Randomized Secure Two-Party Computation for Modular Conversion, Zero Test, Comparison, MOD and Exponentiation

Ching-Hua Yu*† (chinghua.yu@gmail.com) and Bo-Yin Yang* (by@crypto.tw)

*Institute of Information Science, Academia Sinica, Taipei, Taiwan
†National Taiwan University, Taipei, Taiwan

Abstract. When secure arithmetic is required, computation based on secure multiplication (MULT) is much more efficient than computation based on secure boolean circuits. However, a typical application can also require other building blocks, such as comparison, exponentiation and the modulo (MOD) operation. Secure solutions for these functions proposed in the literature rely on bit-decomposition or other bit-oriented methods, which require $O(\ell)$ MULTs for ℓ-bit inputs. In the absence of a known bit-length independent solution, the complexity of the whole computation is often dominated by these non-arithmetic functions.

To resolve the above problem, we start with a general modular conversion, which converts secret shares over distinct moduli. For this, we proposed a probabilistically correct protocol for this with a complexity that is independent of ℓ. Then, we show that when these non-arithmetic functions are based on secure modular conversions, they can be computed in constant rounds and $O(k)$ MULTs, where k is a parameter for an error rate of $2^{-\Omega(k)}$. To promote our protocols to be actively secure, we apply $O(k)$ basic zero-knowledge proofs, which cost at most $O(k)$ exponentiation computation, $O(1)$ rounds and $O(k(\ell+\kappa))$ communication bits, where κ is the security parameter used in the commitment scheme.

Keywords. secure arithmetic, randomized algorithm, modular conversion, comparison, zero test, modulo reduction, modular exponentiation

1 Introduction

Secure two-party computation allows two parties, Alice and Bob, to jointly compute a function $f(\boldsymbol{x^A}, \boldsymbol{x^B})$ without revealing anything about their inputs, where $\boldsymbol{x^A}$ and $\boldsymbol{x^B}$ are secret vectors held by Alice and Bob respectively. Solutions based on boolean circuits [30, 17, 3, 8] provide general feasibility results, while solutions based on secure arithmetic computation, e.g., using homomorphic encryption or oblivious transfer [19, 13, 31], are efficient for addition and multiplication. Specifically, in arithmetic applications, such as data mining, statistical learning and distributed generation of cryptographic keys [20, 6, 4, 21, 11], the second choice is usually considered more practically efficient.

Besides addition and multiplication, several non-arithmetic functions involving integer comparison, zero test, the modulo (MOD) operation and exponentiation also play important roles in many arithmetic applications. To get the best of both worlds, several recent works have used secure arithmetic computation and considered how to compute the non-arithmetic functions efficiently [9, 28, 24, 19]. However, the complexity of the non-arithmetic computation in these works is still much larger than that of secure multiplication. Therefore, even though the arithmetic functions can be computed efficiently, the computation of these non-arithmetic functions may dominate the complexity in many applications. For instance, from the existing results, we can use bit-decomposition, which converts a secret number into a secret binary set [9, 28, 24], as a generic solution to compute non-arithmetic functions in a bit-oriented manner; or we can use more efficient solutions tailored for specific problems, such as integer comparison [26], MOD operation [24, 18] and modular exponentiation [31]. Unfortunately, the complexity of these solutions is still at least $O(\ell)$ secure multiplications[1], where ℓ is the bit-length of the inputs. Hence, there follow two interesting questions. First, it

[1] Using the method in [31], integer exponentiation takes $O(1)$ secure multiplications, but modular exponentiation requires an additional comparison and zero test. Thus, the complexity is still bounded by $O(\ell)$ for general cases.

is open whether there are constant-rounds and probabilistically correct solutions to these problems such that the complexity only depends on a correctness parameter k and has an exponentially low error rate of $2^{\Omega}(k)$. From a theoretical view, such solutions serve as evidences of complexity breakthrough regarding non-arithmetic computation using arithmetic operations and are in accord with the direction of a recent study [29], regarding sub-linear secure comparison (whose complexity is $O(\sqrt{\ell}(k + \log \ell))$ secure multiplications and constant rounds). From a practical view, such solutions provide a trade-off between the efficiency and the error rate. Second, in order to construct solutions that are independent of bit length, we would require some undiscovered schemes which are not bit-oriented, and in particular, we ask whether there is a unified framework, better than bit-decomposition, for integer comparison, zero test, MOD operation and modular exponentiation.

1.1 Problem Statement

We study secure two-party computation of several non-arithmetic functions that involve zero test (equality test), integer comparison, the MOD operation, and modular exponentiation based on the modular conversion protocol. The objective is to find bit-length independent solutions, where the complexity counts the number of secure multiplication.

Instead of using bit-decomposition as the bridge between the arithmetic functions and the non-arithmetic functions, we use *modular conversion*, which converts the shares in $\mathbb{Z}_Q$ into shares in $\mathbb{Z}_P$. The setting of modular conversion is as the follows. Let $Q, P \geq 2$ be two distinct integers, and let x be a secret in $\mathbb{Z}_Q$.[2] In addition, let x^A and x^B be additive shares of x in $\mathbb{Z}_Q$, i.e., $x = x^A + x^B \mod Q$, held by Alice and Bob respectively. Modular conversion converts x^A and x^B into z^A and z^B such that $x = z^A + z^B \mod P$.

On one hand, modular conversion can serve a general purpose. First, modular conversion is related to the MOD operation, which computes the remainder of x divided by an integer P, while the input and the output are both shares in $\mathbb{Z}_Q$. Second, let ℓ be the bit-length of Q. Bit-decomposition can be reduced to $\ell - 1$ parallel MOD operations, computing $x \mod 2^{\ell-1}, ..., x \mod 2$ and some linear combinations. This implies the general use of modular conversion for binary functions. Moreover, because the MOD operation is reducible to modular conversion (discussed in Section 3.7), but not vice versa, modular conversion serves as a more general functionality.

On the other hand, for many well used functions like zero test (equality test), integer comparison, the MOD operation, and modular exponentiation, it may be not necessary to perform full bit-decomposition, as the solutions can be constructed directly by modular conversion.

1.2 Contributions

We propose a new secure two-party modular conversion scheme for general moduli. That is, the input and the output are secret shares in $\mathbb{Z}_Q$ and $\mathbb{Z}_P$ respectively, where Q and P are arbitrary integers greater than or equal to 2. Then, using modular conversion, we derive efficient solutions to several non-arithmetic functions.

Our framework for the general modular conversion protocol is based on a statistically correct computation which relies on the manipulation of secret shares. When the secret shares of an input are generated uniformly, the error rate of the computation is constant. Then, using uniformly random re-sharing k times, we construct a solution with an error rate of $2^{-\Omega(k)}$.

To obtain solutions that would defeat a malicious adversary, we use Multiplicative to Additive Sharing Conversion (or secure multiplication) to construct the protocols of these functions in the

[2] In this paper, without specification, $x \in \mathbb{Z}_Q$ simply means $x \in \{0, ..., Q-1\}$, and $y = x \mod Q$ means $y \in \{0, ..., Q-1\}$.

passive security model. Then we show that these protocols can be promoted to be actively secure by efficient arithmetic zero-knowledge proofs.

In the passive security model, we achieve the following results.

- A general modular conversion protocol that uses $O(k)$ (about $7k + 30$) secure multiplications and $O(1)$ (about 8) rounds, with an error rate of $2^{-\Omega(k)}$ (at most $(7/10)^k$ for all $k \geq 20$).
- A randomized zero-test protocol whose cost is $O(k)$ (about k) secure multiplications and $O(1)$ (about 8) rounds with an error rate of 2^{-k} at most.
- Solutions to the sign function, integer comparison, the MOD operation and modular exponentiation based on the modular conversion and zero test are described. The complexity of all the solutions is all $O(k)$ secure multiplications and O(1) rounds, with an error rate of $2^{-\Omega(k)}$.

In the active security model, we promote our passively secure protocols via specific arithmetic zero-knowledge proofs, which involve at most $O(k)$ basic proofs. The later show the knowledge of a committed value and the knowledge of a multiplicative relationship. The additional costs are at most $O(1)$ rounds of communication, $O(k)$ exponentiation computations and $O(k(\ell+\kappa))$ communication bits, where κ is the security parameter used in the commitment scheme.

1.3 Related Works

Share Conversion is usually regarded as a tool for bridging different kinds of computation. In [9], Damgård, et al. proposed a constant-rounds protocol of bit-decomposition to support integer comparison, zero test, the MOD operation and exponentiation in arithmetic circuits. The communication complexity of the bit-decomposition protocol in [9] is $O(\ell \log \ell)$ (secure multiplications). Subsequently, it was improved to $O(\ell \log^* \ell)$ in [28]. A statistically secure conversion between integer shares and $\mathbb{Z}_Q$ shares was proposed in [1], and a deterministic conversion from a prime field $\mathbb{Z}_Q$ to $\mathbb{Z}_{Q-1}$ is presented in [31]. Both solutions are efficient compared to secure multiplications; however the former requires the secret $x < \frac{Q}{16} \cdot \frac{2^{-\rho}}{n}$, where ρ is a security parameter, and the latter requires $x < \frac{Q}{2}$. Although the solutions do not work for general cases, they motivate the direction of our work. Moreover, in [31], efficient protocols for the transformation between additive and multiplicative shares are proposed to take care the private exponentiation in a non-bit-oriented manner.

Integer Comparison is an important functionality in many arithmetic applications. Besides using bit-decomposition [9, 1], a protocol for secure comparison without bit-decomposition is described in [26]. Recently, Toft [29] proposed a sub-linear protocol for secure two-party comparison (or an extension for two non-colluding parties known in multi-parties.) Their constant-rounds protocol involves $\sqrt{\ell}$ equality tests (in parallel), each of which can be performed by $O(k)$ for a 2^{-k} error rate using a similar scheme to that in [26]. Hence, the total complexity of the protocol is $O(\sqrt{\ell}(k+\log \ell))$.

MOD Operation. In [24], a protocol that uses constant rounds and $O(\ell)$ secure multiplications is proposed for secure MOD operations without using bit-decomposition. Recently, Guajardo et al. [18] introduce a statistically secure solution for modulo reduction. It requires at most $O(n^2\kappa\ell_a)$ secure multiplications where n is the number of parties, κ is a security parameter, and ℓ_a is the bit length of a (public) modulus. However, for general cases, such as $a = Q - 1$ or $a = Q/2$, this solution still requires $O(\ell)$ secure multiplications.

Modular Exponentiation. The first constant-rounds solution proposed by Damgård et al. [9] is based on bit-decomposition, which dominated the complexity. More recently, Yu et al. [31] developed a constant-rounds protocol for secure two-party exponentiation using $O(\ell)$ secure multiplications. Furthermore, the protocol's complexity is dominated by modular reduction, which we will improve to derive a better result.

For these non-arithmetic problems for secure two or multi-party computation, there is no known arithmetic solution whose communication complexity is independent of ℓ (counting the number of secure multiplications). This motivates the study of this paper.

2 Preliminaries

2.1 Passive Security and Secure Arithmetic Computation for Two Parties

We assume the computation involves two non-colluding parties, Alice and Bob, communicating over a public channel. First, we informally define passive security for generic two-party computation.

Passive Security. For a functionality $f(\cdot,\cdot)$, a protocol is said to be *passively secure* or *semi-honest secure* if, for any honest-but-curious adversary $\mathcal{A}$ who corrupts Alice, there exists a probabilistic polynomial time simulator $\mathcal{S}$ that, given the inputs and the randomness of $\mathcal{A}$, can produce a view of $\mathcal{A}$ which is (statistically/computationally) indistinguishable from the real interaction with Bob. A similar rationale should hold if Bob is corrupted. For a standard formal definition, readers may refer to [16].

Instead of using generic boolean circuits, we construct protocols based on arithmetic operations, i.e., additions and multiplications of (sufficiently large) finite fields/rings. This setting is particularly suitable for arithmetic applications, where most computation involves arithmetic operations, such as the computation of integers or finite field/ring elements. In addition, for a modular design, we assume that all the inputs and outputs of the protocols are secret shares.

Secret Shares. To protect a secret, such as a private value, a widely used method hides the secret in secret shares. In two-party computation, secret shares are usually expressed as an additive form. For example, let x be a secret in $\mathbb{Z}_Q$. If Alice and Bob hold x^A and x^B respectively such that $x = x^A + x^B \mod Q$, Alice's (resp. Bob's) view on x^A (resp. x^B) should be indistinguishable from a uniform random value in $\mathbb{Z}_Q$, so that Alice (resp. Bob) cannot learn anything about x from x^A (resp. x^B). In this scenario, x is called a shared secret. To formulate the computation based on secret shares, all the inputs and outputs of a protocol are expressed as shared secrets, e.g, $(y^A, y^B) \leftarrow f(A(x^A), B(x^B))$, where the superscript A (resp. B) of x^A (resp. x^B) denotes Alice's (resp. Bob's) share of x.

Because of the additive sharing form, the addition of two shared secrets can be computed locally, as can any linear combination of secrets. On the other hand, the multiplication of two secrets would involve communication in this setting. Furthermore, in secure two-party computation, as well as secure multiplication, a closely related and more elementary protocol, namely computing the additive shares of the multiplication of two private values held by Alice and Bob respectively, is often used, as described below.

Multiplicative to Additive Sharing ($\mathsf{M2A}$). $\mathsf{M2A}$ is a basic functionality in secure two-party computation. Let x^A and x^B be the secret inputs of Alice and Bob respectively. An $\mathsf{M2A}$ protocol outputs y^A and y^B, held by Alice and Bob respectively, such that $y^A + y^B = x^A \cdot x^B \mod Q$. We formulate this procedure as $(y^A, y^B) \leftarrow \mathsf{M2A}_Q(A(x^A), B(x^B))$.

Secure Multiplication (MULT). MULT is a functionality that securely computes the multiplication of two shared secrets. For example, Alice and Bob have x^A, y^A and x^B, y^B respectively as inputs, where $x = x^A + x^B \mod Q$ and $y = y^A + y^B \mod Q$; and they get z^A and z_B respectively as outputs such that $z^A + z^B = x \cdot y \mod Q = (x^A + x^B) \cdot (y^A + y^B) \mod Q$. Since $x^A y^A$ and $x^B y^B$ can be computed locally by Alice and Bob respectively, the procedure involves two invocations of $\mathsf{M2A}$ for $x^A y^B$ and $x^B y^A$. We formulate the procedure as $(z^A, z^B) \leftarrow \mathsf{MULT}_Q(A(x^A, y^A), B(x^B, y^B))$.

Multiplication Oracle. There are several ways to implement M2A and MULTbased on distinct primitive assumptions. For example, M2A can be implemented by homomorphic encryption systems or oblivious transfers, using only a few ciphertexts [19, 13, 31], and the computational assumption should follow the primitives. In addition, for simplicity, as well as consistency and ease of comparison with previous works, we assume M2A and MULT are secure and take them as the cost units. We regard the computation of one MULT or M2A as one communication round; however concrete schemes may require a few rounds and a few ciphertexts.

2.2 Active Security

The active security model in this paper follows the standard definition of security against a malicious adversary given in [16]. In this model, the corrupted party (Alice or Bob) can deviate arbitrarily from the prescribed protocol. For two-party computation, secret shares alone cannot guarantee the active security, since a corrupted party may falsify his shares. Hence, we also utilize an integer commitment scheme.

Commitment. A commitment scheme allows the sender to send a commitment, an encrypted value, to the receiver without revealing the specific value (called the hiding property). When the sender subsequently decommits/opens the value, the scheme should ensure that the value committed previously is the only one that the receiver can validate from the commitment (called the binding property). Let x be a secret, and denote a commitment of x as $C(x, r)$, where r is an independent random number sampled according to concrete commitment schemes, such as [12, 10, 15], and also as $C(x)$ for short.

To formulate the computation based on the commitments, we assume that input shares and output shares are accompanied by their commitments, e.g., $(y^A, C(y^A), y^B, C(y^B)) \leftarrow f(A(x^A, C(x^B)), B(x^B, C(x^A)))$. The private random numbers, messages, and inputs of the sub-protocols should also be committed.

Then, to convert a passively secure protocol to an actively secure one, we "compile" the protocol into zero-knowledge proofs/protocols. That is, Alice and Bob should prove to each other that they are running the protocol honestly to each other.

Zero-Knowledge Proof. Zero-knowledge proofs allow each party to prove a statement without revealing anything other than the veracity of the statement. Moreover, if the statement is true, the honest verifier will be convinced; and if the statement is false, the cheating party will not be able to convinced the verifier that it is true. Zero-knowledge proofs are then based on the operations of the commitments. In this paper, we count on arithmetic/integer based proofs. We describe the details in Section 4.

2.3 Some Primitives.

Random Invertible Secret. In [2], Bar-Ilan and Beaver propose a method that allows the parties to share a random, unknown, invertible field element. However, the method is mainly applicable to the multiparty and information-theoretical security model. For the two-party case, further commitments and zero-knowledge proofs should be appended. Alternatively, the following simple implementation can be used for the two-party case. To generate a random invertible secret in a prime field $\mathbb{Z}_Q$ in the two-party case, Alice and Bob pick uniformly random values a^A and a^B respectively in $\mathbb{Z}_Q^*$. Then, they invoke an $\mathsf{M2A}_Q$ to compute $r^A + r^B = a^A \cdot a^B \mod Q$. In the active security model, Alice and Bob need to prove that a^A and a^B are non-zero elements. To this end, they first generate $a^A, a^B, b^A, b^B \in_R \mathbb{Z}_Q^*$ accompanied by the commitments $C(a^A), C(a^B), C(b^A), C(b^B)$. Then, they

open and prove $t^A = a^A \cdot b^A$ and $t^B = a^B \cdot b^B$ respectively. If t^A or t^B is zero, the procedure terminates; otherwise they invoke $\mathsf{M2A}_Q$ once to compute r^A and r^B such that $r^A + r^B = a^A \cdot a^B \mod Q$. In addition, when a^A or a^B is picked uniformly in $\mathbb{Z}_Q^*$, $r = r^A + r^B \mod Q$ is a uniformly random value in $\mathbb{Z}_Q^*$. The cost of the whole procedure is less than that of one (actively secure) MULT_Q. We regard it as 1 round in the same way as MULT.

Moreover, let $s^A = (a^A)^{-1} \mod Q$ and $s^B = (a^B)^{-1} \mod Q$. Alice and Bob can compute r'^A and r'^B such that $r'^A + r'^B = s^A \cdot s^B = r^{-1} \mod Q$ by one additional invocation of $\mathsf{M2A}_Q$. In the active security model, a knowledge proof of $s^A \cdot a^A = 1$ and $s^B \cdot a^B = 1$ is appended. The total cost of generating the secret shares of r and r^{-1} is at most 2 MULT_Q's (in parallel).

Polynomial Function Evaluation (POLY). Given a non-zero shared secret $x = x^A + x^B \mod Q$ as the input, we use the following method to compute a polynomial function $f(x) = \sum_{i=1}^{k} c_i x^i$, where $c_1, ..., c_k$ are public constants. First, Alice and Bob generate random invertible secrets s^A and s^B respectively and compute $(s^A)^{-1}, ..., (s^A)^{-k}$, $(s^B)^{-1}, ..., (s^B)^{-k}$ respectively (along with their commitments in the active security model). For $i = 1, ..., k$, Alice and Bob compute $(t_i^A, t_i^B) \leftarrow \mathsf{M2A}_Q(A((s^A)^{-i}), B((s^B)^{-i}))$. Then, using two $\mathsf{M2A}$ (in parallel), they compute and reveal $a = s^A \cdot s^B \cdot (x^A + x^B) \mod Q = (s^A x^A) \cdot s^B + s^A \cdot (s^B x^B)$. Finally, they locally compute $z^A = \sum_{i=1}^{k} c_i a^i t_i^A$ and $z_i^B = \sum_{i=1}^{k} c_i a^i t_i^B$ respectively as the outputs. The total cost is less than the cost of $k+4$ $\mathsf{M2A}_Q$'s in 3 rounds. We formulate it as $(z^A, z^B) \leftarrow \mathsf{POLY}_Q(f(x), A(x^A), B(x^B))$. Note that POLY is only used in the cases where the input secret x is non-zero.

3 Randomized Protocols with Passive Security

3.1 Our Framework

In this section, we explain how several non-arithmetic functions can be implemented by using secure arithmetic computation in the passive security model.

First, we construct a protocol for modular conversion that converts shares in $\mathbb{Z}_Q$ into shares in $\mathbb{Z}_P$. Let x be a secret in $\mathbb{Z}_Q$, and let x^A and x^B be the additive shares of x in $\mathbb{Z}_Q$ (i.e., $x = x^A + x^B \mod Q$) held by Alice and Bob respectively. Modular conversion converts x^A and x^B into z^A and z^B such that $x = z^A + z^B \mod P$.

To obtain a *general* modular conversion, we first build more limited protocols that only work with the input secret in a certain range. Then, we tweak them to ensure their validity regardless of the values of the secret shares. With a uniformly random re-sharing, we can derive a randomized algorithm of modular conversion that is statistically correct with no constraint on the input secret.

In addition to the general modular conversion protocol, a randomized zero test protocol can be built by using modular conversion protocols of limited validity. Using the modular protocols as primitives, we also build several applications that involve sign test, integer comparison and MOD operations. The applications form the centerpiece of an efficient modular exponentiation scheme, as shown by the protocol hierarchy in Fig. 1.

3.2 Modular Conversion with Input Constraints

We first implement the modular conversion protocols with some constraints on the input secret x. If an input satisfies the constraints, it can be assumed that partial information about the input secret is known. Specifically, when the sign of the input (whether $x < Q/2$ or $x \geq Q/2$) is known, the modular conversion of x from Q to P for arbitrary Q and P can be computed efficiently. Using this information to determine whether a wrap-around modulo Q occurs in $x = x^A + x^B \mod Q$, only

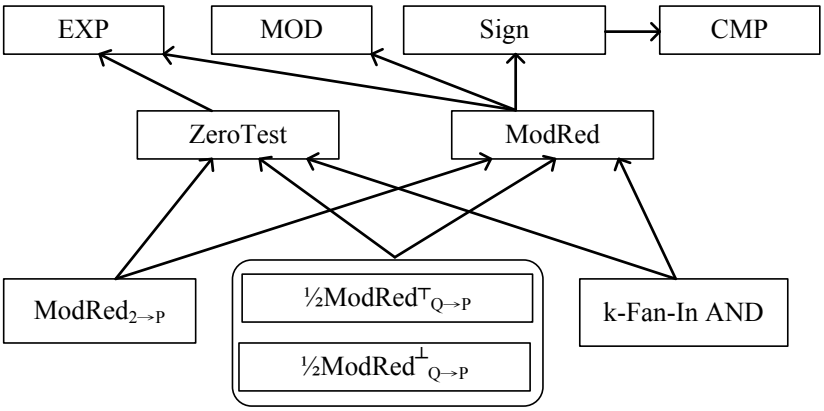

Fig. 1. Protocol hierarchy

one invocation of $\mathsf{M2A}_P$ is required. Hence, based on this observation, we implement two function-limited protocols: one $\frac{1}{2}$modular conversion protocol for the case where $x < Q/2$ ($\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$) and one for the case where $x \geq Q/2$ ($\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}$), as shown in Fig. 2. Moreover, when Q is a small constant, the modular conversion of x from Q to an arbitrary P can be computed using $O(1)$ MULT_P's, deterministically and validly without input constraints. Specifically, when $Q = 2$, the implementation of $\mathsf{ModCNV}_{2\to P}$ only requires one $\mathsf{M2A}_P$, as shown in Fig. 2.

Correctness. We only reason the case where $x < Q/2$ ($\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$) because the case where $x \geq Q/2$ ($\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}$) is similar, and the correctness of $\mathsf{ModCNV}_{2\to P}$ is easy to follow. First, note that $x^A + x^B$ can only be $x + Q$ or x depending on whether a wrap-around modulo Q occurs. If $x^A, x^B < Q/2$, this can only be the latter one, i.e., $x = x^A + x^B$. If $x^A, x^B \geq Q/2$, it is also sure that $x = x^A + x^B - Q$. Otherwise, if $x^A < Q/2, x^B \geq Q/2$ or $x^A \geq Q/2, x^B < Q/2$, we have $Q/2 \leq x^A + x^B < Q/2 + Q$. Specifically, when $x < Q/2$, $x^A + x^B$ can only be $x + Q$.Therefore, when $x < Q/2$, only the case of $x^A, x^B < Q/2$ does not involve a wrap-around modulo Q. This corresponds to the boolean test in Step 3, where $z'^A + z'^B \mod P = b^A b^B = (x^A < Q/2) \wedge (x^B < Q/2)$. Since we have $x = x^A + x^B + ((z'^A + z'^B \mod P) - 1)Q$, which implies that $x \mod P = (x^A + z'^A Q \mod P) + (x^B + (z'^B - 1)Q \mod P) \mod P$, protocol $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ is valid for $x < Q/2$.

Security and Complexity. The communication part of the above protocols only involves one invocation of $\mathsf{M2A}_P$, so the security and efficiency follows from those of $\mathsf{M2A}_P$.

3.3 *m*-Fan-In AND

An m-fan-in AND (AND^m_Q) operation is often applied with parallel boolean tests. That is, a series of parallel tests are performed first and output m secret values, each of which is either 0 or 1 , shared in $\mathbb{Z}_Q$. Then, AND^m_Q is computed to determine whether all the m secrets are true. We use this functionality in our construction of general modular conversion, where we apply several randomized tests in parallel.

The functionality of AND^m_Q is similar to the unbounded fan-in multiplication in [2], which, however, only supports non-zero inputs and the finite fields. By contrast, the inputs of AND^m_Q can be 0, and Q can be an arbitrary integer greater than or equal to 2. Hence, instead of applying the unbounded fan-in multiplication directly, we use polynomial function evaluation and take an

Two half valid modular conversion protocols:

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ abd Bob holds $x^B \in \mathbb{Z}_Q$, such that $x = x^A + x^B \mod Q$. Q, P are two public integers, $Q > 2, P \geq 2$.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B = x \mod P$.

Case 1: $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$, valid for $x < \frac{Q}{2}$:

1. Alice computes a number b^A locally such that $b^A = 1$ if $x^A < \frac{Q}{2}$, otherwise 0.
2. Bob computes a number b^B locally such that $b^B = 1$ if $x^B < \frac{Q}{2}$, otherwise 0.
3. Run $(z'^A, z'^B) \leftarrow \mathsf{M2A}_P(A(b^A), B(b^B))$.
4. Alice computes $z^A = (x^A + z'^A \cdot Q) \mod P$, and Bob computes $z^B = (x^B + (z'^B - 1) \cdot Q) \mod P$

Case 2: $\frac{1}{2}\mathsf{ModCNV}^{\bot}_{Q\to P}(A(x^A), B(x^B))$, valid for $x \geq \frac{Q}{2}$:

1. Alice computes a number b^A locally such that $b^A = 1$ if $x^A \geq \frac{Q}{2}$, otherwise 0.
2. Bob computes a number b^B locally such that $b^B = 1$ if $x^B \geq \frac{Q}{2}$, otherwise 0.
3. Run $(z'^A, z'^B) \leftarrow \mathsf{M2A}_P(A(b^A), B(b^B))$.
4. Alice computes $z^A = (x^A - z'^A \cdot Q) \mod P$, and Bob computes $z^B = (x^B - z'^B \cdot Q) \mod P$

$\mathsf{ModCNV}_{2\to P}(A(x^A), B(x^B))$ modular conversion protocol for the case where $x \in Z_2$:

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_2$ and Bob holds $x^B \in \mathbb{Z}_2$ such that $x = x^A + x^B \mod 2$. P is a public integer, $P > 2$.
- **Outputs:** Alice obtains $z^A \in Z_P$ and Bob obtains $z^B \in Z_P$ such that $z^A + z^B = x \mod P$.

1. Run $(z'^A, z'^B) \leftarrow \mathsf{M2A}_P(A(x_A), B(x_B))$.
2. Alice computes $z^A = (x^A - 2 \cdot z'^A) \mod P$, and Bob computes $z^B = (x^B - 2 \cdot z'^B) \mod P$.

Fig. 2. Protocol $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$, $\frac{1}{2}\mathsf{ModCNV}^{\bot}_{Q\to P}(A(x^A), B(x^B))$ and $\mathsf{ModCNV}_{2\to P}(A(x^A), B(x^B))$

m-fan-in AND operation as an $(m+1)$-degree polynomial function over $\mathbb{Z}_Q$. In addition, in order to support a general modulus (for all $Q \geq 2$) when $Q \leq m+1$ or when Q is not a prime, we should first convert the shares in $\mathbb{Z}_Q$ to shares in $\mathbb{Z}_P$ for a sufficiently large prime P so that the polynomial function can be correctly evaluated.

The protocol is shown in Fig. 3. Note that when $Q > m+1$ and Q is a prime, the modular conversion in Step 1 and 5 can be omited. The correctness of the protocol is proven by the following theorem.

Theorem 1 (Correctness). *For $i = 1$ to m, let x_i^A and x_i^B be shares of x_i in $\mathbb{Z}_Q$, i.e., $x_i = x_i^A + x_i^B \mod Q$. If $x_i \in \{0, 1\}$, then $(z^A, z^B) \leftarrow \mathsf{AND}^m_Q(A(x_1^A, ..., x_m^A), B(x_1^B, ..., x_m^B))$ form shares of $(x_1 \wedge ... \wedge x_m)$ in $\mathbb{Z}_Q$.*

Proof.

1. In Step 1 of the protocol, when $Q > 2$, since $x \in \{0, 1\} < Q/2$, from the validity of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$, x_i is successfully converted into shares in $\mathbb{Z}_P$ (i.e., $x = x_i'^A + x_i'^B \mod P$). Similarly when $Q = 2$, from the validity of $\mathsf{ModCNV}_{2\to P}$, the conversion is also successful.
2. In Step 2, $y = 1 + \sum_{i=1}^{k} x_i' \pmod P$ is a non-zero value.
3. In Step 3, since P is a prime greater than $k+1$, $f(x)$ is a $(k+1)$-degree polynomial function with $(k+1)$ distinct roots: $1, ..., k$ and $(k!)^{-1} - k - 1$. Since y is in $\{1, ..., k+1\}$, it implies that if $y = k+1$, $f(y) = 1$; otherwise, $f(y) = 0$. Because $y = k+1$ if and only if $x_1 = ... = x_k = 1$, $f(y)$ is a valid evaluation of $(x_1 \wedge ... \wedge x_k)$ in $\mathbb{Z}_P$.

4. In Step 4, since P is a prime, POLY_P can be evaluated correctly.
5. In Step 5, since $x \in \{0,1\} < P/2$, from the validity of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{P\to Q}$, the protocol outputs a pair of shares of $(x_1 \wedge ... \wedge x_k)$ in $\mathbb{Z}_Q$. □

Security and Complexity. The communication part of AND^m_Q involves $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ (or $\mathsf{ModCNV}_{2\to P}$ when $Q = 2$), POLY_P and $\frac{1}{2}\mathsf{ModCNV}^{\top}_{P\to Q}$. Hence, the security of AND^m_Q follows from the security of these sub-protocols. The total cost is about 5 rounds and $m + 5$ $\mathsf{M2A}_P$ and 1 $\mathsf{M2A}_Q$.

- **Inputs:** Alice holds $x^A_1, ..., x^A_m \in \mathbb{Z}_Q$ and Bob holds $x^B_1, ..., x^B_m \in \mathbb{Z}_Q$ such that $x_i = x^A_i + x^B_i \mod Q, x_i \in \{0,1\}, \forall i = 1, ...m$. Q is a public number.
- **Outputs:** Alice obtains $z^A \in Z_Q$ and Bob obtains $z^B \in Z_Q$ such that $z^A + z^B = \wedge^m_{i=1} x_i \mod Q$.

Let P be the smallest prime $> m + 1$.

1. For $i = 1, ..., m$, Alice and Bob compute $(x'^A_i, x'^B_i) \leftarrow \mathsf{ModCNV}_{2\to P}(A(x^A_i), B(x^B_i))$ when $Q = 2$, else $(x'^A_i, x'^B_i) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A_i), B(x^B_i))$.
2. For $i = 1, ..., m$, Alice computes $y^A = 1 + \sum^m_{i=1} x'^A_i$, and Bob computes $y^B = \sum^m_{i=1} x'^B_i$. Note that the shared secret $y = y^A + y^B \mod P$ is non-zero.
3. Set a polynomial function $f(x) = \sum^m_{i=1} c_i x^i = (x + (m!)^{-1} - m - 1) \cdot \prod^m_{i=1}(x - i) \mod P$, where $c_1, ..., c_m$ are public constants.
4. Alice and Bob compute $(z'^A, z'^B) \leftarrow \mathsf{POLY}_P(f(y^A + y^B), A(y^A), B(y^B))$.
5. Alice and Bob compute $(z^A, z^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{P\to Q}(A(z'^A), B(z'^B))$.

Fig. 3. m-fan-in AND protocol: $\mathsf{AND}^m_Q(A(x^A_1, ...x^A_m), B(x^B_1, ..., x^B_m))$

3.4 General Modular Conversion

Next we describe a protocol for general modular conversion based on the protocols for modular conversion with input constraints and the protocol for the m-fan-in AND operation. We focus on the case where Q is an odd number. The extension regarding an arbitrary Q ($Q \geq 2$) is discussed in Appendix A.

Recall that, while $\mathsf{ModCNV}_{2\to Q}$ is always valid, the validity of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ is only guaranteed when $x < Q/2$. When $x \geq Q/2$, the validity of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ depends on the value of the shares. Thus, we have the following lemma.

Lemma 1. *Let $Q > 2, P \geq 2$, where $P \nmid Q$. Assume that $\frac{Q}{2} \leq x < Q$ and let x^A and x^B be a pair of shares of x. If x^A is a random number sampled according to the uniform distribution in $\mathbb{Z}_Q$, then the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$ would be $\frac{2(x - \lfloor (Q-1)/2 \rfloor)}{Q}$.*

Proof. For all x in a ring $\mathbb{Z}_Q$, there are Q distinct pairs of (x^A, x^B): $(0, x), ..., (Q - 1, x - Q + 1 \mod Q)$. Moreover, for all x between $\frac{Q-1}{2}$ and $Q - 1$, there are $2(x - \lfloor (Q-1)/2 \rfloor)$ distinct pairs of (x^A, x^B) such that $(x^A < \frac{Q}{2}, x^B \geq \frac{Q}{2})$ or $(x^A \geq \frac{Q}{2}, x^B < \frac{Q}{2})$, which are cases where $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$ fails. If $\frac{Q}{2} \leq x < Q$, the output of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ would vary from the correct output by $(-Q \mod P)$. Hence, for all $\frac{Q}{2} \leq x < Q$, if x^A is a uniformly random number in $\{0, ..., Q - 1\}$, and if $P \nmid Q$, the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$ would be $\frac{2}{Q}\left(x - \left\lfloor \frac{Q-1}{2} \right\rfloor\right)$. □

Specifically, when Q is an odd number, for all $\frac{Q}{2} \leq x < Q$, there always exists at least one pair of shares (x^A, x^B) such that $(x^A < \frac{Q}{2}, x^B \geq \frac{Q}{2})$ or $(x^A \geq \frac{Q}{2}, x^B < \frac{Q}{2})$; and there always exists at least

one pair of shares such that ($x^A < \frac{Q}{2}, x^B < \frac{Q}{2}$). The former case results in a fault in $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$, and the latter works properly in $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$. This implies that $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ can serve as a randomized test. Therefore, we then use uniformly random re-sharing to generate several pairs of shares of x. If $x < Q/2$, $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ would return the same result for all derived pairs of shares; otherwise, with a non-negligible probability, at least one pair of shares would have a different result from the others. However, if x is close to $Q/2$ (resp. $Q-1$), since the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ is close to 0 (resp. 1) , all the k tests of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ would be all correct (resp. wrong) with a high probability. On the other hand, if x is close to $3Q/4$, the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ would be close to $1/2$; and, with high probability, at least one of the k pairs of shares would have a different result. Hence, we test $x + iQ/\sigma \mod Q$ for $i = 0, ..., \sigma - 1$ in parallel. Because at least one of $x, x + Q/\sigma ..., x + (\sigma - 1)Q/\sigma \mod Q$ is close to $3Q/4$, there is a high probability that we can identify it successfully (in a private way). Thus, we have the following lemma.

Lemma 2. *Let $Q > 2, P \geq 2$, where $P \nmid Q$, and let $x \in \mathbb{Z}_Q$. For $i = 0, ..., \sigma - 1$, let $x_i = x + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor$; and let $(x^A_{i,1}, x^B_{i,1}), ..., (x^A_{i,k}, x^B_{i,k})$ be k pairs of shares of x_i, where for all j, $x^A_{i,j}$ is a uniformly and independently random number in $\mathbb{Z}_Q$. In addition, let $(y^A_{i,j}, y^B_{i,j})$ be the outputs of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A_{i,j}), B(x^B_{i,j}))$ and $y_{i,j} = y^A_{i,j} + y^B_{i,j} \mod Q$. Then, the possibility that $y_{i,1} = ... = y_{i,k}$ for all $i = 0, ..., \sigma - 1$ is at most $\left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$.*

Proof. Note that there always exists $x^* \in \{x_0, ..., x_{\sigma-1}\}$ such that $\frac{3Q}{4} - \frac{Q}{2\sigma} \leq x^* \leq \frac{3Q}{4} + \frac{Q}{2\sigma}$. For $j = 1, ..., k$, let (y^{*A}_j, y^{*B}_j) be the outputs of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^{*A}_j), B(x^{*B}_j))$, and let $y^*_j = y^{*A}_j + y^{*B}_j \mod Q$. From Lemma 1, if x^{*A}_j is a uniformly random value in $\mathbb{Z}_Q$, the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ would be between $\frac{1}{2} - \frac{1}{\sigma}$ and $\frac{1}{2} + \frac{1}{\sigma}$. Hence, we have the probability that $y^*_1 = ... = y^*_k$ is at most $\left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$. □

As a result, at least one of $x, x + \lfloor Q/\sigma \rfloor, ..., x + \lfloor (\sigma - 1)Q/\sigma \rfloor \mod Q$ can be identified to be greater than or equal to $Q/2$ with high probability (in a private way). Then, by utilizing $\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}$, which is valid in this interval, we can derive a valid modular conversion of x from modulo Q to P with high probability. Our general modular conversion protocol is shown in Fig. 4. We describe the correctness of this randomized protocol in the following theorem.

Theorem 2 (Correctness). *Let $Q > 2$ and $P \geq 2$ be two numbers, where Q is odd; and let $x^A, x^B \in \mathbb{Z}_Q$ be the secret shares of x in $\mathbb{Z}_Q$ i.e., $x = x^A + x^B \mod Q$. $(z^A, z^B) \leftarrow \mathsf{ModCNV}_{Q\to P}(A(x^A), B(x^B), k)$ form shares of x in $\mathbb{Z}_P$ with an error rate at most $\left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$, where σ is the optimum parameter in $\mathsf{ModCNV}_{Q\to P}$.*

Proof.

1. Note $x^A + x^B = x$ or $x + Q$. If $P|Q$, $x + Q \mod P = x \mod P$. Hence, we have $x \mod P = (x^A + x^B) \mod P = (x^A \mod P) + (x^B \mod P) \mod P$, and the output is valid.
2. In Step 2(b) of the protocol, because r_j is generated uniformly and independently, $x^A_{i,j} = x^A_i - r_j$ will also be a uniformly random number in $\mathbb{Z}_Q$.
3. In Step 2(c), let $y_{i,j} = y^A_{i,j} + y^B_{i,j} \mod 2$. Since $2 \nmid Q$, by Lemma 2, the probability that $y_{i,1} = ... = y_{i,k}$ for all $i = 0, ...\sigma - 1$ is at most $\left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$.
4. In Steps 3 and 4, let $a_i = a^A_i + a^B_i \mod 2$. Since AND^*_2 is valid (by Lemma 1), the probability that at least one of $a_0, ..., a_{\sigma-1}$ is equal to 1 is at least $1 - \left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$.
5. In Step 5, since $\mathsf{ModCNV}_{2\to P}$ is always valid, we have $b^A_i + b^B_i \mod P = a_i$.

6. In Step 6, we compute $\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}$ for $x_i, i = 0$ to $\sigma - 1$. Specifically, if $b_i^A + b_i^B \mod P = 1$, then $x_i \geq Q/2$ and $(w_i'^A, w_i'^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}(A(x_i^A), B(x_i^B))$ is valid.
7. In Step 7, note that, if $x^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor < Q$, no wrap-around modulo Q occurs when x_i^A is computed. Therefore, $x^A \mod P = (x^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \mod P) - (\left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \mod P) = x_i^A - \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \mod P$. Otherwise, if $x^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \geq Q$, we also have $x^A \mod P = (x^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor - Q \mod P) - (\left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor - Q \mod P) = x_i^A + Q - \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \mod P$.
8. Hence, finally in Step 8, when at least one of $b_0, ..., b_{\sigma-1}$ is 1, we will obtain a valid pair of (z^A, z^B), with a probability of at least $1 - \left(\frac{1}{2} + \frac{1}{\sigma}\right)^k$. □

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$ such that $x = x^A + x^B \mod Q$. $Q > 2$ and $P \geq 2$ are two public numbers. Q is odd.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B = x \mod P$.

If $P|Q$, Alice and Bob output $z^A = x^A \mod P$ and $z^B = x^B \mod P$ respectively;
else, let σ be a parameter for the optimum adjustment and k be a parameter for the error rate.

1. For $i = 0, ..., \sigma - 1$, Alice sets $\alpha_i = (x_i^A + \lfloor i \cdot \frac{Q}{\sigma} \rfloor \geq Q)$ and $x_i^A = x^A + \lfloor i \cdot \frac{Q}{\sigma} \rfloor - \alpha_i Q$, and Bob sets $x_i^B = x^B$.
2. Alice and Bob run the following procedure for $j = 1, ..., k$ in parallel:
   (a) Generate a public random number r_j (uniformly and independently).
   (b) For each i, Alice and Bob locally compute $x_{i,j}^A \leftarrow x_i^A + r_j \mod Q$ and $x_{i,j}^B \leftarrow x_i^B - r_j \mod Q$ respectively.
   (c) For each i, run $(y_{i,j}^A, y_{i,j}^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}(A(x_{i,j}^A), B(x_{i,j}^B))$.
3. For each i, compute $(u_i^A, u_i^B) \leftarrow \mathsf{AND}_2^k(A(y_{i,1}^A, ..., y_{i,k}^A)), B(y_{i,1}^B, ..., y_{i,k}^B))$ and $(v_i^A, v_i^B) \leftarrow \mathsf{AND}_2^k(A(1 - y_{i,1}^A, ..., 1 - y_{i,k}^A)), B(-y_{i,1}^B, ..., -y_{i,k}^B))$.
4. For each i, compute $(a_i^A, a_i^B) \leftarrow \mathsf{MULT}_2(A(1 - u_i^A, 1 - v_i^A), B(-u_i^B, -v_i^B))$.
5. For each i, run $(b_i^A, b_i^B) \leftarrow \mathsf{ModCNV}_{2\to P}(a_i^A, a_i^B)$.
6. For each i, run $(w_i'^A, w_i'^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}(A(x_i^A), B(x_i^B))$.
7. For each i, Alice sets $w_i^A = w_i'^A - \lfloor i \cdot \frac{Q}{\sigma} \rfloor + \alpha_i Q \mod P$, and Bob sets $w^B = w'^B$.
8. Compute z^A and z^B such that $z^A + z^B = b_0 w_0 + \bar{b}_0 b_1 w_1 + ... + (\bar{b}_0 ... \bar{b}_{\sigma-2} b_{\sigma-1} w_{\sigma-1}) \mod P$ using $(1 + \sigma)\sigma/2$ MULT_P (where $\bar{b}_i = 1 - b_i$, $b_i = b_i^A + b_i^B \mod P$ and $w_i = w_i^A + w_i^B \mod P$).

Fig. 4. General modular conversion protocol: $\mathsf{ModCNV}_{Q\to P}(A(x^A), B(x^B), k)$

Security. Besides generating k public random numbers, which do not contain any information about the secret values, the communication only takes place in the sub-protocols. Hence, the security also follows from that of the sub-protocols.

Complexity and Parameter Optimization. Note that Step 8 can be computed by using $(1 + \sigma)\sigma/2$ MULT_P in $\log_2 \sigma$ parallel rounds, and Steps 5 and 6 can be computed in parallel. The communication part of the protocol involves generating k random public numbers, $k\sigma$ $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$, σ AND_2^k, k MULT_2, k $\mathsf{ModCNV}_{2\to P}$, k $\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}$, and $(1+\sigma)\sigma/2$ MULT_P. For simplicity, we estimate the cost by the number of MULT_P, counting the cost of 2 $\mathsf{M2A}_P$ as 1 MULT_P, and note that the cost of MULT_2 is no more than MULT_P. The total cost is at most $(1 + \sigma)\sigma/2 + k\sigma + 2k + 3\sigma$ MULT_P and $5 + \lceil \log_2 \sigma \rceil$ rounds. Since σ is a constant, the complexity is $O(k)$ MULT_P and $O(1)$ rounds.

Furthermore, σ (and k) can be chosen to optimize the cost. Suppose we expect that the protocol to have an error rate of at most $2^{-c}, c > 0$. According to Theorem 2,

$$\left(\frac{1}{2}+\frac{1}{\sigma}\right)^k \leq 2^{-c} \implies k \geq \frac{-c}{\log_2(\frac{1}{2}+\frac{1}{\sigma})},$$

so the total cost would be at most

$$\frac{(7+\sigma)\sigma}{2} + \frac{-c(\sigma+2)}{\log_2(\frac{1}{2}+\frac{1}{\sigma})} \ \mathsf{MULT}_P\text{'s}.$$

Hence, for optimization purpose, we choose $\sigma = 3$ for $0 < c \leq 1$, $\sigma = 4$ for $2 \leq c \leq 9$ and $\sigma = 5$ for all $c \geq 10$. We summarize the complexity with this optimization strategy in the following lemma.

Lemma 3. *The optimum parameter σ in $\mathsf{ModCNV}_{Q\to P}$ is always less than or equal to 5; and the cost of $\mathsf{ModCNV}_{Q\to P}$ is at most $7k+30$ MULT_P's and 8 rounds with an error rate $\left(\frac{7}{10}\right)^k$ at most when $k \geq 20$.*

3.5 Zero Test

Nishide and Ohta [26] proposed a non-bitwise zero test by calculating Legendre symbol multiple times. Given an odd prime Q, the solution is based on the observation that if $a = 0$," $\left(\frac{a+r}{Q}\right) = \left(\frac{r}{Q}\right)$" is always true; and if $a \neq 0$, it is valid with a probability of about $1/2$ for a random value r uniformly sampled from $\mathbb{Z}_Q$. The method requires $12k$ MULT_Q with an error rate of $\left(\frac{1}{2}\right)^k$. Instead of comparing the Legendre symbol, we provide an even more lightweight solution for two-party computation based on $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$. Here, we consider the case where Q is a prime. An extension for an arbitrary modulus is described in Appendix A.

First, note that the validity of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ is only guaranteed when the input secret is less than $Q/2$. In addition, by Lemma 1, the error rate increases linearly with x when $x \geq Q/2$ (when the shares of x are sampled uniformly from $\mathbb{Z}_Q$). However, because the counts of odd and even numbers in $[Q/2, Q-1]$ are the same, when x is sampled uniformly from $\mathbb{Z}_Q$ (or $\mathbb{Z}^*_Q$), $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ has approximately the same probability of returning 1 or returning 0. The following lemma states the condition more accurately.

Lemma 4. *Let Q be an odd number ($Q > 2$), r be a random number uniformly sampled from $\mathbb{Z}^*_Q$, r^A be a random number uniformly sampled from $\mathbb{Z}_Q$, and $r^B = r - r^A \mod Q$. In addition, let (z^A, z^B) be an output of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}(A(r^A), B(r^B))$ and $z = z^A + z^B \mod 2$. Then, when $Q = 4a+1$, $Pr[z=1] = \frac{Q+1}{2Q}$; otherwise, (when $Q = 4a-1$), $Pr[z=1] = \frac{Q^2+1}{2Q(Q-1)}$.*

Proof. For either $Q = 4a+1$ or $Q = 4a-1$, the count of odd numbers less than $Q/2$ and the count of odd numbers greater than $Q/2$ are both a. For all $r < Q/2$, $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}(A(r^A), B(r^B))$ always returns a valid result; however, for all $r \geq Q/2$, since r^A is uniformly sampled from $\mathbb{Z}_Q$, by Lemma 1, the error rate of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}(A(r^A), B(r^B))$ is $\frac{2}{Q}\left(r - \left\lfloor\frac{Q-1}{2}\right\rfloor\right)$.

Hence, when $Q = 4a+1$, we have

$$\begin{aligned} Pr[z=1] &= \frac{1}{Q-1}\left(a + \frac{Q-2}{Q} + \frac{4}{Q} + \frac{Q-6}{Q} + ... + \frac{3}{Q} + \frac{Q-1}{Q}\right) \\ &= \frac{1}{(Q-1)Q}\left(\frac{(Q-1)Q}{4} + (4+8+...+Q-1) + (3+7+...+Q-2)\right) = \frac{Q+1}{2Q}. \end{aligned}$$

Otherwise, when $Q = 4a - 1$, we have

$$Pr[z = 1] = \frac{1}{Q-1}\left(a + \frac{2}{Q} + \frac{Q-4}{Q} + \frac{6}{Q} + ... + \frac{3}{Q} + \frac{Q-1}{Q}\right)$$
$$= \frac{1}{(Q-1)Q}\left(\frac{(Q+1)Q}{4} + (2+6+...+Q-1) + (3+7+...+Q-4)\right) = \frac{Q^2+1}{2Q(Q-1)}.$$

□

Combining this with the fact that because Q is a prime, $f(r) = rx : \mathbb{Z}_Q^* \to \mathbb{Z}_Q^*$ is isomorphic. Therefore, we provide an efficient randomized protocol through random re-sharing, as shown in Fig. 5. Note that when $Q = 2$, (x=0) is simply $1 - x \mod 2$. Otherwise, when Q is an odd prime, the correctness of $\mathsf{ZeroTest}_Q$ is proved by the following theorem.

Theorem 3 (Correctness). *Let Q be an odd prime, and let (x^A, x^B) be a pair of shares of x in $\mathbb{Z}_Q$. Then, $(z^A, z^B) \leftarrow \mathsf{ZeroTest}_Q(A(x^A), B(x^B), k)$ forms a pair of shares of (x=0) with an error rate less than $(\frac{1}{2})^k$.*

Proof. In Step 1 of the protocol, let $x_i = x_i^A + x_i^B \mod Q$ and $w_i = w_i^A + w_i^B \mod Q$.

1. If $x = 0$, then for all $r_i \in \mathbb{Z}_Q^*$, $x_i = r_i x \mod Q = 0 < Q/2$; therefore, $w^A + w^B \mod Q = w'^A + w'^B \mod 2 = 1$.
2. If $x \neq 0$, since r_i is a uniformly random number in $\mathbb{Z}_Q^*$, so is $x_i = r_i x \mod Q$. Moreover, because s_i is a uniformly random number in $\mathbb{Z}_Q$, by Lemma 4, $Pr[w_i = 1] > \frac{1}{2}$.
3. Hence, if $x \neq 0$, $Pr[z^A + z^B \mod Q = 1] = Pr[w'^A + w'^B \mod 2 = 1] = Pr[w_i - 0, \forall i = 1, ..., k] < \left(\frac{1}{2}\right)^k$. □

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$, such that $x = x^A + x^B \mod Q$. Q is a public prime.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B \mod Q = 1$ if $x = 0$, otherwise 0.

When $Q = 2$: output $z^A \leftarrow 1 - x^A$ and $z^B \leftarrow x^B$,
else (when $Q > 2$):

1. Alice and Bob run the following procedure for $i = 1, ..., k$ in parallel:
   (a) Generate a public random number s_i (uniformly and independently).
   (b) Generate a public non-zero random number r_i (uniformly and independently).
   (c) Alice and Bob locally compute $x_i^A \leftarrow r_i x^A + s_i \mod Q$ and $x_i^B \leftarrow r_i x^B - s_i \mod Q$ respectively.
   (d) Run $(w_i^A, w_i^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}_{Q\to 2}^{\top}(A(x_i^A), B(x_i^B))$.
2. Run $(w'^A, w'^B) \leftarrow \mathsf{AND}_2^k(A(1 - w_1^A, ..., 1 - w_k^A), B(-w_1^A, ..., -w_k^A))$.
3. Run $(z^A, z^B) \leftarrow \mathsf{ModCNV}_{2\to Q}(A(w'^A), B(w'^B))$.

Fig. 5. Zero test protocol: $\mathsf{ZeroTest}_Q(A(x^A), B(x^B), k)$

Security and Complexity. The communication part of the protocol involves generating k public $s_i \in_R \mathbb{Z}_Q$ and k public $r_i \in_R \mathbb{Z}_Q^*$, k $\frac{1}{2}\mathsf{ModCNV}_{Q\to 2}^{\top}$, 1 AND_2^k, and 1 $\mathsf{ModCNV}_{2\to Q}$. Since no public s_i or r_i contain any information about the secrets, the security of the protocol follows from that of the sub-protocols. Besides, since the cost of $\mathsf{M2A}_2$ is not more than that of $\mathsf{M2A}_Q$, and a MULT_Q involves two $\mathsf{M2A}_Q$'s, for simplicity, we estimate the cost by the number of MULT_Q. The total cost is approximately k MULT_Q's and 8 rounds.

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$ such that $x = x^A + x^B \mod Q$, where Q is a public odd number.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B \mod Q = 1$ if $x \leq \lfloor \frac{Q}{2} \rfloor$, otherwise 0.

1. Alice and Bob locally compute $x'^A = 2x^A \mod Q$ and $x'^B = 2x^B \mod Q$ respectively.
2. Run $(y^A, y^B) \leftarrow \mathsf{ModCNV}_{Q\to 2}(A(x'^A), B(x'^B), k)$.
3. Run $(z^A, z^B) \leftarrow \mathsf{ModCNV}_{2\to Q}(A(1-y^A), B(y^B))$.

Fig. 6. Sign test protocol: $\mathsf{SIGN}_Q(A(x^A), B(x^B))$

3.6 Sign Test.

Let $1, ..., \lfloor Q/2 \rfloor$ correspond to positive numbers, and let $\lceil (Q+1)/2 \rceil, ..., Q-1$ correspond to negative numbers $(-\lceil (Q+1)/2 \rceil, ..., -1)$. For all x in $\mathbb{Z}_Q$, we compute the sign of x through general modular conversion. Here, we consider the case where Q is an odd number. An extension for an arbitrary Q is discussed in Appendix A.

A simple implementation of the protocol is described in Fig. 6. The protocol's validity follows that of $\mathsf{ModCNV}_{Q\to 2}$ and $\mathsf{ModCNV}_{2\to Q}$ and the fact that, for all $x < Q/2$, $2x$ $(0 \leq 2x < Q)$ is even; and for all $x \geq Q/2$, $2x - Q$ $(0 \leq 2x - Q < Q)$ is odd.

Security and Complexity. Since SIGN_Q only involves $\mathsf{ModCNV}_{Q\to 2}$ and $\mathsf{ModCNV}_{2\to Q}$, the security follows from those sub-protocols. Hence, by Lemma 3, the output of SIGN_Q is valid with a probability of at least $1 - (7/10)^k$; and the cost is about $7k + 30$ MULT_2's and 1 $\mathsf{M2A}_Q$ and 9 rounds.

3.7 Applications

In this section, we consider several applications of the modular conversion protocols and the zero test protocol. Using our implementation of SIGN_Q, the complexity of the applications is at most $O(k)$ MULT_Q's and $O(1)$ rounds, with an error rate of $2^{-\Omega(k)}$ at most.

Integer Comparison (CMP_Q). Integer comparison accesses whether $(x \leq y)$ or $(x \geq y)$, etc. When the input secrets x and y are known to be in the interval $[0, Q/2]$, the computation is just $Sign_Q(A(x^A - y^A \mod Q), B(x^B - y^B \mod Q))$. Otherwise, if x and y are known to be in Z_Q, similar to the derivation in [26], CMP_Q can be reduced to a cubic combination of $Sign_Q(A(x^A), B(x^B))$, $Sign_Q(A(y^A), B(y^B))$ and $Sign_Q(A(x^A - y^A \mod Q), B(x^B - y^B \mod Q))$ by 2 rounds and 3 MULT_Q's [3]. Hence, the whole computation mainly involves 3 invocations of SIGN_Q.

Exponentiation (EXP_Q) Yu et al. proposed [31], an efficient method for conducting secure two-party modular exponentiation in an arithmetic way. In the scheme, the secret base is first converted into multiplicative shares over $\mathbb{Z}_Q^*$ and the secret exponent is converted into additive shares over $\mathbb{Z}_{Q-1}$. Then, the computation can be expressed as $(x + (x{=}0))^{(y \mod Q-1)} - (x{=}0) \mod Q$. In the authors' implementation, the scheme can be computed efficiently except zero test $(x{=}0)$ and modular conversion (converting the shares of y from Q to $Q-1$) components. The cost of the computation is estimated to be 5 rounds and 12 $\mathsf{M2A}_Q$'s, or equivalently about 6 MULT_Q's. Hence, our zero test and general modular conversion solutions can complement the implementation in [31].

[3] Let $\alpha = (x{\leq}Q/2)$, $\beta = (y{\leq}Q/2)$ and $\gamma = (x - y{\leq}Q/2)$. We have $(x{\leq}y) = \alpha\overline{\beta} + \overline{\alpha}\overline{\beta}\overline{\gamma} + \alpha\beta\overline{\gamma} = \alpha(\beta + \gamma - 2\beta\gamma) + 1 - \beta - \gamma + \beta\gamma$

MOD Operation ($\mathsf{MOD}_{Q,P}$) Here, MOD operation is defined as the remainder of an integer division of a secret x, shared over $\mathbb{Z}_Q$, with a public divisor P. In [9, 24], this function is called modulo reduction, which. This function is related to the modular conversion. With input $x^A, x^B \in \mathbb{Z}_Q$, $\mathsf{MOD}_{Q,P}$ is expected to output $z^A, z^B \in \mathbb{Z}_Q$ such that $z^A + z^B \mod Q = x \mod P$, where $x = x^A + x^B \mod Q$. That is, the input and the output are both shares in $\mathbb{Z}_Q$. However, it can be reduced to two invocations of the general modular conversion protocol, $\mathsf{ModCNV}_{Q\to P}$ and $\mathsf{ModCNV}_{P\to Q}$.

4 Active Security

In this section, we explained how the passively secure protocols discussed in Section 3 can be made actively secure by efficient arithmetic-based proofs. Under the active security model, Alice and Bob run a protocol to prove that they are being honest to each other. Specifically, for each message and each output of the protocol, as well as the inputs of the sub-protocol, they should prove that the respective values are generated properly according to the protocol.

Notation. Let $ZKP(P(\text{pre-information}),V(\text{pre-information}): \text{statement})$ denotes the zero-knowledge proof of the statement, where $P(\text{pre-information})$ and $V(\text{pre-information})$ denote the information known by the prover and the verifier respectively.

First, we choose an integer commitment scheme with additive homomorphism, e.g., [12, 10, 15], and use the following basic proofs as building blocks.

A1. $ZKP(P(x, r_1, r_2), V(F_1, F_2) : F_1 = C(x, r_1), F_2 = C(x, r_2))$
A2. $ZKP(P(x_1, ..., x_k, a_0, ..., a_k, r_1, ..., r_k, s), V(F_1, ..., F_k, G, a_0, ..., a_k) :$
$F_1 = C(x_1, r_1), ..., F_k = C(x_k, r_k), G = C(a_0 + \sum_{i=1}^{k} a_i x_i, s))$
A3. $ZKP(P(x, y, r_1, r_2, r_3), V(F_1, F2, F_3) : F_1 = C(x, r_1), F2 = C(y, r_2), F_3 = E(xy, r_3))$

A1 proves that two commitments hide the same secret x. A2 proves the relationship of an arbitrary linear combination. Note that when the commitment scheme is additively homomorphic, both the prover and the verifier can compute $G' = C(a_0 + \sum_{i=1}^{k} a_i x_i, s')$ from $F_1 = C(x_1, r_1), ..., F_k = C(x_k, r_k)$ locally. Hence, A2 can be reduced to A1 by showing that G' and G hide the same secret. A3 shows the multiplication relationship. Concrete schemes of A1 and A3 can be found in e.g., [15, 5, 7, 10]. In the schemes, each of the above proofs requires $O(1)$ rounds, $O(1)$ exponentiation, and $O(\ell + \kappa)$ communication bits, where ℓ is the bit-length of the secret, and κ is a security parameter defined by the schemes.

Second, besides the above basic proofs, we require the proofs of three specific relationships: $x \geq 0$, $x \in \{a, a+1, ..., b\}$, and $z = x \mod p$. The second and third proofs are based on the first one, for which concrete schemes can be found in [5, 22]. We provide an alternative, more lightweight protocol, together with the protocols of the second and third proofs, as shown in Fig. 7. Each of these protocols can be reduced to $O(1)$ invocations of the basic proofs (A2 and A3). Since the proofs of B2 and B3 in Figure 7 are straightforward, we only prove the validity of B1. The proof is based on Gauss's lemma for triangular numbers.

Lemma 5. *(Gauss, 1797, See [14] for reference) Every positive number can be represented as the sum of at most three triangular numbers.*

Theorem 4. *If A2 and A3 are valid, B1 is also valid.*

B1. $ZKP(P(x,r), V(F = C(x,r)) : x \geq 0)$

1. The prover decomposes x into three triangular numbers, $\triangle_1, \triangle_2, \triangle_3$, where $\triangle_i = 2^{-1}(a_i^2 + a_i)$, and $a_i \in \mathbb{Z}$ for $i = 1, 2, 3$.
2. The prover sends the commitments of $\triangle_1, \triangle_2, \triangle_3$ to the verifier, and proves the linear combination of $x = \triangle_1 + \triangle_2 + \triangle_3$ using A2.
3. For $i = 1$ to 3, the prover sends the commitments of a_i to the verifier and proves $\triangle_i = 2^{-1}(a_i^2 + a_i)$ using A2 and A3.

B2. $ZKP(P(x,r), V(F = C(x,r)) : x \in \{a, a+1, ..., b\})$

1. The prover proves $x - a \geq 0$ and $b - x \geq 0$ using two invocations of B1.

B3. $ZKP(P(x, r_1, z, r_2, p), V(F_1 = C(x, r_1), F_2 = C(z, r_2), p) :, z = x \mod p)$

1. The prover proves $0 \leq z \leq p - 1$ using B2.
2. Let $x = z + pd$. The prover sends the commitment of d to the verifier, and proves $x = z + pd$ using A2 and A3.

Fig. 7. Zero-Knowledge Proofs

Proof. By Lemma 5, every integer can be decomposed into at most three triangular numbers. Note that, for all $a \in \mathbb{Z}, a^2 + a \geq 0$ and $2|a^2 + a$. Hence, for all $i = 1$ to 3, the proof of $\triangle_i = \frac{a_i^2 + a_i}{2}$ (by A2 and A3) implies that $\triangle_i$ is either zero or a positive integer. Consequently, we also have $\triangle_1 + \triangle_2 + \triangle_3 \geq 0$. In addition, the proof of $x = \triangle_1 + \triangle_2 + \triangle_3$ (by A2) implies that $x \geq 0$. Therefore, when A2 and A3 are valid, B1 is also valid. □

Furthermore, Rabin and Shallit [27] proposed a randomized algorithm that decomposes x into at most three triangular numbers. When an empirically reasonable conjecture, which we call the Rabin-Shallit conjecture[4], is true, the expected running time of their algorithm is only $O(\log^2 x)$. Specifically, when x is decomposed into less than three triangular numbers, in B2, the prover needs to set the remaining $a_i = \triangle_i = 0$.

Using the above zero-knowledge proofs, we can "compile" our passively secure protocols in Section 3 as actively secure ones. Here, we assume that the underlying $\mathsf{M2A}_Q(A(x^A), B(x^B))$ is an actively secure protocol, and its inputs and outputs are accompanied by commitments. Readers may refer to [23, 19, 13] for the details of this elementary protocol. In the following, we enumerate the required zero-knowledge proofs of the protocols discussed in Section 3.

- $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q \to P}$ (Fig. 2): Alice proves $b^A = (x \leq \left\lfloor \frac{Q}{2} \right\rfloor)$ and $z^A = (x^A + z'^A \cdot Q) \mod P$. The first part of the statement can be rewritten as $x = b^A \cdot \lfloor Q/2 \rfloor + a$ with $b^A \in \{0, 1\}$. Hence, to prove this statement, Alice first sends a commitment $C(b^A, r_b)$ to Bob. Then Alice proves the veracity of $z^A = x^A + z'^A \cdot Q$ by one A2 and the veracity of $(b^A)^2 = b^A$ (which implies $b^A \in \{0, 1\}$) by one A3 and one A1. Finally, since a commitment of $x^A + z'^A \cdot Q$ can be computed instantly from the commitment of x^A and z'^A, the second part of the statement can be proved by one B3. Bob's proofs are similar.
- $\frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q \to P}$ (Fig. 2): This is similar to the case of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q \to P}$.
- $\mathsf{ModCNV}_{2 \to Q}$ (Fig. 2): In Step 2, Alice and Bob prove $z^A = (x^A - 2 \cdot z'^A) \mod Q$ and $z^B = (x^B - 2 \cdot z'^B) \mod Q$ respectively using B3.
- AND^k_Q (Fig. 3): In Step 2, Alice and Bob can compute the commitment of y^A and y^B locally. The other parts of this protocol are only comprised of the sub-protocols.

[4] Every number of the form $8t + 3, t \geq 1$, can be expressed as the sum of a square and twice a prime.[27]

$\mathsf{ModCNV}_{Q\to P}$ (Fig. 4): In Step 1, for $i = 1, ..., \sigma - 1$, Alice sends a commitment of α_i to Bob and proves the veracity of $\alpha_i = (x_i^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor \geq Q)$ using a similar way of proving the veracity of $b^A = (x \leq \left\lfloor \frac{Q}{2} \right\rfloor)$ in $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ by one A1, one A2 and one A3. In addition, Alice proves the veracity of $x_i^A = x^A + \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor - \alpha_i Q$ by one A2. In Step 2(a), we use a standard method to generate an unbiased public random number. That is, for each j, Alice and Bob generate uniformly random r_j^A and r_j^B respectively and commit them to each other. Then, they decommit these two values to compute $r_j = r_j^A + r_j^B \mod Q$. Because of the binding property of the commitment scheme and since at least one party is honest, we obtain an unbiased value r_j. Then, in Step 2(b), for $i = 1, ..., \sigma - 1, j = 1, ...k$, Alice and Bob run B3 σk times to prove that $x_{i,j}^A = x_i^A + r_j \mod Q$ and $x_{i,j}^B = x_i^B - r_j \mod Q$ respectively. In Step 7, for $i = 1, ..., \sigma - 1$, Alice proves the veracity of $w_i'^A = w_i^A - \left\lfloor i \cdot \frac{Q}{\sigma} \right\rfloor + \alpha_i Q$ by one A2. Since σ is a constant less than or equal to 5, these proofs involve $O(k)$ basic proofs.

$\mathsf{ZeroTest}_Q$ (Fig. 5): When $Q = 2$, Alice and Bob prove $z^A = 1 + s^A - (x^A)^2$ and $z^B = s^B - (x^B)^2$ respectively by using A2 and A3. When $Q > 2$, in Step 1(a), Alice and Bob generate public random numbers s_i using the same method as that in $\mathsf{ModCNV}_{Q\to P}$. In Step 1(b), to generate a public random number in $\mathbb{Z}_Q^*$, Alice and Bob first sample uniform random numbers r_i^A and r_i^B in $\mathbb{Z}_Q^*$, and commit them to each other. When they decommit the random numbers, since at least one party is honest, at least one of r_i^A and r_i^B is uniformly sampled from $\mathbb{Z}_Q^*$. Hence, if both r_i^A and r_i^B are non-zero, Alice and Bob can compute a uniform random $r_i = r_i^A \cdot r_i^B \mod Q$. If $r_i^A = 0$, which reveals Alice's malice, Bob can stop the protocol, or they can prescribe that the protocol uses $r_i = r_i^B$ in this case and continue. Then, in Step 1(c), for $i = 1, ..., k$, Alice and Bob use B3 to prove that $x_i^A = r_i x^A + s_i \mod Q$ and $x_i^B = r_i x^B - s_i \mod Q$ respectively.

SIGN_Q (Fig. 6): In Step 1, Alice and Bob run B3 to prove that $x'^A = 2x^A \mod Q$ and $x'^B = 2x^B \mod Q$ respectively.

Each of the passive protocols in Section 3 involves at most $O(k)$ statements of A2, A3, B2 and B3. Hence, we can conclude that since each of thes protocols involves $O(k)$ basic proofs, the cost of their proofs is at most $O(1)$ rounds, $O(k)$ exponentiations and $O(k(\ell + \kappa))$ communication bits.

Acknowledgement

The authors gratefully thank Kai-Min Chung and Feng-Hao Liu for their helpful discussion and promotion of this study.

References

1. J. Algesheimer, J. Camenisch, and V. Shoup. Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In *Proc. 22nd CRYPTO*, pages 417–432, 2002.
2. J. Bar-Ilan and D. Beaver. Non-cryptographic fault-tolerant computing in a constant number of rounds of interaction. In *Proc. 8th PODC*, pages 201–209, 1989.
3. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. 20th STOC*, pages 1–10, 1988.
4. P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft. Secure multiparty computation goes live. In *Proc. 13th Financial Cryptography and Data Security*, pages 325–343, 2009.
5. F. Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT'00*, pages 431–444, 2000.

6. P. Bunn and R. Ostrovsky. Secure two-party k-means clustering. In *Proc. 14th CCS*, pages 486–497, 2007.
7. J. Camenisch and M. Michels. A group signature scheme based on an RSA-variant. In *Tech. Rep. RS-98-27, BRICS*, 1998.
8. D. Chaum, C. Crepéau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proc. 20th STOC*, pages 11 – 19, 1988.
9. I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft. Unconditionally secure constant-Rounds multi-party computation for equality, comparison, bits and exponentiation. In *Proc. 3rd TCC*, pages 285–304, 2006.
10. I. Damgård and E. Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, pages 125–142, 2002.
11. I. Damgård and G. L. Mikkelsen. Efficient robust and constant-round distributed RSA key generation. In *Proc. 7th TCC*, pages 183–200, 2010.
12. I. Damgård and J. B. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor iacr. In *Proc. 22nd Crypto*, pages 581–596, 2002.
13. I. Damgård and C. Orlandi. Multiparty computation for dishonest majority: from passive to active security at low cost. In *Proc. 30th CRYPTO*, pages 558–576, 2010.
14. W. Duke. Some Old Problems and New Results about Quadratic Forms. *Notices of the American Mathematical Society*, 44(2):190 – 196, 1997.
15. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Proc. 17th Crypto*, pages 16–30, 1997.
16. O. Goldreich. *Foundations of Cryptography - Volume II, Basic Applications*. Cambridge University Press, 2004.
17. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game - A completeness theorem for protocols with honest majority. In *Proc. 19th STOC*, pages 218–229, 1987.
18. J. Guajardo, B. Mennink, and B. Schoenmakers. Modulo reduction for paillier encryptions and application to secure statistical analysis (extended abstract). In *Proc. 14th Financial Cryptography and Data Security*, volume 6052, pages 375–382, 2010.
19. Y. Ishai, M. Prabhakaran, and A. Sahai. Secure Arithmetic Computation with No Honest Majority. In *Proc. 6th TCC*, pages 294–314, 2009.
20. Y. Lindell and B. Pinkas. Privacy-preserving data mining. *Journal of the Cryptology*, 15(3):177–206, 2002.
21. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. *Journal of the ACM*, 1(1):59–98, 2009.
22. H. Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In *Asiacrypt'03*, pages 398–415, 2003.
23. P. Mohassel and E. Weinreb. Efficient secure linear algebra in the presence of covert or computationally unbounded adversaries. In *Proc. 28th CRYPTO*, volume 5157, pages 481–496, 2008.
24. C. Ning and Q. Xu. Multiparty computation for modulo reduction without bit-decomposition and a generalization to bit-decomposition. In *Asiacrypt'10*, volume 6477, page 487, 2010.
25. C. Ning and Q. Xu. Constant-rounds, linear multi-party computation for exponentiation and modulo reduction with perfect security. In *Asiacrypt'11*, 2011.
26. T. Nishide and K. Ohta. Multiparty computation for interval, equality, and comparison without bit-decomposition protocol. In *Proc. 10th PKC*, pages 343–360, 2007.
27. J. O. Rabin and J. Shallit. Randomized algorithms in number theory. *Communications on Pure and Applied Mathematics*, 39(S1):239 – 256, 1986.
28. T. Toft. Constant-rounds, almost-linear bit-decomposition of secret shared values. In *CT-RSA '09*, pages 357–371, 2009.
29. T. Toft. Sub-linear, secure comparison with two non-colluding parties. In *Proc. 14th PKC*, volume 6571, pages 174–191, 2011.
30. A. C.-C. Yao. How to generate and exchange secrets. In *Proc. 27th FOCS*, pages 162–167, 1986.
31. C.-H. Yu, S. S. M. Chow, K.-M. Chung, and F.-H. Liu. Efficient secure two-party exponentiation. In *CT-RSA '11*, pages 17–32, 2011.

A Protocols for An Arbitrary Modulus

We explained how to compute modular conversion, zero test and sign test when Q is a prime in Section 3. As mentioned in Sections 3.5, 3.6 and 3.7, there are constraints on Q in the $\mathsf{ModCNV}_{Q\to P}$ (Fig. 4), $\mathsf{ZeroTest}_Q$ (Fig. 5) and SIGN_Q (Fig. 6) protocols; however, with some modifications, the constraints can be removed without increasing the complexity. In the following, we discuss the extension of protocols for the case of an arbitrary $Q \geq 2$.

Modular Conversion. In Fig. 4, Q should be odd. When Q is even, the testing of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ in Step 2(c) will always return a valid result irrespective of whether $x < Q/2$, so the strategy of probabilistic test will fail. Hence, for an arbitrary $Q \geq 2$, we make the following modification.

1. We replace $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ in Step 2(c) with $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P'}$, where P' is a small prime and $P' \nmid Q$.
2. Now each $y_{i,j}$ can only be $x_i \mod P'$ or $x_i + Q \mod P'$. The latter case only occurs when $x_i \geq Q/2$. We can map this distinction back to 0 and 1 in the original scenario in Fig. 4 by computing $t_{i,j} = (Q^{-2}) \cdot (y_{i,j} - y_{i,j+1})^2 \mod P'$ privately with one $\mathsf{MULT}_{P'}$. Since P' is a prime and $P' \nmid Q$, $Q^{-2} \mod P'$ exists and is a non-zero number. As a result, $t_{i,j}$ can only be 0 or 1, but the latter case only occurs when $x_i \geq Q/2$.
3. Then, similar to Step 3 in Fig. 4, we compute $\mathsf{AND}^{k-1}_{P'}$ for all $t_{i,j}$ and for all $1-t_{i,j}$. Moreover, if we choose P' as the smallest prime $> k$ and coprime to Q, the first and last steps of $\mathsf{AND}^{k-1}_{P'}$ in Fig. 3 can be omitted.
4. The remaining parts of the procedure are the same as in Fig. 4.

Regarding the cost, although we use $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P'}$ k times, the first and last steps of $\mathsf{AND}^{k-1}_{P'}$ in Fig. 3 are not executed in the modified scheme. Hence, the additional cost is about k $\mathsf{MULT}_{P'}$. The steps are described in detail in Fig. 8.

General modular conversion Protocol (for a general Q):

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$ such that $x = x^A + x^B \mod Q$. $Q > 2$ and $P \geq 2$ are two public numbers.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B = x \mod P$.

If $P|Q$, Alice and Bob output $z^A \leftarrow x^A \mod P$ and $z^B \leftarrow x^B \mod P$ respectively;
else, let σ be a parameter for the optimum adjustment and k be a parameter for the error rate. Besides, let P' be the smallest prime $> k$ and $c = Q^{-2} \mod P'$.

1. For $i = 0, ..., \sigma - 1$, Alice sets $\alpha_i = (x_i^A + \lfloor i \cdot \frac{Q}{\sigma} \rfloor \geq Q)$ and $x_i^A = x^A + \lfloor i \cdot \frac{Q}{\sigma} \rfloor - \alpha_i Q$, and Bob sets $x_i^B = x^B$.
2. Alice and Bob run the following procedure for $j = 1, ..., k$ in parallel:
   (a) Generate a public random number r_j (uniformly and independently).
   (b) For each i, Alice and Bob locally compute $x_{i,j}^A \leftarrow x_i^A + r_j \mod Q$ and $x_{i,j}^B \leftarrow x_i^B - r_j \mod Q$ respectively.
   (c) For each i, run $(y_{i,j}^A, y_{i,j}^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P'}(A(x_{i,j}^A), B(x_{i,j}^B))$.
3. For each i and $j = 1, ..., k-1$, compute $(t_{i,j}^A, t_{i,j}^B) \leftarrow \mathsf{MULT}_{P'}(A(c \cdot (y_{i,j}^A - y_{i,j+1}^A), y_{i,j}^A - y_{i,j+1}^A), B(c \cdot (y_{i,j}^B - y_{i,j+1}^B), y_{i,j}^B - y_{i,j+1}^B)$.
4. For each i, compute $(u_i^A, u_i^B) \leftarrow \mathsf{AND}^{k-1}_{P'}(A(t_{i,1}^A, ..., t_{i,k-1}^A)), B(t_{i,1}^B, ..., t_{i,k-1}^B))$ and $(v_i^A, v_i^B) \leftarrow \mathsf{AND}^{k-1}_{P'}(A(1 - t_{i,1}^A, ..., 1 - t_{i,k-1}^A)), B(-t_{i,1}^B, ..., -t_{i,k-1}^B))$.
5. For each i, compute $(a_i^A, a_i^B) \leftarrow \mathsf{MULT}_{P'}(A(1 - u_i^A, 1 - v_i^A), B(-u_i^B, -v_i^B))$.
6. For each i, run $(b_i^A, b_i^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{P'\to P}(a_i^A, a_i^B)$.
7. For each i, run $(w_i'^A, w_i'^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\perp}_{Q\to P}(A(x_i^A), B(x_i^B))$.
8. For each i, Alice sets $w_i^A = w_i'^A - \lfloor i \cdot \frac{Q}{\sigma} \rfloor + \alpha_i Q \mod P$, and Bob sets $w^B = w'^B$.
9. Compute z^A and z^B such that $z^A + z^B = b_0 w_0 + \bar{b}_0 b_1 w_1 + ... + (\bar{b}_0 ... \bar{b}_{\sigma-2} b_{\sigma-1} w_{\sigma-1}) \mod P$ using $(1+\sigma)\sigma/2$ MULT_P (where $\bar{b}_i = 1 - b_i$, $b_i = b_i^A + b_i^B \mod P$ and $w_i = w_i^A + w_i^B \mod P$).

Fig. 8. Protocol $\mathsf{ModCNV}_{Q\to P}(A(x^A), B(x^B), k)$ for a general Q

Zero Test. In Fig. 5, Q should be a prime. When Q is a composite number, two issues arise: 1) $rx \mod Q$ can be zero, even if $x \neq 0$; and 2)for an even Q, since $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to 2}$ is always valid, the strategy of probabilistic test fails. However, these issues are not difficult to solve. Let P be the smallest prime $> 2Q$. For an arbitrary composite Q, we run $(y^A, y^B) \leftarrow \mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$ first. Then, we replace (x^A, x^B) with (y^A, y^B) and Q with P in Fig. 5, but we still run $\mathsf{ModCNV}_{2\to Q}$ in the last step. Note that when $x = 0$, $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ is always valid and outputs a shared 0. When $x \neq 0$, $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$ can output a shared x or a shared $x + Q \mod P$, which are both non-zero. Hence, this strategy retains the validity of the zero test. If, for simplicity, we leave out the observation that the size of P is 1 bit longer than Q, the additional cost is only one $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}$. The steps are described in detail in Fig. 9.

Sign Test. In Fig. 5, Q should be odd. When Q is even, because $x' = 2x \mod Q$ is always even irrespective of whether $x < \lfloor Q/2 \rfloor$, and y is always 0. Hence the strategy of probabilistic test fails. To resolve the issue, when Q is even, we run $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to Q+1}$ followed by the sign test in $\mathbb{Z}_{Q+1}$. By the definition of positive and negative numbers in Section 3.6, the positive and negative numbers in Z_Q correspond to those in Z_{Q+1}. However, when $x \geq Q/2$, the result of $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to Q+1}$ can be wrong and the output is $x - 1$. The result of the sign test can only be wrong when $x = \lfloor Q/2 \rfloor + 1$. Hence, we perform an additional zero test of $x - \lfloor Q/2 \rfloor - 1$ to exclude this case. The additional cost is one $\frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to Q+1}$ and one $\mathsf{ZeroTest}_Q$. The steps are described in detail in Fig. 10.

Zero Test Protocol (for a general Q):

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$ such taht $x = x^A + x^B \mod Q$. Q is a public number ≥ 2.
- **Outputs:** Alice obtains z^A and Bob obtains z^B such that $z^A + z^B \mod Q = 1$ if $x = 0$, otherwise 0.

When $Q = 2$: output $z^A \leftarrow 1 - x^A$ and $z^B \leftarrow x^B$,
else (when $Q > 2$), let P be the smallest prime $> 2Q$:

1. Run $(y^A, y^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{Q\to P}(A(x^A), B(x^B))$.
2. Alice and Bob run the following procedure for $i = 1, ..., k$ in parallel:
   (a) Generate a public random number s_i.
   (b) Generate a public non-zero random number r_i.
   (c) Locally Alice and Bob compute $x_i^A \leftarrow r_i y^A + s_i \mod P$ and $x_i^B \leftarrow r_i y^B - s_i \mod P$ respectively.
   (d) Run $(w_i^A, w_i^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{P\to 2}(A(x_i^A), B(x_i^B))$.
3. Run $(w'^A, w'^B) \leftarrow \mathsf{AND}_2^k(A(1 - w_1^A, ..., 1 - w_k^A), B(-w_1^A, ..., -w_k^A))$.
4. Run $(z^A, z^B) \leftarrow \mathsf{ModCNV}_{2\to Q}(A(w'^A), B(w'^B))$.

Fig. 9. Protocol $\mathsf{ZeroTest}_Q(A(x^A), B(x^B), k)$ for a general Q

B Comparison of the Protocols' Complexities

In Table B, we summarize our results of two-party omputation of general modular convrsion, zero test, integer comparison, modular exponentiation and modulo reduction, and compare the complexity with existing solutions. All of these solutions are constructed based on a secure $\mathsf{MULT}/\mathsf{M2A}$. Hence the multiparty protocols in [26, 25, 24] can also be applied to the two-party problems here.

Sign Test Protocol (for an even Q):

- **Inputs:** Alice holds $x^A \in \mathbb{Z}_Q$ and Bob holds $x^B \in \mathbb{Z}_Q$ such that $x = x^A + x^B \mod Q$, where Q is a public even number.
- **Outputs:** Alice obtains z^A, and Bob obtains z^B such that $z^A + z^B \mod Q = 1$ if $x \leq \lfloor \frac{Q}{2} \rfloor$, otherwise 0.

1. Run $(s^A, s^B) \leftarrow \mathsf{ZeroTest}_Q(A(x^A - \frac{Q}{2} - 1), B(x^B))$.
2. Run $(u^A, u^B) \leftarrow \frac{1}{2}\mathsf{ModCNV}^{\top}_{Q \to Q+1}(A(x^A), B(x^B))$.
3. Alice and Bob locally compute $x'^A = 2u^A \mod (Q+1)$ and $x'^B = 2u^B \mod (Q+1)$ respectively.
4. Run $(y^A, y^B) \leftarrow \mathsf{ModCNV}_{Q+1 \to 2}(A(x'^A), B(x'^B), k)$.
5. Run $(v^A, v^B) \leftarrow \mathsf{ModCNV}_{2 \to Q}(A(1 - y^A), B(y^B))$.
6. Run $(w^A, w^B) \leftarrow \mathsf{MULT}_Q(A(1 - s^A, v^A), B(-s^B, v^B))$.
7. Alice and Bob locally compute $z^A = s^A + w^A \mod Q$ and $z^B = s^B + w^B \mod Q$ respectively.

Fig. 10. Protocol $\mathsf{SIGN}'_Q(A(x^A), B(x^B))$ for an even Q

Problem	Solution	Rounds	Complexity (MULT_Q)	Error Rate
General Modular Conversion	Section 3.4	8	7k+30 *	$(7/10)^k$
Zero Test	[26]	8	81ℓ	deterministic
	[26]	4	$12k$	$(1/2)^k$
	Section 3.5	8	k	$(1/2)^k$
Integer Comparison	[26]	15	$297\ell + 5$	deterministic
	[29]	O(1)	$O(\sqrt{\ell}(k + \log \ell))$	$(1/2)^k$
	Section 3.7	11	$21k + 96$ **	$(7/10)^k$
Modular Exponentiation	[25]	24	$162\ell + 46\sqrt{\ell} + 28$	deterministic
	Section 3.7, based on [31]	13	$8k + 36$	$(7/10)^k$
MOD (modulo reduction)	[24]	22	$354\ell + 3$	deterministic
	Section 3.7	8	$7k + 30$	$(7/10)^k$

Table 1. Comparison of our results and those of the protocols in [26, 25, 24, 31, 29] for computing additive shares over $\mathbb{Z}_Q$. ($\ell = \log Q$ is the length of the elements in the field $\mathbb{Z}_Q$; k is the correctness parameter for exponentially low error rates.)

* The cost of $\mathsf{ModCNV}_{Q \to P}$ is about $7k + 30$ MULT_P's.

** The cost of CMP_Q is about $21k + 90$ MULT_2's and 6 MULT_Q's, upper-bounded by the complexity of $21k + 96$ MULT_Q's.