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

Huijia Lin[1(✉)], Tianren Liu[1], and Hoeteck Wee[2,3]

[1] University of Washington, Seattle, USA
{rachel,tianrenl}@cs.washington.edu
[2] NTT Research, California, USA
[3] 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) [5,12,18,23] 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 [6,17] may be broadly classified as follows:

H. Lin and T. Liu—Supported by NSF grants CNS-1528178, CNS-1514526, CNS-1652849 (CAREER), CNS-2026774, a Hellman Fellowship, a JP Morgan Research Award, the Defense Advanced Research Projects Agency (DARPA) and Army Research Office (ARO) under Contract No. W911NF-15-C-0236, and a subcontract No. 2017-002 through Galois. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

H. Wee—Research supported in part by ERC Project aSCEND (H2020 639554).

R. Pass and K. Pietrzak (Eds.): TCC 2020, LNCS 12551, pp. 502–531, 2020.
https://doi.org/10.1007/978-3-030-64378-2_18

- protocols for $\mathbf{NC^1}$ achieving information-theoretic security tolerating $t < n/2$ adversarial parties [2];
- protocols for polynomial-size circuits $\mathbf{P/poly}$ achieving computational security tolerating $t < n/2$ adversarial parties, assuming the existence of one-way functions [1,2];
- protocols for polynomial-size circuits $\mathbf{P/poly}$ achieving computational security tolerating $t < n$ adversarial parties, assuming the existence of oblivious transfer [6,17].

All of these constructions follow the same high-level "round collapsing" strategy introduced in [15]. 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 $\mathbf{NC^1}$ 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 [10,11] 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 themselve can be generated efficiently in the pre-processing phase [8,9]. We provide a comparison of our results with the state of the art in Fig. 1 and Fig. 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 staring from [15].

Our Techniques. The crux of our protocols, following [2,19,20], 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.

Reference	Class	IT/Comp	Corruption	Black-box field	Adaptive
[19,20,5]	arith-$\mathbf{NC^1}$	IT	$t < n/3$	yes	yes
[13,4,23]	$\mathbf{P/poly}$	Comp	$t < n/3$	–	yes
[16]	$\mathbf{NC^1}$	IT	$t < n/2$	–	yes*
[2]	$\mathbf{NC^1}$	IT	$t < n/2$	no	no
[2,1,16]	$\mathbf{P/poly}$	Comp	$t < n/2$	–	no
this work	arith-$\mathbf{NC^1}$	IT	$t < n/2$	yes	yes
this work	$\mathbf{P/poly}$	Comp	$t < n/2$	–	yes

Fig. 1. Summary of semi-honest 2-round MPC protocols with a honest majority. All of the constructions for $\mathbf{P/poly}$ (starting with [13]) make black-box use of a PRG. The protocol by [16] handles only a constant number of parties. * They did not fully specify the adaptive simulator.

Reference	Class	Model	Assumptions	Adaptive
[6,17]	$\mathbf{P/poly}$	standard	OT	no
[7]	$\mathbf{P/poly}$	standard	adaptive OT	yes
[16]	$\mathbf{P/poly}$	standard	NIOT	yes*
[21]	$\mathbf{P/poly}$	n-ary correlations	OWF	yes
this work	arith-NC1	2-ary correlated randomness	IT	yes
this work	$\mathbf{P/poly}$	2-ary correlated randomness	OWF	yes

Fig. 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.

We construct MPRE directly without using "inner" multi-round MPC. We observe that the [20] 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 [5] 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 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 [19,20], 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 [2], 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 [2] for a n-party functionality $f(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is a randomized function $\hat{f}(\mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{r}_1, \ldots, \mathbf{r}_n)$ with the following properties:

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

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

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

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

  by distribution, where $\mathbf{r}_1, \ldots, \mathbf{r}_n$ 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 $h_1, \ldots, h_n$ such that $\hat{f}$ can be expressed as a degree d function of $h_1(\mathbf{x}_1, \mathbf{r}_1), \ldots, h_n(\mathbf{x}_n, \mathbf{r}_n)$. 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 $\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{r}_1, \ldots, \mathbf{r}_n$ 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 $L_i(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ 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

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

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, μ);
– P_4 samples w_1, w_5 and pre-computes w_1w_5 so that the encoding has effective degree two.

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

$$\begin{aligned}&\hat{f}((x, \mu), a, b, \perp\ ;\ \mathbf{w}) \\ &\qquad = g\Big(\big(x, \mu, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}\big),\ a,\ b,\ \big(w_1w_5, w_2^{(4)}, w_3^{(4)}, w_4^{(4)}\big)\Big) = (1)\end{aligned}$$

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 Π and a, b, one can compute matching $w_1 = \Pi[1,1] + a$ and $w_5 = \Pi[3,3] + b$.

MPRE for $x_1x_2x_3$ 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 Eq. (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

$$x_1, x_2, x_3 \;\mapsto x_1x_2x_3$$

- P_2 and P_3 hold $(w_1, b^{(1)}), (w_5, b^{(5)})$ such that $b^{(1)} + b^{(5)} = w_1w_5$;
- 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:

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

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 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 $t \le n$.*

MPRE for $x_1x_2x_3$ for honest majority. Next, we build a n-party MPRE with effective degree 2 for

$$x_1,\ x_2,\ x_3,\ \underbrace{\perp, \ldots, \perp}_{n-3} \;\mapsto\; x_1x_2x_3$$

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 $i = 1, 2, \ldots, n$, 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 $Y(1), \ldots, Y(n)$ via polynomial interpolation. In summary, the MPRE is the parallel composition of n gadget MPRE and hence have effective degree 2.

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

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 sketc.h the security proof for the setting where the last t parties $P_{n-t+1}, \ldots, P_n$ 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 $Q_2(i), Q_3(i), i = n-t+1, \ldots, n$. 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 (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 $(\mathsf{SimO}, \mathsf{SimI})$ in which $\mathsf{SimO}(f(\mathbf{x}_1, \ldots, \mathbf{x}_n))$ simulates the encoding $\hat{f}(\mathbf{x}_1, \ldots, \mathbf{x}_n; \mathbf{r}_1, \ldots, \mathbf{r}_n)$, and $\mathsf{SimI}(i, \mathbf{x}_i)$ simulates $\mathbf{r}_i$, in the order that the adaptive adversary corrupts parties.

Putting Pieces Together for $\mathbf{NC^1}$. 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 $\mathbf{NC^1}$ computation in the same model, while preserving the effective degree. The IK02 randomized encoding [20] for arith-NC1 allows for transforming a function g in $\mathbf{NC^1}$ by a degree-3 polynomial $\hat{g}$, such that, $\hat{g}(\mathbf{x}_1, \cdots \mathbf{x}_n \ ; \ \mathbf{r})$ reveals only $g(\mathbf{x}_1, \cdots, \mathbf{x}_n)$ 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.

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

The above is an effective-degree-3 MPRE for arithmetic $\mathbf{NC^1}$. 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 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 $\mathbf{P/poly}$ with effective degree 3 that makes black-box use of a PRG G, using the techniques introduced in [4,13]. 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 $s^{(i)}_{j,0}, s^{(i)}_{j,1}$ corresponding the wire having value 0 or 1; the two labels for wire j is then set to $\ell_{j,b} = s^{(1)}_{j,b} \| \ldots \| s^{(n)}_{j,b}$. 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,

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

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:

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

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 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 $[n] := \{1, 2, \ldots, n\}$. For any set $S \subseteq [n]$ and vector $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, where $\mathbf{x}_i$ itself can be a vector, let $\mathbf{x}[S]$ denote the indexed set $(\mathbf{x}_i)_{i \in S}$. 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* $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}$*, 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* Π *for a* n*-party functionality* f *consists of* n *algorithms* $(C_i)_{i \in [n]}$*. An execution of* Π *with inputs* $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n) \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ *and security parameter* 1^λ *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* $\mathsf{st}_i^{(0)} = (\mathbf{x}_i, \mathbf{r}_i)$*.*

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

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

Define the view of party P_i *in the above execution to be* $\textsc{View}_\Pi(1^\lambda, \mathbf{x})[i] = \mathsf{st}_i^{(r)} = (\mathbf{x}_i, \mathbf{r}_i, (m_{i' \to i}^{(j)})_{i' \in [n] \setminus \{i\}, j \in [r]})$*. Let* $\textsc{View}_\Pi(1^\lambda, \mathbf{x})$ *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* $\mathcal{R}'_1 \times \cdots \times \mathcal{R}'_n$*, then in each execution of the protocol,* $(\mathbf{r}'_1, \ldots, \mathbf{r}'_n) \leftarrow \mathcal{D}$ *is sampled by the beginning of the protocol, and each party* P_i *initialize its state as* $\mathsf{st}_i^{(0)} = (\mathbf{x}_i, \mathbf{r}_i, \mathbf{r}'_i)$*.*

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

Remark 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 $\mathbf{x}_1, \ldots, \mathbf{x}_n$ to different outputs $\mathbf{y}_1, \ldots, \mathbf{y}_n$, every party P_i can sample a one-time pad $\mathbf{k}_i$ of appropriate length and jointly compute the augmented functionality mapping $(\mathbf{x}_1, \mathbf{k}_1), \ldots, (\mathbf{x}_n, \mathbf{k}_n)$ to $(\mathbf{y}_1 + \mathbf{k}_1), \ldots, (\mathbf{y}_n + \mathbf{k}_n)$, 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 Π for a functionality $f : \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}$ is* perfectly *or* statistically *correct, if for every input tuple $\mathbf{x} \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ and every security parameter $\lambda \in \mathbb{N}$, the output of every party P_i equals $f(x_1, \ldots, x_n)$, with* probability 1 *or with* overwhelming probability *respectively.*

Definition 4 (Semi-honest Security Against Static Corruption). *A protocol Π for a n-party functionality f is* perfectly*, or* statistically*, or* computationally *semi-honest secure against t-corruption, if there is a PPT simulator* Sim*, such that for every subset $T \subseteq [n]$ of at most t parties, input tuple $\mathbf{x}$, it holds that the real views $\textsc{View}_{\Pi}(\mathbf{x})[T]$ of parties in T and the output of the simulator $\mathsf{Sim}(T, \mathbf{x}[T], f(\mathbf{x}))$ 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 Π 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* 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 Π on input $\mathbf{x}$ using fresh randomness, obtaining parties' views $\textsc{View}_{\Pi}(\mathbf{x})$.*
  The adversary $\mathcal{A}$ adaptively and iteratively queries $\mathsf{Corrupt}(i)$, and receives P_i's view $\textsc{View}_{\Pi}(\mathbf{x})[i]$, 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 $\mathsf{Corrupt}(i)$, invoke the simulator $(\widetilde{\textsc{View}}[i], \mathsf{st}) \leftarrow \mathsf{Sim}(i, \mathbf{x}_i, y, \mathsf{st})$ and sends $\widetilde{\textsc{View}}[i]$ to $\mathcal{A}$, where st is initialized to be empty.*

3.2 (Multi-party) Randomized Encoding

Definition 6 (Randomized Encoding [3,20]). *Let $f : \mathcal{X} \to \mathcal{Y}$ be some function. The* randomized encoding *of f is a function $\hat{f} : \mathcal{X} \times \mathcal{R} \to z\mathcal{Y}$, where $\mathcal{R}$ is the randomness space. A randomized encoding should be both correct and private.*

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

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

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

Definition 7 (Multi-party Randomized Encoding [2]). *Let $f : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}$ 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 $h_i : \mathcal{X}_i \times \mathcal{R}_i \times \mathcal{R}'_i \to \hat{\mathcal{X}}_i$;*
- *Encoding function $\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \to \hat{\mathcal{Y}}$, the degree of $\hat{f}$ is called the* effective degree *of this MPRE.*

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

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

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 $\mathbf{x} = (x_1, \ldots, x_n)$, 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 $\mathbf{x} = (x_1, \ldots, x_n)$, 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 $(r'_1, \ldots, r'_n) \leftarrow \mathcal{D}$; computes*

$\hat{x}_i = h_i(x_i, r_i, r'_i)$ *for* $i \in [n]$ *and* $\hat{y} = \hat{f}(\hat{x}_1, \ldots, \hat{x}_n)$*. 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 $\mathsf{CorruptO}$**:** *Output* $\hat{y}$
Upon $\mathsf{CorruptI}(i)$**:** *Output* r_i, r'_i.

– In the ideal world: *The distinguisher chooses input* $\mathbf{x} = (x_1, \ldots, x_n)$*, 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* $(\mathsf{SimO}, \mathsf{SimI})$.
The challenger allows the distinguisher to adaptively query the following two oracles. The later one can be queried up to t *times.*
Upon $\mathsf{CorruptO}$**:** *Compute* $y = f(\mathbf{x})$ *and output whatever* $\mathsf{SimO}(y)$ *outputs.*
Upon $\mathsf{CorruptI}(i)$**:** *Output what is output by* $\mathsf{SimI}(i, x_i)$.

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 1 shows that they can be composed as a MPRE for f whose encoding function is $\hat{\hat{f}}$. Theorem 1 is adaptive version of Lemma 3.3 and 3.4 in [2]. 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 2 shows that they can be composed as a MPRE for the functionality

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

while preserving the complexity.

Theorem 1 (Sequential Composition). *Assume there is a perfectly (resp. statistically or computationally) adaptively* t*-private MPRE for functionality* $f : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}$*, whose encoding function is* $\hat{f} : \hat{\mathcal{X}}_1 \times \cdots \times \hat{\mathcal{X}}_n \to \hat{\mathcal{Y}}$*. Assume there is a perfectly (resp. statistically or computationally) adaptively* t*-private MPRE for* $\hat{f}$*, whose encoding function is* $\hat{\hat{f}} : \hat{\hat{\mathcal{X}}}_1 \times \cdots \times \hat{\hat{\mathcal{X}}}_n \to \hat{\hat{\mathcal{Y}}}$*. Then there exists a perfectly (resp. statistically or computationally) adaptively* t*-private MPRE for* f *whose encoding function is* $\hat{\hat{f}}$.

Theorem 2 (Parallel Composition). *For each* $j \in [m]$*, assume there is a perfectly (resp. statistically or computationally) adaptively* t*-private MPRE for functionality* $f^{(j)} : \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \to \mathcal{Y}^{(j)}$*, whose encoding function is* $\hat{f}^{(j)} : \hat{\mathcal{X}}_1^{(j)} \times \cdots \times \hat{\mathcal{X}}_n^{(j)} \to \hat{\mathcal{Y}}^{(j)}$*. 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* $f^{(1)}, \ldots, f^{(m)}$

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

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

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

$$l_i(x_1, \ldots, x_n) = (l_i^{(1)}(x_1, \ldots, x_n), \ldots, l_i^{(m)}(x_1, \ldots, x_n)),$$

to the i-th party.

The proof of composition theorems are defered to the full version.

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 Sect. 4.1, we construct a 4-party gadget MPRE; in Sect. 4.2, we construct an MPRE for the 3-party functionality $\mathsf{3MultPlus}_3$ described below, which computes a degree-3 monomial shifted by some linear terms, in the OLE-correlation model; then in Sect. 4.3, we consider the n-party version of the functionality $\mathsf{3MultPlus}_n$ and construct an MPRE for it in the honest majority setting.

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

Finally, 3MultPlus is complete in the sense that MPRE for the 3MultPlus functionalities implies MPRE for general degree-3 functionalities. The proof can be found in the full version. 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 $w_1, \ldots, w_5$, [19,20] show that $(\phi_1, \ldots, \phi_6)$ is a randomized encoding of $abx + \mu + \nu$, where $\phi_1, \ldots, \phi_6$ are defined as

$$\begin{aligned}
\begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} &:= \begin{bmatrix} 1 & w_1 & w_2 \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} a & & \mu + \nu \\ -1 & x & \\ & -1 & b \end{bmatrix} \begin{bmatrix} 1 & w_3 & w_4 \\ & 1 & w_5 \\ & & 1 \end{bmatrix} \\
&= \begin{bmatrix} a - w_1 & \begin{pmatrix} aw_3 + xw_1 \\ -w_3 w_1 - w_2 \end{pmatrix} & \begin{pmatrix} w_2 b - w_2 w_5 - w_1 w_4 \\ + w_1 w_5 x + w_4 a + \mu + \nu \end{pmatrix} \\ -1 & x - w_3 & -w_4 + w_5 x \\ & -1 & b - w_5 \end{bmatrix}.
\end{aligned} \tag{3}$$

[19,20] guarantee that $\phi_1, \ldots, \phi_5$ are i.i.d. uniform despite the value of (a, b, x, μ, ν). 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 $w_1 w_5$, i.e. let $h_4(w_1, w_5, w_2'', w_4'') = (w_1, w_5, w_1 w_5, w_2'', w_4'')$. 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 $(\phi_1, \ldots, \phi_6)$ as defined in Equation (3), with $w_2 := w_2' + w_2''$, $w_4 := w_4' + w_4''$.
Decoding function: On input $(\phi_1, \ldots, \phi_6)$, output $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix}$.

Fig. 3. Effective degree-2 MPRE for the gadget functionality

Lemma 5. *The scheme defined in Fig. 3 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

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

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

– *In the real world:* For input x, a, b, μ, ν
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 $(\phi_1, \ldots, \phi_6)$ according to Eq. (3).
$\mathsf{CorruptO}$**:** Output $\phi_1, \ldots, \phi_6$.
$\mathsf{CorruptI}(1)$**:** Output w_3, w_2', w_4'.
$\mathsf{CorruptI}(2)$**:** Output $\perp$.
$\mathsf{CorruptI}(3)$**:** Output $\perp$.
$\mathsf{CorruptI}(4)$**:** Output w_1, w_5, w_2'', w_4''.
– *In the ideal world:*
At the Outset: Sample random $\phi_1, \ldots, \phi_5$,
Upon $\mathsf{CorruptO}$, $\mathsf{SimO}(y)$**:** Let ϕ_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 $\phi_1, \ldots, \phi_6$.

Upon $\mathsf{CorruptI}(1)$, $\mathsf{SimI}(1,(x,\mu))$**:** 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 $\mathsf{SimI}(4,\nu,(a,b))$ has learned a and has sampled the values of w_1, w_5. Then, set w_2, w_4 to satisfy $\phi_2 = aw_3 + xw_1 - w_3w_1 - w_2$, $\phi_4 = -w_4 + w_5x$, and set $w_2' = w_2 - w_2''$, $w_4' = w_4 - w_4''$.
Output w_3, w_2', w_4'.

Upon $\mathsf{CorruptI}(2)$, $\mathsf{SimI}(2,a)$**:** Output $\perp$.

Upon $\mathsf{CorruptI}(3)$, $\mathsf{SimI}(3,b)$**:** Output $\perp$.

Upon $\mathsf{CorruptI}(4)$, $\mathsf{SimI}(4,\nu,(a,b))$**:** 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 $\mathsf{SimI}(1,(x,\mu))$ has learned x and has sampled the value of w_3. Then, set w_2, w_4 to satisfy $\phi_2 = aw_3 + xw_1 - w_3w_1 - w_2$ and $\phi_4 = -w_4 + w_5x$, 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 $\phi_1, \ldots, \phi_5$.
Let ϕ_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 $w_1, \ldots, w_5$ from Eq. (3). Sample w_2', w_2'' as additive sharing of w_2, Sample w_4', w_4'' as additive sharing of w_4.
$\mathsf{CorruptO}$**:** Output $\phi_1, \ldots, \phi_6$.
$\mathsf{CorruptI}(1)$**:** Output w_3, w_2', w_4'.
$\mathsf{CorruptI}(2)$**:** Output $\perp$.
$\mathsf{CorruptI}(3)$**:** Output $\perp$.
$\mathsf{CorruptI}(4)$**:** 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 $\phi_1, \ldots, \phi_5$. But the simulator cannot compute $w_1, \ldots, w_5$ at the beginning as it doesn't know a, b, x, μ, ν at that moment. Instead, the simulator compute each of $w_1, \ldots, w_5$ 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.

4.2 MPRE for 3-Party 3MultPlus Using OLE Correlation

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

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

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

For randomly sampled $w_1, \ldots, w_5$, [19,20] show that $(\phi_1, \ldots, \phi_6)$ is a randomized encoding of $x_1x_2x_3 + \alpha + \beta + \gamma$, where $\phi_1, \ldots, \phi_6$ are defined as

$$\begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} := \begin{bmatrix} 1 & w_1 & w_2 \\ & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} x_1 & & \alpha+\beta+\gamma \\ -1 & x_2 & \\ & -1 & x_3 \end{bmatrix} \begin{bmatrix} 1 & w_3 & w_4 \\ & 1 & w_5 \\ & & 1 \end{bmatrix}$$
$$= \begin{bmatrix} x_1 - w_1 & \begin{pmatrix} x_1w_3 + x_2w_1 \\ -w_3w_1 - w_2 \end{pmatrix} & \begin{pmatrix} w_2x_3 - w_2w_5 - w_1w_4 + w_4x_1 \\ +w_1w_5x_2 + \alpha + \beta + \gamma \end{pmatrix} \\ -1 & x_2 - w_3 & -w_4 + w_5x_2 \\ & -1 & x_3 - w_5 \end{bmatrix}. \tag{4}$$

[19,20] guarantee that $\phi_1, \ldots, \phi_5$ 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

$$\begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix} = \begin{bmatrix} x_1 - w_1 & \begin{pmatrix} x_1w_3 + x_2w_1 \\ -w_3w_1 - w_2 \end{pmatrix} & \begin{pmatrix} w_2x_3 - w_2w_5 - w_1w_4 + w_4x_1 \\ +(b^{(1)} + b^{(3)})x_2 + \alpha + \beta + \gamma \end{pmatrix} \\ -1 & x_2 - w_3 & -w_4 + w_5x_2 \\ & -1 & x_3 - w_5 \end{bmatrix}. \tag{5}$$

MPRE for the 3-party functionality $\mathsf{3MultPlus}_3$

Local randomness: P_i samples $w_2^{(i)}, w_3^{(i)}, w_4^{(i)} \leftarrow \mathbb{F}$.
Correlated randomness: P_1 is given $(w_1, b^{(1)}) \in \mathbb{F}^2$, and P_3 is given $(w_5, b^{(3)}) \in \mathbb{F}^2$, for random $(w_1, b^{(1)}, w_5, b^{(3)}) \in \mathbb{F}^4$ satisfying $w_1w_5 = b^{(1)} + b^{(3)}$.
Preprocessing function: Preprocessing is not necessary. I.e., h_i is the identity function for $i \in \{1, 2, 3\}$.
Encoding function: On input the outputs of all h_i, output $(\phi_1, \ldots, \phi_6)$ as defined in Equation (5), with $w_2 := \sum_{i\in[3]} w_2^{(i)}$, $w_3 := \sum_{i\in[3]} w_3^{(i)}$, $w_4 := \sum_{i\in[3]} w_4^{(i)}$.
Decoding function: On input $(\phi_1, \ldots, \phi_6)$, output $\det \begin{bmatrix} \phi_1 & \phi_2 & \phi_6 \\ -1 & \phi_3 & \phi_4 \\ & -1 & \phi_5 \end{bmatrix}$.

Fig. 4. Effective degree-2 MPRE for the $\mathsf{3MultPlus}_3$ functionality

Lemma 6. *The MPRE in Fig. 4 for the 3-party functionality* $\mathsf{3MultPlus}_3$ *has effective degree* 2 *and tolerates any number of corruptions, in the OLE-correlation model.*

Proof. The correctness is straight forward,

$$\mathsf{Dec}(\phi_1,\ldots,\phi_6) = \det\begin{bmatrix}\phi_1 & \phi_2 & \phi_6\\ -1 & \phi_3 & \phi_4\\ & -1 & \phi_5\end{bmatrix} = \det\begin{bmatrix}x_1 & & \alpha+\beta+\gamma\\ -1 & x_2 & \\ & -1 & x_3\end{bmatrix}$$
$$= x_1x_2x_3 + \alpha + \beta + \gamma.$$

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 $b^{(1)} + b^{(3)} = w_1w_5$, 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 $(\phi_1,\ldots,\phi_6)$ according to Eq. (5).

  $\mathsf{CorruptO}$**:** Output $\phi_1,\ldots,\phi_6$.

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

  $\mathsf{CorruptI}(2)$**:** Output $w_2^{(2)}, w_3^{(2)}, w_4^{(2)}$.

  $\mathsf{CorruptI}(3)$**:** Output $w_5, b^{(3)}, w_2^{(3)}, w_3^{(3)}, w_4^{(3)}$.
- *In the ideal world:*

  At the Outset: Sample random $\phi_1,\ldots,\phi_5$,

  Upon $\mathsf{CorruptO}$, $\mathsf{SimO}(y)$**:** Let ϕ_6 be the unique value that $\det\left[\begin{smallmatrix}\phi_1 & \phi_2 & \phi_6\\ -1 & \phi_3 & \phi_4\\ & -1 & \phi_5\end{smallmatrix}\right] = y$. Output $\phi_1,\ldots,\phi_6$.

  Upon $\mathsf{CorruptI}(1)$, $\mathsf{SimI}(1,(x_1,\alpha))$**:** 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 $\mathsf{SimI}(3,(x_3,\gamma))$ has set the values of $w_5, b^{(3)}$. Set $b^{(1)} = w_1w_5 - b^{(3)}$.

  If both P_2 and P_3 are corrupted, subroutines $\mathsf{SimI}(2,(x_2,\beta))$, $\mathsf{SimI}(3,(x_3,\gamma))$ have set $w_j^{(2)}, w_j^{(3)}$ for $j \in \{2,3,4\}$. Then solve w_2, w_3, w_4 from Eq. (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 $\mathsf{CorruptI}(2)$, $\mathsf{SimI}(2,(x_2,\beta))$**:** If both P_1 and P_3 are corrupted, subroutines $\mathsf{SimI}(1,(x_1,\beta))$, $\mathsf{SimI}(3,(x_3,\gamma))$ have set $w_j^{(1)}, w_j^{(3)}$ for $j \in \{2,3,4\}$. Then solve w_2, w_3, w_4 from Eq. (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 $\mathsf{CorruptI}(3)$, $\mathsf{SimI}(3,(x_3,\gamma))$**:** 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 $\mathsf{SimI}(1,(x_1,\alpha))$ 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 $\mathsf{SimI}(1,(x_1,\beta))$, $\mathsf{SimI}(2,(x_2,\gamma))$ have set $w_j^{(1)}, w_j^{(2)}$ for $j \in \{2,3,4\}$. Then solve w_2, w_3, w_4 from Eq. (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 $\phi_1, \ldots, \phi_5$.
Let ϕ_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 $w_1, \ldots, w_5$ from Eq. (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 $w_j^{(1)} + w_j^{(2)} + w_j^{(3)} = w_j$.
$\mathsf{CorruptO}$**:** Output $\phi_1, \ldots, \phi_6$.
$\mathsf{CorruptI}(1)$**:** Output $w_1, b^{(1)}, w_2^{(1)}, w_3^{(1)}, w_4^{(1)}$.
$\mathsf{CorruptI}(2)$**:** Output $w_2^{(2)}, w_3^{(2)}, w_4^{(2)}$.
$\mathsf{CorruptI}(3)$**:** 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 $\phi_1, \ldots, \phi_5$. But the simulator cannot compute $w_1, \ldots, w_5$ by solving (4) at the beginning as it doesn't know x_1, x_2, x_3, α, β, γ 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.

4.3 MPRE for n-Party 3MultPlus with Honest Majority

We construct an MPRE (Fig. 5) for the n-party functionality

$$\mathsf{3MultPlus}_n : ((x_1,\alpha),(x_2,\beta),(x_3,\gamma),\underbrace{\perp,\ldots,\perp}_{n-3}) \mapsto x_1x_2x_3 + \alpha + \beta + \gamma$$

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

Additional Notation. Let $\mathbb{F}$ be a field that $|\mathbb{F}| > n$, let $1, \ldots, n$ 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, $m = \mathrm{rec}(t, (i_1, \sigma_1) \ldots, (i_{t+1}, \sigma_{t+1}))$ denotes the procedure for reconstructing the constant term from $t + 1$ points on the polynomial via interpolation. For convenience, we also denote by $\mathcal{P}(t, m) \mid (i_1, \sigma_1) \ldots, (i_s, \sigma_s)$ the set of polynomials $Q \in \mathcal{P}(t, m)$ such that $Q(i_1) = \sigma_1, \ldots, Q(i_s) = \sigma_s$, for $s \leq t + 1$.

Protocol Overview. We decompose the computation of $x_1x_2x_3 + \alpha + \beta + \gamma$, into two parts $x_1x_2x_3 + z + s$ 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 $s = \mathrm{rec}(n - 1, (1, S(1)), \ldots, (n, S(n))$.

Observe that

$$Y := x_1Q_2Q_3 + Z + S \in \mathcal{P}(n - 1, x_1x_2x_3 + z + s) \ .$$

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

$$((x_1, Z(i)),\ Q_2(i),\ Q_3(i),\ S(i)) \ \mapsto\ x_1Q_2(i)Q_3(i) + Z(i) + S(i) = Y(i) \ ,$$

from which the output party can reconstruct Y and the constant term $x_1x_2x_3 + z + s$.

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 $x_1x_2x_3 + z + s$, which is random thanks to the randomization via Z. Now, suppose the adversary additional 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,

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

together with the linear term $\alpha + \beta + \gamma + z + s$. A formal description is in Fig. 5. It is easy to see that $\hat{F}$ has effective degree 2 since $\hat{f}$ has effective degree ≤ 2.

MPRE $\hat{F}$ for the n-party gadget functionality

This scheme uses the following tools:

- $\hat{f}$ is the effective degree-2 MPRE for the 4-party gadget in Section 4.1.

Local Randomness and Preprocessing: Parties locally sample and do:
- P_1 samples $z \leftarrow \mathbb{F}, Z \leftarrow \mathcal{P}(n-1, z)$.
- P_2 samples $Q_2 \leftarrow \mathcal{P}(t, x_2)$.
- P_3 samples $Q_3 \leftarrow \mathcal{P}(t, x_3)$.
- $\forall i \in [n]$, P_i samples $S(i) \leftarrow \mathbb{F}$.
- $\forall i \in [n]$, P_1, P_2, P_3, P_i sample $\mathbf{r}_1^{(i)}, \mathbf{r}_i^{(i)}$ respectively and performs local preprocessing for the i'th invocation of MPRE $\hat{f}$ below.

Encoding: Compute the encoding function of $\hat{f}$ and output:

$$\forall i\ \hat{\mathbf{y}}_i = \hat{f}((x_1, Z(i)),\ Q_2(i),\ Q_3(i),\ S(i)\ ;\ \mathbf{r}_1^{(i)}, \perp, \perp, \mathbf{r}_i^{(i)})$$
$$\hat{y} = \alpha + \beta + \gamma - s - z\ ,$$

where $s = \mathrm{rec}(n-1, (i, S(i))_i)$.

Decoding: For every i, decode $\hat{\mathbf{y}}_i$ to obtain $Y(i) = x_1Q_2(i)Q_3(i) + Z(i) + S(i)$. Output $\mathrm{rec}(n-1, (i, Y(i))_i) + \hat{y}$.

Fig. 5. Effective degree-2 MPRE $\hat{F}$ for the n-party gadget functionality

Lemma 7. *The MPRE scheme in Fig. 5 for the n-party functionality* $\mathsf{3MultPlus}_n$ *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 ℓ. By the adaptive security of $\hat{f}$ and Theorem 2, we have that the parallel composition of all n invokations of $\hat{f}$ and the linear function ℓ is an MPRE $\hat{G}$ for the following composed functionality G:

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

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 $(\mathsf{SimI}_G, \mathsf{SimO}_G)$ be its simulator. Below, we use this simulator to construct the simulator $(\mathsf{SimI}_F, \mathsf{SimO}_F)$ for $\hat{F}$.

Overview. The encoding of $\hat{F}$ consists of encoding of $\hat{f}$ and the output of ℓ with appropriate input / output. The job of $(\mathsf{SimI}_F, \mathsf{SimO}_F)$ is: *1)* simulate the input / output of calls to $\hat{G}$, i.e., calls to $\hat{f}$ and $\hat{\ell}$, and *2)* invoke $(\mathsf{SimI}_G, \mathsf{SimO}_G)$ to simulate the encoding and local randomness of all calls to $\hat{f}$ and ℓ. 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 (Proof of Lemma 7). We start with the formal description of the simulator.

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

- 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 $O \leftarrow \mathcal{P}(n-1, \tau) \mid (i_1, Y(i_1)), \ldots, (i_s, Y(i_s))$ conditioned on the list of fixed $(i_j, Y(i_j))$'s from previous step. (Note that $s \le t$ points are fixed.)

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

Upon $\mathsf{CorruptI}(1)$, $\mathsf{SimI}(x_1, \alpha)$**:**

- 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 $\mathsf{SimI}_G(x_1, \alpha, (Z(i))_i, S(1))$.

Upon $\mathsf{CorruptI}(2)$, $\mathsf{SimI}(x_2, \beta)$**:**

- $\forall i$, if P_i is already corrupted, $Q_2(i)$ is already fixed.
- Sample $Q_2 \leftarrow \mathcal{P}(t, x_2) \mid (i_1, Q_2(i_1)), \ldots, (i_s, Q_2(i_s))$, conditioned on the list of fixed $(i_j, Q_2(i_j))$. (Note that this can be done as $s \le t$ 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 $\mathsf{SimI}_G(x_2, \beta, (Q_2(i))_i, S(2))$.

Upon $\mathsf{CorruptI}(3)$, $\mathsf{SimI}(x_3, \gamma)$**:** Same as in $\mathsf{SimI}(x_2, \beta)$:

- $\forall i$, if P_i is already corrupted, $Q_3(i)$ is already fixed.
- Sample $Q_3 \leftarrow \mathcal{P}(t, x_3) \mid (i_1, Q_3(i_1)), \ldots, (i_s, Q_3(i_s))$, 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 $\mathsf{Siml}_G(x_3, \gamma, (Q_3(i))_i, S(3))$.

Upon $\mathsf{Corruptl}(i)$, $\mathsf{Siml}(\perp)$) 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 $Q_3 \leftarrow \mathcal{P}(t, x_3) \mid (i_1, Q_3(i_1)), \ldots, (i_s, Q_3(i_s))$, 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 $\mathsf{Siml}_G(S(i), Q_2(i), Q_3(i))$.

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 invokation of $\hat{G}$ are simulated perfectly.

Hybrid. More formally, consider the following hybrid, where input/output of the invokation 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 $x_1, x_3, x_3, \alpha, \beta, \gamma$.

- $\forall i$, 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 $\mathsf{CorruptO}$**:** Send to adversary $\mathsf{SimO}_G(y - \tau, (Y(i))_i)$.
Upon $\mathsf{Corruptl}(1)$**:** Send to adversary $\mathsf{Siml}_G(x_1, \alpha, (Z(i))_i, S(1))$.
Upon $\mathsf{Corruptl}(2)$ Send to adversary $\mathsf{Siml}_G(x_2, \beta, (Q_2(i))_i, S(2))$.
Upon $\mathsf{Corruptl}(3)$**:** Send to adversary $\mathsf{Siml}_G(x_3, \gamma, (Q_3(i))_i, S(3))$.
Upon $\mathsf{Corruptl}(i)$**:** Send to adversary $\mathsf{Siml}_G(S(i), Q_2(i), Q_3(i))$.

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.

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

We lift our effective degree-2 MPRE for degree-3 functionalities constructed in the previous section, to MPRE for $\mathbf{NC^1}$ and $\mathbf{P/poly}$. The transformation uses

the former MPRE to compute degree-3 randomized encodings for $\mathbf{NC^1}$ [20] and for $\mathbf{P/poly}$ [22], and preserves the effective degree. The resulting effective-degree-2 MPRE for $\mathbf{NC^1}$ is information theoretically secure and tolerates any adaptive corruptions, while the resulting MPRE for $\mathbf{P/poly}$ is computationally secure making black box access to a PRG, and tolerates $n-1$ adaptive corruptions.

By our sequential composition theorem (Theorem 1), it is sufficient to construct degree-3 MPRE for $\mathbf{NC^1}$ and for $\mathbf{P/poly}$. The former is constructed in [2]. The later, a $(n-1)$-private degree-3 MPRE for $\mathbf{P/poly}$ that makes black-box use of PRG, has been implicitly constructed in [13]. We will formally analyze the adaptive security of our MPRE for $\mathbf{P/poly}$ in the rest of the section. The adaptive security of the our MPRE for $\mathbf{NC^1}$ is deferred to the full version.

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

Lemma 8. *The scheme in Fig. 6 is a MPRE for* $\mathbf{P/poly}$ *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 $\mathbf{P/poly}$.

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

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

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

$$G^{\mathrm{MP}}(z^{(1)} \| \ldots \| z^{(n)}) := G(z^{(1)}) \oplus \cdots \oplus G(z^{(n)}).$$

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

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 $g_j : \{0,1\} \times \{0,1\} \to \{0,1\}$ 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 $j \in \mathcal{J}_{\text{out}}$.

Proof (Proof of Lemma 8). Scheme is essentially Yao's garbled circuit which uses G^{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 $\mathsf{Corruptl}(i)$, $\mathsf{Siml}(i, (x_j)_{j \in \mathcal{J}_i})$**:** Sets $k_j = x_j \oplus \bar{x}_j$ for all $j \in \mathcal{J}_i$.
Sample random $\hat{s}^{(i)}_{j,0}, \hat{s}^{(i)}_{j,1}$ for all $j \in \mathcal{J}_{\text{in}}$, sample random $k^{(i)}_j$ for all $j \notin \mathcal{J}_{\text{in}}$.
Set $s^{(i)}_{j,\bar{x}_i}$ as the i-th part of z_j and sample random $s^{(i)}_{j,\bar{x}_i \oplus 1}$ for all $j \notin \mathcal{J}_{\text{in}}$.
Output $\left(x_j, k_j\right)_{j \in \mathcal{J}_i}, \left(\hat{s}^{(i)}_{j,0}, \hat{s}^{(i)}_{j,1}\right)_{j \in \mathcal{J}_{\text{in}}}, \left(k^{(i)}_j, s^{(i)}_{j,0}, s^{(i)}_{j,1}\right)_{j \notin \mathcal{J}_{\text{in}}}$.

$(n-1)$-private MPRE for circuit C

Input P_i has x_j for each $j \in \mathcal{J}_i$.

Randomness For each $j \in \mathcal{J}_i$, the i-th party samples $k_j \in \{0,1\}$. For each $j \in \mathcal{J}_{\text{in}}$, the i-th party samples $\hat{s}^{(i)}_{j,0}, \hat{s}^{(i)}_{j,1} \in \{0,1\}^{2(\mu+1)\cdot d(j)}$. For each $j \notin \mathcal{J}_{\text{in}}$, the i-th party samples $k^{(i)}_j \in \{0,1\}, s^{(i)}_{j,0}, s^{(i)}_{j,1} \in \{0,1\}^{\mu}$.

Preprocessing function: The i-th party computes $\hat{s}^{(i)}_{j,0} = G(s^{(i)}_{j,0}), \hat{s}^{(i)}_{j,1} = G(s^{(i)}_{j,1})$ for all $j \notin \mathcal{J}_{\text{in}}$.

That is,

$$h_i\Big(\big(x_j,k_j\big)_{j\in\mathcal{J}_i}, \big(\hat{s}^{(i)}_{j,0},\hat{s}^{(i)}_{j,1}\big)_{j\in\mathcal{J}_{\text{in}}}, \big(k^{(i)}_j, s^{(i)}_{j,0}, s^{(i)}_{j,1}\big)_{j\notin\mathcal{J}_{\text{in}}}\Big)$$
$$:= \Big(\big(x_j,k_j\big)_{j\in\mathcal{J}_i}, \big(\hat{s}^{(i)}_{j,0},\hat{s}^{(i)}_{j,1}\big)_{j\in\mathcal{J}_{\text{in}}}, \big(k^{(i)}_j, s^{(i)}_{j,0}, s^{(i)}_{j,1}, G(s^{(i)}_{j,0}), G(s^{(i)}_{j,1})\big)_{j\notin\mathcal{J}_{\text{in}}}\Big).$$

Encoding function: Define $k_j := \bigoplus_{i\in[n]} k^{(i)}_j$, $\hat{s}_{j,b} := \bigoplus_{i\in[n]} \hat{s}^{(i)}_{j,b}$ for $j \in [m]$; $s_{j,b} := s^{(1)}_{j,b} \| \ldots \| s^{(n)}_{j,b}$ for $j \notin \mathcal{J}_{\text{in}}$. The encoding consists of

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

for each $j \in \mathcal{J}_{\text{in}}$; and

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

for each $j \notin \mathcal{J}_{\text{in}}$, $b_1, b_2 \in \{0,1\}$; and k_j for each $j \in \mathcal{J}_{\text{out}}$.

Decoding function: For each $j \notin \mathcal{J}_{\text{in}}$, compute

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

and $\hat{z}_j = G^{\text{MP}}(z_j)$. Output $k_j \oplus \bar{x}_j$ for each $j \in \mathcal{J}_{\text{out}}$.

Fig. 6. Computational MPRE for **P/poly** using Black-box PRG

Upon $\mathsf{CorruptO}$, $\mathsf{SimO}((x_j)_{j\in\mathcal{J}_{\text{out}}})$**:** 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 ($t \in \{0, \frac{1}{2}, 1, \frac{3}{2}, \ldots, m\}$):*

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 $j \le t$, 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 $\mathsf{CorruptI}(i)$**:** Sample random $\hat{s}^{(i)}_{j,0}, \hat{s}^{(i)}_{j,1}$ for all $j \in \mathcal{J}_{\text{in}}$. Sample random $k^{(i)}_j$ for all $j \notin \mathcal{J}_{\text{in}}$. Set $s^{(i)}_{j,b}$ as the i-th part of $s_{j,b}$ for all $j \notin \mathcal{J}_{\text{in}}$.
Send $\left(x_j, k_j\right)_{j \in \mathcal{J}_i}, \left(\hat{s}^{(i)}_{j,0}, \hat{s}^{(i)}_{j,1}\right)_{j \in \mathcal{J}_{\text{in}}}, \left(k^{(i)}_j, s^{(i)}_{j,0}, s^{(i)}_{j,1}\right)_{j \notin \mathcal{J}_{\text{in}}}$ to the adversary.

Upon $\mathsf{CorruptO}$**:** For each $j \notin \mathcal{J}_{\text{in}}$ that $j \le t + \frac{1}{2}$, 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, $k^{(1)}_j, \ldots, k^{(n)}_j$ are i.i.d. random boolean, and $k_j := k^{(1)}_j \oplus \cdots \oplus k^{(n)}_j$, $\bar{x}_j := k_j \oplus x_j$; while in the 0-th hybrid world, $\bar{x}_j$ and $k^{(i)}_j$ for all corrupted party i are randomly sampled, and $k_j := \bar{x}_j \oplus x_j$. There 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 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

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

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

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, $s_{j,\bar{x}_j\oplus 1} = s^{(1)}_{j,\bar{x}_j\oplus 1} \| \ldots \| s^{(n)}_{j,\bar{x}_j\oplus 1}$ are randomly sampled and $\hat{s}_{j,\bar{x}_j\oplus 1}$ is determined by $\hat{s}_{j,\bar{x}_j\oplus 1} = G^{\mathrm{MP}}(s_{j,\bar{x}_j\oplus 1}) = \bigoplus_i G(s^{(i)}_{j,\bar{x}_j\oplus 1})$. 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 $s^{(i^*)}_{j,\bar{x}_j\oplus 1}$ is only used to generate $\hat{s}_{j,\bar{x}_j\oplus 1}$, thus it is computational indistinguishable if $G(s^{(i^*)}_{j,\bar{x}_j\oplus 1})$ is replaced by uniform randomness. Replacing $G(s^{(i^*)}_{j,\bar{x}_j\oplus 1})$ 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.

6 Two-Round MPC

As what we are going to show in Lemma 9, 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 $\mathbf{NC^1}$ 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 Sect. 6.1 and 6.2.

Lemma 9. *Let $(\hat{f}, h_1, \ldots, h_n)$ 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 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 (prefect, 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.*

The proof is deferred to the full version.

6.1 Honest Majority and Plain Model

In the honest majority setting, the BGW [5] protocol when restricted to computing degree-2 polynomials has only two rounds. The adaptive security of BGW is proved in [14].

Lemma 10. *For any degree-2 functionality f, the BGW protocol computes f in 2-round and tolerates adaptive minority corruptions.*

6.2 Honest Minority and 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.

Input P_1 has $x_1, z_1 \in \mathbb{F}$; P_2 has $x_2, z_2 \in \mathbb{F}$.
OLE Correlation Sample random $a_1, a_2, b_1, b_2 \in \mathbb{F}$ such that $a_1 a_2 = b_1 + b_2$.
P_1 has $a_1, b_1 \in \mathbb{F}$, P_2 has $a_2, b_2 \in \mathbb{F}$.
Round 1 P_1 sends $m_{1,1} := x_1 - a_1$ to P_2. P_2 sends $m_{2,1} := x_2 - a_2$ to P_1.
Round 2 P_1 sends $m_{1,1}$ and $m_{1,2} := m_{2,1} x_1 + b_1 + z_1$ to the receiver.
P_2 sends $m_{2,1}$ and $m_{2,2} := m_{1,1} x_2 + b_2 + z_2$ to the receiver.
The receiver outputs $m_{1,2} + m_{2,2} - m_{1,1} m_{2,1}$.

Fig. 7. 2-round MPC for 2MULTPlus in OLE correlation model

Lemma 11. *The 2-round MPC described in Figure 7 is a adaptive secure MPC protocol for the following functionality*

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

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 $x_1x_2 + z_1 + z_2$, and a_1, a_2 are the randomness of the randomness encoding. The formal proof is deferred to the full version.

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

1. Ananth, P., Choudhuri, A.R., Goel, A., Jain, A.: Round-optimal secure multiparty computation with honest majority. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 395–424. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_14
2. Applebaum, B., Brakerski, Z., Tsabary, R.: Perfect secure computation in two rounds. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 152–174. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_6
3. Applebaum, B., Ishai, Y., Kushilevitz, E.: Cryptography in NC^0. SIAM J. Comput. **36**(4), 845–888 (2006)
4. Beaver, D., Micali, S., Rogaway, P.: The round complexity of secure protocols. In: Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, pp. 503–513 (1990). https://doi.org/10.1145/100216.100287
5. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pp. 351–371 (1988). https://doi.org/10.1145/62212.62213
6. Benhamouda, F., Lin, H.: k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 500–532. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_17
7. Benhamouda, F., Lin, H., Polychroniadou, A., Venkitasubramaniam, M.: Two-round adaptively secure multiparty computation from standard assumptions. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 175–205. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_7
8. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y.: Compressing vector OLE. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15–19, 2018, pp. 896–912. ACM (2018). https://doi.org/10.1145/3243734.3243868
9. Boyle, E., Couteau, G., Gilboa, N., Ishai, Y., Kohl, L., Scholl, P.: Efficient pseudorandom correlation generators from ring-LPN. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 387–416. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_14
10. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptology **13**(1), 143–202 (2000). https://doi.org/10.1007/s001459910006

11. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, pp. 639–648. ACM (1996) https://doi.org/10.1145/237814.238015
12. Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols. In: Proceedings of the twentieth Annual ACM Symposium on Theory of Computing, pp. 11–19. ACM (1988). https://doi.org/10.1145/62212.62214
13. Damgård, I., Ishai, Y.: Constant-round multiparty computation using a black-box pseudorandom generator. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 378–394. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_23
14. Damgård, I., Nielsen, J.B.: Adaptive versus static security in the UC model. In: Chow, S.S.M., Liu, J.K., Hui, L.C.K., Yiu, S.M. (eds.) ProvSec 2014. LNCS, vol. 8782, pp. 10–28. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12475-9_2
15. Garg, S., Gentry, C., Halevi, S., Raykova, M.: Two-round secure MPC from indistinguishability obfuscation. In: Lindell, Y. (ed.) TCC 2014. LNCS, vol. 8349, pp. 74–94. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_4
16. Garg, S., Ishai, Y., Srinivasan, A.: Two-round MPC: information-theoretic and black-box. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 123–151. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_5
17. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 468–499. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_16
18. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pp. 218–229. ACM (1987). https://doi.org/10.1145/28395.28420
19. Ishai, Y., Kushilevitz, E.: Randomizing polynomials: a new representation with applications to round-efficient secure computation. In: Proceedings 41st Annual Symposium on Foundations of Computer Science, pp. 294–304. IEEE (2000). https://doi.org/10.1109/SFCS.2000.892118
20. Ishai, Y., Kushilevitz, E.: Perfect constant-round secure computation via perfect randomizing polynomials. In: Widmayer, P., Eidenbenz, S., Triguero, F., Morales, R., Conejo, R., Hennessy, M. (eds.) ICALP 2002. LNCS, vol. 2380, pp. 244–256. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45465-9_22
21. Ishai, Y., Mittal, M., Ostrovsky, R.: On the message complexity of secure multiparty computation. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10769, pp. 698–711. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_24
22. Yao, A.C.C.: Protocols for secure computations. In: 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982), pp. 160–164. IEEE (1982). https://doi.org/10.1109/SFCS.1982.38
23. Yao, A.C.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (sfcs 1986), pp. 162–167. IEEE (1986). https://doi.org/10.1109/SFCS.1986.25