'_1, \dots, \mathbf{r}'_n) \leftarrow \mathcal{D}$  is sampled by the beginning of the protocol, and each party  $P_i$  initialize its state as  $\text{st}_i^{(0)} = (\mathbf{x}_i, \mathbf{r}_i, \mathbf{r}'_i)$ .

Below, we suppress the appearance of the security parameter  $1^\lambda$ , which is assumed implicitly.

**Remark 3.1.** We remark that the above definition considers the same output for all parties. It can be generalized to the case where each party has a different output. From a protocol design point of view, it is without loss of generality to consider a common output: To compute function  $f$  mapping  $\mathbf{x}_1, \dots, \mathbf{x}_n$  to different outputs  $\mathbf{y}_1, \dots, \mathbf{y}_n$ , every party  $P_i$  can sample a one-time pad  $\mathbf{k}_i$  of appropriate length and jointly compute the augmented functionality mapping  $(\mathbf{x}_1, \mathbf{k}_1), \dots, (\mathbf{x}_n, \mathbf{k}_n)$  to  $(\mathbf{y}_1 + \mathbf{k}_1), \dots, (\mathbf{y}_n + \mathbf{k}_n)$ , where  $\mathbf{k}_i$ 's and  $+$  should be defined appropriately for the specific functionality  $f$ . For instance, if  $f$  is a Boolean computation,  $\mathbf{k}_i$ 's should be random strings and  $+$  is XOR, and if  $f$  is an arithmetic computation over a finite field,  $\mathbf{k}_i$ 's should be random vectors and  $+$  over the field.

**Definition 3** (MPC Correctness). A protocol  $\Pi$  for a functionality  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$  is *perfectly* or *statistically* correct, if for every input tuple  $\mathbf{x} \in \mathcal{X}_1 \times \dots \times \mathcal{X}_n$  and every security parameter  $\lambda \in \mathbb{N}$ , the output of every party  $P_i$  equals  $f(x_1, \dots, x_n)$ , with *probability 1* or with *overwhelming probability* respectively.

**Definition 4** (Semi-Honest Security against Static Corruption). A protocol  $\Pi$  for a  $n$ -party functionality  $f$  is *perfectly*, or *statistically*, or *computationally* semi-honest secure against  $t$ -corruption,

if there is a PPT simulator  $\text{Sim}$ , such that for every subset  $T \subseteq [n]$  of at most  $t$  parties, input tuple  $\mathbf{x}$ , it holds that the real views  $\text{VIEW}_{\Pi}(\mathbf{x})[T]$  of parties in  $T$  and the output of the simulator  $\text{Sim}(T, \mathbf{x}[T], f(\mathbf{x}))$  are *identically distributed*, or *statistically close*, or *computationally indistinguishable* respectively.

**Semi-honest Adaptive Security** In the adaptive corruption model, a semi-honest adversary is allowed to choose which party to corrupt next adaptively (up to  $t$  corruptions) depending on its current view, which includes the views of previously corrupted parties. Correspondingly, the simulator for adaptive adversaries is an interactive stateful algorithm that responds to adversary's corruption requests with simulated views, generated from the inputs and output of corrupted parties.

**Definition 5** (Semi-honest Security against Adaptive Corruption.). A protocol  $\Pi$  for a  $n$ -party functionality  $f$  is *perfectly*, or *statistically*, or *computationally* semi-honest adaptively secure against  $t$ -corruption, if there is a PPT interactive and stateful simulator  $\text{Sim}$ , such that, for every adversary  $\mathcal{A}$  (PPT in the computational setting, computationally unbounded otherwise), input tuple  $\mathbf{x}$ , the outputs of the following two experiments are *identically distributed*, or *statistically close*, or *computationally indistinguishable* respectively.

- *In the real world:* The challenger runs an execution of  $\Pi$  on input  $\mathbf{x}$  using fresh randomness, obtaining parties' views  $\text{VIEW}_{\Pi}(\mathbf{x})$ .

The adversary  $\mathcal{A}$  adaptively and iteratively queries  $\text{Corrupt}(i)$ , and receives  $P_i$ 's view  $\text{VIEW}_{\Pi}(\mathbf{x})[i]$ , up to at most  $t$  corruptions.

Return  $\mathcal{A}$ 's output.

- *In the simulation:* Proceed identically as in the real world, except that upon  $\mathcal{A}$ 's request  $\text{Corrupt}(i)$ , invoke the simulator  $(\widetilde{\text{VIEW}}[i], \text{st}) \leftarrow \text{Sim}(i, \mathbf{x}_i, y, \text{st})$  and sends  $\widetilde{\text{VIEW}}[i]$  to  $\mathcal{A}$ , where  $\text{st}$  is initialized to be empty.

### 3.2 (Multi-Party) Randomized Encoding

**Definition 6** (Randomized Encoding [IK02, AIK04]). Let  $f : \mathcal{X} \rightarrow \mathcal{Y}$  be some function. The *randomized encoding* of  $f$  is a function  $\hat{f} : \mathcal{X} \times \mathcal{R} \rightarrow \hat{\mathcal{Y}}$ , where  $\mathcal{R}$  is the randomness space. A randomized encoding should be both correct and private.

**Correctness** There is a decoding function  $\text{Dec}$  such that for all  $x \in \mathcal{X}, r \in \mathcal{R}$ , it holds that

$$\text{Dec}(\hat{f}(x; r)) = f(x).$$

**Privacy** There exists a efficient randomized simulation algorithm  $\text{Sim}$  such that for any  $x \in \mathcal{X}$ , the distribution of  $\text{Sim}(f(x))$  is identical to that of  $f(x; r)$ . The privacy can be relaxed to statistical privacy (resp. computational privacy), if the  $\text{Sim}(f(x))$  and  $\hat{f}(x; r)$  are statistically close (resp. computational indistinguishable).

**Definition 7** (Multi-Party Randomized Encoding [ABT18]). Let  $f : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \rightarrow \mathcal{Y}$  be some  $n$ -party functionality. A *multi-party randomized encoding* (MPRE) of  $f$  consists of

- Input space  $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$  and output space  $\mathcal{Y}$ ;

- 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  $h_i : \mathcal{X}_i \times \mathcal{R}_i \times \mathcal{R}'_i \rightarrow \hat{\mathcal{X}}_i$ ;
- Encoding function  $\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \rightarrow \hat{\mathcal{Y}}$ , the degree of  $\hat{f}$  is called the *effective degree* of this MPRE.

Such that for any input  $(x_1, \dots, x_n)$ , the encoding  $\hat{f}(h_1(x_1, r_1, r'_1), \dots, h_n(x_n, r_n, r'_n))$  represents  $y = f(x_1, \dots, x_n)$  in the following sense:

**Correctness** There exists a decoding function  $\text{Dec} : \hat{\mathcal{Y}} \rightarrow \mathcal{Y}$ , such that for any input  $(x_1, \dots, x_n) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ , randomness  $(r_1, \dots, r_n) \in \mathcal{R}_1 \times \cdots \times \mathcal{R}_n$  and correlated randomness  $(r'_1, \dots, r'_n)$  in the support of  $\mathcal{D}$ , the corresponding encodings  $\hat{y} = \hat{f}(h_1(x_1, r_1, r'_1), \dots, h_n(x_n, r_n, r'_n))$  satisfies that  $f(x_1, \dots, x_n) = \text{Dec}(\hat{y})$ .

**Semi-honest Adaptive  $t$ -Privacy** The MPRE is perfectly (resp. statistically or computationally) secure against  $t$  adaptive corruptions if there exists an adaptive simulator such that the following real world and ideal world are perfectly (resp. statistically or computationally) indistinguishable.

In both the real world and the ideal world, the distinguisher first chooses input  $\mathbf{x} = (x_1, \dots, x_n)$ , and sends it to the challenger. Then the distinguisher can make queries and tries to guess which world it is.

- *In the real world:* The distinguisher chooses input  $\mathbf{x} = (x_1, \dots, x_n)$ , and sends it to the challenger. The challenger samples local randomness  $r_i \leftarrow \mathcal{R}_i$  for each  $i \in [n]$  and correlated randomness  $(r'_1, \dots, r'_n) \leftarrow \mathcal{D}$ ; computes  $\hat{x}_i = h_i(x_i, r_i, r'_i)$  for  $i \in [n]$  and  $\hat{y} = \hat{f}(\hat{x}_1, \dots, \hat{x}_n)$ . In short, the challenger follows the protocol.

The challenger allows the distinguisher to adaptively query the following two oracles. The later one can be queried up to  $t$  times.

**Upon CorruptO:** Output  $\hat{y}$

**Upon CorruptI( $i$ ):** Output  $r_i, r'_i$ .

- *In the ideal world:* The distinguisher chooses input  $\mathbf{x} = (x_1, \dots, x_n)$ , and sends it to the challenger. The challenger does nothing other than stores the input. The queries are answered by the simulator, which is a randomized stateful algorithm (SimO, SimI).

The challenger allows the distinguisher to adaptively query the following two oracles. The later one can be queried up to  $t$  times.

**Upon CorruptO:** Compute  $y = f(\mathbf{x})$  and output whatever SimO( $y$ ) outputs.

**Upon CorruptI( $i$ ):** Output what is output by SimI( $i, x_i$ ).

### 3.3 Composition of MPREs

If there is a MPRE for  $f$  whose encoding function is  $\hat{f}$ , together with a MPRE for  $\hat{f}$  whose encoding function is  $\hat{\hat{f}}$ . Then Theorem 3.2 shows that they can be composed as a MPRE for  $f$  whose encoding function is  $\hat{\hat{f}}$ . Theorem 3.2 is adaptive version of Lemma 3.3 and 3.4 in [ABT18]. Such composition is useful when  $\hat{\hat{f}}$  is simpler than  $\hat{f}$ .

If there are MPREs for  $f_1, f_2$ . W.l.o.g., assume their input domain are the same. Then Theorem 3.3 shows that they can be composed as a MPRE for the functionality

$$f(x_1, \dots, x_n) = (f_1(x_1, \dots, x_n), f_2(x_1, \dots, x_n))$$

while preserving the complexity.

**Theorem 3.2** (Sequential Composition). *Assume there is a perfectly (resp. statistically or computationally) adaptively  $t$ -private MPRE for functionality  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$ , whose encoding function is  $\hat{f} : \hat{\mathcal{X}}_1 \times \dots \times \hat{\mathcal{X}}_n \rightarrow \hat{\mathcal{Y}}$ . Assume there is a perfectly (resp. statistically or computationally) adaptively  $t$ -private MPRE for  $\hat{f}$ , whose encoding function is  $\hat{\hat{f}} : \hat{\hat{\mathcal{X}}}_1 \times \dots \times \hat{\hat{\mathcal{X}}}_n \rightarrow \hat{\hat{\mathcal{Y}}}$ . Then there exists a perfectly (resp. statistically or computationally) adaptively  $t$ -private MPRE for  $f$  whose encoding function is  $\hat{\hat{f}}$ .*

*Proof.* Say the MPRE for  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$ , call it the outer MPRE, has

- Local randomness spaces  $\mathcal{R}_1^{\text{outer}}, \dots, \mathcal{R}_n^{\text{outer}}$ ;
- (if any) Correlated randomness space  $\mathcal{R}_1^{\text{outer}'} \times \dots \times \mathcal{R}_n^{\text{outer}'}$  and the distribution  $\mathcal{D}^{\text{outer}}$ ;
- Local preprocessing functions  $h_1^{\text{outer}}, \dots, h_n^{\text{outer}}$ ;
- Encoding function  $\hat{f} : \hat{\mathcal{X}}_1 \times \dots \times \hat{\mathcal{X}}_n \rightarrow \hat{\mathcal{Y}}$
- Decoding function  $\text{Dec}^{\text{outer}} : \hat{\mathcal{Y}} \rightarrow \mathcal{Y}$ ;
- Simulator  $\text{Sim}^{\text{outer}} = (\text{SimO}^{\text{outer}}, \text{SimI}^{\text{outer}})$ .

Similarly, say the MPRE for  $\hat{f} : \hat{\mathcal{X}}_1 \times \dots \times \hat{\mathcal{X}}_n \rightarrow \hat{\mathcal{Y}}$ , the inner MPRE, has

- Local randomness spaces  $\mathcal{R}_1^{\text{inner}}, \dots, \mathcal{R}_n^{\text{inner}}$ ;
- (if any) Correlated randomness space  $\mathcal{R}_1^{\text{inner}'} \times \dots \times \mathcal{R}_n^{\text{inner}'}$  and the distribution  $\mathcal{D}^{\text{inner}}$ ;
- Local preprocessing functions  $h_1^{\text{inner}}, \dots, h_n^{\text{inner}}$ ;
- Encoding function  $\hat{\hat{f}} : \hat{\hat{\mathcal{X}}}_1 \times \dots \times \hat{\hat{\mathcal{X}}}_n \rightarrow \hat{\hat{\mathcal{Y}}}$
- Decoding function  $\text{Dec}^{\text{inner}} : \hat{\hat{\mathcal{Y}}} \rightarrow \hat{\mathcal{Y}}$ ;
- Simulator  $\text{Sim}^{\text{inner}} = (\text{SimO}^{\text{inner}}, \text{SimI}^{\text{inner}})$ .

Then the scheme in Figure 3 is a MPRE for  $f$  whose encoding function is  $\hat{\hat{f}}$ .

The correctness of the resulting MPRE follows from the correctness of the inner and outer MPRE. In short,

$$\begin{aligned} & \text{Dec}(\hat{\hat{f}}(\dots, h_i(x_i, r_i^{\text{outer}}, r_i^{\text{inner}}, r_i^{\text{outer}'}, r_i^{\text{inner}'}), \dots)) \\ &= \underset{\text{(by definition)}}{\text{Dec}^{\text{outer}}(\text{Dec}^{\text{inner}}(\hat{\hat{f}}(\dots, h_i^{\text{inner}}(h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'}), r_i^{\text{inner}}, r_i^{\text{inner}'}), \dots)))} \\ &= \underset{\text{(correctness of inner MPRE)}}{\text{Dec}^{\text{outer}}(\hat{f}(\dots, h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'}), \dots))} \\ &= \underset{\text{(correctness of outer MPRE)}}{f(\dots, x_i, \dots)}. \end{aligned}$$

The adaptive  $t$ -privacy is the indistinguishability between the real world and the ideal world.

- *In the real world.* The adversary chooses the input  $(x_1, \dots, x_n)$ , then the adversary's queries are answered as the follows

<p>MPRE yielded by sequential composition</p> <p><b>Local randomness:</b> <math>P_i</math> samples <math>r_i^{\text{outer}} \leftarrow \mathcal{R}_1^{\text{outer}}, r_i^{\text{inner}} \leftarrow \mathcal{R}_i^{\text{inner}}</math></p> <p><b>Correlated randomness:</b> <math>P_i</math> gets <math>r_i^{\text{outer}'}, r_i^{\text{inner}'}</math> such that <math>(r_1^{\text{outer}'}, \dots, r_n^{\text{outer}'})</math> is sampled from <math>\mathcal{D}^{\text{outer}}</math> and <math>(r_1^{\text{inner}'}, \dots, r_n^{\text{inner}'})</math> is sampled from <math>\mathcal{D}^{\text{inner}}</math></p> <p><b>Preprocessing function <math>h_i</math>:</b></p> $h_i(x_i, r_i^{\text{outer}}, r_i^{\text{inner}}, r_i^{\text{outer}'}, r_i^{\text{inner}'}) = h_i^{\text{inner}}(h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'}), r_i^{\text{inner}}, r_i^{\text{inner}'})$ <p><b>Encoding function:</b> <math>\hat{f}</math>.</p> <p><b>Decoding function:</b> <math>\text{Dec}(\hat{y}) = \text{Dec}^{\text{outer}}(\text{Dec}^{\text{inner}}(\hat{y}))</math>.</p>
---

Figure 3: Sequential Composition of MPREs

- On the outset:** Sample  $r_i^{\text{outer}} \leftarrow \mathcal{R}_i^{\text{outer}}, r_i^{\text{inner}} \leftarrow \mathcal{R}_i^{\text{inner}}$  for  $i \in [n]$ .  
Sample  $(r_1^{\text{outer}'}, \dots, r_n^{\text{outer}'}) \leftarrow \mathcal{D}^{\text{outer}}, (r_1^{\text{inner}'}, \dots, r_n^{\text{inner}'}) \leftarrow \mathcal{D}^{\text{inner}}$ .  
Compute  $\hat{x}_i = h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'})$ ,  $\hat{x}_i = h_i^{\text{inner}}(\hat{x}_i, r_i^{\text{inner}}, r_i^{\text{inner}'})$  for  $i \in [n]$ .  
Compute  $\hat{y} = \hat{f}(\hat{x}_1, \dots, \hat{x}_n)$ ,  $\hat{y} = \text{Dec}^{\text{inner}}(\hat{y})$ ,  $y = \text{Dec}^{\text{outer}}(\hat{y})$ .

**CorruptI(i):** Output  $r_i^{\text{outer}}, r_i^{\text{inner}}, r_i^{\text{outer}'}, r_i^{\text{inner}'}$ .

**CorruptO:** Output  $\hat{y}$ .

- *In the ideal world.* The adversary chooses the input  $(x_1, \dots, x_n)$ , then the adversary's queries are answered as the follows

**CorruptI(i):** Let  $(r_i^{\text{outer}}, r_i^{\text{outer}'}) \leftarrow \text{SimI}^{\text{outer}}(i, x_i)$ .

Compute  $\hat{x}_i = h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'})$ , let  $(r_i^{\text{inner}}, r_i^{\text{inner}'}) \leftarrow \text{SimI}^{\text{inner}}(i, \hat{x}_i)$ .

Output  $r_i^{\text{outer}}, r_i^{\text{inner}}, r_i^{\text{outer}'}, r_i^{\text{inner}'}$ .

**CorruptO:** Compute  $y = f(x_1, \dots, x_n)$ . Let  $\hat{y} \leftarrow \text{SimO}^{\text{outer}}(y)$ . Let  $\hat{\hat{y}} \leftarrow \text{SimO}^{\text{inner}}(\hat{y})$ . Output  $\hat{\hat{y}}$ .

The indistinguishability is proved by a hybrid argument. We introduce the middle world.

- *In the middle world.* The adversary chooses the input  $(x_1, \dots, x_n)$ , then the adversary's queries are answered as the follows

**On the outset:** Sample  $r_i^{\text{outer}} \leftarrow \mathcal{R}_i^{\text{outer}}$  for  $i \in [n]$ . Sample  $(r_1^{\text{outer}'}, \dots, r_n^{\text{outer}'}) \leftarrow \mathcal{D}^{\text{outer}}$ .

Compute  $\hat{x}_i = h_i^{\text{outer}}(x_i, r_i^{\text{outer}}, r_i^{\text{outer}'})$  for  $i \in [n]$ .

Compute  $\hat{y} = \hat{f}(\hat{x}_1, \dots, \hat{x}_n)$ ,  $y = \text{Dec}^{\text{outer}}(\hat{y})$ .

**CorruptI(i):** Let  $(r_i^{\text{inner}}, r_i^{\text{inner}'}) \leftarrow \text{SimI}^{\text{inner}}(i, \hat{x}_i)$ .

Output  $r_i^{\text{outer}}, r_i^{\text{inner}}, r_i^{\text{outer}'}, r_i^{\text{inner}'}$ .

**CorruptO:** Let  $\hat{y} \leftarrow \text{SimO}^{\text{inner}}(\hat{y})$ . Output  $\hat{y}$ .

Then as long as the adversary call **CorruptI** at most  $t$  times, the security of the inner MPRE implies the indistinguishability between the real world and the middle world, the security of the outer MPRE implies the indistinguishability between the real world and the ideal world.  $\square$

**Theorem 3.3 (Parallel Composition).** For each  $j \in [m]$ , assume there is a perfectly (resp. statistically or computationally) adaptively  $t$ -private MPRE for functionality  $f^{(j)} : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}^{(j)}$ , whose encoding function is  $\hat{f}^{(j)} : \hat{\mathcal{X}}_1^{(j)} \times \dots \times \hat{\mathcal{X}}_n^{(j)} \rightarrow \hat{\mathcal{Y}}^{(j)}$ . Then there exists a perfectly (resp. statistically or computationally) adaptively  $t$ -private MPRE for  $f$  whose encoding function is  $\hat{f}$ , where  $f$  concatenate the outputs of  $f^{(1)}, \dots, f^{(m)}$

$$f(x_1, \dots, x_n) := (f^{(1)}(x_1, \dots, x_n), \dots, f^{(m)}(x_1, \dots, x_n))$$

and  $\hat{f}$  is the concatenation of  $\hat{f}^{(1)}, \dots, \hat{f}^{(m)}$

$$\hat{f}((\hat{x}_1^{(1)}, \dots, \hat{x}_1^{(m)}), \dots, (\hat{x}_n^{(1)}, \dots, \hat{x}_n^{(m)})) = (\hat{f}^{(1)}(\hat{x}_1^{(1)}, \dots, \hat{x}_n^{(1)}), \dots, \hat{f}^{(m)}(\hat{x}_1^{(m)}, \dots, \hat{x}_n^{(m)})).$$

Additionally, if the MPRE for  $f^{(j)}$  has leakage  $L_i^{(j)} : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{L}_i^{(j)}$  to  $P_i$  for  $i \in [n], j \in [m]$ , then the resulting MPRE has leakage  $L_i : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{L}_i^{(1)} \times \dots \times \mathcal{L}_i^{(m)}$ ,

$$L_i(x_1, \dots, x_n) = (L_i^{(1)}(x_1, \dots, x_n), \dots, L_i^{(m)}(x_1, \dots, x_n)),$$

to the  $i$ -th party.

*Proof.* Say the MPRE for  $f_j : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$  has

- Local randomness spaces  $\mathcal{R}_1^{(j)}, \dots, \mathcal{R}_n^{(j)}$ ;
- (if any) Correlated randomness space  $\mathcal{R}_1^{(j)'} \times \dots \times \mathcal{R}_n^{(j)'}$  and the distribution  $\mathcal{D}^{(j)}$ ;
- Local preprocessing functions  $h_1^{(j)}, \dots, h_n^{(j)}$ ;
- Encoding function  $\hat{f}^{(j)} : \hat{\mathcal{X}}_1^{(j)} \times \dots \times \hat{\mathcal{X}}_n^{(j)} \rightarrow \hat{\mathcal{Y}}^{(j)}$
- Decoding function  $\text{Dec}^{(j)} : \hat{\mathcal{Y}}^{(j)} \rightarrow \mathcal{Y}^{(j)}$ ;
- (if any) Leakage functions  $L_i : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{L}_i^{(1)} \times \dots \times \mathcal{L}_i^{(m)}$ ;
- Simulator  $\text{Sim}^{(j)} = (\text{SimO}^{(j)}, \text{SimI}^{(j)})$ .

Then the scheme in Figure 4 is a MPRE for  $f$  whose encoding function is  $\hat{f}$ .

The correctness of the resulting MPRE follows from the correctness of the given MPREs. In short

$$\begin{aligned} & \text{Dec}(\hat{f}(\dots, h_i(x_i, r_i^{(1)}), \dots, r_i^{(m)}, r_i^{(1)'}, \dots, r_i^{(m)'}, \dots)) \\ &= \text{Dec}(\hat{f}(\dots, (h_i^{(1)}(x_i, r_i^{(1)}, r_i^{(1)'}), \dots, h_i^{(m)}(x_i, r_i^{(m)}, r_i^{(m)'}), \dots)) \\ &= \text{Dec}((\dots, h_i^{(1)}(x_i, r_i^{(1)}, r_i^{(1)'}), \dots), \dots, (\dots, h_i^{(m)}(x_i, r_i^{(m)}, r_i^{(m)'}), \dots)) \\ &= (\text{Dec}^{(1)}(\dots, h_i^{(1)}(x_i, r_i^{(1)}, r_i^{(1)'}), \dots), \dots, \text{Dec}^{(m)}(\dots, h_i^{(m)}(x_i, r_i^{(m)}, r_i^{(m)'}), \dots)) \\ &= (f^{(1)}(\dots, x_i, \dots), \dots, f^{(m)}(\dots, x_i, \dots)). \end{aligned}$$

The adaptive  $t$ -privacy is proved by hybrid argument. For  $j = 0, 1, \dots, m$ , define the  $j$ -th hybrid world as

- *In the  $j$ -th hybrid world.* The adversary chooses the input  $(x_1, \dots, x_n)$ , then the adversary's queries are answered as

MPRE yielded by parallel composition
<b>Local randomness:</b> $P_i$ samples $r_i^{(j)} \leftarrow \mathcal{R}_i^{(j)}$ for each $j \in [m]$
<b>Correlated randomness:</b> $P_i$ gets $r_i^{(j) \prime}$ such that $(r_1^{(j) \prime}, \dots, r_n^{(j) \prime})$ is sampled from $\mathcal{D}^{(j)}$ for each $j \in [m]$
<b>Preprocessing function <math>h_i</math>:</b>
$h_i(x_i, r_i^{(1)}, \dots, r_i^{(m)}, r_i^{(1) \prime}, \dots, r_i^{(m) \prime}) = (h_i^{(1)}(x_i, r_i^{(1)}, r_i^{(1) \prime}), \dots, h_i^{(m)}(x_i, r_i^{(m)}, r_i^{(m) \prime})).$
<b>Encoding function <math>\hat{f}</math>:</b>
$\hat{f}((\hat{x}_1^{(1)}, \dots, \hat{x}_1^{(m)}), \dots, (\hat{x}_n^{(1)}, \dots, \hat{x}_n^{(m)})) = (\hat{f}^{(1)}(\hat{x}_1^{(1)}, \dots, \hat{x}_n^{(1)}), \dots, \hat{f}^{(m)}(\hat{x}_1^{(m)}, \dots, \hat{x}_n^{(m)})).$
<b>Decoding function:</b> $\text{Dec}(\hat{y}^{(1)}, \dots, \hat{y}^{(m)}) = (\text{Dec}^{(1)}(\hat{y}^{(1)}), \dots, \text{Dec}^{(m)}(\hat{y}^{(m)})).$

Figure 4: Parallel Composition of MPRES

**On the outset:** For each  $j' \leq j$ , do the following:

Sample  $r_i^{(j')} \leftarrow \mathcal{R}_i^{(j')}$  for  $i \in [n]$ , sample  $(r_1^{(j') \prime}, \dots, r_n^{(j') \prime}) \leftarrow \mathcal{D}^{(j')}$ .

Compute  $\hat{x}_i^{(j')} = h_i^{(j')}(x_i, r_i^{(j')}, r_i^{(j') \prime})$  for  $i \in [n]$ .

Compute  $\hat{y}^{(j')} = \hat{f}^{(j')}(\hat{x}_1^{(j')}, \dots, \hat{x}_n^{(j')})$ .

**CorruptI( $i$ ):** For each  $j' > j$ , let  $(r_i^{(j')}, r_i^{(j') \prime}) \leftarrow \text{SimI}^{(j')}(i, x_i, L_i^{(j')}(x_1, \dots, x_n))$ .

Output  $(r_i^{(1)}, \dots, r_i^{(m)}, r_i^{(1) \prime}, \dots, r_i^{(m) \prime})$ .

**CorruptO:** For each  $j' > j$ , compute  $y^{(j')} = f^{(j')}(x_1, \dots, x_m)$ , let  $\hat{y}^{(j')} \leftarrow \text{SimO}^{(j')}(y^{(j')})$ .

Output  $(\hat{y}^{(1)}, \dots, \hat{y}^{(m)})$ .

The  $m$ -th hybrid is the real world, the 0-th hybrid is the ideal world. In particular, the simulator is

- *In the ideal world.* The simulator is the stateful algorithm

**Upon CorruptI( $i$ ), SimI( $i, x_i, (L_i^{(1)}, \dots, L_i^{(m)})$ ):** (Here  $(L_i^{(1)}, \dots, L_i^{(m)})$  is the leakage.)

Let  $(r_i^{(j)}, r_i^{(j) \prime}) \leftarrow \text{SimI}^{(j)}(i, x_i, L_i^{(j)})$  for each  $j \in [m]$ .

Output  $(r_i^{(1)}, \dots, r_i^{(m)}, r_i^{(1) \prime}, \dots, r_i^{(m) \prime})$ .

**Upon CorruptO, SimO( $y^{(1)}, \dots, y^{(m)}$ ):** Let  $\hat{y}^{(j)} \leftarrow \text{SimO}^{(j)}(y^{(j)})$  for each  $j \in [m]$ .

Output  $(\hat{y}^{(1)}, \dots, \hat{y}^{(m)})$ .

The security of the MPRE for  $f^{(j)}$  implies the indistinguishability between the  $j$ -th hybrid world and the  $(j - 1)$ -th hybrid world. □

## 4 MPRE for Degree-3 Polynomials

In this section, we build MPRE for degree-3 polynomials in two settings: (i) honest majority, and (ii) OLE correlations. Our road-map is as follows: In Section 4.1, we construct a 4-party gadget

MPRE; in Section 4.2, we construct an MPRE for the 3-party functionality  $3\text{MultPlus}_3$  described below, which computes a degree-3 monomial shifted by some linear terms, in the OLE-correlation model; then in Section 4.3, we consider the  $n$ -party version of the functionality  $3\text{MultPlus}_n$  and construct an MPRE for it in the honest majority setting.

$$\begin{aligned} 3\text{MultPlus}_3 &: ((x_1, \alpha), (x_2, \beta), (x_3, \gamma)) \mapsto x_1x_2x_3 + \alpha + \beta + \gamma \\ 3\text{MultPlus}_n &: ((x_1, \alpha), (x_2, \beta), (x_3, \gamma), \underbrace{\perp, \dots, \perp}_{n-3}) \mapsto x_1x_2x_3 + \alpha + \beta + \gamma \end{aligned}$$

Finally, in Section 4.4, we show that MPRE for the  $3\text{MultPlus}$  functionalities is sufficient for obtaining MPRE for general degree-3 functionalities. All our MPRE have effective degree 2.

#### 4.1 Our 4-Party Gadget MPRE with leakage

Fix a field  $\mathbb{F}$ . We begin with a MPRE with leakage for the following 4-party gadget function

$$((x, \mu), a, b, \nu) \mapsto abx + \mu + \nu.$$

For randomly sampled  $w_1, \dots, w_5$ , [IK00, IK02] show that  $(\phi_1, \dots, \phi_6)$  is a randomized encoding of  $abx + \mu + \nu$ , where  $\phi_1, \dots, \phi_6$  are defined as

$$\begin{aligned} \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} &:= \begin{bmatrix} 1 & w_1 & w_2 \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} a & & \mu + \nu \\ -1 & x & \\ & -1 & b \end{bmatrix} \begin{bmatrix} 1 & w_3 & w_4 \\ & 1 & w_5 \\ & & 1 \end{bmatrix} \\ &= \begin{bmatrix} a - w_1 & w_3a + w_1x - w_1w_3 - w_2 & w_1w_5x + w_2b + w_4a - w_2w_5 - w_1w_4 + \mu + \nu \\ -1 & x - w_3 & w_5x - w_4 \\ & -1 & b - w_5 \end{bmatrix}. \end{aligned} \quad (3)$$

[IK00, IK02] guarantee that  $\phi_1, \dots, \phi_5$  are i.i.d. uniform despite the value of  $(a, b, x, \mu, \nu)$ . We would like to transfer this randomized encoding into an effective degree-2 MPRE with leakage.

fully-private MPRE for the gadget functionality

**Local randomness:**  $P_1$  samples  $w_3, w'_2, w'_4 \leftarrow \mathbb{F}$ ;  $P_4$  samples  $w_1, w_5, w''_2, w''_4 \leftarrow \mathbb{F}$ .

**Preprocessing function  $h_i$ :**  $P_4$  locally computes  $w_1w_5$ , i.e. let  $h_4(w_1, w_5, w''_2, w''_4) = (w_1, w_5, w_1w_5, w''_2, w''_4)$ . Other parties simply outputs their local input and randomness, i.e.,  $h_i$  for  $i \neq 4$  is the identity function.

**Encoding function  $g$ :** On input the outputs of all  $h_i$ , output  $(\phi_1, \dots, \phi_6)$  as defined in Equation (3), with  $w_2 := w'_2 + w''_2$ ,  $w_4 := w'_4 + w''_4$ .

**Decoding function:** On input  $(\phi_1, \dots, \phi_6)$ , output  $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix}$ .

Figure 5: Effective degree-2 MPRE for the gadget functionality

**Lemma 4.1.** *The scheme defined in Figure 5 is an MPRE for the following 4-party gadget function*

$$((x, \mu), a, b, \nu) \mapsto abx + \mu + \nu$$

*with the following properties:*

(I) it has effective degree 2;

(II) tolerates any number of corruptions with leakage  $L_4((x, \mu), a, b, \nu) = (a, b)$ .

*Proof.* The correctness is straight forward. The decoding function is the determinant of the matrix in (3), thus

$$\text{Dec}(\phi_1, \dots, \phi_6) = \det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = \det \begin{bmatrix} a & & \mu + \nu \\ -1 & x & \\ & -1 & b \end{bmatrix} = abx + \mu + \nu.$$

For the adaptive privacy, we need to define the simulator.

- *In the real world:* For input  $x, a, b, \mu, \nu$

**At the outset:** Sample random  $w_1, w_3, w_5, w'_2, w''_2, w'_4, w''_4$ , compute  $w_2 = w'_2 + w''_2, w_4 = w'_4 + w''_4$ , compute  $(\phi_1, \dots, \phi_6)$  according to Equation (3).

**CorruptO:** Output  $\phi_1, \dots, \phi_6$ .

**Corruptl(1):** Output  $w_3, w'_2, w'_4$ .

**Corruptl(2):** Output  $\perp$ .

**Corruptl(3):** Output  $\perp$ .

**Corruptl(4):** Output  $w_1, w_5, w''_2, w''_4$ .

- *In the ideal world:*

**At the outset:** Sample random  $\phi_1, \dots, \phi_5$ ,

**Upon CorruptO, SimO( $y$ ):** Let  $\phi_6$  be the unique value that  $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = y$ . Output  $\phi_1, \dots, \phi_6$ .

**Upon Corruptl(1), Siml(1, ( $x, \mu$ )):** Set  $w_3$  as the unique value that  $\phi_3 = x - w_3$ .

If  $P_4$  is not corrupted yet, sample  $w'_2, w'_4$  at random.

If  $P_4$  is already corrupted, subroutine **Siml(4,  $\nu, (a, b)$ )** has learned  $a$  and has sampled the values of  $w_1, w_5$ . Then, set  $w_2, w_4$  to satisfy  $\phi_2 = w_3a + w_1x - w_1w_3 - w_2, \phi_4 = w_5x - w_4$ , and set  $w'_2 = w_2 - w''_2, w'_4 = w_4 - w''_4$ .

Output  $w_3, w'_2, w'_4$ .

**Upon Corruptl(2), Siml(2,  $a$ ):** Output  $\perp$ .

**Upon Corruptl(3), Siml(3,  $b$ ):** Output  $\perp$ .

**Upon Corruptl(4), Siml(4,  $\nu, (a, b)$ ):** Set  $w_1, w_5$  to satisfy  $\phi_1 = a - w_1, \phi_5 = b - w_5$ .

If  $P_1$  is not corrupted yet, sample  $w''_2, w''_4$  at random.

If  $P_1$  is already corrupted, subroutine **Siml(1, ( $x, \mu$ ))** has learned  $x$  and has sampled the value of  $w_3$ . Then, set  $w_2, w_4$  to satisfy  $\phi_2 = w_3a + w_1x - w_1w_3 - w_2$  and  $\phi_4 = w_5x - w_4$ , set  $w''_2 = w_2 - w'_2, w''_4 = w_4 - w'_4$ .

Output  $w_1, w_5, w''_2, w''_4$ .

To formally show that adversary cannot distinguish between the real world and the ideal world, we introduce a middle world.

- *In the middle world:*

**At the outset:** Sample random  $\phi_1, \dots, \phi_5$ .

Let  $\phi_6$  be the unique value that  $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = abx + \mu + \nu$ .

Solve  $w_1, \dots, w_5$  from Equation (3). Sample  $w'_2, w''_2$  as additive sharing of  $w_2$ , Sample  $w'_4, w''_4$  as additive sharing of  $w_4$ .

**CorruptO:** Output  $\phi_1, \dots, \phi_6$ .

**Corruptl(1):** Output  $w_3, w'_2, w'_4$ .

**Corruptl(2):** Output  $\perp$ .

**Corruptl(3):** Output  $\perp$ .

**Corruptl(4):** Output  $w_1, w_5, w''_2, w''_4$ .

The real world is indistinguishable from the middle world, due to the security of the randomized encoding in (3).

Comparing the ideal world with the middle world, the only difference is that the computation is deferred in the ideal world: Same as the real world, the simulator in the ideal samples random  $\phi_1, \dots, \phi_5$ . But the simulator cannot compute  $w_1, \dots, w_5$  at the beginning as it doesn't know  $a, b, x, \mu, \nu$  at that moment. Instead, the simulator compute each of  $w_1, \dots, w_5$  once it has the necessary information, using exactly the method as the middle world (i.e. by solving (3)). Thus the ideal world is also indistinguishable from the middle world.  $\square$

## 4.2 MPRE for 3-Party 3MultPlus using OLE correlation

In this section, we construct an MPRE for the three party functionality

$$3\text{MultPlus}_3 : ((x_1, \alpha), (x_2, \beta), (x_3, \gamma)) \mapsto x_1x_2x_3 + \alpha + \beta + \gamma$$

that has effective degree 2 and tolerates any number of corruptions in the OLE-correlation model.

For randomly sampled  $w_1, \dots, w_5$ , [IK00, IK02] show that  $(\phi_1, \dots, \phi_6)$  is a randomized encoding of  $x_1x_2x_3 + \alpha + \beta + \gamma$ , where  $\phi_1, \dots, \phi_6$  are defined as

$$\begin{aligned} \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} &:= \begin{bmatrix} 1 & w_1 & w_2 \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} x_1 & & \alpha + \beta + \gamma \\ -1 & x_2 & \\ & -1 & x_3 \end{bmatrix} \begin{bmatrix} 1 & w_3 & w_4 \\ & 1 & w_5 \\ & & 1 \end{bmatrix} \\ &= \begin{bmatrix} x_1 - w_1 & \begin{pmatrix} w_3x_1 + w_1x_2 \\ -w_1w_3 - w_2 \end{pmatrix} & \begin{pmatrix} w_1w_5x_2 + w_2x_3 + w_4x_1 \\ -w_2w_5 - w_1w_4 + \alpha + \beta + \gamma \end{pmatrix} \\ -1 & x_2 - w_3 & w_5x_2 - w_4 \\ & -1 & x_3 - w_5 \end{bmatrix}. \end{aligned} \tag{4}$$

[IK00, IK02] guarantee that  $\phi_1, \dots, \phi_5$  are i.i.d. uniform despite the value of  $(x_1, x_2, x_3, \alpha + \beta + \gamma)$ . We would like to transfer this randomized encoding into an effective degree-2 MPRE using OLE correlated randomness.

Notice that  $w_1w_5x_2$  is the only degree-3 monomial in the randomized encoding, and  $w_1, w_5$  belong to the randomness of the randomized encoding. Thus if  $w_1, w_5$  are sampled from OLE correlated randomness, monomial  $w_1w_5x_2$  can be transferred into a degree-2 term. More precisely, let  $(w_1, b^{(1)}, w_5, b^{(3)}) \in \mathbb{F}^4$  be sampled from OLE correlation, it holds that  $w_1w_5 = b^{(1)} + b^{(3)}$ . The marginal distribution of  $(w_1, w_5)$  is still uniform; and  $w_1w_5x_2$  equals  $(b^{(1)} + b^{(3)})x_2$ , which is a

degree-2 term. Then the randomized encoding has “effective” degree 2 as it can be computed from

$$\begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = \begin{bmatrix} x_1 - w_1 & \begin{pmatrix} w_3 x_1 + w_1 x_2 \\ -w_1 w_3 - w_2 \end{pmatrix} & \begin{pmatrix} (b^{(1)} + b^{(3)})x_2 + w_2 x_3 + w_4 x_1 \\ -w_2 w_5 - w_1 w_4 + \alpha + \beta + \gamma \end{pmatrix} \\ -1 & x_2 - w_3 & w_5 x_2 - w_4 \\ & -1 & x_3 - w_5 \end{bmatrix}. \quad (5)$$

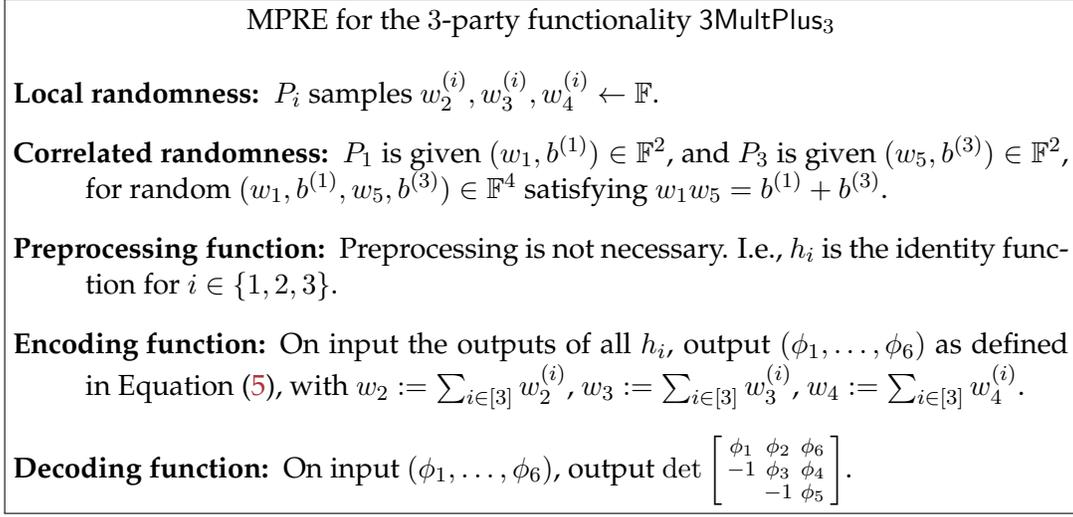


Figure 6: Effective degree-2 MPRE for the 3MultPlus<sub>3</sub> functionality

**Lemma 4.2.** *The MPRE in Figure 6 for the 3-party functionality 3MultPlus<sub>3</sub> has effective degree 2 and tolerates any number of corruptions, in the OLE-correlation model.*

*Proof.* The correctness is straight forward,

$$\text{Dec}(\phi_1, \dots, \phi_6) = \det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = \det \begin{bmatrix} x_1 & & \alpha + \beta + \gamma \\ -1 & x_2 & \\ & -1 & x_3 \end{bmatrix} = x_1 x_2 x_3 + \alpha + \beta + \gamma.$$

For the adaptive privacy, we need to define the simulator.

- *In the real world:* For input  $x_1, x_2, x_3, \alpha, \beta, \gamma$

**At the outset:** Sample random  $w_1, w_5, w_2^{(1)}, w_2^{(2)}, w_2^{(3)}, w_3^{(1)}, w_3^{(2)}, w_3^{(3)}, w_4^{(1)}, w_4^{(2)}, w_4^{(3)} \in \mathbb{F}$ , sample random  $b^{(1)}, b^{(3)}$  that  $b^{(1)} + b^{(3)} = w_1 w_5$ , compute  $w_2 = \sum_{i \in [3]} w_2^{(i)}$ ,  $w_3 = \sum_{i \in [3]} w_3^{(i)}$ ,  $w_4 = \sum_{i \in [3]} w_4^{(i)}$ , compute  $(\phi_1, \dots, \phi_6)$  according to Equation (5).

**CorruptO:** Output  $\phi_1, \dots, \phi_6$ .

**Corruptl(1):** Output  $w_1, b^{(1)}, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}$ .

**Corruptl(2):** Output  $w_2^{(2)}, w_3^{(2)}, w_4^{(2)}$ .

**Corruptl(3):** Output  $w_5, b^{(3)}, w_2^{(3)}, w_3^{(3)}, w_4^{(3)}$ .

- *In the ideal world:*

**At the outset:** Sample random  $\phi_1, \dots, \phi_5$ ,

**Upon CorruptO, SimO( $y$ ):** Let  $\phi_6$  be the unique value that  $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ -1 & \phi_5 & \end{bmatrix} = y$ . Output  $\phi_1, \dots, \phi_6$ .

**Upon Corruptl(1), Siml(1, ( $x_1, \alpha$ )):** Set  $w_1$  to satisfy  $\phi_1 = x_1 - w_1$ .

If  $P_3$  is not corrupted yet, sample  $b^{(1)}$  at random.

If  $P_3$  is already corrupted, subroutine **Siml(3, ( $x_3, \gamma$ ))** has set the values of  $w_5, b^{(3)}$ . Set  $b^{(1)} = w_1 w_5 - b^{(3)}$ .

If both  $P_2$  and  $P_3$  are corrupted, subroutines **Siml(2, ( $x_2, \beta$ ))**, **Siml(3, ( $x_3, \gamma$ ))** have set  $w_j^{(2)}, w_j^{(3)}$  for  $j \in \{2, 3, 4\}$ . Then solve  $w_2, w_3, w_4$  from Equation (4) and set  $w_j^{(1)} = w_j - w_j^{(2)} - w_j^{(3)}$  for  $j \in \{2, 3, 4\}$ .

If at least one of  $P_2, P_3$  is not corrupted yet, sample  $w_2^{(1)}, w_3^{(1)}, w_4^{(1)} \in \mathbb{F}$ .

Output  $w_1, b^{(1)}, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}$ .

**Upon Corruptl(2), Siml(2, ( $x_2, \beta$ )):** If both  $P_1$  and  $P_3$  are corrupted, subroutines **Siml(1, ( $x_1, \beta$ ))**, **Siml(3, ( $x_3, \gamma$ ))** have set  $w_j^{(1)}, w_j^{(3)}$  for  $j \in \{2, 3, 4\}$ . Then solve  $w_2, w_3, w_4$  from Equation (4) and set  $w_j^{(2)} = w_j - w_j^{(1)} - w_j^{(3)}$  for  $j \in \{2, 3, 4\}$ .

If at least one of  $P_1, P_3$  is not corrupted yet, sample  $w_2^{(2)}, w_3^{(2)}, w_4^{(2)} \in \mathbb{F}$ .

Output  $w_2^{(2)}, w_3^{(2)}, w_4^{(2)}$ .

**Upon Corruptl(3), Siml(3, ( $x_3, \gamma$ )):** Set  $w_5$  to satisfy  $\phi_5 = x_3 - w_5$ .

If  $P_1$  is not corrupted yet, sample  $b^{(3)}$  at random.

If  $P_1$  is already corrupted, subroutine **Siml(1, ( $x_1, \alpha$ ))** has set the values of  $w_1, b^{(1)}$ . Set  $b^{(3)} = w_1 w_5 - b^{(1)}$ .

If both  $P_1$  and  $P_2$  are corrupted, subroutines **Siml(1, ( $x_1, \beta$ ))**, **Siml(2, ( $x_2, \gamma$ ))** have set  $w_j^{(1)}, w_j^{(2)}$  for  $j \in \{2, 3, 4\}$ . Then solve  $w_2, w_3, w_4$  from Equation (4) and set  $w_j^{(3)} = w_j - w_j^{(1)} - w_j^{(2)}$  for  $j \in \{2, 3, 4\}$ .

If at least one of  $P_1, P_2$  is not corrupted yet, sample  $w_2^{(3)}, w_3^{(3)}, w_4^{(3)} \in \mathbb{F}$ .

Output  $w_5, b^{(3)}, w_2^{(3)}, w_3^{(3)}, w_4^{(3)}$ .

To show the indistinguishability between the real world and the ideal world, we introduce a middle world.

- *In the middle world:* For input  $x_1, x_2, x_3, \alpha, \beta, \gamma$

**At the outset:** Sample random  $\phi_1, \dots, \phi_5$ .

Let  $\phi_6$  be the unique value that  $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ -1 & \phi_5 & \end{bmatrix} = y$ .

Solve  $w_1, \dots, w_5$  from Equation (4).

Sample random  $b^{(1)}, b^{(3)}$  that  $b^{(1)} + b^{(3)} = w_1 w_5$ . For each of  $j \in \{2, 3, 4\}$ , sample random  $w_j^{(1)}, w_j^{(2)}, w_j^{(3)}$  that  $w_j^{(1)} + w_j^{(2)} + w_j^{(3)} = w_j$ .

**CorruptO:** Output  $\phi_1, \dots, \phi_6$ .

**Corruptl(1):** Output  $w_1, b^{(1)}, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}$ .

**Corruptl(2):** Output  $w_2^{(2)}, w_3^{(2)}, w_4^{(2)}$ .

**Corruptl(3):** Output  $w_5, b^{(3)}, w_2^{(3)}, w_3^{(3)}, w_4^{(3)}$ .

The real world is indistinguishable from the middle world, due to the security of the randomized encoding in (3).

Comparing the ideal world with the middle world, the only difference is that the computation is deferred in the ideal world: Same as the real world, the simulator in the ideal samples random  $\phi_1, \dots, \phi_5$ . But the simulator cannot compute  $w_1, \dots, w_5$  by solving (4) at the beginning as it doesn't know  $x_1, x_2, x_3, \alpha, \beta, \gamma$  at that moment. Instead, the simulator compute  $w_1$  once it knows  $x_1$ ; compute  $w_5$  once it knows  $x_3$ ; and compute  $w_2, w_3, w_4$  once it knows all the inputs. Thus the ideal world is also indistinguishable from the middle world.  $\square$

### 4.3 MPRE for $n$ -Party 3MultPlus with Honest Majority

We construct an MPRE (Figure 7) for the  $n$ -party functionality

$$3\text{MultPlus}_n : ((x_1, \alpha), (x_2, \beta), (x_3, \gamma), \underbrace{\perp, \dots, \perp}_{n-3}) \mapsto x_1 x_2 x_3 + \alpha + \beta + \gamma$$

that has effective degree 2 and tolerates minority corruptions. The construction requires  $|\mathbb{F}| > n$ .

**Additional notations.** Let  $\mathbb{F}$  be a field that  $|\mathbb{F}| > n$ , let  $1, \dots, n$  denote  $n$  distinct non-zero elements in  $\mathbb{F}$ . Denote by  $\mathcal{P}(t, m)$  the set of degree- $t$  polynomials  $P$  with constant term  $m$  over  $\mathbb{F}$ , so that  $Q \leftarrow \mathcal{P}(t, m)$  refers to sampling a random degree- $t$  polynomial  $Q$  whose constant term is  $m$ . In addition,  $m = \text{rec}(t, (i_1, \sigma_1) \dots, (i_{t+1}, \sigma_{t+1}))$  denotes the procedure for reconstructing the constant term from  $t + 1$  points on the polynomial via interpolation. For convenience, we also denote by  $\mathcal{P}(t, m) \mid (i_1, \sigma_1) \dots, (i_s, \sigma_s)$  the set of polynomials  $Q \in \mathcal{P}(t, m)$  such that  $Q(i_1) = \sigma_1, \dots, Q(i_s) = \sigma_s$ , for  $s \leq t + 1$ .

**Protocol overview.** We decompose the computation of  $x_1 x_2 x_3 + \alpha + \beta + \gamma$ , into two parts  $x_1 x_2 x_3 + z + s$  and  $\alpha + \beta + \gamma - z - s$  where  $z$  is sampled by  $P_1$  and  $s$  is jointly sampled by all  $n$  parties. Since the second term is linear, we focus on designing an MPRE for the first part.

- $P_1$  samples  $z \leftarrow \mathbb{F}, Z \leftarrow \mathcal{P}(n - 1, z)$ .
- $P_2$  samples  $Q_2 \leftarrow \mathcal{P}(t, x_2)$  and  $P_3$  samples  $Q_3 \leftarrow \mathcal{P}(t, x_3)$ .
- $P_i$  samples  $S(i) \leftarrow \mathbb{F}$ , for every  $i \in [n]$ . Let  $s = \text{rec}(n - 1, (1, S(1)), \dots, (n, S(n)))$ .

Observe that

$$Y := x_1 Q_2 Q_3 + Z + S \in \mathcal{P}(n - 1, x_1 x_2 x_3 + z + s).$$

Here, we rely on the fact that  $2t \leq n - 1$ . Then, for each  $i = 1, 2, \dots, n$ , parties  $P_1, P_2, P_3, P_i$  can run the gadget MPRE described in Section 4.1 to compute  $Y(i)$

$$((x_1, Z(i)), Q_2(i), Q_3(i), S(i)) \mapsto x_1 Q_2(i) Q_3(i) + Z(i) + S(i) = Y(i),$$

from which the output party can reconstruct  $Y$  and the constant term  $x_1 x_2 x_3 + z + s$ .

*Security Intuition.* We can prove security of this protocol for up to  $t < n/2$  corruptions. Consider two cases: If  $P_1$  is not corrupted, or corrupted. In the first case, the view of the output party consists of  $\alpha + \beta + \gamma - z - s$ , and a degree  $n - 1$  polynomial  $Y$  with constant term  $x_1 x_2 x_3 + z + s$ ,

which is random thanks to the randomization via  $Z$ . Now, suppose the adversary additionally corrupts  $t$  parties, excluding  $P_1$ ; call this set of parties  $T$ . Then, security of the gadget MPRE tells us that the adversary also learns  $\{Q_2(i), Q_3(i), S(i) : i \in T\}$ . Suppose for now  $2, 3 \notin T$ . By the property of Shamir's secret sharing, this leaks no additional information about  $x_1, x_2, x_3$  to the adversary. Now, if  $2 \in T$ , then the adversary also learns  $Q_2$ , but that is okay since it already learns  $x_2$ ; the same argument applies to  $3 \in T$ .

In the second case that  $P_1$  is corrupted and adversary learns  $x_1$  and all  $Z(i)$ 's, the polynomial  $Y$  is still a random degree- $(n-1)$  polynomial with constant term  $x_1x_2x_3 + z + s$  thanks to the randomization via  $S$ . If the adversary corrupts at set  $T$  of  $t$  parties, including  $P_1$ , and learns  $\{Q_2(i), Q_3(i), S(i) : i \in T\}$ , Shamir's secret sharing, again protects  $x_2, x_3$  from being leaked to the adversary.

**Protocol Specification** In short, the MPRE  $\hat{F}$  for  $x_1x_2x_3 + \alpha + \beta + \gamma$  simply computes  $n$  4-party gadget MPRE,

$$\hat{f}((x_1, Z(i)), Q_2(i), Q_3(i), S(i)) \text{ for all } i \in [n]$$

together with the linear term  $\alpha + \beta + \gamma + z + s$ . A formal description is in Figure 7. It is easy to see that  $\hat{F}$  has effective degree 2 since  $\hat{f}$  has effective degree  $\leq 2$ .

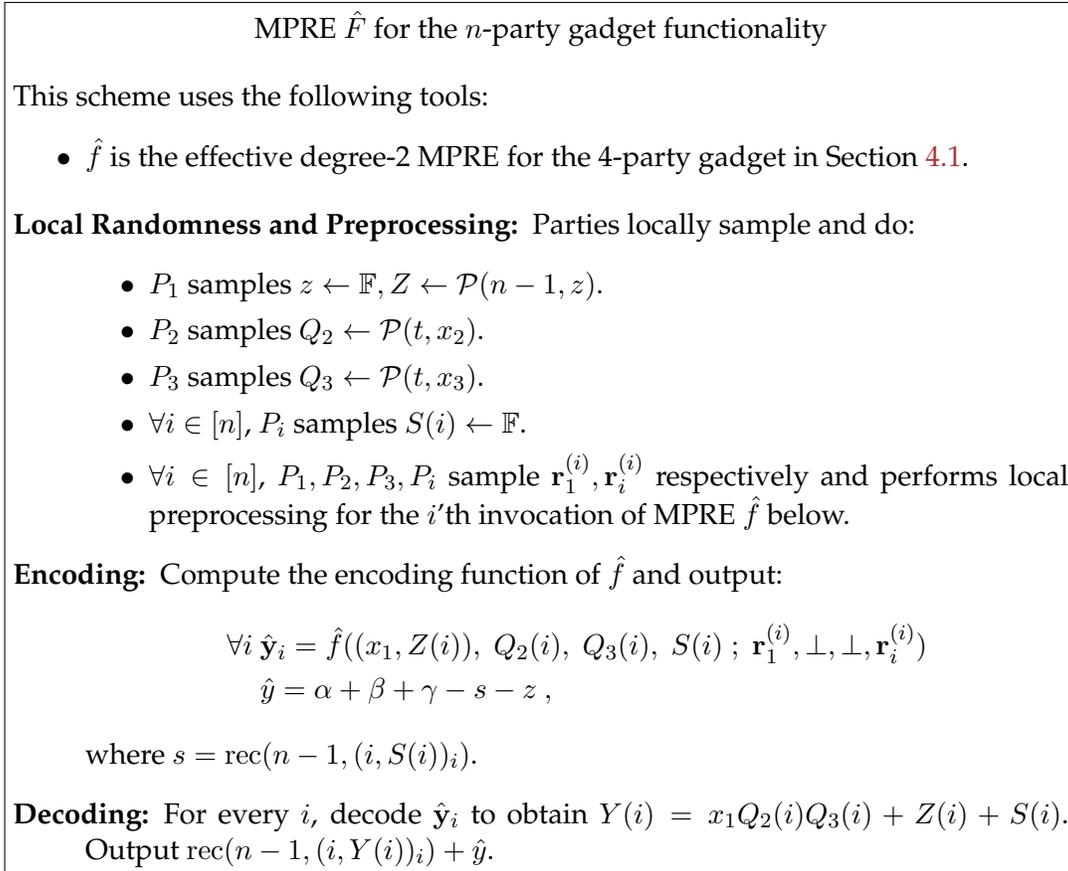


Figure 7: Effective degree-2 MPRE  $\hat{F}$  for the  $n$ -party gadget functionality

**Lemma 4.3.** *The MPRE scheme in Figure 7 for the  $n$ -party functionality  $3\text{MultPlus}_n$  has effective degree 2 and satisfies  $t$ -adaptive privacy for  $t < n/2$ . The construction requires  $|\mathbb{F}| > n$ .*

**Simulator.** Observe that the MPRE  $\hat{F}$  invokes the 4-party gadget MPRE  $\hat{f}$  for  $n$  times and computes a linear function  $\ell$ . By the adaptive security of  $\hat{f}$  and Theorem 3.3, we have that the parallel composition of all  $n$  invocations of  $\hat{f}$  and the linear function  $\ell$  is an MPRE  $\hat{G}$  for the following composed functionality  $G$ :

$$G : \left( (x_1, \alpha, (Z(i))_i, S(1)), (x_2, \beta, (Q_2(i))_i, S(2)), (x_3, \gamma, (Q_3(i))_i, S(3)), \dots, S(i), \dots, S(n) \right) \\ \mapsto \alpha + \beta + \gamma - s - z, (Y(i))_i .$$

The leakage function of  $\hat{f}$  gives the leakage function of  $\hat{G}$ , which is  $L_i$  leaking  $(Q_2(i), Q_3(i))$  to  $P_i$  for every  $i$ .  $\hat{G}$  is secure against  $t < n/2$  adaptive corruption. Let  $(\text{Siml}_G, \text{SimO}_G)$  be its simulator. Below, we use this simulator to construct the simulator  $(\text{Siml}_F, \text{SimO}_F)$  for  $\hat{F}$ .

*Overview.* The encoding of  $\hat{F}$  consists of encoding of  $\hat{f}$  and the output of  $\ell$  with appropriate input / output. The job of  $(\text{Siml}_F, \text{SimO}_F)$  is: 1) simulate the input / output of calls to  $\hat{G}$ , i.e., calls to  $\hat{f}$  and  $\ell$ , and 2) invoke  $(\text{Siml}_G, \text{SimO}_G)$  to simulate the encoding and local randomness of all calls to  $\hat{f}$  and  $\ell$ . Task 1) requires simulating  $Y(i)$ , a random  $n$ -out-of- $n$  Shamir sharing of  $x_1x_2x_3 + z + s$  belonging to the output of encoding, all  $Z(i)$  belonging to  $P_1$ , each  $S(i)$  belonging to  $P_i$ , and each  $Q_2(i), Q_3(i)$  belonging to  $P_2, P_3$  respectively, and leaked to  $P_i$ . Consistency between  $Y(i)$  and  $Z(i), S(i)$  is maintained by “programming” the variable that is simulated the last. This can be done as  $S(i), Z(i)$  are all marginally random and provide enough degree of freedom for programming even if all parties were corrupted. Consistency between simulating  $Q_2(i), Q_3(i)$  when  $P_2, P_3$  are corrupted and when  $P_i$  is corrupted can be maintained, thanks to the fact that at most  $t$  parties are corrupted and  $Q_2, Q_3$  have degree  $t$  with constant term  $x_2, x_3$ .

*Proof of Lemma 4.3.* We start with the formal description of the simulator.

**Upon CorruptO, SimO**( $y = x_1x_2x_3 + \alpha + \beta + \gamma$ ):

- Sample  $\tau \leftarrow \mathbb{F}$ .
- $\forall i$ , if  $P_1, P_i$  are already corrupted,  $Y(i) = x_1Q_2(i)Q_3(i) + Z(i) + S(i)$  is fixed.
- Sample  $O \leftarrow \mathcal{P}(n-1, \tau) \mid (i_1, Y(i_1)), \dots, (i_s, Y(i_s))$  conditioned on the list of fixed  $(i_j, Y(i_j))$ 's from previous step. (Note that  $s \leq t$  points are fixed.)

Send to adversary  $\text{SimO}_G(y - \tau, (Y(i))_i)$ .

**Upon Corruptl(1), Siml**( $x_1, \alpha$ ):

- Sample  $S(1) \leftarrow \mathbb{F}$ .
- $\forall i$ , if  $P_i$  and the output party are already corrupted, find the unique  $Z(i)$  that satisfies the equation  $Y(i) = x_1Q_2(i)Q_3(i) + Z(i) + S(i)$ .

Send to adversary  $\text{Siml}_G(x_1, \alpha, (Z(i))_i, S(1))$ .

**Upon Corruptl(2), Siml**( $x_2, \beta$ ):

- $\forall i$ , if  $P_i$  is already corrupted,  $Q_2(i)$  is already fixed.
- Sample  $Q_2 \leftarrow \mathcal{P}(t, x_2) \mid (i_1, Q_2(i_1)), \dots, (i_s, Q_2(i_s))$ , conditioned on the list of fixed  $(i_j, Q_2(i_j))$ . (Note that this can be done as  $s \leq t$  points are fixed, and  $Q_3$  has degree  $t$ .)
- if  $P_1$  and the output party are already corrupted, find the unique  $S(2)$  that satisfies the equation  $Y(2) = x_1Q_2(2)Q_3(2) + Z(2) + S(2)$ .

Send to adversary  $\text{Siml}_G(x_2, \beta, (Q_2(i))_i, S(2))$ .

**Upon Corruptl(3),  $\text{Siml}(x_3, \gamma)$ :** Same as in  $\text{Siml}(x_2, \beta)$ :

- $\forall i$ , if  $P_i$  is already corrupted,  $Q_3(i)$  is already fixed.
- Sample  $Q_3 \leftarrow \mathcal{P}(t, x_3) \mid (i_1, Q_3(i_1)), \dots, (i_s, Q_3(i_s))$ , conditioned on the list of fixed  $(i_j, Q_3(i_j))$ .
- if  $P_1$  and the output party are already corrupted, find the unique  $S(3)$  that satisfies the equation  $Y(3) = x_1 Q_2(3) Q_3(3) + Z(3) + S(3)$ .

Send to adversary  $\text{Siml}_G(x_3, \gamma, (Q_3(i))_i, S(3))$ .

**Upon Corruptl( $i$ ),  $\text{Siml}(\perp)$  for  $i \notin \{1, 2, 3\}$ :**

- If  $P_2$  and/or  $P_3$  is already corrupted,  $Q_2$  and/or  $Q_3$  are fixed. Otherwise, sample  $Q_2(i), Q_3(i) \leftarrow \mathbb{F}$ .
- Sample  $Q_3 \leftarrow \mathcal{P}(t, x_3) \mid (i_1, Q_3(i_1)), \dots, (i_s, Q_3(i_s))$ , conditioned on the list of fixed  $(i_j, Q_3(i_j))$ .
- if  $P_1$  and the output party are already corrupted, find the unique  $S(i)$  that satisfies the equation  $Y(i) = x_1 Q_2(i) Q_3(i) + Z(i) + S(i)$ .

Send to adversary  $\text{Siml}_G(S(i), Q_2(i), Q_3(i))$ .

**Correctness of Simulation** We argue that the view of the adversary in the real world and simulation are identically distributed following from the simulation security of  $\hat{G}$  and the fact that the input / output of the invocation of  $\hat{G}$  are simulated perfectly.

*Hybrid.* More formally, consider the following hybrid, where input / output of the invocation of  $\hat{G}$  is generated at the beginning as in the real world, while the encoding of  $\hat{G}$  is still simulated.

**At the outset:** With knowledge of  $x_1, x_2, x_3, \alpha, \beta, \gamma$ .

- $\forall i$ , sample  $Z(i) \leftarrow \mathbb{F}$ . Let  $z = Z(0)$ .
- Sample  $Q_2 \leftarrow \mathcal{P}(t, x_2)$ .
- Sample  $Q_3 \leftarrow \mathcal{P}(t, x_3)$ .
- $\forall i$ , sample  $S(i) \leftarrow \mathbb{F}$ . Let  $s = S(0)$ .
- $\forall i$ , compute  $Y(i) = x_1 Q_2(i) Q_3(i) + Z(i) + S(i)$ .
- Compute  $\tau = x_1 x_2 x_3 + z + s$ , and  $y = x_1 x_2 x_3 + \alpha + \beta + \gamma$ .

**Upon CorruptO:** Send to adversary  $\text{SimO}_G(y - \tau, (Y(i))_i)$ .

**Upon Corruptl(1):** Send to adversary  $\text{Siml}_G(x_1, \alpha, (Z(i))_i, S(1))$ .

**Upon Corruptl(2):** Send to adversary  $\text{Siml}_G(x_2, \beta, (Q_2(i))_i, S(2))$ .

**Upon Corruptl(3):** Send to adversary  $\text{Siml}_G(x_3, \gamma, (Q_3(i))_i, S(3))$ .

**Upon Corruptl( $i$ ):** Send to adversary  $\text{Siml}_G(S(i), Q_2(i), Q_3(i))$ .

The only difference between the above hybrid and the real world is whether the encoding of  $\hat{G}$  is simulated or not, it follows from the security of  $\hat{G}$  that the views of the adversary are identically distributed. The only difference between the hybrid and the simulation is whether the input / output of the call to  $\hat{G}$  is generated at the beginning with knowledge of  $x_1, x_2, x_3, \alpha, \beta, \gamma$  or generated in a delayed fashion. Since these two ways of generation yield the same distribution, the hybrid and simulation are also identically distributed. We conclude that the real world and simulation are identically distributed.  $\square$

#### 4.4 dMultPlus is Complete among Degree- $d$ Functionalities

We show that the following  $d$ -party functionality dMultPlus from [GIS18, BGI<sup>+</sup>18] is complete.

$$\text{dMultPlus} : (x_1, z_1), \dots, (x_d, z_d) \mapsto x_1 \cdots x_d + z_1 + \cdots + z_d.$$

It is complete for degree- $d$  functionalities in the following sense: if there is a MPRE for dMultPlus, then there is a MPRE for any degree- $d$  functionality that has the same complexity. Since we already have sequential composition (Theorem 3.2), it is sufficient to prove the following lemma.

**Lemma 4.4.** *Let  $d \in \{2, 3\}$ . For any degree- $d$  functionality  $f$  over a field  $\mathbb{F}$ , there exists a MPRE  $(\hat{f}, h_1, \dots, h_n)$  for  $f$ , such that*

- *the MPRE tolerates arbitrary number of adaptive corruptions;*
- *every output bit of the MPRE is either linear or of the form<sup>1</sup>*

$$(x_1, z_1), \dots, (x_d, z_d) \mapsto x_1 \cdots x_d + z_1 + \cdots + z_d.$$

*Proof.* By the parallel composition (Theorem 3.3), we can assume w.l.o.g. that  $f$  is a polynomial. Since we are considering fully adaptive security, we can assume w.l.o.g. that each party only has one input bit. I.e. the input is  $(x_1, \dots, x_n) \in \mathbb{F}^n$ , and the output of  $f$  is

$$f(x_1, \dots, x_n) = \sum_{j=1}^m c_j x_{i_{j,1}} \cdots x_{i_{j,d}}.$$

Let  $\mathcal{I}_j := \{i_{j,1}, \dots, i_{j,d}\}$ .

Then Figure 8 construct an MPRE for  $f$ . Its correctness follows from

$$\sum_{j=1}^m c_j u_j - s = \sum_{j=1}^m c_j (x_{i_{j,1}} \cdots x_{i_{j,d}} + z_{j,i_{j,1}} + \cdots + z_{j,i_{j,d}}) - s = \sum_{j=1}^m c_j x_{i_{j,1}} \cdots x_{i_{j,d}} = f(x_1, \dots, x_n).$$

The privacy is guaranteed by the following simulator.

**At the outset:** Samples random  $u_1, \dots, u_m$ .

**Upon Corrupt $l(i)$ , Sim $l(i, x_i)$ :** For every  $j$  such that  $i \in \mathcal{I}_j$ ,

- If there exists another party in  $\mathcal{I}_j$  that is not corrupted, sample  $z_{j,i}$  at random.
- If all other parties in  $\mathcal{I}_j$  are corrupted, then  $z_{j,\iota}$  has been set by subroutine Sim $l(\iota, x_\iota)$  for all  $\iota \in \mathcal{I}_j \setminus \{i\}$ . set  $z_{j,i} = u_j - \sum_{\iota \in \mathcal{I}_j \setminus \{i\}} z_{j,\iota}$ .

**Output**  $(z_{j,i})_{j \text{ s.t. } i \in \mathcal{I}_j}$ .

**Upon Corrupt $O$ , Sim $O(y)$ :** Set  $s = \sum_{j=1}^m c_j u_j - y$ .

**Output**  $(u_1, \dots, u_m, s)$ .  $\square$

<sup>1</sup>This function is called dMultPlus in [GIS18].

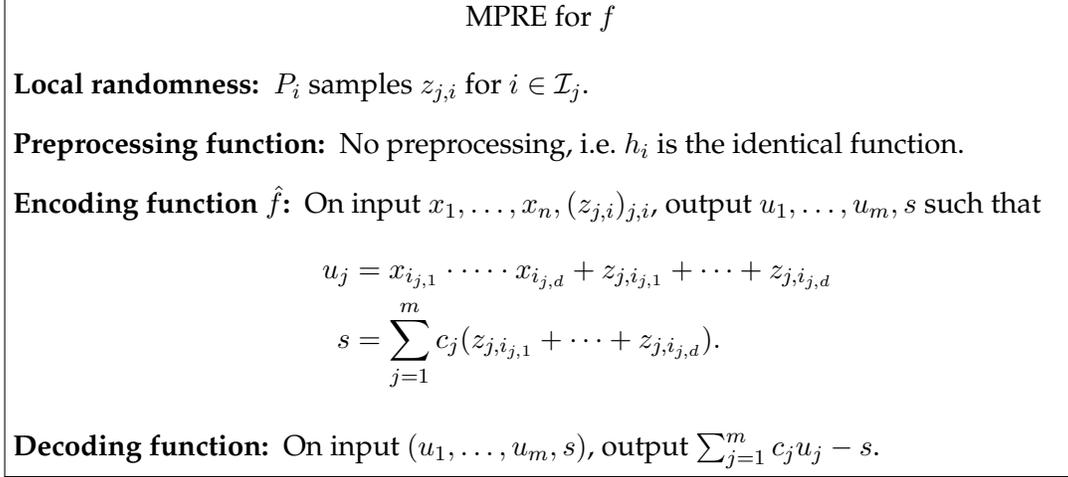


Figure 8: MPRE for degree- $k$  functionalities

## 5 MPRE for $\text{NC}^1$ and $\text{P/poly}$

We lift our effective degree-2 MPRE for degree-3 functionalities constructed in the previous section, to MPRE for  $\text{NC}^1$  and  $\text{P/poly}$ . The transformation uses the former MPRE to compute degree-3 randomized encodings for  $\text{NC}^1$  [IK02] and for  $\text{P/poly}$  [Yao82], and preserves the effective degree. The resulting effective-degree-2 MPRE for  $\text{NC}^1$  is information theoretically secure and tolerates any adaptive corruptions, while the resulting MPRE for  $\text{P/poly}$  is computationally secure making black box access to a PRG, and tolerates  $n - 1$  adaptive corruptions.

By our sequential composition theorem (Theorem 3.2), it is sufficient to construct degree-3 MPRE for  $\text{NC}^1$  and for  $\text{P/poly}$ . The former is constructed in [ABT18]. The later, a  $(n - 1)$ -private degree-3 MPRE for  $\text{P/poly}$  that makes black-box use of PRG, has been implicitly constructed in [DI05]. We will formally analyze their adaptive security in the rest of the section.

### 5.1 MPRE for $\text{NC}^1$

**Lemma 5.1.** *For any arithmetic  $\text{NC}^1$  functionality over field  $\mathbb{F}$ , it has a MPRE such that*

- *the MPRE is perfectly secure against adaptive corruptions;*
- *the MPRE has effective degree 3 over  $\mathbb{F}$ .*

We know that any arithmetic  $\text{NC}^1$  functionality has a degree-3 randomized encoding [IK00, IK02]. The randomized encoding of Ishai and Kushilevitz has a nice property that given the input and the encoding, one can easily compute the randomness. Such property can be formalized as the adaptive security of randomized encodings.

Any adaptive randomized encoding implies a fully-adaptive MPRE, the proof is similar to Proposition 3.2 in [ABT18]. Thus Lemma 5.1 follows as a corollary.

**Definition 8** (adaptive randomized encoding). Let  $\hat{f} : \mathcal{X} \times \mathcal{R} \rightarrow \hat{\mathcal{Y}}$  be a randomized encoding for function  $f : \mathcal{X} \rightarrow \mathcal{Y}$ . Randomized encoding  $\hat{f}$  is *adaptive* if there exist simulation algorithms  $\text{SimI}, \text{SimO}$  such that the following two distribution are identical for any  $x \in \mathcal{X}$

- the distribution of  $(r, \hat{y})$ : first let  $\hat{y} \leftarrow \text{SimO}(f(x))$ , then let  $r \leftarrow \text{SimI}(x, \hat{y})$ ;

- the distribution of  $(r, \hat{y})$ : first sample  $r \leftarrow \mathcal{R}$ , then set  $\hat{y} = \hat{f}(x; r)$ .

**Lemma 5.2.** *If  $\hat{f}^{\text{RE}}$  is an randomized encoding for functionality  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_n \rightarrow \mathcal{Y}$ , then the scheme in Figure 9 is a MPRE for  $f$  such that*

- the MPRE is secure against  $n - 1$  adaptive corruptions;
- if  $\hat{f}^{\text{RE}}$  is an adaptive randomized encoding, the MPRE is secure against any number of adaptive corruptions;
- the effective degree of the MPRE is the same as the degree of  $\hat{f}^{\text{RE}}$ .

*Proof.* The correctness follows directly from the correctness of  $\hat{f}^{\text{RE}}$ .

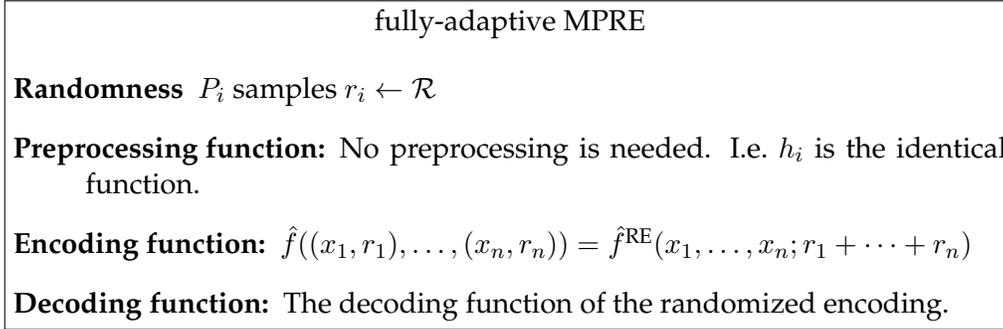


Figure 9: MPRE induced by a randomized encoding

Let  $(\text{SimI}^{\text{RE}}, \text{SimO}^{\text{RE}})$  be the adaptive simulator of  $\hat{f}^{\text{RE}}$ . If there is no adaptive simulator, let  $\text{SimO}^{\text{RE}}$  be the simulator of  $\hat{f}^{\text{RE}}$ . The adaptive simulator of the MPRE works as the follows

**Upon CorruptI( $i$ ), SimI( $i, x_i$ ):**

- If there exists at least one other party that is not corrupted, sample  $r_i \leftarrow \mathcal{R}$ , output  $r_i$ .
- Otherwise, the simulator has learned  $x_1, \dots, x_n$ . W.l.o.g. we can assume subroutine  $\text{SimO}(y)$  has been invoked, thus  $\hat{y}$  has been set.  
Let  $r \leftarrow \text{SimI}^{\text{RE}}(x_1, \dots, x_n, \hat{y})$ . Let  $r_i = r - \sum_{i' \neq i} r_{i'}$ . Output  $r_i$ .

**Upon CorruptO, SimO( $y$ ):** Let  $\hat{y} \leftarrow \text{SimO}^{\text{RE}}(y)$ . Output  $\hat{y}$ .

Due to the security of the randomized encoding, the ideal world is indistinguishable from the real world. □

## 5.2 Computational MPRE for P/poly based on Black-box PRG

**Lemma 5.3.** *The scheme in Figure 10 is a MPRE for P/poly such that*

- the MPRE uses PRG as a black-box;
- the MPRE is computationally secure against  $n - 1$  adaptive corruptions;
- the MPRE has effective degree 3 over boolean field.

**Proof Overview** The construction is similar to Yao’s garbled circuits. Yao’s garbled circuits can be viewed as a degree-3 computational randomized encoding for **P/poly**.

Recall that in Yao’s garbled circuits, the construction involves many pairs of the form

$$(s_j, \hat{s}_j),$$

so that they need to satisfy the following properties

- $s_j$  is uniformly random;
- $\hat{s}_j$  is longer than  $s_j$  and can be deterministically computed from  $s_j$ ;
- if  $s_j$  is hidden,  $\hat{s}_j$  is computationally indistinguishable from uniform distribution.

PRG exactly fits the requirements. In Yao’s garbled circuits,  $s_j$  is sampled at random, and  $\hat{s}_j := G(s_j)$ , where  $G$  is a PRG.

To convert Yao’s garbled circuit into a computational MPRE, the label  $s_j$  should be jointly sampled by all parties. For the MPRE to be secure,  $\hat{s}_j$  should be indistinguishable from uniform randomness as long as at least one party’s local randomness is hidden. Moreover, for the MPRE to have low effective degree, PRG should be only be used in the preprocessing phase.

A natural construction that satisfies all the requirements is

- $s_j := s_j^{(1)} \| \dots \| s_j^{(n)}$ , where  $s_j^{(i)}$  is locally sampled by the  $i$ -th party;
- $\hat{s}_j := G(s_j^{(1)}) \oplus \dots \oplus G(s_j^{(n)})$ .

Denote the mapping from  $s_j$  to  $\hat{s}_j$  by  $G^{\text{MP}}$ , i.e.

$$G^{\text{MP}}(z^{(1)} \| \dots \| z^{(n)}) := G(z^{(1)}) \oplus \dots \oplus G(z^{(n)}).$$

Under the new notation,  $\hat{s}_j = G^{\text{MP}}(s_j)$ .

**Circuit definition** To rigorously state our MPRE, we formalize the notations for functionality in **P/poly**. A boolean circuit is specified by a directed acyclic graph. The nodes in the graph are indexed by numbers in  $[m]$ , each represents a wire in the circuit.

- For any  $j \in [m]$ , let  $x_j$  denote the wire value of the  $j$ -th wire.
- Let  $\mathcal{J}_i$  denote the input wires of the  $i$ -th party. For each  $j \in \mathcal{J}_i$ , the  $i$ -th party knows the value of  $x_j$ . Let  $\mathcal{J}_{\text{in}} := \bigcup_i \mathcal{J}_i$  denote all the input wires.
- Any wire other than the input wires is the output of a gate. Let  $j_1, j_2 < j$  denotes the input wires of the gate ( $j_1, j_2$  are implicit functions of  $j$ ), let  $g_j : \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$  be the corresponding gate function. Thus  $x_j = g_j(x_{j_1}, x_{j_2})$ .
- For each wire  $j$ , let  $d(j)$  denote the fan-out of the wire.
- Let  $\mathcal{J}_{\text{out}}$  denote all the output wires. Thus the circuit output consists of  $x_j$  for all  $j \in \mathcal{J}_{\text{out}}$ .

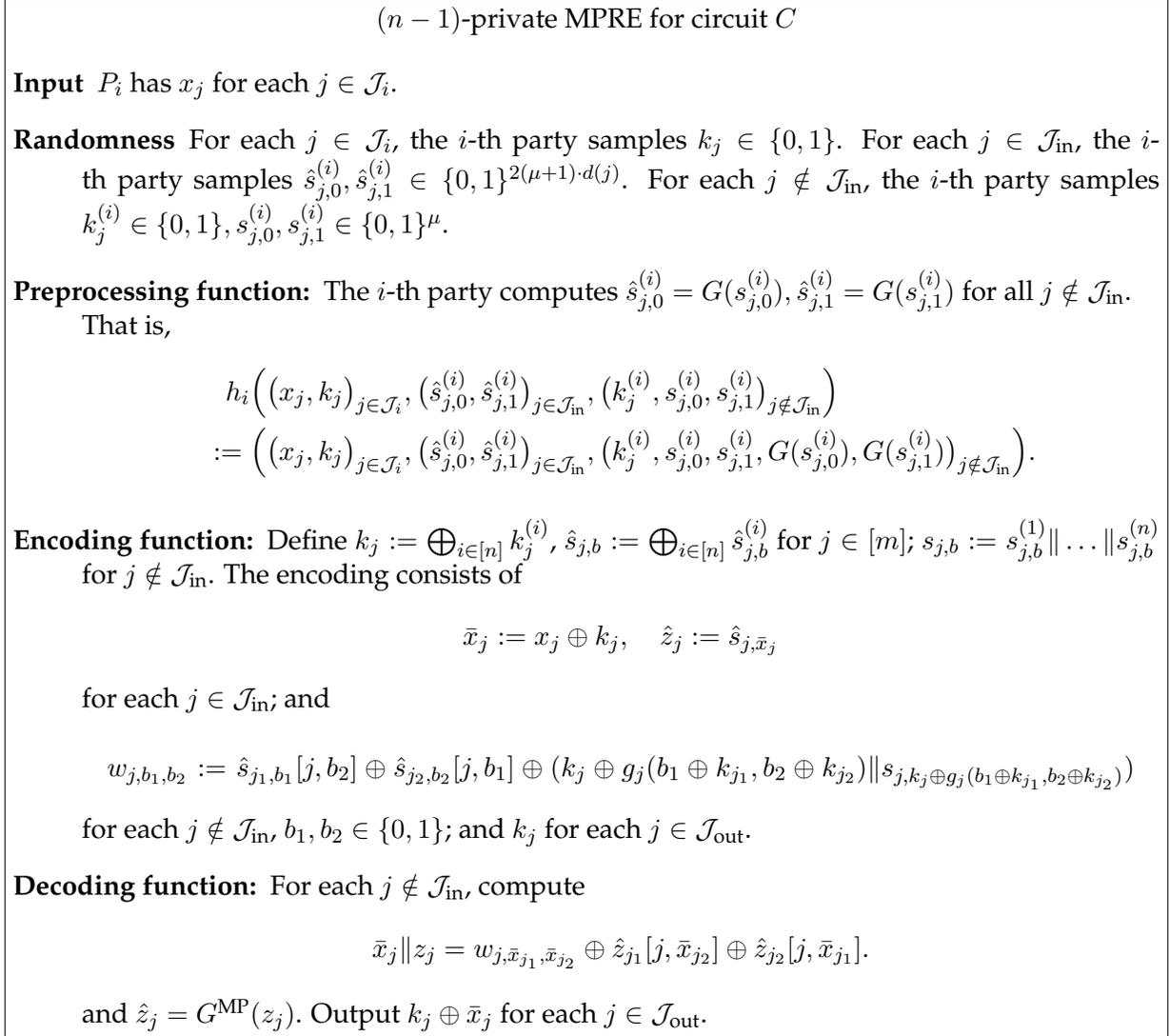


Figure 10: Computational MPRE for **P/poly** using Black-box PRG

*Proof of Lemma 5.3.* Scheme is essentially Yao's garbled circuit which uses  $G^{\text{MP}}$  as PRG.  $k_j$  is the permutation bit of the  $j$ -th wire.  $s_{j,0}, s_{j,1}$  are the wire keys of the  $j$ -th wire.  $(w_{j,0,0}, w_{j,0,1}, w_{j,1,0}, w_{j,1,1})$  is the table associated with the  $j$ -th gate. Thus both the correctness and privacy can be proved in a similar fashion as garbled circuit.

The correctness is implied from the statement that

$$\bar{x}_j := x_j \oplus k_j, \quad \hat{z}_j := \hat{s}_{j, \bar{x}_j} \tag{6}$$

for all  $j \in [m]$ . The statement can be proved by induction. For any  $j \in \mathcal{J}_{\text{in}}$ , (6) is directly guaranteed by the encoding function. For any  $j \notin \mathcal{J}_{\text{in}}$ , assume the statement holds for  $j_1, j_2$  – the two

input wire of the  $j$ -th gate, then

$$\begin{aligned}
\bar{x}_j \| z_j &= w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}} \oplus \hat{s}_{j_1, \bar{x}_{j_1}} [j, \bar{x}_{j_2}] \oplus \hat{s}_{j_2, \bar{x}_{j_2}} [j, \bar{x}_{j_1}] \\
&= (k_j \oplus g_j(\bar{x}_{j_1} \oplus k_{j_1}, \bar{x}_{j_2} \oplus k_{j_2})) \| s_{j, k_j \oplus g_j(\bar{x}_{j_1} \oplus k_{j_1}, \bar{x}_{j_2} \oplus k_{j_2})} \\
&= k_j \oplus g_j(x_{j_1}, x_{j_2}) \| s_{j, k_j \oplus g_j(x_{j_1}, x_{j_2})} \\
&= k_j \oplus x_j \| s_{j, k_j \oplus x_j},
\end{aligned}$$

thus  $\bar{x}_j = k_j \oplus x_j$ ,  $z_j = s_{j, k_j \oplus x_j} = s_{j, \bar{x}_j}$  and  $\hat{z}_j = G^{\text{MP}}(z_j) = G^{\text{MP}}(s_{j, \bar{x}_j}) = \hat{s}_{j, \bar{x}_j}$ . As the consequence, for each  $j \in \mathcal{J}_{\text{out}}$ , the decoding function will output  $\bar{x}_j \oplus k_j$ , which equals the right output  $x_j$ .

For adaptive privacy, the simulator in *the ideal world* works as the follows

**At the outset:** Sample  $\bar{x}_j \leftarrow \{0, 1\}$  for all  $j \in [m]$ , sample random  $\hat{z}_j$  for all  $j \in \mathcal{J}_{\text{in}}$ , sample random  $z_j$  and sets  $\hat{z}_j = G^{\text{MP}}(z_j)$  all  $j \notin \mathcal{J}_{\text{in}}$ .

**Upon CorruptI( $i$ ), SimI( $i, (x_j)_{j \in \mathcal{J}_i}$ ):** Sets  $k_j = x_j \oplus \bar{x}_j$  for all  $j \in \mathcal{J}_i$ .

Sample random  $\hat{s}_{j,0}^{(i)}, \hat{s}_{j,1}^{(i)}$  for all  $j \in \mathcal{J}_{\text{in}}$ , sample random  $k_j^{(i)}$  for all  $j \notin \mathcal{J}_{\text{in}}$ .

Set  $s_{j, \bar{x}_i}^{(i)}$  as the  $i$ -th part of  $z_j$  and sample random  $s_{j, \bar{x}_i \oplus 1}^{(i)}$  for all  $j \notin \mathcal{J}_{\text{in}}$ .

Output  $(x_j, k_j)_{j \in \mathcal{J}_i}, (\hat{s}_{j,0}^{(i)}, \hat{s}_{j,1}^{(i)})_{j \in \mathcal{J}_{\text{in}}}, (k_j^{(i)}, s_{j,0}^{(i)}, s_{j,1}^{(i)})_{j \notin \mathcal{J}_{\text{in}}}$ .

**Upon CorruptO, SimO( $(x_j)_{j \in \mathcal{J}_{\text{out}}}$ ):** Sets  $k_j = x_j \oplus \bar{x}_j$  for all  $j \in \mathcal{J}_{\text{out}}$ .

For each  $j \notin \mathcal{J}_{\text{in}}$ , the simulator sets

$$w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}} = \hat{z}_{j_1} [j, \bar{x}_{j_2}] \oplus \hat{z}_{j_2} [j, \bar{x}_{j_1}] \oplus (\bar{x}_j \| z_j),$$

and samples random  $w_{j, b_1, b_2}$  for  $(b_1, b_2) \neq (\bar{x}_{j_1}, \bar{x}_{j_2})$ .

Output  $(\bar{x}_j, \hat{z}_j)_{j \in \mathcal{J}_{\text{in}}}, (w_{j, b_1, b_2})_{j \notin \mathcal{J}_{\text{in}}, b_1, b_2 \in \{0, 1\}}, (k_j)_{j \in \mathcal{J}_{\text{out}}}$  to the adversary.

In order to show the real world and the ideal world are computationally indistinguishable from the adversary's view, we define a sequence of  $2m + 1$  hybrid worlds. In the  $t$ -th hybrid world ( $t \in \{0, \frac{1}{2}, 1, \frac{3}{2}, \dots, m\}$ ):

**At the outset:** The adversary decides input  $(x_j)_{j \in \mathcal{J}_{\text{in}}}$ .

Sample  $\bar{x}_j \in \{0, 1\}$  for all  $j \in [m]$ , sample random  $\hat{z}_j$  for all  $j \in \mathcal{J}_{\text{in}}$ , sample random  $z_j$  and sets  $\hat{z}_j = G^{\text{MP}}(z_j)$  all  $j \notin \mathcal{J}_{\text{in}}$ .

For all  $j \in [m]$ , set  $k_j = x_j \oplus \bar{x}_j$ . For all  $j \in \mathcal{J}_{\text{in}}$ , set  $\hat{s}_{j, \bar{x}_j} = \hat{z}_j$  and sample random  $\hat{s}_{j, \bar{x}_j \oplus 1}$ . For all  $j \notin \mathcal{J}_{\text{in}}$ , set  $s_{j, \bar{x}_j} = z_j$ , set  $\hat{s}_{j, \bar{x}_j} = G^{\text{MP}}(s_{j, \bar{x}_j})$ ,  $\hat{z}_j = G^{\text{MP}}(z_j)$ , thus  $\hat{z}_j = \hat{s}_{j, \bar{x}_j}$ .

set  $s_{j, \bar{x}_j} = z_j$ , sample random  $s_{j, \bar{x}_j \oplus 1}$ .

For each  $j \notin \mathcal{J}_{\text{in}}$  that  $j \leq t$ , sample random  $s_{j, \bar{x}_j \oplus 1}, \hat{s}_{j, \bar{x}_j \oplus 1}$ .

For each  $j \notin \mathcal{J}_{\text{in}}$  that  $j > t$ , sample random  $s_{j, \bar{x}_j \oplus 1}$ , set  $\hat{s}_{j, \bar{x}_j \oplus 1} = G^{\text{MP}}(s_{j, \bar{x}_j \oplus 1})$ .

**Upon CorruptI( $i$ ):** Sample random  $\hat{s}_{j,0}^{(i)}, \hat{s}_{j,1}^{(i)}$  for all  $j \in \mathcal{J}_{\text{in}}$ . Sample random  $k_j^{(i)}$  for all  $j \notin \mathcal{J}_{\text{in}}$ . Set  $s_{j,b}^{(i)}$  as the  $i$ -th part of  $s_{j,b}$  for all  $j \notin \mathcal{J}_{\text{in}}$ .

Send  $(x_j, k_j)_{j \in \mathcal{J}_i}, (\hat{s}_{j,0}^{(i)}, \hat{s}_{j,1}^{(i)})_{j \in \mathcal{J}_{\text{in}}}, (k_j^{(i)}, s_{j,0}^{(i)}, s_{j,1}^{(i)})_{j \notin \mathcal{J}_{\text{in}}}$  to the adversary.

**Upon CorruptO:** For each  $j \notin \mathcal{J}_{\text{in}}$  that  $j \leq t + \frac{1}{2}$ , set

$$w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}} = \hat{z}_{j_1}[j, \bar{x}_{j_2}] \oplus \hat{z}_{j_2}[j, \bar{x}_{j_1}] \oplus (\bar{x}_j \| z_j),$$

and sample random  $w_{j, b_1, b_2}$  for  $(b_1, b_2) \neq (\bar{x}_{j_1}, \bar{x}_{j_2})$ .

For each  $j \notin \mathcal{J}_{\text{in}}$  that  $j > t + \frac{1}{2}$ , set

$$w_{j, b_1, b_2} = \hat{s}_{j_1, b_1}[j, b_2] \oplus \hat{s}_{j_2, b_2}[j, b_1] \oplus (k_j \oplus g_j(b_1 \oplus k_{j_1}, b_2 \oplus k_{j_2}) \| s_{j, k_j \oplus g_j(b_1 \oplus k_{j_1}, b_2 \oplus k_{j_2})})$$

for  $b_1, b_2 \in \{0, 1\}$ .

The simulator sends  $(\bar{x}_j, \hat{z}_j)_{j \in \mathcal{J}_{\text{in}}}, (w_{j, b_1, b_2})_{j \notin \mathcal{J}_{\text{in}}, b_1, b_2 \in \{0, 1\}}, (k_j)_{j \in \mathcal{J}_{\text{out}}}$  to the adversary.

The ideal world is computationally indistinguishable from the real world, because 1) the real world is indistinguishable from the 0-th hybrid world; 2) the ideal world is indistinguishable from the  $m$ -th hybrid world; 3) the  $j$ -th hybrid world is computationally indistinguishable from the  $(j - 1)$ -th hybrid world.

*The real world is indistinguishable from the 0-th hybrid world* as they are essentially the same. E.g. in the real world,  $k_j^{(1)}, \dots, k_j^{(n)}$  are i.i.d. random boolean, and  $k_j := k_j^{(1)} \oplus \dots \oplus k_j^{(n)}$ ,  $\bar{x}_j := k_j \oplus x_j$ ; while in the 0-th hybrid world,  $\bar{x}_j$  and  $k_j^{(i)}$  for all corrupted party  $i$  are randomly sampled, and  $k_j := \bar{x}_j \oplus x_j$ . These two methods of sampling yield the same distribution.

*The ideal world is indistinguishable from the  $m$ -th hybrid world.* Compared with the  $m$ -th hybrid world, the only difference of the ideal world is that some computation is deferred. E.g. in the  $m$ -th hybrid world, it sets  $k_j := \bar{x}_j \oplus x_j$  at the beginning; while in the ideal world, the simulator can only set  $k_j$  after  $x_j$  is given.

*The  $(j - 1)$ -th hybrid world. is indistinguishable from the  $(j - \frac{1}{2})$ -th hybrid world.* The only difference between them is how  $w_{j, 0, 0}, w_{j, 0, 1}, w_{j, 1, 0}, w_{j, 1, 1}$  are generated.

As for  $w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}}$ , we have

$$\begin{aligned} & w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}} \quad (\text{in the } (j - \frac{1}{2})\text{-th hybrid world}) \\ &= \hat{z}_{j_1}[j, \bar{x}_{j_2}] \oplus \hat{z}_{j_2}[j, \bar{x}_{j_1}] \oplus (\bar{x}_j \| z_j) \\ &= \hat{s}_{j_1, \bar{x}_{j_1}}[j, \bar{x}_{j_2}] \oplus \hat{s}_{j_2, \bar{x}_{j_2}}[j, \bar{x}_{j_1}] \oplus (\bar{x}_j \| s_{j, \bar{x}_j}) \\ &= \hat{s}_{j_1, \bar{x}_{j_1}}[j, \bar{x}_{j_2}] \oplus \hat{s}_{j_2, \bar{x}_{j_2}}[j, \bar{x}_{j_1}] \oplus (k_j \oplus g_j(\bar{x}_{j_1} \oplus k_{j_1}, \bar{x}_{j_2} \oplus k_{j_2}) \| s_{j, k_j \oplus g_j(\bar{x}_{j_1} \oplus k_{j_1}, \bar{x}_{j_2} \oplus k_{j_2})}) \\ &= w_{j, \bar{x}_{j_1}, \bar{x}_{j_2}} \quad (\text{in the } (j - 1)\text{-th hybrid world}). \end{aligned}$$

For the other three terms,  $w_{j, b_1, b_2}$  for  $(b_1, b_2) \neq (\bar{x}_{j_1}, \bar{x}_{j_2})$ , we have

$$\begin{aligned} & w_{j, b_1, b_2} \quad (\text{in the } (j - 1)\text{-th hybrid world}) \\ &= \hat{s}_{j_1, b_1}[j, b_2] \oplus \hat{s}_{j_2, b_2}[j, b_1] \oplus (k_j \oplus g_j(b_1 \oplus k_{j_1}, b_2 \oplus k_{j_2}) \| s_{j, k_j \oplus g_j(b_1 \oplus k_{j_1}, b_2 \oplus k_{j_2})}). \end{aligned}$$

Notice that in the  $(j - 1)$ -th hybrid world,  $\hat{s}_{j_1, \bar{x}_{j_1} \oplus 1}, \hat{s}_{j_2, \bar{x}_{j_2} \oplus 1}$  are fresh randomness that are only used to generate  $w_{j, \bar{x}_{j_1} \oplus 1, \bar{x}_{j_2} \oplus 1}, w_{j, \bar{x}_{j_1}, \bar{x}_{j_2} \oplus 1}, w_{j, \bar{x}_{j_1} \oplus 1, \bar{x}_{j_2}}$ . Thus it's equivalent to sampling  $w_{j, b_1, b_2}$  for  $(b_1, b_2) \neq (\bar{x}_{j_1}, \bar{x}_{j_2})$  at random as they are already one-time padded by fresh randomness, which is exactly how they are generated in the  $(j - \frac{1}{2})$ -th hybrid world.

*The last piece is the computational indistinguishability between the  $j$ -th hybrid world and the  $(j - \frac{1}{2})$ -th hybrid world.* The only difference between them is how  $\hat{s}_{j, \bar{x}_j \oplus 1}$  is generated.

In the  $(j - \frac{1}{2})$ -th hybrid world,  $s_{j, \bar{x}_{j \oplus 1}} = s_{j, \bar{x}_{j \oplus 1}}^{(1)} \parallel \dots \parallel s_{j, \bar{x}_{j \oplus 1}}^{(n)}$  are randomly sampled and  $\hat{s}_{j, \bar{x}_{j \oplus 1}}$  is determined by  $\hat{s}_{j, \bar{x}_{j \oplus 1}} = G^{\text{MP}}(s_{j, \bar{x}_{j \oplus 1}}) = \bigoplus_i G(s_{j, \bar{x}_{j \oplus 1}}^{(i)})$ . As we are proving  $(n - 1)$ -privacy, the adversary cannot corrupts all parties. Let  $i^*$  denote a party currently not corrupted by the adversary. Notice that  $s_{j, \bar{x}_{j \oplus 1}}^{(i^*)}$  is only used to generate  $\hat{s}_{j, \bar{x}_{j \oplus 1}}$ , thus it is computational indistinguishable if  $G(s_{j, \bar{x}_{j \oplus 1}}^{(i^*)})$  is replaced by uniform randomness. Replacing  $G(s_{j, \bar{x}_{j \oplus 1}}^{(i^*)})$  by uniform randomness is equivalent to sampling  $\hat{s}_{j, \bar{x}_{j \oplus 1}}$  at random, which is how  $\hat{s}_{j, \bar{x}_{j \oplus 1}}$  is generated in the  $j$ -th hybrid world.  $\square$

## 6 Two-Round MPC

As what we are going to show in Lemma 6.1, an effective-degree-2 adaptive MPRE for functionality  $f$  and an adaptive 2-round MPC for any degree-2 functions will imply an adaptive 2-round MPC for the functionality  $f$ . In previous sections, we construct effective degree-2 MPRE for  $\text{NC}^1$  and  $\mathbf{P/poly}$  under different settings. The last step is to construct adaptive 2-round MPC protocols for degree-2 functionalities in these settings, which are Section 6.1 and 6.2.

**Lemma 6.1.** *Let  $(\hat{f}, h_1, \dots, h_n)$  be a MPRE for functionality  $f$  that tolerates  $t$  adaptive corruptions. Assume there is a MPC protocol for  $\hat{f}$  that tolerates  $t$  adaptive corruptions. Then there exists a MPC protocol (Figure 11) for  $f$  such that*

- the resulting MPC protocol has the same round and communication complexity as the MPC protocol for  $\hat{f}$ ;
- the resulting MPC protocol tolerates  $t$  adaptive corruptions; the type of the simulation security (perfect, statistical or computational) align with that of the MPRE for  $f$  and MPC for  $\hat{f}$ ;
- if the MPC for  $\hat{f}$  or the MPRE for  $f$  uses correlated randomness, the resulting MPC uses the same correlated randomness.

*Proof.* The correctness is straight-forward. By the correctness of the MPC for  $\hat{f}$ , the receiver will receive  $\hat{y} = \hat{f}(\hat{x}_1, \dots, \hat{x}_n)$ . The receiver outputs  $\text{Dec}(\hat{y})$ , which equals  $f(x_1, \dots, x_n)$  due to the correctness of the MPRE.

Let  $(\text{Siml}^{\text{MPRE}}, \text{SimO}^{\text{MPRE}})$  be the adaptive simulator of the MPRE, and let  $(\text{Siml}^{\text{MPC}}, \text{SimO}^{\text{MPC}})$  be the adaptive simulator of the MPC protocol for  $\hat{f}$ . Then the simulator for the MPC protocol in Figure 11 works as the follows

**Upon CorruptI( $i$ ), Siml( $i, x_i$ ):** Let  $r_i, r'_i \leftarrow \text{Siml}^{\text{MPRE}}(x_i)$ . Set  $\hat{x}_i = h_i(x_i, r_i, r'_i)$ .

Let  $\text{View}_i \leftarrow \text{Siml}^{\text{MPC}}(\hat{x}_i)$ . Output  $\text{View}_i, r_i, r'_i$ .

**Upon CorruptO, SimO( $y$ ):** Let  $\hat{y} \leftarrow \text{SimO}^{\text{MPRE}}(y)$ .

Let  $\text{View}_{\text{receiver}} \leftarrow \text{SimO}^{\text{MPC}}(\hat{y})$ . Output  $\text{View}_{\text{receiver}}$ .

The security can be formally proved by a hybrid argument. Consider a middle world where the MPRE part (i.e.  $r_i, r'_i, \hat{x}_i, \hat{y}$ ) is generated as in the real world, and the rest are generated by the simulator  $(\text{Siml}^{\text{MPC}}, \text{SimO}^{\text{MPC}})$ .

The ideal world is indistinguishable from the middle world due to the privacy of the MPRE and the correctness of the MPC for  $\hat{f}$ . The real world is indistinguishable from the middle world due to the privacy of the MPC for  $\hat{f}$ .  $\square$

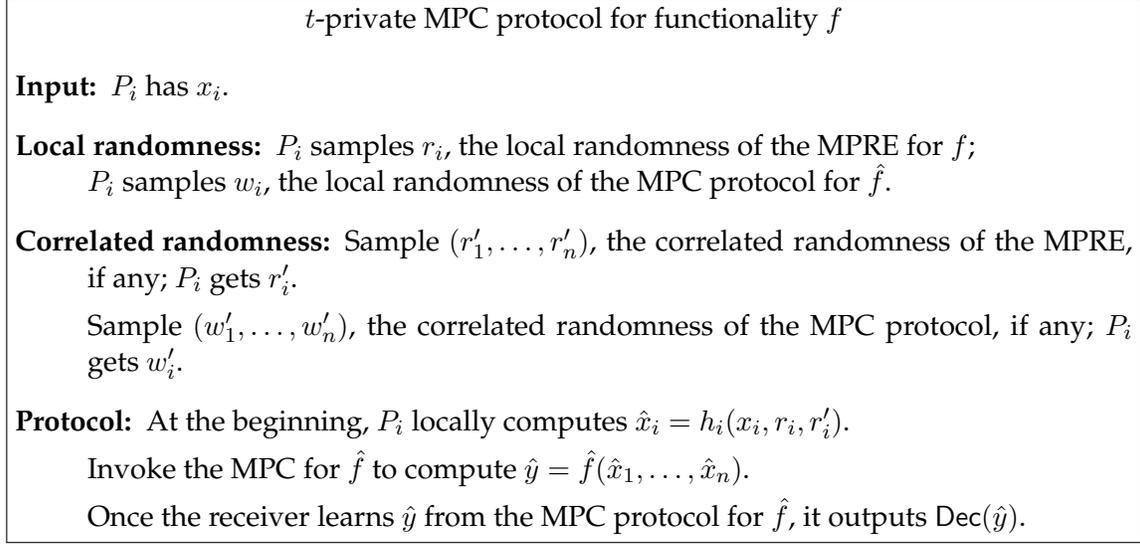


Figure 11: A MPC protocol implied by MPRE

## 6.1 Honest Majority & Plain Model

In the honest majority setting, the BGW [BGW88] protocol when restricted to computing degree-2 polynomials has only two rounds. The adaptive security of BGW is proved in [DN14].

**Lemma 6.2.** *For any degree-2 functionality  $f$ , the BGW protocol computes  $f$  in 2-round and tolerates adaptive minority corruptions.*

## 6.2 Honest Minority & OLE Correlations

We now construct a very simple adaptively secure MPC protocol using OLE-correlation for the following 2MultPlus functionality, which is sufficient for computing any degree-2 polynomials.

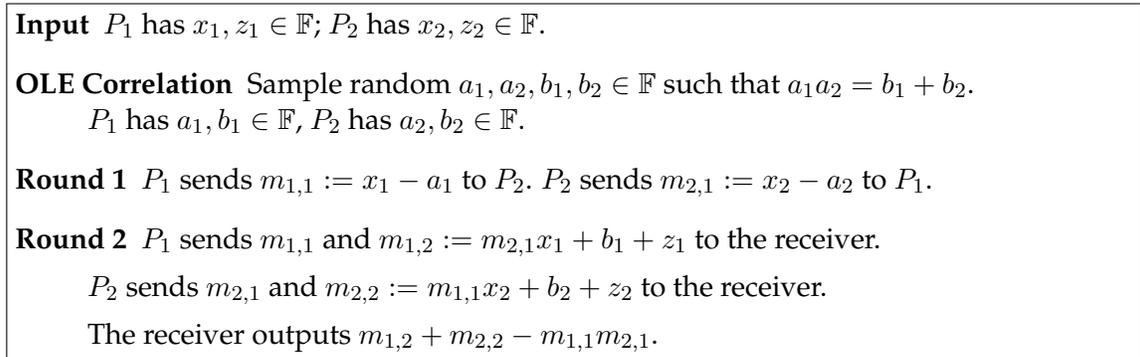


Figure 12: 2-round MPC for 2MULTPlus in OLE correlation model

**Lemma 6.3.** *The 2-round MPC described in Figure 12 is a adaptive secure MPC protocol for the following functionality*

$$2\text{MultPlus} : ((x_1, z_1), (x_2, z_2)) \mapsto x_1 x_2 + z_1 + z_2$$

*and it tolerates an arbitrary number of corruptions.*

**Proof Overview** The scheme can also be explained as a randomized encoding for branching program. As  $(b_1, b_2)$  is the additive secret sharing of  $a_1 a_2$ , the receiver essentially learns  $m_{1,1}, m_{2,1}$  and  $m_{1,2} + m_{2,2}$ .

As

$$\begin{bmatrix} 1 & a_1 \\ & 1 \end{bmatrix} \begin{bmatrix} x_1 & z_1 + z_2 \\ -1 & x_2 \end{bmatrix} \begin{bmatrix} 1 & a_2 \\ & 1 \end{bmatrix} = \begin{bmatrix} m_{1,1} & m_{1,2} + m_{2,2} \\ -1 & m_{2,1} \end{bmatrix},$$

the message received by the receiver is a randomized encoding of  $x_1 x_2 + z_1 + z_2$ , and  $a_1, a_2$  are the randomness of the randomness encoding.

*Proof.* The correctness of the protocol follows directly from

$$\begin{aligned} & m_{1,2} + m_{2,2} - m_{1,1} m_{2,1} \\ &= x_1 x_2 - x_1 a_2 + b_1 + z_1 x_1 x_2 - x_2 a_1 + b_2 + z_2 - (x_1 x_2 - x_1 a_2 - a_1 x_2 + a_1 a_2) \\ &= x_1 x_2 + z_1 + z_2. \end{aligned}$$

The view of the adversary can be (adaptively) simulated as the following simulator

**On the outset:** Sample  $m_{1,1}, m_{2,1} \in \mathbb{F}$  at random.

**Siml** $(i, (x_i, z_i))$ : Set  $a_i = x_i - m_{i,1}$ .

Sample random  $m_{i,2}$  if it is not set by subroutine **SimO** $(y)$ . If the other party has been corrupted, we can assume w.l.o.g. that subroutine **SimO** $(y)$  has set  $m_{i,2}$ .

Solve  $b_i$  from  $m_{i,2} := m_{3-i,1} x_i + b_i + z_i$ .

Output  $a_i, b_i, m_{3-i,1}$ .

**SimO** $(y)$ :

- If both  $P_1, P_2$  are corrupted: w.l.o.g. subroutine **SimO** $(y)$  have already been invoked.
- If  $P_i$  is corrupted,  $P_{3-i}$  is not corrupted: Subroutine **Siml** $(i, (x_i, z_i))$  has set  $m_{i,2}$ . Solve  $m_{3-i,2}$  from  $m_{1,2} + m_{2,2} - m_{1,1} m_{2,1} = y$ .
- If neither  $P_1$  nor  $P_2$  is corrupted: Sample random  $m_{1,2}, m_{2,2}$  such that  $m_{1,2} + m_{2,2} = y + m_{1,1} m_{2,1}$ .

Output  $m_{1,1}, m_{1,2}, m_{2,1}, m_{2,2}$ .

In both the real world and the ideal world,

- $m_{1,1}, m_{2,1}$  are randomly sampled;
- $(m_{1,2}, m_{2,2})$  is additive secret sharing of  $y + m_{1,1} m_{2,1}$ ;
- The rest (i.e.  $a_1, b_1, a_2, b_2$ ) can be determined by  $x_1, z_1, x_2, z_2, m_{1,1}, m_{1,2}, m_{2,1}, m_{2,2}$ . □

### Acknowledgements.

We thank Yuval Ishai for insightful discussions. Part of this work was done while the authors were visiting the Simons Institute for the Theory of Computing.

## 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.
- [ACGJ18] Prabhanjan Ananth, Arka Rai Choudhuri, Aarushi Goel, and Abhishek Jain. Round-optimal secure multiparty computation with honest majority. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 395–424. Springer, Heidelberg, August 2018.
- [AIK04] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in  $NC^0$ . In *45th FOCS*, pages 166–175. IEEE Computer Society Press, October 2004.
- [BCG<sup>+</sup>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, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 387–416. Springer, 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, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 896–912. ACM, 2018.
- [BGI<sup>+</sup>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.
- [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.
- [Can00] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, January 2000.

- [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.
- [CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multiparty computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996.
- [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.
- [DN14] Ivan Damgård and Jesper Buus Nielsen. Adaptive versus static security in the UC model. In Sherman S. M. Chow, Joseph K. Liu, Lucas Chi Kwong Hui, and Siu-Ming Yiu, editors, *Provable Security - 8th International Conference, ProvSec 2014, Hong Kong, China, October 9-10, 2014. Proceedings*, volume 8782 of *Lecture Notes in Computer Science*, pages 10–28. Springer, 2014.
- [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.
- [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.
- [GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.
- [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.
- [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.
- [IMO18] Yuval Ishai, Manika Mittal, and Rafail Ostrovsky. On the message complexity of secure multiparty computation. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part I*, volume 10769 of *LNCS*, pages 698–711. Springer, Heidelberg, March 2018.
- [Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.