Efficient Additive Randomized Encodings for String Oblivious Transfer: A Core Primitive for General Functions

Masaya Yoshimura[1], Kyoichi Asano[2⋆], Yugo Kasashima[1], Mitsugu Iwamoto[1], and Yohei Watanabe[1,2]

[1] The University of Electro-Communications, 1–5–1 Chofugaoka, Chofu, Tokyo, 182–8585, Japan
{yoshimura, kasashima, mitsugu, watanabe}@uec.ac.jp
[2] National Institute of Advanced Industrial Science and Technology, 2–3–26 Aomi, Koto-ku, Tokyo, 135–0064, Japan
kyoichi.asano@aist.go.jp

Abstract. Additive Randomized Encodings (AREs) provide a lightweight route to non-interactive secure computation: each party locally produces a randomized encoding of its input, and an evaluator learns only the function value from the sum of these encodings. Prior frameworks for constructing AREs for general functions ultimately hinge on an efficient ARE for string oblivious transfer (SOT), making SOT the core efficiency bottleneck. We revisit this final step and present improved ARE constructions for SOT. First, we streamline the recent PKE-based approach by directly constructing the required one-sided ARE (OSARE) for SOT. This yields a perfectly correct, statistically one-sided secure OSARE for SOT of size $\mathcal{O}(\lambda)$, improving the intermediate overhead and, consequently, the overall size of the resulting PKE-based ARE. Second, we give a pairing-free ARE for SOT under the Squaring DDH assumption. Our construction is compatible with Halevi et al.'s efficient equality-to-SOT methodology: we build an ARE for Rabin-OT that embeds a message in the equality-checking procedure and then transform it to SOT while incurring only constant-factor communication overhead and no additional correctness error. The resulting ARE for SOT has size $\mathcal{O}(\lambda)$ and negligible correctness error.

1 Introduction

More than four decades have passed since Yao [20] introduced the notion of secure multi-party computation (MPC), which has become a central area of research in cryptography. *Non-interactive MPC* [4, 13] enables n parties to collaboratively compute a function f over their joint inputs $(x_1, x_2, \ldots, x_n)$ by independently sending a single online message each to a central server, referred to as the evaluator. Upon receiving all messages, the evaluator computes the output $f(x_1, x_2, \ldots, x_n)$ while learning nothing beyond the output itself. Although non-interactive MPC typically requires a trusted setup, such as correlated randomness or a public-key infrastructure, the shuffle model, recently studied in the contexts of differential privacy (DP) [9] as well as private information retrieval and secure aggregation [1, 11, 12, 15, 16], enables the realization of non-interactive MPC without a trusted setup. In this model, the evaluator receives all messages in a randomly permuted order; that is, the messages are anonymized.

In this paper, we focus on *additive randomized encodings* (AREs) [14], which are closely related to non-interactive MPC in the shuffle model [16]. An ARE for an n-input function f is defined over an Abelian group $\mathbb{G}$ and enables each party i to locally compute a randomized encoding $\hat{x}_i$ of its input x_i. A little bit unlike in the shuffle model, the evaluator receives only the sum encoding $\hat{y} \coloneqq \hat{x}_1 + \cdots + \hat{x}_n$, from which it can decode the value $f(x_1, \ldots, x_n)$. Crucially, the sum encoding $\hat{y}$ reveals no information beyond the output itself.

Prior Frameworks for Constructing AREs for General Functions. All the existing construction frameworks [6, 7, 14] for AREs ultimately trace back to the seminal work of Halevi et al. [14]. Their approach begins with constructing AREs $\mathsf{ARE}_{f_{\mathrm{eq}}}$ for the *equality function* f_{eq}, which

⋆ Most of the work was done while also at the University of Electro-Communications, Japan.

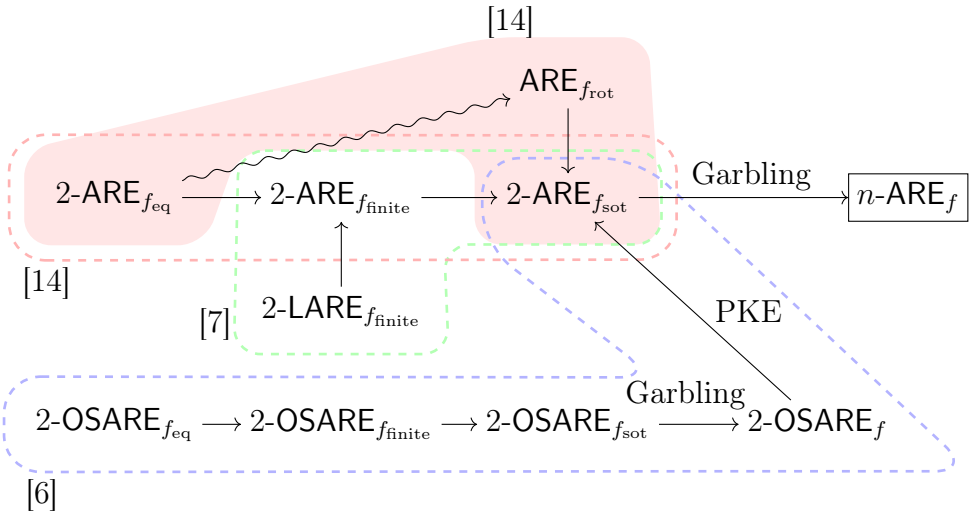

Fig. 1. An illustrated diagram of existing transformations for n-ARE for general functions f. f_{eq}, f_{finite}, f_{rot}, and f_{sot} are equality, finite, Rabin-OT, and SOT functions, respectively, and the prefix of (OS/L)ARE indicates the number of inputs (i.e., the number of parties in ARE) for the corresponding function. The red, blue, and green dashed regions represent the generic transformations to f_{sot} proposed by Halevi et al. [14], Bitansky et al. [6], and Bitansky et al. [7], respectively. The solid red region highlights a more efficient variant of Halevi et al.'s construction, where $\mathsf{ARE}_{f_{\mathrm{rot}}}$ is directly constructed based on $\mathsf{ARE}_{f_{\mathrm{eq}}}$ (represented by a wavy arrow).

tests whether two inputs coincide. This construction is then extended to *finite functions* (or, small functions) f_{finite}, i.e., Boolean functions defined over polynomial-size domains. Crucially, the class of finite functions contains the *string oblivious transfer* (SOT) function f_{sot}, defined by

$$f_{\mathrm{sot}} : \{0,1\} \times \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}.$$

By instantiating an ARE $\mathsf{ARE}_{f_{\mathrm{sot}}}$ for SOT f_{sot} and combining it with a garbling scheme [5, 18, 20, 21], one obtains AREs ARE_f for arbitrary functions f. While subsequent works [6, 7] improved and refined this methodology through relaxed notions termed *one-sided ARE* (OSARE) [6] and *leaky ARE* (LARE) [7], the core paradigm has remained unchanged: employing $\mathsf{ARE}_{f_{\mathrm{sot}}}$ together with a garbling scheme[3] to realize AREs ARE_f for arbitrary, general functions f. Except for the inefficient approach of placing f_{finite} in parallel for each output bit, garbling schemes are needed to lift AREs from restricted function classes, such as equality functions f_{eq} or finite functions f_{finite}, to general functions f. Therefore, *the efficiency of ARE constructions for the SOT function f_{sot} plays a crucial role in the overall efficiency of AREs for arbitrary functions f.*

Figure 1 summarizes the prior construction frameworks for ARE_f. As illustrated, they all indeed converge to the same final step for obtaining AREs for arbitrary functions f.

1.1 Our Focus: AREs for the SOT Function

In this paper, motivated by the observation that existing frameworks ultimately rely on AREs $\mathsf{ARE}_{f_{\mathrm{sot}}}$ for the SOT function, we investigate more efficient constructions of $\mathsf{ARE}_{f_{\mathrm{sot}}}$, with the aim of improving the overall efficiency of AREs ARE_f for general functions. Informally, the size of an ARE is measured by the total bit-length of all parties' encodings (see Definition 2 in Section 3.2).

[3] The state-of-the-art work [7] used the term *decomposable randomized encodings* (DREs), which are sometimes referred to as garbling schemes in the literature.

Table 1. Efficiency comparisons between Bitansky et al.'s ARE for SOT and ours. $|C|$ denotes the number of gates of a circuit C. PKE stands for public-key encryption. The sizes are derived by employing the Bitansky et al.'s transformation [6].

Scheme	Size	Correctness Error	Assumption	Pairing
BEG25 [6]	$O(\lambda^3 + \lambda(\lvert C_g\rvert + \lvert C_e\rvert))^{\P}$	0	PKE	–
Ours (§4)	$O(\lambda^2 + \lambda(\lvert C_g\rvert + \lvert C_e\rvert))^{\P}$	0	PKE	–

¶ C_g and C_e are Boolean circuits computing functions g and e, which are used for single-key functional encryption to lift OSARE to ARE, respectively (see [6] for their formal definitions). Such an FE can be constructed from plain public-key encryption [19]. Note that the input lengths of g and e are $O(\lambda)$.

Table 2. Efficiency comparisons between Bitansky et al.'s concrete construction and ours for OSARE for SOT.

Scheme	Size	Correctness Error	Assumption	Pairing
BEG25 [6]	$O(\lambda^2)$	0	–	–
Ours (§4)	$O(\lambda)$	0	–	–

Halevi et al. [14] constructed AREs for f_{sot} by first building $\mathsf{ARE}_{f_{\mathrm{eq}}}$ and extending it to finite functions f_{finite}, a class that contains the SOT function. Under a pairing-based assumption, this approach yields $\mathsf{ARE}_{f_{\mathrm{sot}}}$ of size $O(\lambda^2)$. They further proposed a more efficient variant based on the Rabin-OT function, achieving size $O(\lambda)$. Bitansky et al. [7] reduced the computational assumption to the computational Diffie–Hellman (CDH) assumption over traditional groups, at the cost of an $O(\lambda^2)$ blow-up in size. In the statistical setting, Bitansky et al. [7] introduced LARE and demonstrated the existence of statistically secure AREs, though with size $O(\lambda^3)$. Although a size of $O(\lambda)$ bits appears necessary to ensure a sufficient level of security, there are currently no ARE constructions for the SOT function that achieve $O(\lambda)$ size without relying on pairings.

From the perspective of computational assumptions, one cannot avoid the transformation of Bitansky et al. [6], which relies only on public-key encryption (PKE) as a building block. However, the resulting $\mathsf{ARE}_{f_{\mathrm{sot}}}$ is significantly less efficient than the aforementioned constructions, since single-key functional encryption, used to lift $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ to $\mathsf{ARE}_{f_{\mathrm{sot}}}$, substantially increases the size.

One can see efficiency comparisons of the above constructions in Tables 1 and 3.

1.2 Our Contributions

In this paper, we improve the efficiency of $\mathsf{ARE}_{f_{\mathrm{sot}}}$, the core primitive underlying constructions of AREs for arbitrary functions.

First, we propose a more efficient construction of $\mathsf{ARE}_{f_{\mathrm{sot}}}$ that relies only on the existence of public-key encryption (PKE) (see Table 1 for details). Our key idea is to construct $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ concretely without additional computational assumptions. Compared to the original $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ scheme [6], our construction achieves an $O(\lambda)$ factor improvement in size. We also provide a detailed comparison between our $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ scheme and that of Bitansky et al. in Table 2.

Second, we present the most efficient construction of $\mathsf{ARE}_{f_{\mathrm{sot}}}$ under specific computational assumptions. Our scheme achieves the same asymptotic size as Halevi et al.'s construction [14] while relying on the squaring decisional Diffie–Hellman (SDDH) assumption, which can be viewed as the additive SDDH (ASDDH) assumption over non-bilinear groups. Since Halevi et al.'s scheme relies on ASDDH over bilinear groups, our result eliminates the need for pairings. See Table 3 for a comparison with previous schemes.

Table 3. An efficiency comparison among concrete constructions of AREs for SOT. (A)SDDH and CDH represent (additive) squaring decisional and computational Diffie–Hellman assumptions, respectively.

Scheme	Size	Correctness Error‡	Assumption	Pairing
HIKR23 [14]	$O(\lambda)$	$O(2^{-\lambda})$	ASDDH	✓
BEG25 [6]†	$O(\lambda^3)$	$O(2^{-\lambda})$	CDH	–
BEGI25 [7]	$O(\lambda^3)$	$O(\lambda \cdot 2^{-\lambda})$	–	–
Ours (§5)	$O(\lambda)$	$O(2^{-\lambda})$	SDDH	–

† This is the ARE size obtained when applying the correctness amplification technique [6, Lemma 4.5] to achieve $O(2^{-\lambda})$-correctness.
‡ Under computational assumptions, perfect correctness can be achieved for any scheme, either by adopting the random string model [10], or by incurring a polynomial increase in communication overhead [3, 8].

2 Technical Overview

In this section, we present our ideas for constructing an efficient $\mathsf{ARE}_{f_{\mathrm{sot}}}$. As stated in the previous section, we propose two constructions: the first constructs an OSARE for SOT directly, to reduce the overhead incurred by the transformation of Bitansky et al. [6]; the second proposes an alternative concrete construction of ARE for SOT, which can be used with the efficient transformation of Halevi et al. [14]. At first, in Section 2.1, we describe the two approaches for constructing $\mathsf{ARE}_{f_{\mathrm{sot}}}$ proposed by Halevi et al. [14]. Then, in Section 2.2, we revisit Bitansky et al.'s PKE-based transformation, which is based on Halevi et al.'s approach, and present a more efficient approach by constructing the intermediate primitive directly. Finally, in Section 2.3, we consider an efficient construction of $\mathsf{ARE}_{f_{\mathrm{sot}}}$ based on the CDH assumption, which is equivalent to the squaring CDH assumption, proposed by Bitansky et al. [6].

2.1 Two Main Approaches for Constructing $\mathsf{ARE}_{f_{\mathrm{sot}}}$

We explain the two constructions of ARE for SOT $\mathsf{ARE}_{f_{\mathrm{sot}}}$ presented by Halevi et al. Their approaches use an ARE scheme for the two-party equality function f_{eq} over the domain D, defined below.

$$f_{\mathrm{eq}} \colon D \times D \to \{\mathit{equal}, \mathit{not\text{-}equal}\},\ f_{\mathrm{eq}}(x,y) = \begin{cases} \mathit{equal} & \text{if } x = y \\ \mathit{not\text{-}equal} & \text{otherwise} \end{cases}$$

They proposed the following two approaches to construct $\mathsf{ARE}_{f_{\mathrm{sot}}}$ from $\mathsf{ARE}_{f_{\mathrm{eq}}}$.

- Method 1: First construct an ARE scheme for any finite function from equality, and then instantiate it for SOT.
- Method 2: Construct an ARE scheme for SOT directly from equality.

The difference between Method 1 and Method 2 is that Method 1 can employ an arbitrary ARE scheme for f_{eq}, whereas Method 2 cannot; this distinction directly affects the communication complexity.

Method 1. For Method 1, any ARE $\mathsf{ARE}_{f_{\mathrm{eq}}}$ scheme can be transformed into an ARE scheme for any finite function $f : D_1 \times D_2 \to \{0,1\}$ whose output is a single bit. Regarding communication complexity, assuming $|D_1| \leq |D_2|$, the resulting scheme incurs a communication cost that is $|D_1|$

times that of the ARE for equality. In particular, this method yields SOT whose communication complexity is 2λ times that of the ARE for equality (recall that $f_{\mathrm{sot}} : \{0,1\} \times \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$).

Method 2. In contrast, Method 2 exploits the specific $\mathsf{ARE}_{f_{\mathrm{eq}}}$ construction of Halevi et al., which is based on a pairing $\hat{e}\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and transforms it first into an ARE scheme for Rabin-OT (ROT), and subsequently into an ARE scheme for SOT, incurring only a constant-factor overhead in communication complexity. We first describe a transformation to obtain an ARE scheme for ROT, defined as

$$f_{\mathrm{rot}}\colon \{0,1\} \times (\{0,1\} \times \{0,1\}^{\lambda}) \to \{0,1\}^{\lambda},\ f_{\mathrm{rot}}((b,m),c) = \begin{cases} m & \text{if } b = c \\ R & \text{otherwise.} \end{cases}$$

Concretely, in the $\mathsf{ARE}_{f_{\mathrm{eq}}}$ scheme of Halevi et al., given the sum of the encodings obtained by the evaluator, denoted by $(\hat{z}_0, \hat{z}_1, \hat{z}_2) \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_T$, the evaluator determines whether the inputs of the two parties are equal by checking whether $\hat{e}(\hat{z}_0, \hat{z}_1) = \hat{z}_2$. Here, since $\hat{e}(\hat{z}_0, \hat{z}_1) \in \mathbb{G}_T$, for a message $M \in \mathbb{G}_T$, one can modify the scheme so that instead of outputting $\hat{z}_2$, it outputs $\hat{z}'_2 = \hat{z}_2 \oplus M$. In this case, the evaluator obtains z_0, z_1 such that $\hat{e}(\hat{z}_0, \hat{z}_1) = z_2$ if and only if the parties' inputs are equal. Moreover, from $\hat{z}'_2 - \hat{e}(z_0, z_1)$, the evaluator can recover M. By setting the input domains of the two parties to $\{0,1\}$ and randomizing appropriately, one can construct an ARE for the Rabin-OT function. Using this ARE for ROT, it is possible to transform it into SOT. The details are deferred to Section 5.2.

2.2 Efficient One-Sided ARE for SOT

Bitansky et al. [6] showed that ARE can be constructed from any PKE scheme. More concretely, they introduced a new notion called One-Sided ARE (OSARE) and showed that any OSARE for an arbitrary function f can be transformed into an ARE for f using a PKE scheme. They then construct an information-theoretically secure OSARE for equality and obtain an OSARE for a general function f by applying the transformation similar to that of Halevi et al. [14].

Suffer from Efficient ARE Construction from PKE. However, it seems difficult to obtain an efficient ARE from the OSARE proposed by Bitansky et al. One of the major obstacles arises in the step of deriving an $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ from their $\mathsf{OSARE}_{f_{\mathrm{eq}}}$. Informally, OSARE weakens the security notion of standard two-party ARE by failing to guarantee the privacy of one party's input. For the formal definition, see Definition 3 in Section 3.3. In particular, to obtain an efficient OSARE for SOT, it is unclear how to apply Halevi et al.'s efficient Method 2 transformation from $\mathsf{OSARE}_{f_{\mathrm{eq}}}$ to $\mathsf{OSARE}_{f_{\mathrm{sot}}}$. In the following, we explain why Halevi et al.'s technique in Method 2 cannot be directly applied to Bitansky et al.'s OSARE for equality.

We examine the information-theoretic two-party OSARE for equality constructed by Bitansky et al. Let Party 1's encoding $\hat{x}$ on input i, be given by

$$\hat{x} \coloneqq (u_1, \ldots, u_{i-1}, 0, u_{i+1}, \ldots, u_d),$$

, where $d = |D|$, and let Party 2's encoding $\hat{y}$ on input j be

$$\hat{y} \coloneqq (0, \ldots, v, \ldots, 0),$$

where $u_1, \ldots, u_d, v \xleftarrow{\$} \mathbb{Z}_q \setminus \{0\}$, and the value v appears in the j-th position of $\hat{y}$. In the decoding algorithm, given the component-wise sum of the encodings, denoted by $(\hat{z}_1, \ldots, \hat{z}_d)$, the output is *not-equal* if at least one coordinate is zero, and *equal* otherwise.

We next consider applying Method 2. However, as we explain below, this appears to be difficult. The reason is that the equality-checking mechanism in Bitansky et al.'s OSARE differs significantly from that in Halevi et al.'s ARE for equality. In Halevi et al.'s construction, the evaluator checks equality by performing a specific algebraic relation on the sum of the encodings. In contrast, Bitansky et al.'s construction determines equality by checking whether any coordinate of the summed vector is zero. Because of this structural difference, there is no apparent way to embed a message into Bitansky et al.'s OSARE for equality such that the message is revealed precisely when the inputs are equal. Therefore, since Method 2 does not seem applicable to Bitansky et al.'s construction, obtaining an $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ requires applying Method 1, which incurs a communication complexity of $O(\lambda^2)$.

A New Approach: Direct OSARE Scheme for OT. It has been observed that the method of efficiently obtaining an $\mathsf{ARE}_{f_{\mathrm{sot}}}$ from Halevi et al.'s $\mathsf{ARE}_{f_{\mathrm{eq}}}$ cannot be applied to the OSARE for equality proposed by Bitansky et al. Therefore, we take a completely different approach from that of Bitansky et al. to construct an OSARE for SOT. Specifically, instead of deriving it from an $\mathsf{OSARE}_{f_{\mathrm{eq}}}$, we directly construct an $\mathsf{OSARE}_{f_{\mathrm{sot}}}$. Here, the function f_{sot} is defined as follows:

$$f_{\mathrm{sot}} \colon \{0,1\} \times \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}, f_{\mathrm{sot}}(c,(s_0,s_1)) = s_c$$

We begin by considering a trivial ARE construction for f_{sot} that guarantees correctness only and provides no security whatsoever. Let Party 1's input be $c \in \{0,1\}$, and define Party 1's encoding $\hat{x}$ as

$$\hat{x} \coloneqq (0,0,c).$$

Let Party 2's inputs be s_0, s_1, and define Party 2's encoding $\hat{y}$ as

$$\hat{y} \coloneqq (s_0, s_1, 0).$$

Then the evaluator obtains the sum of the encodings $\hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2)$ given by

$$\hat{z} \coloneqq (s_0, s_1, c).$$

As the decoding algorithm, the evaluator inspects $\hat{z}_2$ and outputs $\hat{z}_0 = s_0$ if $\hat{z}_2 = 0$, and $\hat{z}_1 = s_1$ if $\hat{z}_2 = 1$. It is immediate that the correctness of the ARE is satisfied. However, this construction clearly provides no security. Informally, for example, when $c = 1$, the value s_0 corresponding to $c = 0$ is revealed, and thus information about the non-selected message leaks.

Conversely, in the above trivial construction that provides no security guarantees, it suffices to ensure that when $c = 1$, the value s_0 is not revealed to the evaluator, and when $c = 0$, the value s_1 is not revealed. It is important to note that the ARE considered here is one-sided. In particular, we aim to protect the privacy of Party 2's inputs, while making no guarantee about the privacy of Party 1's input. Thus, the value c may be revealed, and moreover, the output of the function f_{sot}, namely s_c, may also be revealed from the sum of the encodings. Motivated by this observation, we modify the trivial construction as follows.

Let Party 1's input be $c \in \{0,1\}$, sample $r \xleftarrow{\$} \{0,1\}^{\lambda}$, and define Party 1's encoding $\hat{x}$ as

$$\hat{x} \coloneqq \begin{cases} (0,r,c) \text{ if } c = 0, \\ (r,0,c) \text{ otherwise.} \end{cases}$$

Let Party 2's inputs be s_0, s_1, and define Party 2's encoding $\hat{y}$ as

$$\hat{y} \coloneqq (s_0, s_1, 0).$$

Then the evaluator obtains the sum of the encodings $\hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2)$ given by

$$\hat{z} \coloneqq \begin{cases} (s_0, s_1 + r, c) \text{ if } c = 0, \\ (s_0 + r, s_1, c) \text{ otherwise.} \end{cases}$$

Observing the above construction, we see that Party 1's input is always revealed from the sum of the encodings $\hat{z}$. This is an inherent feature of the construction. In contrast, in Bitansky et al.'s ARE for equality, one party's input is revealed only in the *not-equal* case, while in the *equal* case, neither party's input is revealed. We instead ultimately focus on the fact that one-sided security is a significantly weaker notion. By exploiting this weaker security requirement more aggressively, we obtain a more efficient construction of one-sided ARE.

2.3 Efficient ARE for SOT from SDDH

Finally, we explain an efficient construction of ARE from the Squaring DDH (SDDH) assumption. Our first observation is to examine the communication complexity of the $\mathsf{ARE}_{f_{\mathrm{eq}}}$ construction based on the (pairing-free) CDH assumption proposed by Bitansky et al. [6]. In their construction, the additive encodings are of the form

$$\begin{aligned} (\hat{x}_0, \hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4) &\coloneqq (r, g^{rx}, g^{r^2 x}, 0, 0), \\ (\hat{y}_0, \hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4) &\coloneqq (s, g^{-sy}, g^{-s^2 y} + [R], S, \langle \mathrm{bin}([R]), S \rangle). \end{aligned}$$

To check equality, the evaluator obtains the sum

$$(\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3, \hat{z}_4) = (g^{r+s}, g^{rx-sy}, g^{r^2 x - s^2 y + R}, S, \langle \mathrm{bin}([R]), S \rangle)$$

and, checks whether $\langle \mathrm{bin}(\hat{z}_2 - \hat{z}_0 \hat{z}_1), \hat{z}_3 \rangle = \hat{z}_4$, which always occurs when $x = y$, and with probability exactly $1/2$ otherwise. They employ parallel repetition to amplify correctness; to achieve a correctness probability of $\delta \in (0, 1)$, it is necessary to run $O(\log(1/\delta))$ parallel copies, which significantly increases the communication complexity. Moreover, their construction does not allow us to apply Halevi et al.'s efficient transformation from $\mathsf{ARE}_{f_{\mathrm{eq}}}$ to $\mathsf{ARE}_{f_{\mathrm{sot}}}$ (Method 2). This construction further applies the Goldreich–Levin hardcore bit theorem to reduce the security to the Squaring CDH assumption (which is in turn equivalent to the standard CDH assumption [2, 17]). However, when reducing to the CDH assumption, the above construction only performs a 1-bit comparison (via an inner product of bit strings). Thus, in addition to the loss in correctness probability, we cannot apply Halevi et al.'s efficient transformation from $\mathsf{ARE}_{f_{\mathrm{eq}}}$ to $\mathsf{ARE}_{f_{\mathrm{sot}}}$ (i.e., Method 2 described above), which embeds a λ-bit string into the equality-checking procedure.

Therefore, rather than reducing to the CDH assumption, we modify the construction to be compatible with Halevi et al.'s efficient transformation method (Method 2). More concretely, we first construct an ARE for ROT that embeds a message $M \in \mathbb{G}$, and then apply Halevi et al.'s transformation to obtain an ARE for SOT. As a result, our construction employs the Squaring DDH assumption instead of the Squaring CDH assumption.

3 Preliminaries

3.1 Notation

Throughout the paper, λ denotes a security parameter. For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\} \subseteq \mathbb{N}$. For $a, b \in \mathbb{N}$ with $a \leq b$, $[a, b]$ denotes the set $\{a, a + 1, \ldots, b\}$. For a finite set S, $s \xleftarrow{\$} S$ denotes a

sampling of an element s from S uniformly at random. Probabilistic polynomial time is abbreviated as PPT. For discrete probability distributions X and Y over support A, their statistical distance (or total variation distance) is $\Delta(X,Y) = \frac{1}{2}\sum_{a\in A}|\Pr[X=a]-\Pr[Y=a]|$. Let $X \approx_\delta Y$ denote the fact that $\Delta(X,Y) \leq \delta$. Let $\approx_c$ denote computational indistinguishability. Let $\mathcal{U}(X)$ denote the uniform distribution over the set X. In particular, when $X = \{0,1\}^\lambda$, we write $\mathcal{U}_\lambda$.

3.2 Additive Randomized Encodings

Definition 1 (Additive Randomized Encoding [14]). *Let $f\colon (\{0,1\}^\ell)^n \to \{0,1\}^m$ be a n-party function. A n-party ARE scheme for f consists of three PPT algorithms $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^n, 1^\ell) \to \mathsf{pp}$ *is a setup algorithm that, given the security parameter λ, the number of parties n, and the input length ℓ of the function f, generates public parameters* pp*. The public parameters include 1^λ, n, ℓ, and an explicit description of an Abelian group $(\mathbb{G}, +)$.*
- $\mathsf{Enc}_{\mathsf{pp}}(x_i, i) \to \hat{x}_i$ *is an encoding algorithm that, given an input $x_i \in \{0,1\}^\ell$ of party i and an index $i \in [n]$ outputs an encoding $\hat{x}_i \in \mathbb{G}$.*
- $\mathsf{Dec}_{\mathsf{pp}}(\hat{y}) \to y$ *is a decoding algorithm that given $\hat{y} \in \mathbb{G}$ outputs y.*

We require the following properties:

1. *Correctness: Π is ϵ-correct, for a function $\epsilon(\lambda)$, if for all $\lambda, n, \ell \in \mathbb{N}$, and $x_1, \ldots x_n \in \{0,1\}^\ell$:*

$$\Pr\left[\begin{array}{ll}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^n, 1^\ell); & \\ \hat{x}_i \leftarrow \mathsf{Enc}_{\mathsf{pp}}(x_i, i); & : \mathsf{Dec}_{\mathsf{pp}}(\hat{y}) = f(x_1, \ldots, x_n) \\ \hat{y} = \sum_{i=1}^n \hat{x}_i & \end{array}\right] \geq 1 - \epsilon(\lambda),$$

   where summation is taken over the group $\mathbb{G}$ specified by pp*.*
2. *Security: Π is δ-secure if there exists a PPT simulator* Sim *such that for all $\lambda \in \mathbb{N}$, $x_1, \ldots x_n \in \{0,1\}^\ell$:*

$$(\mathsf{pp}, \hat{y}) \approx_\delta \mathsf{Sim}(1^\lambda, f(x_1, \ldots x_n)),$$

   where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \hat{x}_i \leftarrow \mathsf{Enc}_{\mathsf{pp}}(x_i, i)$, and $\hat{y} = \sum_{i=1}^n \hat{x}_i$.

We say that security is perfect if $\delta = 0$. Moreover, we say that security is computational if statistical indistinguishability is replaced by computational indistinguishability, denoted by $\approx_c$.

Definition 2 (ARE Size [7]). *We say that n-party ARE over a group $\mathbb{G}$ has size (at most) c if:*

- $\mathbb{G} = \mathbb{G}_1 \times \cdots \times \mathbb{G}_t$ *for some (explicit and efficient) groups $\mathbb{G}_i$*
- *For any party $i \in [n]$, there is a set $S_i \subseteq [t]$ such that the party's encodings $\hat{x}_i$ are supported on $\mathbb{G}_{S_i} \times \{0\}_{\overline{S_i}}$. Namely, each party encodes into a subset of coordinates (when $\mathbb{G}$ is viewed as a product).*
- $\sum_{i\in[n]} \log|\mathbb{G}_{S_i}| := \sum_{i\in[n],\, j\in S_i} \log|\mathbb{G}_j| \leq c$.

We often use expressions such as Party i to directly send some value (e.g., C) to the evaluator in the context of ARE. Formally, this can be interpreted within the ARE framework as Party i encoding $C \in \mathbb{G}$, while all other parties encode $0 \in \mathbb{G}$. Hence, this description does not deviate from the ARE setting. In particular, a party may send nothing with respect to some coordinate j (i.e., in $\mathbb{G}_j$), and the notation $\{0\}$ above captures precisely this situation.

Two-Party OSARE for SOT $\mathsf{OSARE}_{f_{\mathbf{sot}}}$

$\underline{\mathsf{Enc}(1^\lambda, c \in \{0,1\}, 1)}$:

- Sample $r \xleftarrow{\$} \{0,1\}^\lambda$.
- Output $\hat{x} = \begin{cases} (0, r, 0) & \text{if } c = 0 \\ (r, 0, 1) & \text{otherwise.} \end{cases}$

$\underline{\mathsf{Enc}(1^\lambda, (s_0, s_1) \in \{0,1\}^{2\lambda}, 2)}$:

- Output $\hat{y} = (s_0, s_1, 0)$.

$\underline{\mathsf{Dec}(1^\lambda, \hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2))}$:

- Output $\hat{z}_0$ if $\hat{z}_2 = 0$, otherwise $\hat{z}_1$.

Fig. 2. OSARE for SOT $\mathsf{OSARE}_{f_{\mathrm{sot}}}$

3.3 One-Sided Additive Randomized Encodings

Definition 3 (One-Sided Additive Randomized Encoding [6]). *Let $f\colon \{0,1\}^{\ell_1} \times \{0,1\}^{\ell_2} \to \{0,1\}^m$ be a 2-party function. An ARE scheme $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ for f, defined for two parties, is said to be a one-sided ARE if it satisfies the same correctness as an ARE and the security defined below:*

1. *<u>One-Sided Security:</u> Π is δ-secure if there exists a PPT simulator Sim such that for all $\lambda \in \mathbb{N}$, all $x \in \{0,1\}^{\ell_1}$, and all $y \in \{0,1\}^{\ell_2}$:*

$$(\mathsf{pp}, \hat{z}) \approx_\delta \mathsf{Sim}(1^\lambda, x, f(x,y)),$$

where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \hat{x} \leftarrow \mathsf{Enc}_{\mathsf{pp}}(x, 1), \hat{y} \leftarrow \mathsf{Enc}_{\mathsf{pp}}(y, 2)$, and $\hat{z} = \hat{x} + \hat{y}$.

We say that security is perfect if $\delta = 0$. Moreover, we say that security is computational if statistical indistinguishability is replaced by computational indistinguishability, denoted by $\approx_c$.

4 Efficient OSARE for SOT $\mathsf{OSARE}_{f_{\mathbf{sot}}}$

We consider the one-sided ARE for the string-OT function f_{sot}, defined below.

$$f_{\mathrm{sot}}\colon \{0,1\} \times \{0,1\}^{2\lambda} \to \{0,1\}^\lambda, f_{\mathrm{sot}}(c, (s_0, s_1)) = s_c$$

Then, we describe our construction of one-sided ARE $\mathsf{OSARE}_{f_{\mathrm{sot}}}$ for the string-OT function in Figure 2. As in the construction in Figure 2, we do not require a setup; instead, we make the encoding and decoding algorithms take the security parameter as explicit input.

Theorem 1. *The scheme $\mathsf{OSARE}_{f_{sot}}$ in Figure 2 is perfectly correct.*

Proof. It is trivial that the scheme satisfies perfect correctness. For all $(c, s_0, s_1) \in \{0,1\} \times \{0,1\}^{2\lambda}$ and all $r \in \{0,1\}^\lambda$, the sum of the encodings $\hat{z}$ is formed as follows by construction:

$$\hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2) = \begin{cases} (s_0, s_1 + r, 0) & \text{if } c = 0 \\ (s_0 + r, s_1, 1) & \text{otherwise.} \end{cases}$$

Then, the decoder outputs $\hat{z}_2$, which equals s_c. Hence, the scheme is perfectly correct. □

Theorem 2. *The scheme* $\mathsf{OSARE}_{f_{sot}}$ *in Figure 2 is statistically one-sided secure.*

Proof. In the one-sided security setting, we allow leakage of one party's input; in particular, the receiver's choice bit c is not required to be hidden. Accordingly, it suffices to simulate the view given c and the functionality output $f_{\mathrm{sot}}(c, (s_0, s_1)) = s_c$.

We define the simulator Sim as follows: On input $(1^\lambda, c, s_c)$, it samples $s' \xleftarrow{\$} \{0,1\}^\lambda$ and outputs $(s_0, s', 0)$ if $c = 0$, otherwise $(s', s_1, 1)$. To see that this is statistically indistinguishable from the real execution, note that in the real distribution, if $c = 0$ then $\hat{z}_1 = s_1 + r$ for uniform $r \xleftarrow{\$} \{0,1\}^\lambda$, and hence $\hat{z}_1$ is uniform over $\{0,1\}^\lambda$, which is equivalently distributed as s'. If $c = 1$ then $\hat{z}_0 = s_0 + r$ for uniform $r \xleftarrow{\$} \{0,1\}^\lambda$, and hence $\hat{z}_0$ is uniform over $\{0,1\}^\lambda$, which is equivalently distributed as s'. Therefore, in either case, the output of $\mathsf{Sim}(1^\lambda, c, s_c)$ simulates the real distribution. □

Theorem 3. *The size of the scheme* $\mathsf{OSARE}_{f_{sot}}$ *in Figure 2 is* $O(\lambda)$.

Proof. According to Definition 2 in Section 3.2, letting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3)$ denote the groups used for the encodings of each party in the OSARE scheme in Fig. 2, the size of the ARE is given as follows.

$$\begin{aligned}
\sum_{i\in[n]} \log|\mathbb{G}_{S_i}| &:= \sum_{i\in[n],\, j\in S_i} \log|\mathbb{G}_j| \\
&= \sum_{j\in S_1} \log|\mathbb{G}_j| + \sum_{j\in S_2} \log|\mathbb{G}_j| \\
&= \begin{cases} (\log|\mathbb{G}_2| + \log|\mathbb{G}_3|) + (\log|\mathbb{G}_1| + \log|\mathbb{G}_2|) & \text{if } c = 0, \\ (\log|\mathbb{G}_1| + \log|\mathbb{G}_3|) + (\log|\mathbb{G}_1| + \log|\mathbb{G}_2|) & \text{if } c = 1. \end{cases} \\
&= O(\lambda).
\end{aligned}$$

□

5 Efficient ARE for SOT from SDDH

In this section, we propose an ARE scheme for SOT based on the Squaring decisional Diffie–Hellman (SDDH) assumption. This construction adapts the CDH-based construction of Bitansky et al. [6] to obtain an ARE for Rabin-OT. Finally, we present a transformation from ARE for Rabin-OT to ARE for SOT. The idea of transforming into ARE for Rabin-OT originates from Halevi et al. [14].

5.1 Efficient ARE for Rabin-OT $\mathsf{ARE}_{f_{\mathrm{rot}}}$ from SDDH

Definition 4 (Squaring decisional Diffie-Hellman assumption). *The squaring decision DH holds for a parameter-generator* $\mathcal{G}$*, if for every* $\lambda \in \mathbb{N}$*, the following two distribution ensembles are computationally indistinguishable:*

$$(g, g^x, g^{x^2}) \text{ and } (g, g^x, g^r)$$

where $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^\lambda)$ *and* $x, r \xleftarrow{\$} \mathbb{Z}_q$.

Lemma 1 (Based on Lemma 4.2 of [6]). *Assuming Squaring DDH, given* $\mathsf{pp} = (\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^\lambda)$*, distinct* $x, y \in [1, \frac{q-1}{2}]$ *where* $x \neq y, r, s \xleftarrow{\$} \mathbb{Z}_q$*,* $t_1 = r + s, t_2 = g^{rx-sy}$*, then* $t_3 = g^{r^2x - s^2y}$ *is computationally indistinguishable.*

Proof. Most of the proof follows [6, Lemma 4.2]. Letting $\tau_1 = r + s$ and $\tau_2 = rx - sy$, we have

$$r = \frac{y\tau_1 + \tau_2}{x+y},\ s = \frac{x\tau_1 - \tau_2}{x+y}.$$

Then, r and s are eliminated from t_3.

$$t_3 = g^{\alpha\cdot\tau_1\tau_2+\beta\cdot\tau_1^2+\gamma\cdot\tau_2^2}$$

where,

$$\alpha(x,y) = \frac{4xy}{(x+y)^2},\ \beta(x,y) = \frac{xy(y-x)}{(x+y)^2},\ \gamma(x,y) = \frac{x-y}{(x+y)^2} \tag{1}$$

and τ_1, τ_2 are uniform in $\mathbb{Z}_q$ and independent. We note that $x + y \neq 0$ since $x, y \in [1, \frac{q-1}{2}]$ and $\alpha, \beta, \gamma \neq 0$ since $x \neq y$. Now, suppose there exists a PPT adversary $\mathcal{A}$ such that, given $(\mathsf{pp}, \tau_1, g^{\tau_2}, t_3)$ where $(\tau_1, \tau_2) \leftarrow \mathbb{Z}_q^2$,

- In the real case, namely when $t_3 = g^{\alpha\tau_1\tau_2+\beta\tau_1^2+\gamma\tau_2^2}$, the adversary outputs 1,
- In the random case, namely when t_3 is uniform in $\mathbb{G}$, the adversary outputs 0,

with a non-negligible distinguishing advantage.

Then, we construct a PPT adversary $\mathcal{B}$ that solves the SDDH problem using $\mathcal{A}$ as a subroutine. On input $(\mathsf{pp}, g^{\rho_1}, g^{\rho_2})$, the adversary $\mathcal{B}$ proceeds as follows:

1. Samples $u \xleftarrow{\$} \mathbb{Z}_q$, $x, y \xleftarrow{\$} [1, \frac{q-1}{2}]$ where $x \neq y$.
2. Computes $\alpha = \alpha(x,y)$, $\beta = \beta(x,y)$, $\gamma = \gamma(x,y)$ according to Eq. (1).
3. Computes $t_3' = (g^{\rho_1})^{\alpha u} \cdot g^{\beta\cdot u^2} \cdot (g^{\rho_2})^{\gamma}$.
4. Outputs $b' \leftarrow \mathcal{A}(\mathsf{pp}, u, g^{\rho_1}, t_3')$.

We show that $\mathcal{B}$ solves the SDDH problem with non-negligible probability, equal to the success probability of $\mathcal{A}$. In the real case, suppose $\rho_2 = \rho_1^2$. Then

$$t_3' = (g^{\rho_1})^{\alpha u} \cdot g^{\beta u^2} \cdot (g^{\rho_1^2})^{\gamma} = g^{\alpha u\rho_1+\beta u^2+\gamma\rho_1^2}.$$

Setting $\tau_1 = u$ and $\tau_2 = \rho_1$ since $\mathcal{A}$'s input, we obtain

$$t_3' = g^{\alpha\tau_1\tau_2+\beta\tau_1^2+\gamma\tau_2^2}.$$

Hence, the input to $\mathcal{A}$ is distributed exactly as in the real case. Therefore, whenever $\mathcal{A}$ outputs 1, $\mathcal{B}$ also outputs 1.

In the random case, suppose ρ_2 is uniform and independent of ρ_1. Then

$$t_3' = g^{\alpha u\rho_1+\beta u^2+\gamma\rho_2}.$$

The equality

$$t_3' = g^{\alpha u\rho_1+\beta u^2+\gamma\rho_1^2}$$

holds if and only if

$$\gamma(\rho_2 - \rho_1^2) \equiv 0 \pmod{q}.$$

Since $\gamma \neq 0$ and q is prime, this occurs if and only if $\rho_2 = \rho_1^2$, which happens with probability $1/q$. Therefore, in the random case, $\mathcal{A}$ outputs 1 with probability at most $1/q$.

Consequently, $\mathcal{B}$ solves the SDDH problem with the same distinguishing advantage as $\mathcal{A}$, except with probability $1/q$. □

Two-Party ARE for Rabin-OT $\mathsf{ARE}_{f_{\mathbf{rot}}}$ from SDDH

$\underline{\mathsf{Setup}(1^\lambda)}$:

- Set $\mathsf{pp} = (\mathbb{G}, q, g, E) \leftarrow \mathcal{G}$.

$\underline{\mathsf{Enc}(1^\lambda, c \in \{0,1\}, 1)}$:

- Sample $r \xleftarrow{\$} \mathbb{Z}_q$.
- Output $\hat{x} = (r, g^{r(c+1)}, g^{r^2(c+1)}, 0)$.

$\underline{\mathsf{Enc}(1^\lambda, (b \in \{0,1\}, m \in \{0,1\}^\lambda), 2)}$:

- Sample $s \xleftarrow{\$} \mathbb{Z}_q, M \xleftarrow{\$} \mathbb{G}$.
- Output $\hat{y} = (s, g^{-s(b+1)}, M \cdot g^{-s^2(b+1)}, m \oplus E(M))$.

$\underline{\mathsf{Dec}(1^\lambda, \hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3))}$:

- Output $m = \hat{z}_3 \oplus E(\hat{z}_2/\hat{z}_1^{\hat{z}_0})$.

Fig. 3. Two-Party ARE for Rabin-OT $\mathsf{ARE}_{f_{\mathrm{rot}}}$ from SDDH.

We describe a scheme $\mathsf{ARE}_{f_{\mathrm{rot}}}$ for the Rabin-OT function. It is convenient to consider Rabin-OT as the following randomized function:

$$f_{\mathrm{rot}} \colon \{0,1\} \times (\{0,1\} \times \{0,1\}^\lambda) \to \{0,1\}^\lambda, f_{\mathrm{rot}}(c,(b,m)) = \begin{cases} m \text{ if } b = c \\ R \text{ otherwise} \end{cases}$$

where $R \in \{0,1\}^\lambda$ is (close to) uniform and independent of m. This construction is obtained by slightly modifying the Two-Party Equality ARE $\mathsf{ARE}_{f_{\mathrm{eq}}}$ for CDH construction of Bitansky et al. [6] so that it can be adapted to Rabin-OT. Additionally, we use an "encoding" $E : \mathbb{G} \to \{0,1\}^\lambda$ with the property that $E(M)$ for a uniform $M \in \mathbb{G}$ yields an almost-uniform bit-string in $\{0,1\}^\lambda$ following the idea of Halevi et al. [14].

Theorem 4. *The scheme $\mathsf{ARE}_{f_{rot}}$ is correct.*

Proof. For input $b, c \in \{0,1\}$ and additive encoding $\hat{z} = (\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3)$, we have

$$\begin{aligned} \hat{z}_2/\hat{z}_1^{\hat{z}_0} &= M \cdot g^{r^2(c+1)-s^2(b+1)} / \{g^{r(c+1)-s(b+1)}\}^{(r+s)} \\ &= M \cdot g^{rs(b-c)} \end{aligned}$$

- If $b = c$, then $\hat{z}_3 \oplus E(\hat{z}_2/\hat{z}_1^{\hat{z}_0}) = (m \oplus E(M)) \oplus E(M) = m$
- If $b \neq c$, then $\hat{z}_3 \oplus E(\hat{z}_2/\hat{z}_1^{\hat{z}_0}) = (m \oplus E(M)) \oplus E(M \cdot g^{\pm rs})$, which is nearly uniform in $\{0,1\}^\lambda$. □

Theorem 5. *The scheme $\mathsf{ARE}_{f_{rot}}$ is computationally secure.*

Proof. We define the simulator Sim as follows:

- When inputs are sender's message m, $\mathsf{Sim}(1^\lambda, m)$ sample $r', s', M' \leftarrow \mathbb{Z}_q$ uniformly at random compute $m' = m \oplus E(g^{M'-s'r'})$ and outputs $(\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3) = (r', g^{s'}, g^{M'}, m')$.
- When inputs are random message R, $\mathsf{Sim}(1^\lambda, m)$ sample $r', s', M' \leftarrow \mathbb{Z}_q, m' \leftarrow \{0,1\}^\lambda$ uniformly at random and outputs $(\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3) = (r', g^{s'}, g^{M'}, m')$.

In both cases, the values $(\hat{z}_0, \hat{z}_1, \hat{z}_2) = (r+s, g^{r(c+1)-s(b+1)}, g^{M+r^2(c+1)-s^2(b+1)})$ are independent and uniformly random over their respective domains. Since $c \neq -b$, $0 < c+1, b+1 < q/2$, the value $\hat{z}_0$ and $\hat{z}_1$ are linearly independent.

Here, when $c = b$, correctness implies that m' is identically distributed to $\hat{z}_3$.

When $c \neq b$, by Lemma 1, the value $\hat{z}_2/M = g^{r^2(c+1)-s^2(b+1)} = g^{\pm rs}$ is indistinguishable from a uniformly random element in the domain $\mathbb{G}$. Hence, $\hat{z}_2 = M \cdot g^{\pm rs}$ is indistinguishable from $R \cdot g^{\pm rs}$, where $R \xleftarrow{\$} \mathbb{G}$. It follows that $\hat{z}_3 = m \oplus E(M)$ is indistinguishable from $m \oplus E(R)$ given $\hat{z}_0, \hat{z}_1$, and $\hat{z}_2$. Therefore, we have

$$(\hat{z}_0, \hat{z}_1, \hat{z}_2, \hat{z}_3) \approx_c (\mathcal{U}(\mathbb{Z}_q), \mathcal{U}(\mathbb{G}), \mathcal{U}(\mathbb{G}), \mathcal{U}_\lambda)\,. \qquad \square$$

Theorem 6. *The size of the scheme* $\mathsf{ARE}_{f_{rot}}$ *in Figure 3 is* $O(\lambda)$.

Proof. According to Definition 2 in Section 3.2, letting $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, \mathbb{G}_4)$ denote the groups used for the encodings of each party in the ARE scheme in Fig. 3, the size of the ARE is given as follows.

$$\begin{aligned}
\sum_{i\in[n]} \log|\mathbb{G}_{S_i}| &:= \sum_{i\in[n],\, j\in S_i} \log|\mathbb{G}_j| \\
&= \sum_{j\in S_1} \log|\mathbb{G}_j| + \sum_{j\in S_2} \log|\mathbb{G}_j| \\
&= (\log|\mathbb{G}_1| + \log|\mathbb{G}_2| + \log|\mathbb{G}_3|) \\
&\quad + (\log|\mathbb{G}_1| + \log|\mathbb{G}_2| + \log|\mathbb{G}_3| + \log|\mathbb{G}_4|) \\
&= O(\lambda).
\end{aligned}$$

$\square$

5.2 Efficient ARE for SOT $\mathsf{ARE}_{f_{\mathrm{sot}}}$ from $\mathsf{ARE}_{f_{\mathrm{rot}}}$

Finally, we present a transformation from ARE for Rabin-OT to ARE for String-OT. The transformation described below essentially follows [14, Section 5.1.3].

The parties run two instances of the Rabin-OT scheme in parallel, where:

- Party 1 uses the same choice bit c in both instances.
- Party 2 with inputs $s_0, s_1 \in \{0,1\}^\lambda$ chooses a random bit $b \xleftarrow{\$} \{0,1\}$ and uses input (b, s_b) in the first instance and input $(1-b, s_{1-b})$ in the second.

In addition to the above, the participant sends the following, corresponding to the previously chosen choice bit $b, c \in \{0,1\}$, within the ARE framework:

- Party 1 sends the encoding (c, c)
- Party 2 sends the encoding $(b, 1-b)$

Let $(\hat{z}_0, \hat{z}_1)$ denote the sum of the encodings in the ARE for f_{rot}, and let $(\hat{z}_2, \hat{z}_3)$ denote the sum of the encodings in the additional ARE for the one-bit XOR function. The evaluator performs decoding as follows:

- If $\hat{z}_2 = 0$, output $\mathsf{Dec}_{f_{\mathrm{rot}}}(\hat{z}_0)$.
- If $\hat{z}_3 = 0$, output $\mathsf{Dec}_{f_{\mathrm{rot}}}(\hat{z}_1)$.

Here, we assume that $\hat{z}_2, \hat{z}_3 \in \{0,1\}$. That is, the second and third components lie in $\{0,1\}$, and we expect that when the corresponding encodings are both equal to 1, their sum evaluates to 0.

By making this modification, we avoid the use of the λ-bit verification string employed in Halevi et al.'s transformation. As a result, the communication complexity is reduced (by at most a constant

factor). Moreover, whereas their transformation incurs a correctness error of $2^{-\lambda}$, our transformation introduces no additional correctness error.

Consequently, the correctness of $\mathsf{ARE}_{f_{\mathrm{rot}}}$ is preserved in the resulting $\mathsf{ARE}_{f_{\mathrm{sot}}}$. Regarding security, informally, the sum of the additional encodings $(\hat{z}_2, \hat{z}_3)$ takes the values $(0,1)$ and $(1,0)$ each with probability $1/2$, independently of c. Therefore, the simulator Sim can simulate by flipping a coin and outputting $(\hat{z}_2, \hat{z}_3) = (0,1)$ or $(1,0)$. The pair $(\hat{z}_0, \hat{z}_1)$ is then simulated according to the security proof of the ARE for f_{rot}: if $(\hat{z}_2, \hat{z}_3) = (0,1)$, then $(\hat{z}_0, \hat{z}_1)$ is simulated as $(\mathsf{Sim}(1^\lambda, m), \mathsf{Sim}(1^\lambda, R))$; otherwise, it is simulated as $(\mathsf{Sim}(1^\lambda, R), \mathsf{Sim}(1^\lambda, m))$. This shows that the resulting ARE for f_{sot} is secure.

Moreover, recalling that the above transformation is achieved by a two-fold parallel repetition of the ARE for f_{rot} together with an additional 2-bit encoding from each party, the size of the resulting ARE for f_{sot} is $2|\mathsf{ARE}_{f_{\mathrm{rot}}}| + 4$, where $|\mathsf{ARE}_{f_{\mathrm{rot}}}|$ denotes the size of the underlying ARE for f_{rot}. Therefore, by instantiating with the scheme $\mathsf{ARE}_{f_{\mathrm{rot}}}$ in Figure 3, the size of the resulting ARE for f_{sot} is $2 \cdot O(\lambda) + 4 = O(\lambda)$.

6 Conclusion

In this paper, we studied efficient constructions of AREs for the SOT function, a core building block for constructing AREs for arbitrary functions. We first presented a more efficient construction of $\mathsf{OSARE}_{f_{\mathrm{sot}}}$, improving the size of the previous construction by an $O(\lambda)$ factor without additional assumptions. We also presented a computational construction of $\mathsf{ARE}_{f_{\mathrm{sot}}}$ that achieves the same asymptotic size as the construction of Halevi et al., while relying on the Squaring DDH assumption over non-bilinear groups.

An interesting open problem is to find an efficient direct construction of 2-$\mathsf{ARE}_{f_{\mathrm{eq}}}$ based on a substantially different structure, since known direct constructions, except for special derivations via OSAREs and related notions, appear to rely on similar underlying structures.

Acknowledgments This work was supported by JSPS KAKENHI Grant Numbers JP26K02908, JP26K02910, JP26K00966, JP26H02489, JP23K17455, JP23KJ0968, JST CREST JPMJCR23M2, and the AIP Challenge Program.

References

1. Balle, B., Bell, J., Gascón, A., Nissim, K.: Private summation in the multi-message shuffle model. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. p. 657–676. CCS '20, Association for Computing Machinery, New York, NY, USA (2020)
2. Bao, F., Deng, R.H., Zhu, H.: Variations of diffie-hellman problem. In: ICICS 2003. vol. 2836, pp. 301–312 (2003)
3. Barak, B., Ong, S.J., Vadhan, S.P.: Derandomization in cryptography. In: CRYPTO 2003. vol. 2729, pp. 299–315 (2003)
4. Beimel, A., Gabizon, A., Ishai, Y., Kushilevitz, E., Meldgaard, S., Paskin-Cherniavsky, A.: Non-interactive secure multiparty computation. In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology - CRYPTO 2014. Lecture Notes in Computer Science, vol. 8617, pp. 387–404. Springer (2014)
5. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: the ACM Conference on Computer and Communications Security. pp. 784–796. ACM (2012)
6. Bitansky, N., Erabelli, S., Garg, R.: Additive randomized encodings from public key encryption. In: CRYPTO 2025 (2025)
7. Bitansky, N., Erabelli, S., Garg, R., Ishai, Y.: Shuffling is universal: Statistical additive randomized encodings for all functions. IACR Cryptol. ePrint Arch. p. 1442 (2025), https://eprint.iacr.org/2025/1442, to appear in STOC 2026.
8. Bitansky, N., Vaikuntanathan, V.: A note on perfect correctness by derandomization. In: EUROCRYPT 2017. vol. 10211, pp. 592–606 (2017)

9. Dwork, C.: Differential privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) Automata, Languages and Programming. pp. 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
10. Dwork, C., Naor, M., Reingold, O.: Immunizing encryption schemes from decryption errors. In: EUROCRYPT 2004. vol. 3027, pp. 342–360 (2004)
11. Gascón, A., Ishai, Y., Kelkar, M., Li, B., Ma, Y., Raykova, M.: Computationally secure aggregation and private information retrieval in the shuffle model. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. p. 4122–4136. CCS '24, Association for Computing Machinery, New York, NY, USA (2024)
12. Ghazi, B., Manurangsi, P., Pagh, R., Velingker, A.: Private aggregation from fewer anonymous messages. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 798–827. Springer International Publishing, Cham (2020)
13. Halevi, S., Ishai, Y., Jain, A., Kushilevitz, E., Rabin, T.: Secure multiparty computation with general interaction patterns. In: Sudan, M. (ed.) Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016. pp. 157–168. ACM (2016)
14. Halevi, S., Ishai, Y., Kushilevitz, E., Rabin, T.: Additive randomized encodings and their applications. In: CRYPTO 2023 (2023)
15. Ishai, Y., Kelkar, M., Lee, D., Ma, Y.: Information-Theoretic Single-Server PIR in the Shuffle Model. In: Aggarwal, D. (ed.) 5th Conference on Information-Theoretic Cryptography (ITC 2024). Leibniz International Proceedings in Informatics (LIPIcs), vol. 304, pp. 6:1–6:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2024)
16. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Cryptography from anonymity. In: FOCS2006 (2006)
17. Maurer, U.M., Wolf, S.: The relationship between breaking the diffie-hellman protocol and computing discrete logarithms. SIAM J. Comput. **28**(5), 1689–1721 (1999)
18. Rosulek, M., Roy, L.: Three halves make a whole? beating the half-gates lower bound for garbled circuits. In: Advances in Cryptology - CRYPTO 2021. pp. 94–124 (2021)
19. Sahai, A., Seyalioglu, H.: Worry-free encryption: functional encryption with public keys. In: Proceedings of the 17th ACM conference on Computer and communications security (2010)
20. Yao, A.C.: How to generate and exchange secrets (extended abstract). In: 27th Annual Symposium on Foundations of Computer Science (1986)
21. Zahur, S., Rosulek, M., Evans, D.: Two halves make a whole - reducing data transfer in garbled circuits using half gates. In: Advances in Cryptology - EUROCRYPT 2015. pp. 220–250 (2015)