Oblivious Transfers and Intersecting Codes

Gilles Brassard, Claude Crépeau, and Miklós Sántha

Abstract—**Assume $\mathcal{A}$ owns t secret k-bit strings. She is willing to disclose one of them to $\mathcal{B}$, at his choosing, provided he does not learn anything about the other strings. Conversely, $\mathcal{B}$ does not want $\mathcal{A}$ to learn which secret he chose to learn. A protocol for the above task is said to implement One-out-of-t String Oblivious Transfer, denoted $\binom{t}{1}$-OT$_2^k$. This primitive is particularly useful in a variety of cryptographic settings. An apparently simpler task corresponds to the case $k = 1$ and $t = 2$ of two 1-bit secrets: this is known as One-out-of-two Bit Oblivious Transfer, denoted $\binom{2}{1}$-OT$_2$. We address the question of implementing $\binom{t}{1}$-OT$_2^k$ assuming the existence of a $\binom{2}{1}$-OT$_2$. In particular, we prove that unconditionally secure $\binom{2}{1}$-OT$_2^k$ can be implemented from $\Theta(k)$ calls to $\binom{2}{1}$-OT$_2$. This is optimal up to a small multiplicative constant. Our solution is based on the notion of *self-intersecting* codes. Of independent interest, we give several efficient new constructions for such codes. Another contribution of this paper is a set of information-theoretic definitions for correctness and privacy of unconditionally secure oblivious transfer.**

Index Terms— **Cryptography, oblivious transfer, intersecting codes, algebraic-geometric codes, information theory.**

I. Introduction

The equivalence between cryptographic primitives is a major research topic [6], [11], [16], [30], [12], [25], [13], [31], [18], [15], and [17]. A large number of cryptographic protocols have been shown equivalent to one another.

One-out-of-two String Oblivious Transfer, denoted $\binom{2}{1}$-OT$_2^k$ is a primitive that originates with [43] (under the name of "multiplexing"), a paper that marked the birth of quantum cryptography. According to this primitive, one party $\mathcal{A}$ owns two secret k-bit strings w_0 and w_1, and another party $\mathcal{B}$ wants to learn w_c for a secret bit c of his choice. $\mathcal{A}$ is willing to collaborate provided that $\mathcal{B}$ does not learn any information about $w_{\bar{c}}$, but $\mathcal{B}$ will not participate if $\mathcal{A}$ can obtain information about c. Independently from [43] but inspired by [39], a natural restriction of this primitive was introduced subsequently in [20] with applications to contract signing protocols: One-out-of-two Bit Oblivious Transfer, denoted $\binom{2}{1}$-OT$_2$, concerns the case $k = 1$ in which w_0 and w_1 are single-bit secrets, generally called b_0 and b_1 in that case. In the other direction, a natural *extension* of $\binom{2}{1}$-OT$_2^k$ called ANDOS (for All-or-Nothing Disclosure of Secrets) was introduced in [6] but is denoted $\binom{t}{1}$-OT$_2^k$ in the current paper: here $\mathcal{A}$ owns t secret k-bit strings $w_0, w_1, \cdots, w_{t-1}$ and $\mathcal{B}$ wants to learn w_c for a secret integer $0 \le c < t$ of his choice. It must be impossible for $\mathcal{B}$ to obtain information on more than one w_i and for $\mathcal{A}$ to obtain information about which secret $\mathcal{B}$ learned. ANDOS has applications to mental poker [10], voting [38], zero-knowledge proofs [2], [33], [41], exchange of secrets [7], and identification [21], [17], to name just a few.

The main contribution of [6] was a reduction of $\binom{t}{1}$-OT$_2^k$ to $\binom{2}{1}$-OT$_2$, i.e., an efficient two-party protocol to achieve ANDOS based on the assumption of the existence of a protocol for the simpler type of oblivious transfer. The fact that the more general ANDOS can be reduced to $\binom{2}{1}$-OT$_2$ is not surprising because a number of authors [23], [24], [30], [12], [25], and [15], have later shown that $\binom{2}{1}$-OT$_2$ is sufficient to implement *any* two-party computation. Nevertheless, our direct reductions are interesting because of their greater efficiency. Even more efficient direct reductions of this kind were subsequently given in [19]. In the remainder of this section, we review the basic intuition behind the reductions presented in [6]. Section II defines formally the type of functions needed to achieve our goal, which we call *zigzag* functions, and it gives formal definitions for oblivious transfer based on information-theoretic considerations. Section III reduces the problem of finding efficient zigzags to the notion of self-intersecting codes [8]. Section IV surveys techniques for the construction of efficient self-intersecting codes based on our earlier work in [19]. Finally, Section V discusses open problems together with a new research direction we are currently investigating: an alternative approach based on privacy amplification [4] to the problem considered in this paper and to a natural generalization. The Appendixes contain proofs of the main theorems of this paper.

A. Oblivious String Transfer: The Basic Reduction

For any n-bit string x, let x^i denote the ith bit of x. For a set $I = \{i_1, i_2, \cdots, i_m\}$ such that $1 \le i_1 < i_2 < \cdots < i_m \le n$, we define x^I to be the concatenation $x^{i_1} x^{i_2} \cdots x^{i_m}$. Assume $\mathcal{A}$ and $\mathcal{B}$ dispose of a safe protocol to accomplish $\binom{2}{1}$-OT$_2$. They wish to perform $\binom{2}{1}$-OT$_2^k$ over the two k-bit strings w_0 and w_1. First observe that performing $\binom{2}{1}$-OT$_2$ on each pair w_0^i, w_1^i, $1 \le i \le k$, in order to implement a $\binom{2}{1}$-OT$_2^k$ fails dramatically since a cheating $\mathcal{B}$ can get w_0^1 and w_1^2 for instance, i.e., partial information about both w_0 and w_1.

Assume instead that we have a function f: $\{0, 1\}^n \to \{0, 1\}^k$ with the property that for every two input strings x_0, x_1 and every disjoint sets $I, J \subseteq \{1, 2, \cdots, n\}$, seeing

Manuscript received January 5, 1996; revised April 15, 1996. G. Brassard was supported in part by Canada's NSERC and Québec's FCAR. C. Crépeau was supported in part by Québec's FCAR. M. Sántha was supported in part by the Alexander von Humboldt Fellowship and by the ESPRIT Working Group 7097 RAND.

G. Brassard is with the Département IRO, Université de Montréal, C.P. 6128, Montréal, Qué., K3C 3J7 Canada.

C. Crépeau is with the Département IRO, Université de Montréal, C.P. 6128, Montréal, Qué., H3C 3J7 Canada, and with the École Normale Supérieure, CNRS URA 1327, Laboratoire d'Informatique de l'École Normale Supérieure, 75230 Paris Cedex 05, France.

M. Sántha is with Laboratoire de Recherche en Informatique, Université Paris-Sud, Bâtiment 490, 91405 Orsay, France.

Publisher Item Identifier S 0018-9448(96)07518-9.

the bits x_0^I and $x_1^{\bar{I}}$ releases information on at most one of $f(x_0)$ or $f(x_1)$. We could then use the following protocol to achieve $\binom{2}{1}$-OT_2^k.

Protocol 1.1: $\binom{2}{1}$-$\mathrm{OT}_2^k(w_0, w_1)(c)$

```
1) $\mathcal{A}$ picks random $x_0, x_1 \in \{0, 1\}^n$ such that
   $f(x_0) = w_0$ and $f(x_1) = w_1$.
2) $\overset{n}{\underset{i=1}{\mathbf{DO}}}$ $\mathcal{A}$ transfers $z^i \leftarrow \binom{2}{1}$-$\mathrm{OT}_2(x_0^i, x_1^i)(c)$ to $\mathcal{B}$.
3) $\mathcal{B}$ recovers $w_c$ by computing $f(z)$.
```

This protocol works since an honest $\mathcal{B}$ gets $x_c^{\{1, \cdots, n\}}$ and $x_{\bar{c}}^{\emptyset}$, which gives him full information on w_c and no information on $w_{\bar{c}}$. By the definition of f, on the other hand, a cheating $\mathcal{B}$ who gets x_0^I and $x_1^{\bar{I}}$ for some $I \neq \emptyset \neq \bar{I}$ cannot get information on both $f(x_0)$ and $f(x_1)$ simultaneously.

Based on the above protocol, one of the main purposes of this paper is to discuss efficient solutions to the problem of building such functions f.

1) A Simple Example: Consider the following function f: $\{0, 1\}^3 \rightarrow \{0, 1\}^2$ that can be used to accomplish $\binom{2}{1}$-OT_2^2 from $\binom{2}{1}$-OT_2, where $\oplus$ is used to denote the exclusive–or.

$$f(x^1, x^2, x^3) = (x^1 \oplus x^2, x^2 \oplus x^3).$$

If we use this function with the above protocol, we see that if $\mathcal{B}$ gets x_0^1, x_0^2, x_0^3 at Step 2) he will be able to compute $(w_0^1, w_0^2) = f(x_0^1, x_0^2, x_0^3)$, and, similarly if he gets x_1^1, x_1^2, x_1^3. But if he gets x_0^I and $x_1^{\bar{I}}$ for some $I \neq \emptyset \neq \bar{I}$, then one of I or $\bar{I}$ (call it Y) must be such that $\#Y = 1$. Note that the partial functions $f(x^1, *, *)$, $f(*, x^2, *)$, and $f(*, *, x^3)$ give all four possible outputs given the four possible inputs. Therefore, given only one bit of x, nothing is known about the output of f, i.e., the corresponding w.

B. All-or-Nothing Disclosure of Secrets

Assume now $\mathcal{A}$ and $\mathcal{B}$ dispose of a safe protocol to accomplish $\binom{2}{1}$-OT_2^k. They wish to perform $\binom{t}{1}$-OT_2^k over the k-bit strings $w_0, w_1, \cdots, w_{t-1}$. The idea is for $\mathcal{A}$ to choose $t-2$ random k-bit strings $x_1, x_2, \cdots, x_{t-2}$. Using $\binom{2}{1}$-OT_2^k, $\mathcal{A}$ offers $\mathcal{B}$ to choose to learn one of w_0 or x_1. Then, she offers him to choose between $w_1 \oplus x_1$ or $x_2 \oplus x_1$, and then between $w_2 \oplus x_2$ or $x_3 \oplus x_2$, and so on. (A special case is made for w_{t-1}—see the formal description of the protocol below.) If $\mathcal{B}$ wants to learn w_0, he must do so in the first instance of $\binom{2}{1}$-OT_2^k, but then all information about x_1 is lost, which precludes learning anything about the other w_i's. If $\mathcal{B}$ wants to learn w_1 instead, he must obtain x_1 in the first instance of $\binom{2}{1}$-OT_2^k—thus forsaking all information about w_0—and then $w_1 \oplus x_1$ in the second instance: this enables him to compute $w_1 = x_1 \oplus (w_1 \oplus x_1)$ but at the cost of losing all information about x_2, and therefore about all subsequent w_i's. Similarly, $\mathcal{B}$ can learn any w_i at the expense of $\mathcal{A}$'s other secrets. Clearly, $\mathcal{A}$ does not learn anything about which secret $\mathcal{B}$ chose. A formal description of the protocol follows, in which we consider Boolean expressions such as $c \neq i$ to take binary value 1 when *true* and 0 when *false*. For simplicity, from now on we denote $\vec{w} = w_0, w_1, \cdots, w_{t-1}$.

Protocol 1.2: $\binom{t}{1}$-$\mathrm{OT}_2^k(\vec{w})(c)$

```
1) $\mathcal{A}$ sets $x_0 \leftarrow 0^k$, $x_{t-1} \leftarrow w_{t-1}$ and picks random
   $x_1, \cdots, x_{t-2} \in \{0, 1\}^k$.
2) $\overset{t-2}{\underset{i=0}{\mathbf{DO}}}$ $\mathcal{A}$ transfers to $\mathcal{B}$
   $z_i \leftarrow \binom{2}{1}$-$\mathrm{OT}_2^k(w_i \oplus x_i, x_{i+1} \oplus x_i)(c \neq i)$ .
3) $\mathcal{B}$ sets $\hat{c} \leftarrow \min\{t-2, c\}$ and recovers $w_c$ by
   computing $\bigoplus_{i=0}^{\hat{c}} z_i$.
```

We formally prove in Appendix I that Protocols 1.1 and 1.2 accomplish the task they were designed for in a very strong sense defined in Section II-B.

II. Formal Definitions

In this section, we first give a formal definition for the notion of *zigzag functions*. This notion is crucial to understanding the connection between oblivious transfer and self-intersecting codes. Section II-B introduces information-theoretic definitions for *correctness* and *privacy* of a protocol for oblivious transfer. We prove in Appendix I that the protocols of Section I are correct and private implementations of their respective version of oblivious transfer.

A. Definition of Zigzag Functions

Let F be a finite set.

Definition 2.1: Consider function f: $F^n \rightarrow F^k$. A subset $I \subseteq \{1, 2, \cdots, n\}$ *biases* f if

$$\exists\, w_0, w_1 \in F^k,\ x \in F^n\ [\#\{z \in F^n | z^I = x^I, f(z) = w_0\} \neq \#\{z \in F^n | z^I = x^I, f(z) = w_1\}]. \tag{1}$$

Intuitively, I biases f if knowledge of the entries x^I of an otherwise unknown vector $x \in F^n$ may give information on $f(x)$.

Definition 2.2: An (n, k)*-zigzag* (over F) is a function f: $F^n \rightarrow F^k$ such that

$$\forall I, J \subseteq \{1, 2, \cdots, n\} [I \cap J = \emptyset \Rightarrow I \text{ or } J \text{ does not bias } f] \tag{2}$$

or equivalently

$$\forall I \subseteq \{1, 2, \cdots, n\}\ [I \text{ or } \bar{I} \text{ does not bias } f]. \tag{3}$$

Definition 2.3: Consider a function g: $\mathbb{N} \rightarrow \mathbb{N}$. A family of functions $\{f_i\}_{i \in \mathbb{N}}$ is a $g(k)$*-zigzag family* if for each integer k there exists at least one i such that f_i: $F^{g(k)} \rightarrow F^k$ is a $(g(k), k)$-zigzag. This zigzag family is *good* if $g(k) \in O(k)$ and is *efficient* if there exists an efficient algorithm $\mathcal{D}$ to find at least one such i for any given k. Moreover, $\mathcal{D}$ must produce an efficient algorithm $\mathcal{E}_i$ to compute f_i and an efficient algorithm $\mathcal{C}_i$ to produce random preimages of f_i chosen according to the uniform distribution. Thus for any $w \in F^k$ and $x \in F^{g(k)}$, x may be produced from a call on $\mathcal{C}_i(w)$ only if $f_i(x) = w$, in which case it is produced with probability $2^{k-g(k)}$. Note that algorithm $\mathcal{C}_i$ must be probabilistic for this notion to make sense. In all cases, algorithms are considered efficient provided their expected running time is upper-bounded by a polynomial in the value of k.

In general, algorithm $\mathcal{D}$ may be probabilistic: there may exist several different $(g(k), k)$-zigzags for each k and $\mathcal{D}(k)$ may produce a different one each time it is invoked. Among probabilistic zigzag families, we distinguish between Monte Carlo and Las Vegas families. A *Las Vegas* zigzag family respects the above definition unconditionally: no matter which probabilistic choices are taken by $\mathcal{D}(k)$, it is guaranteed that the resulting function is a $(g(k), k)$-zigzag. A *Monte Carlo* zigzag family allows for a small probability of failure: it may happen that a call on $\mathcal{D}(k)$ produces a function that is not a zigzag. No warning is generally given when this occurs. However, the probability of failure must be exponentially small in k.

In the special case of a *deterministic* zigzag family, for which algorithm $\mathcal{D}$ is deterministic, there is no reason for i to be different from k and we may as well require that $f_k: F^{g(k)} \to F^k$ be a $(g(k), k)$-zigzag for each k.

Our goal is to find efficient $g(k)$-zigzag families for the smallest possible function $g(k)$. Note that efficient zigzag families are precisely what is needed to implement Protocol 1.1 efficiently. Given k, which is the length of the two strings w_0 and w_1 that $\mathcal{A}$ wants to involve in the protocol, $\mathcal{A}$ and $\mathcal{B}$ use $\mathcal{D}(k)$ to agree on f_i, $\mathcal{E}_i$, and $\mathcal{C}_i$; $\mathcal{A}$ uses $\mathcal{C}_i$ to produce x_0 and x_1 from w_0 and w_1 at Step 1); and $\mathcal{B}$ uses $\mathcal{E}_i$ to compute w_c from z at Step 3).

B. Information-Theoretic Definition of Oblivious Transfer

A cryptographic protocol is a multiparty synchronous program that describes for each party the computations to be performed or the messages to be sent to some other party at each point in time. The protocol terminates when no party has any message to send or information to compute. The protocols we describe in this paper all take place between two parties $\mathcal{A}$ and $\mathcal{B}$. We denote by $\bar{\mathcal{A}}$ and $\bar{\mathcal{B}}$ the *honest* programs to be executed by $\mathcal{A}$ and $\mathcal{B}$: honest parties behave according to $\bar{\mathcal{A}}$ and $\bar{\mathcal{B}}$ and no other program. In the following definitions of *correctness* and *privacy* we also consider alternative *dishonest* programs $\tilde{\mathcal{A}}$ and $\tilde{\mathcal{B}}$ executed by $\mathcal{A}$ or $\mathcal{B}$ in an effort to obtain unauthorized information from one another. The definitions specify the result of honest parties interacting together through a specific protocol as well as the possible information leakage of an honest party facing a dishonest party. We are not concerned with the situation where both parties may be dishonest as they can do anything they like in that case; we are only concerned with protecting an honest party against a dishonest party. At the end of each execution of a protocol, each party will issue an "accept" or "reject" verdict regarding their satisfaction with the behavior of the other party. Two honest parties should always issue "accept" verdicts at the end of their interactions. An honest party will issue a "reject" verdict at the end of a protocol if he received some message from the other party of improper format or some message not satisfying certain conditions specified by the protocol. We also implicitly assume certain time limits for each party to issue messages to each other: after a specified amount of time, a party will give up interacting with the other party and issue a "reject" verdict.

As discussed in the Introduction, a $\binom{2}{1}$-OT$_2$ is a cryptographic protocol for two participants that enables a sender $\mathcal{A}$ to transfer one of two bits b_0 or b_1 to a receiver $\mathcal{B}$ who chooses secretly which bit b_c he gets. This is done in an all-or-nothing fashion, which means that $\mathcal{B}$ cannot get partial information about b_0 and b_1 at the same time, however malicious or (computationally) powerful he is, and that $\mathcal{A}$ finds out nothing about the choice c of $\mathcal{B}$. Generalization of $\binom{2}{1}$-OT$_2$ include the $\binom{2}{1}$-OT$_2^k$, in which the bits b_0 and b_1 are replaced by k-bit strings w_0 and w_1, and $\binom{t}{1}$-OT$_2^k$, in which $\mathcal{A}$ has several k-bit strings $w_0, w_1, \cdots, w_{t-1}$ from which $\mathcal{B}$ is given to choose one. The choice c is now from the set $T = \{0, 1, \cdots, t-1\}$.

Formally speaking we describe a two-party protocol that satisfies the following constraints of *correctness* and *privacy*. These notions have been defined before for general protocols by Crépeau [13], Micali and Rogaway [36], and Beaver [1] using simulators. In this paper, we use the language of information theory to express definitions similar to those introduced by Crépeau [14]. Our goal is *not* to discuss definitions for general two-party protocols: we restrict our study to oblivious transfers.

1) Correctness: Let $[P_0, P_1](a)(b)$ be the random variable (since P_0 and P_1 may be probabilistic programs) that describes the outputs obtained by $\mathcal{A}$ and $\mathcal{B}$ when they execute together the programs P_0 and P_1 on respective inputs a and b. Similarly, let $[P_0, P_1]^*(a)(b)$ be the random variable that describes the total information (including not only messages received and issued by the parties but also the result of any local random sampling they may have performed) acquired during the execution of protocol $[P_0, P_1]$ on inputs a, b. Let $[P_0, P_1]_P(a)(b)$ and $[P_0, P_1]_P^*(a)(b)$ be the marginal random variables obtained by restricting the above to only one party P. The latter is often called the *view* of P [26]. In the following definition, the equality sign $(=)$ means that the distributions on the left-hand side and the right-hand side are the same. The symbol "ϵ" stands for the empty string.

Definition 2.4 Correctness: Protocol $[\bar{\mathcal{A}}, \bar{\mathcal{B}}]$ is *correct* for $\binom{t}{1}$-OT$_2^k$ if

- $\forall \vec{w} \in F^{tk}$, $c \in T$
$$\text{Prob}\,\{[\bar{\mathcal{A}}, \bar{\mathcal{B}}](\vec{w})(c) \neq (\epsilon, w_c)\} = 0 \tag{4}$$
- for any program $\tilde{\mathcal{A}}$ there exists a probabilistic program $\tilde{\mathcal{S}}$ s.t. $\forall \vec{w} \in F^{tk}$, $c \in T$

$$([\tilde{\mathcal{A}}, \bar{\mathcal{B}}]_{\mathcal{B}}(\vec{w})(c) | \mathcal{B} \text{ accepts}) = ([\bar{\mathcal{A}}, \bar{\mathcal{B}}]_{\mathcal{B}}(\tilde{\mathcal{S}}(\vec{w}))(c) | \mathcal{B} \text{ accepts}). \tag{5}$$

Intuitively, (4) means that if the protocol is executed as described, it will accomplish the task it was designed for: $\mathcal{B}$ receives word w_c and $\mathcal{A}$ receives nothing. Condition (5) means that in situations in which $\mathcal{B}$ does not abort, $\mathcal{A}$ cannot induce a distribution on $\mathcal{B}$'s output using a dishonest $\tilde{\mathcal{A}}$ that she could not induce simply by changing the input words and then being honest (which she can always do without being detected). This second condition, called *awareness* in [36], is concerned with the future use of the outputs of a protocol. If the output of a protocol is to be used later in another protocol we wish to guarantee that the output distribution of the first protocol corresponds to the expected output distribution prescribed by the task definition. Otherwise, trouble may force a party to abort and thus disclose situational information about past executions of the protocol. Although some authors [36] insist that correctness and privacy **must** be defined together in the general case, the particular case of oblivious transfers allows for independent definitions. Here, no correctness condition involving $\tilde{\mathcal{B}}$ is necessary since $\mathcal{A}$ receives no output.

2) Privacy: Let $\vec{W} = W_0, W_1, \cdots, W_{t-1}$ and C be the random variables taking values over F^{tk} and T that describe $\mathcal{A}$'s and $\mathcal{B}$'s inputs. We assume that both $\mathcal{A}$ and $\mathcal{B}$ are aware of the joint probability distribution of these random variables $P_{\vec{W},C}$. A sample $\vec{w}$, c is generated from that distribution and $\vec{w}$ is provided as $\mathcal{A}$'s secret input while c is provided as $\mathcal{B}$'s secret input.

We assume for the next definition that the reader is familiar with the notion of *entropy* $\boldsymbol{H}(X)$ of a random variable X. The mutual *information* of two random variables X, Y is given by $\boldsymbol{I}(X; Y) = \boldsymbol{H}(X) - \boldsymbol{H}(X|Y)$ and conditioned by a third random variable Z

$$\boldsymbol{I}(X; Y|Z) = \boldsymbol{H}(X|Z) - \boldsymbol{H}(X|Y, Z).$$

Definition 2.5 Privacy: Protocol $[\mathcal{A}, \mathcal{B}]$ is *private* for $\binom{t}{1}$-OT$_2^k$ if $\forall \vec{W} \in F^{tk}$, $C \in T$

- for any program $\tilde{\mathcal{A}}$

$$\boldsymbol{I}(C; [\tilde{\mathcal{A}}, \mathcal{B}]^*_{\mathcal{A}}(\vec{W})(C)|\vec{W}) = 0 \qquad (6)$$

- for any program $\tilde{\mathcal{B}}$ there exists a random variable $\tilde{C} = \xi(C) \in T$ s.t.

$$\boldsymbol{I}(\vec{W}; [\mathcal{A}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(\vec{W})(C)|C, W_{\tilde{C}}) = 0. \qquad (7)$$

The above two conditions are designed to guarantee that each party is limited to the information he or she should get according to the honest task definition. Condition (6) means that $\tilde{\mathcal{A}}$ cannot acquire any information about C through the protocol. Condition (7) means that $\tilde{\mathcal{B}}$ may acquire information about only one of $W_0, W_1, \cdots, W_{t-1}$ through the protocol. In particular, no joint information about the t words may be obtained by the protocol. This is why our condition assumes that $\tilde{\mathcal{B}}$ is given one of the words. We do not require that $\tilde{\mathcal{B}}$ be given W_C because there is no way to prevent him from obtaining any other $W_{\tilde{C}}$ through otherwise honest use of the protocol.

One of the main results of this paper is to provide a transformation of any protocol for $\binom{2}{1}$-OT$_2$ satisfying the above constraints into a protocol for $\binom{t}{1}$-OT$_2^k$ also satisfying these constraints. Please consult Appendix I for proofs that Protocols 1.1 and 1.2 are correct and private when based on a correct and private $\binom{2}{1}$-OT$_2$.

Nevertheless, few protocols for $\binom{2}{1}$-OT$_2$ actually satisfy the above constraints *perfectly*. In general, the above constraints are only satisfied *statistically*; statistical versions of constraints (4), (6), and (7) are obtained by replacing the right-hand side zero by an exponentially decreasing function in a security parameter s and by changing the equality ($=$) in constraint (5) by statistical indistinguishability (for a precise definition of this notion, consult [26], for instance). Protocols 1.1 and 1.2 may also be used to transform a protocol for $\binom{2}{1}$-OT$_2$ satisfying statistical correctness and privacy into a protocol for $\binom{t}{1}$-OT$_2^k$ also satisfying these statistical constraints.

III. Linear Zigzags

In order to generalize the simple example of Section I-A1, we shall look at linear families of functions, i.e., functions $f\colon \mathbb{F}^n \rightarrow \mathbb{F}^k$ that are *defined* by a $k \times n$ matrix M over field $\mathbb{F}$ as $f(x) = Mx$, where both input $x \in \mathbb{F}^n$ and output $f(x) \in \mathbb{F}^k$ are considered as column vectors. For instance, the function f of Section I-A1 is defined by

$$M = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

over the field $\mathbb{F}_2$ of integers modulo 2 in which addition and multiplication correspond to the Boolean exclusive–or and conjunction, respectively.

The use of linear functions has the advantage that it is efficient to compute them and to compute random preimages of arbitrary points in $\mathbb{F}^k$. Therefore, families of linear functions are *efficient* in the sense of Definition 2.3 provided they can be constructed efficiently. In other words, there must be an efficient algorithm $\mathcal{D}$ that produces (the encoding of) a $k \times g(k)$ matrix over $\mathbb{F}$ that defines a $(g(k), k)$-zigzag for each integer k. Different matrices may be produced on different calls on $\mathcal{D}(k)$ if the zigzag family is probabilistic, and it may happen with vanishingly small probability that a call on $\mathcal{D}(k)$ produces a matrix that does *not* define a zigzag at all if the family is allowed to be Monte Carlo.

A. Matrix Characterization

Given a $k \times n$ matrix M over field $\mathbb{F}$ and integer i, $1 \leq i \leq n$, let M^i be the ith column of M. For a set of indices $I = \{i_1, i_2, \cdots, i_m\}$ such that $1 \leq i_1 < i_2 < \cdots < i_m \leq n$, we define M^I to be the matrix obtained by the concatenation of columns $M^{i_1} M^{i_2} \cdots M^{i_m}$. Remember that we defined earlier a similar notion z^I for vectors that restricts z to its components specified by I. Zigzag characterization (3) for $f(x) = Mx$ is equivalent to the characterization given by the following proposition.

Proposition 3.1: A $k \times n$ matrix M over $\mathbb{F}$ defines a linear zigzag if and only if

$$\forall I \subseteq \{1, 2, \cdots, n\} [M^I \text{ or } M^{\bar{I}} \text{ has rank } k]. \qquad (8)$$

Proof: We show that for all I, I does not bias $Mx \Leftrightarrow M^{\bar{I}}$ has rank k.

$$\forall w_0, w_1 \in \mathbb{F}^k,\ x \in \mathbb{F}^n\, [\#\{z \in \mathbb{F}^n | z^I = x^I, Mz = w_0\} = \#\{z \in \mathbb{F}^n | z^I = x^I, Mz = w_1\}]$$

k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17
$\bigcap(k)$	**1**	**3**	**6**	**9**	**13**	**15**	21	25	29	30	41	46	51	53	54	63

Fig. 1. Small values of $\bigcap(k)$. **Bold** values are known to be exact.

$$\begin{aligned}
&\Leftrightarrow \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^I z^I + M^{\bar{I}} z^{\bar{I}} = w_0\}\\
&\quad = \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^I z^I + M^{\bar{I}} z^{\bar{I}} = w_1\}\\
&\Leftrightarrow \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^I x^I + M^{\bar{I}} z^{\bar{I}} = w_0\}\\
&\quad = \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^I x^I + M^{\bar{I}} z^{\bar{I}} = w_1\}\\
&\Leftrightarrow \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^{\bar{I}} z^{\bar{I}} = w_0 - M^I x^I\}\\
&\quad = \#\{z \in \mathbb{F}^n | z^I = x^I,\ M^{\bar{I}} z^{\bar{I}} = w_1 - M^I x^I\}\\
&\Leftrightarrow \#\{z \in \mathbb{F}^n | M^{\bar{I}} z^{\bar{I}} = w_0\}\\
&\quad = \#\{z \in \mathbb{F}^n | M^{\bar{I}} z^{\bar{I}} = w_1\}\\
&\qquad \text{since } w - M^I x^I \text{ is a cyclic}\\
&\qquad \text{permutation of the elements of } \mathbb{F}^k\\
&\Leftrightarrow M^{\bar{I}} \text{ has rank } k. \qquad \blacksquare
\end{aligned}$$

B. A Trivial Lower Bound

Since our goal is to determine the smallest function g for which we can find efficient $g(k)$-zigzag families, we now give a simple lower bound.

Proposition 3.2: For any linear $g(k)$-zigzag family, it must be that $g(k) \geq 2k - 1$.

Proof: For any matrix of size $k \times n$ for $n < 2k - 1$, any subset I such that $\#I = k - 1$ cannot have rank k, nor can $\bar{I}$ since

$$\#\bar{I} = n - \#I = n - k + 1 < 2k - 1 - k + 1 = k. \qquad \blacksquare$$

Note that the bound $g(k) \geq 2k - 1$ applies for any field $\mathbb{F}$.

C. Code Characterization

We now introduce a new characterization of the matrices that define zigzag functions in terms of the words of a code generated by the rows of the matrix. For this purpose we need the following definition.

Definition: We say that two vectors $v_0, v_1 \in \mathbb{F}^n$ *intersect* if they have at least one nonzero component in common, i.e., if there exists an i, $1 \leq i \leq n$, such that $v_0^i v_1^i \neq 0$.

Proposition 3.4: A $k \times n$ matrix M satisfies characterization (8) if and only if M satisfies

$$\forall\, a, b \in \mathbb{F}^k \setminus \{0^k\}[aM \text{ and } bM \text{ intersect}]. \qquad (9)$$

Proof: We actually show $\neg(8) \Leftrightarrow \neg(9)$.

$\Rightarrow$ Suppose $\exists I$ such that M^I and $M^{\bar{I}}$ have rank less than k. Then there exists a nonzero vector a such that $aM^I = 0^{\#I}$ and a nonzero vector b such that $bM^{\bar{I}} = 0^{n-\#I}$. For those two vectors we have that aM and bM do not intersect because for all i, $(aM)^i = 0$, or $(bM)^i = 0$.

$\Leftarrow$ Suppose that $\exists a, b \in \mathbb{F}^k \setminus \{0^k\}$ such that for all i, $(aM)^i = 0$ or $(bM)^i = 0$. Let I be the set of indices such that $(aM)^i = 0$. Clearly, $aM^I = 0^{\#I}$ which implies that M^I has rank less than k. Similarly, we have that $(bM)^i = 0$ for $i \in \bar{I}$. This means that $bM^{\bar{I}} = 0^{n-\#I}$ which implies that $M^{\bar{I}}$ has rank less than k. $\blacksquare$

A Simple Family: The matrix M of Section I-A1 is the generating matrix of a [3, 2, 2] code C with the above intersecting property. Using iterated direct product on C, a family of $[3^s, 2^s, 2^s]$ *intersecting codes* C^s was obtained by Miklós [37], Cohen and Lempel [8], and Brassard, Crépeau, and Robert [6]. The following section is devoted to the general study of these codes.

IV. Intersecting Codes

Consider M as the generating matrix of a linear code. Proposition 3.4 states that M defines an (n, k)-zigzag exactly if the $[n, k, d]$ code generated by M is such that any two codewords c_1, c_2 must intersect (not counting the zero codeword). The minimal distance d of the code is uniquely defined by M but is irrelevant at this point.

Such *self-intersecting codes*[1] [8] have been studied in the past. For instance, Cohen and Lempel [8] have shown that the dual of BCH codes of length $n = 2^m - 1$ and design distance $2t + 1 < \frac{1}{3}2^{m/2} + 3$ are intersecting. Retter [40] showed that most (classical) Goppa codes (see [34]) of rate less than 0.0817 are intersecting.

Define $\bigcap(k) = \min\{n$: there exists a binary $[n, k, d]$ intersecting code. Katona and Srivastava [28] have tabulated the value of $\bigcap(k)$ for $1 \leq k \leq 5$, while Sloane [42] followed by Cohen and Zemor [9] have provided upper bounds on several extra values, as shown in Fig. 1.

Katona and Srivastava also derived a lower bound on the asymptotic behavior of $\bigcap(k)/k$ by combining the McEliece–Rodemich–Rumsey–Welch bound for binary linear codes with the simple bound obtained by observing that if M defines an $[n, k, d]$ intersecting code then $d \geq k$. A corresponding upper bound was given by Komlós (reported in [8] and [37]). These bounds are the following:

$$3.5277 < \liminf_{k\to\infty} \frac{\bigcap(k)}{k} \leq \limsup_{k\to\infty} \frac{\bigcap(k)}{k} < 4.8188.$$

These bounds imply that, asymptotically, binary $[ck, k, d]$ intersecting codes exist for $c > 4.8188$ but not for $c < 3.5277$. Nevertheless, no efficient zigzag family can be inferred from the above results because even if most codes satisfy the property, it is not clear how to obtain efficiently one that is guaranteed to satisfy it.

The current section focuses on the polynomial-time constructability of such zigzag families. First we use in Section IV-A the fact proven in Appendix II that for $\alpha = \log_{4/3} 4 \approx 4.8188$ and any $\gamma > \alpha$, a random $k \times \gamma k$ binary matrix defines a $(\gamma k, k)$-zigzag with probability asymptotically close to 1,

[1] For the remainder of this paper we omit the word "self" as no other type of intersecting codes are considered. We have considered using pairs of intersecting codes but asymptotically we get the same results.

while a random $k \times \lambda k$ binary matrix for any $\lambda < \alpha$ defines a $(\lambda k, k)$-zigzag with probability asymptotically close to 0. In both cases, the convergence is exponentially fast in k. (The first part of this result is implied by the proof of Cohen and Lempel [8] of Komlós' bound.) This yields an obvious $\Theta(k^2)$-time Monte Carlo binary γk-zigzag family for any $\gamma > \alpha$.

Then, we show that the concatenation of intersecting codes yields an intersecting code, and we use this fact in three constructions. Section IV-C applies this technique to the Monte Carlo family derived in Section IV-A to improve it to a $\Theta(k^2)$-time *Las Vegas* binary $2\gamma k$-zigzag family for any $\gamma > \alpha$. Section IV-D presents an "efficient" $O(k^{32})$-time *deterministic* binary $O(k)$-zigzag family based on the algebraic–geometric codes of Goppa [27]. Finally, Section IV-E uses the concatenation method in a construction reminiscent of that of Justesen codes to obtain an $O(k^4)$-time *deterministic* binary $O(k)$-zigzag family.

A. Monte Carlo Construction

In this section, we determine precisely the size of random matrices over $\mathbb{F}_2$ that define binary linear zigzags. If k denotes the number of rows, then there is a threshold function $t(k)$, which is linear in k, such that a random binary matrix with more than $t(k)$ columns defines a linear zigzag with high probability, whereas a random binary matrix with less than $t(k)$ columns does not define a linear zigzag, also with high probability.

Theorem 4.1: Set $\alpha = \log_{4/3} 4$, and let M be a random $k \times n$ matrix over $\mathbb{F}_2$. Then for every constant $\varepsilon > 0$, there exists a constant $0 < \eta < 1$ such that we have the following two propositions:

1) If $n \geq (1+\varepsilon)\alpha k$, then Prob $\{M$ defines a linear zigzag$\}$ $> 1 - \eta^k$, and
2) If $n \leq (1-\varepsilon)\alpha k$, then Prob $\{M$ does not define a linear zigzag$\}$ $> 1 - \eta^k$.

The proof of this theorem may be found in Appendix II. Although the first part of this result is implied by the proof of Cohen and Lempel [8] of Komlós' bound, we nevertheless include it in our proof since it is only one line.

Remark: A similar analysis for the case of matrices over $\mathbb{F}_q$ shows that for any $\varepsilon > 0$, a random $\mathbb{F}_q$ matrix of size $k \times (2+\varepsilon)k$ has asymptotic probability 1 of being a zigzag, as $q \to \infty$. This is optimal according to Proposition 3.2.

Time Complexity: As mentioned before, Proposition 1) yields an efficient Monte Carlo binary γk-zigzag family for any $\gamma > \alpha$. The running time of this construction is $\Theta(k^2)$. On the other hand, Proposition 2) shows that this bound is optimal in the sense that this technique cannot yield a binary λk-zigzag family for any $\lambda < \alpha$.

B. Intersecting Concatenated Codes

In the remainder of this section, we consider several constructions of intersecting codes based on concatenation [22] and [34]. We need the following Lemma, first used implicitly in [19]:

Lemma 4.1: Let C_o be an $[n_o, k_o, d_o]$ intersecting code over $\mathbb{F}_{q^m}$ and C_i be an $[n_i, k_i, d_i]$ intersecting code over $\mathbb{F}_q$ with $m = k_i$. The concatenated code $C = C_o \star C_i$ over $\mathbb{F}_q$ is an $[n_on_i, k_ok_i, \geq d_od_i]$ intersecting code.

Proof: The fact that the resulting code has parameters $[n_on_i, k_ok_i, \geq d_od_i]$ is well known. Consider two nonzero codewords c_0 and c_1 of the concatenated code C. By construction, both c_0 and c_1 are made of n_o blocks of n_i $\mathbb{F}_q$ symbols and must have been obtained through nonzero codewords c_0^o and c_1^o of the outer code C_o. By assumption, c_0^o and c_1^o intersect in at least one position j, thus block j of both c_0 and c_1 are nonzero codewords of the inner code C_i. By assumption, these blocks intersect and thus c_0 and c_1 intersect as well. ■

The following constructions are based on a simple observation of [19]: if M defines an $[n, k, d]$ code for some $d > n/2$ then it must intersect (by a pigeon-hole argument: any two codewords of such a code intersect). Unfortunately, for binary codes, Plotkin's bound [34] implies that $[n, k, n/2+1]$ codes can only exist for $n > 2^k$, which would result in terribly wasteful zigzags. But for larger fields we can exploit this idea and then combine it with concatenation to build intersecting codes over $\mathbb{F}_2$.

C. Las Vegas Construction

Although the result of Section IV-A implies that binary linear $(O(k), k)$-zigzags exist and can be obtained easily by picking one at random, it does not provide an efficient way of building a *guaranteed* zigzag. The problem of checking if a random matrix defines a zigzag seems rather hard: it is trivially solvable in exponential time but no polynomial-time algorithm for this problem is known. Nevertheless, using concatenation we can build from a random matrix defining a zigzag, new matrices exponentially larger with no extra effort to check if they also define a zigzag. Therefore, the exponential time necessary to check if a random matrix defines a zigzag becomes negligible with respect to the full size of the matrix. Kilian [32] inspired by the concatenation method of [19] has exploited this into a Las Vegas construction of intersecting codes that we now describe.

Construction: Consider any $\gamma > \alpha = \log_{4/3} 4$.

- Use a $[2^m, 2^{m-1}, 2^{m-1}+1]$ extended Reed–Solomon code over $\mathbb{F}_{2^m}$ as outer code C_o.
- Pick random $m \times \gamma m$ binary matrices until one is found that defines an intersecting code (this is checked by an exhaustive verification procedure) and use it as inner code C_i.

The resulting $C = C_o \star C_i$ is a $[2\gamma k, k, d]$ binary intersecting code for $k = m2^{m-1}$ and some d. Thus for any choice of $\gamma > \alpha$, this yields a $(2\gamma k, k)$-zigzag when $k = m2^{m-1}$. Although this construction is limited to values of k of that precise form, a similar construction using an incomplete outer Reed–Solomon code yields a similar expansion factor for any value of k. Details of this generalization are left to the reader.

As observed by Cohen and Lempel in [8], since the Reed–Solomon codes are minimum-distance-separable, it is impossible that similar codes with a smaller minimal distance intersect. Since the size of the random inner code is also optimal in the sense of the previous section, we conclude that this technique cannot yield $(2\lambda k, k)$-zigzag for $\lambda < \alpha$.

Time Complexity: By the result of Section IV-A, $m \times \gamma m$ matrices defining an intersecting code always exist and for sufficiently large m, at least half of these matrices have this property. Thus only a constant number of matrices must be tested on the average.

Checking whether or not a matrix M has the property requires to find all the codewords generated by M, which takes time in $O(m2^m)$, and to check that each pair of codewords intersect, which takes time in $O(m2^{2m})$. The total average running time is, therefore, in $O(k^2)$ since $k = m2^{m-1}$.

D. Deterministic Construction à la Goppa

An alternative to Reed–Solomon codes for getting outer codes with $d > n/2$, first put forward by Crépeau and Sántha [19], is to use algebraic-geometric (AG) codes of Goppa [27]. We rely on the work of Katsman, Tsfasman, and Vlăduţ [29] for their polynomial construction.

Proposition 4.2 [35]: If $q = p^{2m}$ for some prime p, it is possible to construct in polynomial time $[N, K, D]$ codes over $\mathbb{F}_q$ with parameters everywhere along the line of equation

$$\frac{K}{N} + \frac{D}{N} = 1 - \frac{1}{\sqrt{q} - 1}.$$

Corollary 4.3: For $q > 9$, some AG code of length N has minimal distance at least $N/2 + 1$.

Construction: Let $q = 2^{2m}$ for $m > 1$, and let $\Gamma_q = \frac{1}{2} - 1/(\sqrt{q} - 1)$.

- Use an $[N, \Gamma_q N, N/2 + 1]$ AG code over $\mathbb{F}_q$ for outer code C_o.
- Use any $[n_i, 2m, d_i]$ intersecting binary code as inner code C_i.

The result $C = C_o \star C_i$ is an $[n_i N, 2m\Gamma_q N, d]$ binary intersecting code with $d \geq d_i(N/2 + 1)$.

Examples: With $m = 2$ this construction yields $[27k/2, k, d]$ binary intersecting codes for $k = 2N/3$, if we use the $[9, 4, 4]$ inner code of the simple family defined in Section III-C. If, as in [19], we restrict our attention to inner codes that are members of that family, the best field is $\mathbb{F}_{256}$ with an expansion factor of 7.7885, that is $m = 4$ yielding $[405k/52, k, d]$ binary intersecting codes for $k = 52N/15$. More recently, through this same construction, Cohen and Zémor [9] have obtained a better expansion factor (6.4138) for $m = 5$ using the binary $[30, 10, 11]$ intersecting inner code of the dual BCH code type [8].

Time Complexity: This is the drawback of this approach: current constructions of AG codes require $O(k^{32})$ operations. Thus although the construction is polynomial-time, it is quite impractical.

E. Deterministic Construction à la Justesen

We now take a deterministic approach similar to the construction of Justesen [34] for efficiently constructable good families of codes. Consider for $\alpha_1, \alpha_2, \cdots, \alpha_a \in \mathbb{F}_{q^m} \setminus \{0^m\}$ the following codes of length $n = (a+1)m$ over $\mathbb{F}_q$:

$$C_{\alpha_1, \alpha_2, \cdots, \alpha_a} = \{[u, \alpha_1 u, \alpha_2 u, \cdots, \alpha_a u]: u \in \mathbb{F}_{q^m}\}.$$

Denote $h_q(x)$ the q-ary entropy function

$$h_q(x) = x \log_q (q-1) - x \log_q x - (1-x) \log_q (1-x).$$

Theorem 4.4: Some of the codes $C_{\alpha_1, \alpha_2, \cdots, \alpha_a}$ of length $n = (a+1)m$ have minimal distance larger than γn as long as $h_q(\gamma) < a/(a+1)$ and m is large enough.

Proof: Notice that if

$$(\alpha_1, \alpha_2, \cdots, \alpha_a) \neq (\alpha_1', \alpha_2', \cdots, \alpha_a')$$

then

$$C_{\alpha_1, \alpha_2, \cdots, \alpha_a} \bigcap C_{\alpha_1', \alpha_2', \cdots, \alpha_a'} = \{0^n\}$$

since any vector $[u, v_1, v_2, \cdots, v_a]$ with nonzero u uniquely defines

$$(\alpha_1, \alpha_2, \cdots, \alpha_a) = (v_1 u^{-1}, v_2 u^{-1}, \cdots, v_a u^{-1}).$$

Therefore, a given vector $[u \neq 0, v_1, v_2, \cdots, v_a]$ belongs to a single $C_{\alpha_1, \alpha_2, \cdots, \alpha_a}$. The number of nonzero words of length $n = (a+1)m$ and weight less or equal to γn is [34]

$$\sum_{i=1}^{\gamma(a+1)m} \binom{(a+1)m}{i} (q-1)^i \leq q^{(a+1)mh_q(\gamma)}.$$

Since each such word belongs to a single code, only $q^{(a+1)mh_q(\gamma)}$ of these codes may have minimal distance smaller than or equal to γn. The total number of $C_{\alpha_1, \alpha_2, \cdots, \alpha_a}$ is $(q^m - 1)^a$. Therefore, codes $C_{\alpha_1, \alpha_2, \cdots, \alpha_a}$ with minimal distance at least γn exist provided $(q^m)^{(a+1)h_q(\gamma)} < (q^m - 1)^a$. This is guaranteed whenever $h_q(\gamma) < a/(a+1)$ and m is large enough that $(q^m)^\delta < q^m - 1$, where $\delta = (a+1)h_q(\gamma)/a < 1$. ■

Corollary 4.5: From the above we get the following for large enough m:

- For $q > 60$, some $[n = 3m, m, d]$ code $C_{\alpha, \beta}$ achieves $d > n/2$.
- For $q > 3$, some $[n = 10m, m, d]$ code $C_{\alpha_1, \alpha_2, \cdots, \alpha_9}$ achieves $d > n/2$.

These codes are intersecting.

Construction: Let $q = 2^b$ for some $b > 0$ and a be such that $h_q(1/2) < a/(a+1)$. Consider any m large enough that

$$q^{(a+1)mh_q(1/2)/a} < q^m - 1.$$

Take $N = mq^m$.

- Use a $[q^m, q^m/2, q^m/2 + 1]$ extended Reed–Solomon code over $\mathbb{F}_{q^m}$ as primary outer code C_o.
- Search through all the $C_{\alpha_1, \alpha_2, \cdots, \alpha_a}$ of length $n = (a+1)m$ over $\mathbb{F}_q$ until an intersecting $[(a+1)m, m, d_i]$ code is found and use it as primary inner code C_i.
- Use any $[n_i, b, d_i]$ intersecting binary code as secondary inner code C_I.

The result $C_O = C_o \star C_i$ is an $[(a+1)N, N/2, d_O]$ intersecting code over $\mathbb{F}_q$, while $C = C_O \star C_I$ is a binary $[(a+1)n_iN, bN/2, d]$ intersecting code for some d_O and d.

Examples:

- For $b = 6$ we get that C_o is a $[64^m, 64^m/2, 64^m/2+1]$ code, C_i a $[3m, m, d_i]$ code, and C_I a $[15, 6, 6]$ code, yielding a $[15k, k, d]$ binary intersecting code, for $k = 3m64^m$.
- For $b = 2$ we get that C_o is a $[4^m, 4^m/2, 4^m/2+1]$ code, C_i a $[10m, m, d_i]$ code, and C_I a $[3, 2, 2]$ code, yielding a $[30k, k, d]$ binary intersecting code, for $k = m4^m$.

As in Section IV-C, a construction using an incomplete outer Reed–Solomon code yields a similar expansion factor for any value of k. Details of this generalization are left to the reader.

Time Complexity: The first example above is the fastest deterministic construction known to the authors of a *good* family of intersecting codes and thus of a *good* zigzag family. For $k = 3m64^m$, in the worst case only $(64^m-1)^2 < (k/\lg k)^2$ codes $C_{\alpha,\beta}$ will be tested for self-intersection. Testing each such code requires comparing 64^{2m} pairs of codewords of length $3m$. This requires $O(k^2/\lg k)$ operations. Therefore, the total running time is in $O(k^4/(\lg k)^3)$ in the worse case.

V. Ongoing Research and Open Questions

We have shown how to construct linear-size zigzags both by probabilistic and deterministic polynomial-time methods. The exact complexity of the decision problem "Given a matrix M, is it the generator of a zigzag?" still has to be determined (the best we can say is that it is in co-NP). Another open problem is to construct over $\mathbb{F}_q$ some zigzag that will do better than the asymptotic bounds of Section IV-A. Finally, we ask if nonlinear functions can generate smaller zigzags than linear functions.

An important fact about the method based on zigzag functions considered in this paper is that, by definition of the zigzag, there is no way for $\mathcal{B}$ to learn information about both w_0 and w_1 even though the zigzag function is known *before* he gets to choose which bits to obtain through the $\binom{2}{1}$-OT_2 instances in Protocol 1.1 (unless a Monte Carlo zigzag family is used). We are currently investigating [5] another approach to the problem of reducing $\binom{2}{1}$-OT_2^k to $\binom{2}{1}$-OT_2, in which $\mathcal{A}$ does not reveal the function to $\mathcal{B}$ until *after* the necessary $\binom{2}{1}$-OT_2's have been performed.

Our new approach is based on *privacy amplification*, a technique invented in [4] and refined in [3]. Assume $\mathcal{A}$ knows a random n-bit string x about which $\mathcal{B}$ has partial information. Privacy amplification allows $\mathcal{A}$ to shrink x to a shorter string y about which $\mathcal{B}$ has an arbitrarily small amount of information even if he knows the recipe used by $\mathcal{A}$ to transform x into y. Intuitively, this can be used to implement $\binom{2}{1}$-$\mathrm{OT}_2^k(w_0, w_1)(c)$ from $\binom{2}{1}$-OT_2 because $\mathcal{A}$ can offer $\mathcal{B}$ to read one of two random strings x_0 or x_1 by a simple sequence of $\binom{2}{1}$-$\mathrm{OT}_2(x_0^i, x_1^i)(c_i)$. Subsequently, $\mathcal{A}$ tells $\mathcal{B}$ how to transform x_0 into w_0 and x_1 into w_1 by way of privacy amplification. An honest $\mathcal{B}$ who accessed all the bits of x_c can reconstruct w_c from this information. But a dishonest $\mathcal{B}$ who accessed some of the bits of x_0 and some of the bits of x_1 will not have enough information on at least one of them to infer any information on the corresponding w or even joint information on both w_0 and w_1.

Privacy amplification allows for a protocol that is simpler, more general, and more efficient than the zigzag-based solution investigated in this paper, but at the cost of a vanishingly small failure probability. More specifically, $2k+s$ instances of $\binom{2}{1}$-OT_2 are sufficient to implement $\binom{2}{1}$-OT_2^k so that the probability that a cheating $\mathcal{B}$ may learn information on both strings is exponentially small in s. This is significantly better than all the methods based on zigzag functions provided a probability of failure is tolerable. Moreover, it allows the implementation of $\binom{2}{1}$-OT_2^k at no extra cost if the underlying $\binom{2}{1}$-OT_2 goes in the other direction, i.e., from $\mathcal{B}$ to $\mathcal{A}$, or if it permits $\mathcal{B}$ to choose not only one bit or the other, but also their exclusive–or. A drawback of this approach is that a new function must be generated and transmitted at each run of the protocol. We postpone our detailed exposition of this alternative technique because our research is still ongoing [5]. In particular, we wish to investigate the extent of its generality.

Appendix I
Proofs of Correctness and Privacy

Both Protocols 1.1 and 1.2 were described for the specific set $F = \{0, 1\}$. As a matter of fact, they would work equally well for any finite set F. Thus in the proofs we do not consider any particular F.

A. Protocol 1.1

We show that given that f is an (n, k)-*zigzag*, Protocol 1.1 is correct and private. We assume the existence of a correct and private subprotocol $[\mathcal{U}, \mathcal{V}]$ for $\binom{2}{1}$-OT_2.

Theorem A1: Protocol 1.1 is correct.

Proof:

Condition (4): Since $[\mathcal{U}, \mathcal{V}]$ is correct for $\binom{2}{1}$-OT_2 by assumption, $\mathcal{B}$ gets the desired values x_c^i at Step 2). By definition of x_0, x_1 at Step 1), it is clear that the value computed at Step 3) is indeed $w_c = f(x_c)$.

Condition (5): By assumption $[\mathcal{U}, \mathcal{V}]$ is correct for $\binom{2}{1}$-OT_2 and therefore for any program $\tilde{\mathcal{U}}_i$ there exists a probabilistic program $\tilde{\mathcal{S}}_i$ s.t. $\forall b_0^i, b_1^i \in F, c \in \{0, 1\}$

$$([\tilde{\mathcal{U}}_i, \mathcal{V}]_{\mathcal{B}}(b_0^i, b_1^i)(c) | \mathcal{B} \text{ accepts}) = ([\mathcal{U}, \mathcal{V}]_{\mathcal{B}}(\tilde{\mathcal{S}}_i(b_0^i, b_1^i))(c) | \mathcal{B} \text{ accepts}). \quad (10)$$

We now describe the program for $\tilde{\mathcal{S}}$ as a function of $\tilde{\mathcal{A}}$. On input (w_0, w_1), for $1 \le i \le n$, run protocol $[\tilde{\mathcal{A}}, \bar{\mathcal{B}}]$ on inputs $(w_0, w_1)(0)$ until the ith execution of $\binom{2}{1}$-OT_2. Call $\tilde{\mathcal{U}}_i$ the behavior of $\tilde{\mathcal{A}}$ from that point on until the end of that execution. Let b_0^i, b_1^i be the inputs to $\binom{2}{1}$-OT_2 used by $\tilde{\mathcal{A}}$ if such inputs exist and otherwise let b_0^i, b_1^i be anything as they are irrelevant anyway. By assumption, there exists $\tilde{\mathcal{S}}_i$ that satisfies (10). $\tilde{\mathcal{S}}$ sets $(z_0^i, z_1^i) \leftarrow \tilde{\mathcal{S}}_i(b_0^i, b_1^i)$. If at any point $\bar{\mathcal{B}}$ aborts then $\tilde{\mathcal{S}}$ aborts as well. Finally, $\tilde{\mathcal{S}}$ returns $f(z_0), f(z_1)$. If $\mathcal{A}$ runs $\tilde{\mathcal{S}}$ and then $\bar{\mathcal{A}}$ instead of running $\tilde{\mathcal{A}}$ then $\mathcal{B}$ will see the same output distribution. ■

Theorem A2: Protocol 1.1 is private.

Proof: First notice that neither Step 1) nor Step 3) involve information transfers from either parties. Thus the only possible step in which $\mathcal{A}$ or $\mathcal{B}$ could learn something about their respective inputs is Step 2).

Condition (6): By assumption $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ is private for $\binom{1}{2}$-OT$_2$ and thus $\forall X_0^i, X_1^i \in F$, $C \in \{0, 1\}$ and any program $\tilde{\mathcal{U}}$

$$\boldsymbol{I}(C; [\tilde{\mathcal{U}}, \bar{\mathcal{V}}]^*_{\mathcal{A}}(X_0^i, X_1^i)(C)|X_0^i, X_1^i) = 0.$$

Therefore, $\mathcal{A}$ learns nothing about C through any of the iterations of Step 2) either. Which means that $\forall W_0, W_1 \in F^k$, $C \in \{0, 1\}$ and for any program $\tilde{\mathcal{A}}$

$$\boldsymbol{I}(C; [\tilde{\mathcal{A}}, \bar{\mathcal{B}}]^*_{\mathcal{A}}(W_0, W_1)(C)|W_0, W_1) = 0.$$

Condition (7): By assumption $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ is private for $\binom{2}{1}$-OT$_2$ and therefore for each i, $\forall X_0^i, X_1^i \in F$, $C \in \{0, 1\}$ and for any program $\tilde{\mathcal{V}}$ there exists a random variable $\tilde{C}_i = \xi_i(C) \in \{0, 1\}$ s.t.

$$\boldsymbol{I}((X_0^i, X_1^i); [\bar{\mathcal{U}}, \tilde{\mathcal{V}}]^*_{\mathcal{B}}(X_0^i, X_1^i)(C)|C, X^i_{\tilde{C}_i}) = 0$$

and therefore

$$\boldsymbol{I}((X_0, X_1);\\ [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|C, X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n}) = 0$$

which implies

$$\boldsymbol{I}((W_0, W_1);\\ [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|C, X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n}) = 0$$

which is the same as

$$\begin{aligned}&\boldsymbol{H}((W_0, W_1)|[\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C),\\ &\quad C, X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n})\\ &\qquad = \boldsymbol{H}((W_0, W_1)|C, X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n}). \quad (11)\end{aligned}$$

From here on, we omit C as everything is conditioned by it. Let $\tilde{C}$ be the random variable such that

$$\tilde{C} = \begin{cases} 0, & \text{if } \{i\colon \tilde{C}_i = 1\} \text{ does not bias } f\\ 1, & \text{otherwise.}\end{cases}$$

Since f is a zigzag only $W_{\tilde{C}}$ is biased by the $X^i_{\tilde{C}_i}$'s and thus there exists a random variable $\hat{W}_{\tilde{C}} = \Phi(W_{\tilde{C}})$ such that

$$\boldsymbol{H}((W_0, W_1)|X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n}) = \boldsymbol{H}((W_0, W_1)|\hat{W}_{\tilde{C}}). \quad (12)$$

Now suppose

$$\boldsymbol{H}((W_0, W_1)|[\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}})\\ < \boldsymbol{H}((W_0, W_1)|\hat{W}_{\tilde{C}}).$$

This would imply by (11) and (12)

$$\boldsymbol{H}((W_0, W_1)|[\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}})\\ < \boldsymbol{H}((W_0, W_1)|[\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C),\\ X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n})$$

which is impossible since $\hat{W}_{\tilde{C}}$ can be deduced from $X^1_{\tilde{C}_1}, X^2_{\tilde{C}_2}, \cdots, X^n_{\tilde{C}_n}$. Thus we also have

$$\boldsymbol{H}((W_0, W_1)|[\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}})\\ = \boldsymbol{H}((W_0, W_1)|\hat{W}_{\tilde{C}}).$$

Now using on both sides of the equality a property of the entropy function

$$\boldsymbol{H}(X|Y) = \boldsymbol{H}(Y|X) + \boldsymbol{H}(X) - \boldsymbol{H}(Y)$$

we obtain

$$\begin{aligned}&\boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}}|(W_0, W_1))\\ &\quad - \boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}}) + \boldsymbol{H}((W_0, W_1))\\ &\qquad = \boldsymbol{H}(\hat{W}_{\tilde{C}}|(W_0, W_1)) - \boldsymbol{H}(\hat{W}_{\tilde{C}}) + \boldsymbol{H}((W_0, W_1))\end{aligned}$$

and thus

$$\boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|(W_0, W_1))\\ = \boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C), \hat{W}_{\tilde{C}}) - \boldsymbol{H}(\hat{W}_{\tilde{C}}).$$

But by definition of the right-hand side

$$\boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|(W_0, W_1))\\ = \boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|\hat{W}_{\tilde{C}}).$$

This clearly implies that the same is true for $W_{\tilde{C}}$

$$\boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|(W_0, W_1))\\ = \boldsymbol{H}([\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|W_{\tilde{C}})$$

leading by inversion of the previous steps to the result

$$\boldsymbol{I}((W_0, W_1); [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]^*_{\mathcal{B}}(W_0, W_1)(C)|C, W_{\tilde{C}}) = 0. \qquad \blacksquare$$

B. Protocol 1.2

We assume the existence of a correct and private subprotocol $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ for $\binom{2}{1}$-OT$_2^k$.

Theorem A3: Protocol 1.2 is correct.

Proof:

Condition (4): Let $\bar{\mathcal{A}}$ and $\bar{\mathcal{B}}$ be as in Protocol 1.2. From the description of the protocol we find

$$[\bar{\mathcal{A}}, \bar{\mathcal{B}}](w_0, w_1, \cdots, w_{t-1})(c) = \left(\epsilon, \bigoplus_{i=0}^{\hat{c}} z_i\right).$$

Thus we have to show $\bigoplus_{i=0}^{\hat{c}} z_i = w_c$. Remember from the protocol that

$$z_i = \begin{cases} w_i \oplus x_i, & \text{if } c = i\\ x_{i+1} \oplus x_i, & \text{if } c \neq i.\end{cases}$$

First, consider the case $c = \hat{c} < t - 1$.

$$\begin{aligned}\bigoplus_{i=0}^{\hat{c}} z_i = \bigoplus_{i=0}^{c} z_i\\ &= z_c \oplus z_{c-1} \oplus \cdots \oplus z_0\\ &= w_c \oplus x_c \oplus x_c \oplus x_{c-1} \oplus \cdots \oplus x_2 \oplus x_1 \oplus x_1 \oplus x_0\\ &= w_c \oplus x_0\\ &= w_c.\end{aligned}$$

Second, consider the case $c = t - 1$.

$$\begin{aligned}\bigoplus_{i=0}^{\hat{c}} z_i = \bigoplus_{i=0}^{t-2} z_i &= x_{t-1} \oplus x_{t-2} \oplus x_{t-2} \oplus x_{t-3} \oplus \cdots \oplus x_2 \oplus x_1\\ &\quad \oplus x_1 \oplus x_0\\ &= x_{t-1} \oplus x_0\\ &= w_{t-1}\\ &= w_c.\end{aligned}$$

Therefore, we find

$$[\bar{\mathcal{A}}, \bar{\mathcal{B}}](w_0, w_1, \cdots, w_{t-1})(c) = (\epsilon, w_c)$$

as required.

Condition (5): By assumption $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ is correct for $\binom{2}{1}$-OT_2^k and therefore for any program $\tilde{\mathcal{U}}_i$ there exists a probabilistic program $\tilde{\mathcal{S}}_i$ s.t. $\forall\, y_0^i, y_1^i \in F^k, e \in \{0, 1\}$

$$\begin{aligned}&([\tilde{\mathcal{U}}_i, \bar{\mathcal{V}}]_{\mathcal{B}}(y_0^i, y_1^i)(e) | \mathcal{B} \text{ accepts})\\ &\qquad = ([\bar{\mathcal{U}}, \bar{\mathcal{V}}]_{\mathcal{B}}(\tilde{\mathcal{S}}_i(y_0^i, y_1^i))(e) | \mathcal{B} \text{ accepts}). \quad (13)\end{aligned}$$

We now describe the program for $\tilde{\mathcal{S}}$ as a function of $\tilde{\mathcal{A}}$. On input $\vec{w}$, for $0 \leq i \leq t-2$, run protocol $[\tilde{\mathcal{A}}, \bar{\mathcal{B}}]$ on inputs $(\vec{w})(0)$ until the ith execution of $\binom{2}{1}$-OT_2^k. Call $\tilde{\mathcal{U}}_i$ the behavior of $\tilde{\mathcal{A}}$ from that point on until the end of that execution. Let y_0^i, y_1^i be the inputs to $\binom{2}{1}$-OT_2^k used by $\tilde{\mathcal{A}}$ if such inputs exist and otherwise let y_0^i, y_1^i be anything as they are irrelevant anyway. By assumption, there exists $\tilde{\mathcal{S}}_i$ that satisfies (13). $\tilde{\mathcal{S}}$ sets $(z_0^i, z_1^i) \leftarrow \tilde{\mathcal{S}}_i(y_0^i, y_1^i)$. If at any point $\bar{\mathcal{B}}$ aborts then $\tilde{\mathcal{S}}$ aborts as well. Finally, $\tilde{\mathcal{S}}$ returns z_0^0, $z_1^0 \oplus z_0^1$, $z_1^1 \oplus z_0^2, \cdots, z_1^{t-3} \oplus z_0^{t-2}$, $z_1^{t-2} \oplus z_0^{t-1}$, $z_1^{t-2} \oplus z_1^{t-1}$. If $\mathcal{A}$ runs $\tilde{\mathcal{S}}$ and then $\bar{\mathcal{A}}$ instead of running $\tilde{\mathcal{A}}$ then $\mathcal{B}$ will see the same output distribution. ∎

Theorem A4: Protocol 1.2 is private.

Proof: As for Protocol 1.1 neither Step 1) nor Step 3) involve information transfers from either parties. Thus the only possible step in which $\mathcal{A}$ or $\mathcal{B}$ could learn something about their respective inputs is Step 2).

Condition (6): By assumption $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ is private for $\binom{2}{1}$-OT_2^k and thus $\forall\, Y_0^i, Y_1^i \in F^k$, $C \in T$ and any program $\tilde{\mathcal{U}}$

$$\boldsymbol{I}(C; [\tilde{\mathcal{U}}, \bar{\mathcal{V}}]_{\mathcal{A}}^*(Y_0^i, Y_1^i)(C \neq i) | Y_0^i, Y_1^i) = 0.$$

Therefore, $\mathcal{A}$ learns nothing about C through any of the iterations of Step 2) either. Which means that $\forall\, \vec{W} \in F^{kt}$, $C \in T$ and for any program $\tilde{\mathcal{A}}$

$$\boldsymbol{I}(C; [\tilde{\mathcal{A}}, \bar{\mathcal{B}}]_{\mathcal{A}}^*(\vec{W})(C) | \vec{W}) = 0.$$

Condition (7): By assumption $[\bar{\mathcal{U}}, \bar{\mathcal{V}}]$ is private for $\binom{2}{1}$-OT_2^k and, therefore, for each i, $\forall\, Y_0^i, Y_1^i \in F^k$, $C \in T$ and for any program $\tilde{\mathcal{V}}$ there exists a random variable $\tilde{C}_i = \xi_i(C) \in \{0, 1\}$ s.t.

$$\boldsymbol{I}((Y_0^i, Y_1^i); [\bar{\mathcal{U}}, \tilde{\mathcal{V}}]_{\mathcal{B}}^*(Y_0^i, Y_1^i)(C \neq i) | C, Y_{\tilde{C}_i}^i) = 0$$

where

$$Y_{\tilde{C}_i}^i = \begin{cases} W_i \oplus X_i, & \text{if } \tilde{C}_i = 0\\ X_{i+1} \oplus X_i, & \text{if } \tilde{C}_i = 1\end{cases}$$

and, therefore,

$$\boldsymbol{I}((\vec{W}, \vec{X}); [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C) | C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2}) = 0$$

which means

$$\boldsymbol{I}(\vec{W}; [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C) | C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2}) = 0$$

which is the same as

$$\begin{aligned}&\boldsymbol{H}(\vec{W} | [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C), C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2})\\ &\qquad = \boldsymbol{H}(\vec{W} | C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2}). \quad (14)\end{aligned}$$

Let $\tilde{C}$ be the random variable such that

$$\tilde{C} = \min_i \{\tilde{C}_i = 0 \quad \text{or} \quad i = t-1\}.$$

By definition of the $Y_{\tilde{C}_i}^i$'s and because of the fact that the X_i's are uniformly selected at random we get

$$\begin{aligned}&\boldsymbol{H}(\vec{W} | C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2})\\ &\quad = \boldsymbol{H}(\vec{W} | C, X_0, X_1, \cdots, X_{\tilde{C}}, W_{\tilde{C}}, Y_{\tilde{C}+1}, \cdots, Y_{\tilde{C}_{t-2}}^{t-2})\\ &\quad = \boldsymbol{H}(\vec{W} | C, W_{\tilde{C}}). \quad (15)\end{aligned}$$

Now suppose

$$\boldsymbol{H}(\vec{W} | [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C), C, W_{\tilde{C}}) < \boldsymbol{H}(\vec{W} | C, W_{\tilde{C}}).$$

This would imply by (14) and (15)

$$\begin{aligned}&\boldsymbol{H}(\vec{W} | [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C), C, W_{\tilde{C}})\\ &\qquad < \boldsymbol{H}(\vec{W} | [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C), C, Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2})\end{aligned}$$

which is impossible since $W_{\tilde{C}}$ can be deduced from

$$Y_{\tilde{C}_1}^1, Y_{\tilde{C}_2}^2, \cdots, Y_{\tilde{C}_{t-2}}^{t-2}.$$

Thus we also have

$$\boldsymbol{H}(\vec{W} | [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C), C, W_{\tilde{C}}) = \boldsymbol{H}(\vec{W} | C, W_{\tilde{C}})$$

or equivalently

$$\boldsymbol{I}(\vec{W}; [\bar{\mathcal{A}}, \tilde{\mathcal{B}}]_{\mathcal{B}}^*(\vec{W})(C) | C, W_{\tilde{C}}) = 0. \quad ∎$$

Appendix II
Proof of Theorem 4.1

Theorem 4.1: Set $\alpha = \log_{4/3} 4$, and let M be a random $k \times n$ matrix over $\mathbb{F}_2$. Then for every constant $\varepsilon > 0$, there exists a constant $0 < \eta < 1$ such that we have the following two propositions:

1) If $n \geq (1+\varepsilon)\alpha k$, then Prob $\{M$ defines a linear zigzag$\}$ $> 1 - \eta^k$, and
2) If $n \leq (1-\varepsilon)\alpha k$, then Prob $\{M$ does *not* define a linear zigzag$\}$ $> 1 - \eta^k$.

Proof: Let $a \neq b$ be two nonzero binary row vectors of length k, and set $S = \{a, b\}$. Clearly, M defines a linear zigzag if and only if for every such S, aM, and bM intersect. Let D_S be the event that aM and bM do not intersect, and let X_S be the associated indicator random variable. As every bit in aM and bM is 1 with probability $1/2$, we have

$$E[X_S] = \text{Prob}\,\{D_S\} = \left(\frac{3}{4}\right)^n.$$

Let us define the random variable $X = \sum_S X_S$. Then M defines a linear zigzag if and only if $X = 0$. By linearity of the expectation,

$$E[X] = \binom{2^k - 1}{2} \left(\frac{3}{4}\right)^n.$$

Proposition 1: $n \geq (1 + \varepsilon)\alpha k$. Then we have

$$\text{Prob}\,\{X > 0\} \leq E[X] < 2^{2k} \left(\frac{3}{4}\right)^n \leq \left(\frac{1}{4}\right)^{\varepsilon k}.$$

Proposition 2: $n \leq (1 - \varepsilon)\alpha k$. Then using Chebishev's inequality, we have:

$$\text{Prob}\,\{X = 0\} \leq \text{Prob}\,\{|X - E[X]| \geq E[X]\} \leq \frac{\text{Var}\,(X)}{E[X]^2}.$$

We will show that $\text{Var}\,(X)$ is exponentially small compared to $E[X]^2$. Let $a_1 \neq b_1$ and $a_2 \neq b_2$ be nonzero row vectors of length k, and set $S_1 = \{a_1, b_1\}$ and $S_2 = \{a_2, b_2\}$. Since X_{S_1} and X_{S_2} are 0–1 random variables, we have:

$$\begin{aligned} \text{Var}(X) &= \sum_{S_1, S_2} \text{cov}\,(X_{S_1}, X_{S_2}) \\ &= \sum_{S_1, S_2} E[X_{S_1}](E[X_{S_2}|X_{S_1} = 1] - E[X_{S_2}]). \end{aligned}$$

If the random variables X_{S_2} and X_{S_1} are not independent, then we will bound $\text{cov}\,(X_{S_1}, X_{S_2})$ from above by $E[X_{S_1}]E[X_{S_2}|X_{S_1} = 1]$. If X_{S_2} and X_{S_1} are independent, then $\text{cov}\,(X_{S_1}, X_{S_2}) = 0$. The proof works out because for most S_1 and S_2 they are indeed independent. We will prove this with the help of the following lemma:

Lemma A1: Let $M = (m_i^j)$ be a random $k \times n$ binary matrix. If for some $t > 0$, $\{a_1, \cdots, a_t\}$ is a linearly independent family of vectors of length k, then $a_1 M, \cdots, a_m M$ are independent random variables.

Proof: Let us fix t binary vectors $\{b_1, \cdots, b_t\}$ of length n. Let A be the $t \times k$ matrix whose ith row is a_i, and similarly let B be the $t \times n$ matrix whose ith row is b_i. We will show that

$$\text{Prob}\,\{AM = B\} = 2^{-tn}.$$

We can suppose without loss of generality that the first t columns of A have rank t. Let us choose anyhow $c_i^j \in \mathbb{F}_2$ for $i = t+1, \cdots, k$ and $j = 1, \cdots, n$, and let us fix $m_i^j = c_i^j$. Let $\hat{A}$ be the truncation of A to its first t columns, and let $\hat{M}$ be the truncation of M to its first t rows. Finally, let $\hat{B} = (\hat{b}_i^j)$ be the $t \times n$ matrix where

$$\hat{b}_i^j = b_i^j + \sum_{l=t+1}^{k} a_i^l c_l^j.$$

The matrix $\hat{A}$ is regular; therefore, we have

$$\text{Prob}\,\{\hat{A}\hat{M} = \hat{B}\} = 2^{-tn}.$$

This implies the result since this is true for every choice of c_i^j. ■

It follows that the random variables X_{S_1} and X_{S_2} are independent when the family of vectors $\mathcal{U} = \{a_1, b_1, a_2, b_2\}$ is linearly independent. Thus we have to consider the covariances of only those random variables X_{S_1} and X_{S_2} where there is some linear dependence among the vectors of $\mathcal{U}$. We will distinguish two cases according to the rank of $\mathcal{U}$.

Case 1: $\text{rank}\,(\mathcal{U}) = 3$. Then there exist coefficients α_1, β_1, α_2, $\beta_2 \in \mathbb{F}_2$, not all 0, such that

$$\alpha_1 a_1 + \beta_1 b_1 + \alpha_2 a_2 + \beta_2 b_2 = 0.$$

The number of such families is $O(2^{3k})$. Since $a_1 \neq b_1$, $a_2 \neq b_2$, and the elements of $\mathcal{U}$ are nonzero vectors, we can suppose without loss of generality that $\alpha_1 = \alpha_2 = 1$. Therefore, we are left with the following possible dependencies among the members of $\mathcal{U}$: $a_2 = a_1$ or $a_2 = a_1 + b_1$ or $a_2 = a_1 + b_2$ or $a_2 = a_1 + b_1 + b_2$.

Let us suppose that $a_1 M$ and $b_1 M$ do not intersect. Then we claim that in case of any of the above dependencies, the probability that $a_2 M$ and $b_2 M$ do not intersect either is at most $(\frac{5}{6})^n$. The proper analysis is quite similar in the four cases, let us consider here in details, for example, the case when $a_2 = a_1$. Let $1 \leq i \leq n$ be an index. The string $a_1^i b_1^i$ with probability $1/3$ takes each of the values 00, 01, and 10. Since $b_2^i = 1$ with probability $1/2$ independently from the value of $a_1^i b_1^i$, we have

$$\text{Prob}\,\{a_2^i b_2^i = 11 | a_1^i b_1^i \neq 11\} = \frac{1}{3} \times 0 + \frac{1}{3} \times 0 + \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}.$$

This is true independently for every i, and the claim follows. Therefore, the total contribution to the variance of these families with respect to $E[X]^2$ is

$$O\left(\frac{2^{3k}(\frac{3}{4})^n(\frac{5}{6})^n}{E[X]^2}\right) = O\left(\left(\frac{(\frac{10}{9})^\alpha}{2}\right)^k\right).$$

Case 2: $\text{rank}\,(\mathcal{U}) = 2$. Then $S_1 = S_2$, and the number of such families is $\binom{2^k - 1}{2}$. In this case

$$E[X_{S_1}]E[X_{S_2}|X_{S_1} = 1] = E[X_{S_1}]$$

and the total contribution to the variance with respect to $E[X]^2$ is at most

$$\frac{1}{E[X]} = O\left(\left(\tfrac{1}{4}\right)^{\varepsilon k}\right).$$ ■

Acknowledgment

The authors wish to thank P. Barbaroux, J. Buchmann, I. B. Damgård, W. Fernandez De La Vega, A. Joux, L. Salvail, and J. Stern for their help, comments, and suggestions. They especially wish to thank J.-M. Robert for his earlier collaboration on this topic, O. Goldreich and S. Micali for suggesting characterization (8), J. Kilian for the construction from Section IV-C, and D. Mayers and J. van de Graaf for helpful discussions on the definitions of Section II-B. C. Crépeau wishes to thank S. Goldwasser for introducing him to Goppa codes, P. Elias for providing an extensive bibliography on the topic, and G. Cohen for valuable information about his earlier research on intersecting codes.

References

[1] D. Beaver, "Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority," *J. Cryptol.*, vol. 4, no. 2, pp. 75–122, 1991.

[2] M. Bellare and S. Micali, "Non-interactive oblivious transfer and applications," in *Advances in Cryptology: Crypto '89 Proc.* Berlin: Springer-Verlag, 1990, pp. 547–559.

[3] C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Maurer, "Generalized privacy amplification," *IEEE Trans. Inform. Theory*, vol. 41, no. 6, pp. 1915–1923, Nov. 1995.

[4] C. H. Bennett, G. Brassard, and J.-M. Robert, "Privacy amplification by public discussion," *SIAM J. Comput.*, vol. 17, no. 2, pp. 210–229, Apr. 1988.

[5] G. Brassard and C. Crépeau, "Oblivious transfers and privacy amplification," unpublished manuscript, 1996.

[6] G. Brassard, C. Crépeau, and J.-M. Robert, "Information theoretic reductions among disclosure problems," in *Proc. 27th Annu. IEEE Symp. on Foundations of Computer Science*, 1986, pp. 168–173.

[7] R. Cleve, "Controlled gradual disclosure schemes for random bits and their applications," in *Advances in Cryptology: Crypto '89 Proc.* Berlin: Springer-Verlag, 1990, pp. 573–588.

[8] G. Cohen and A. Lempel, "Linear intersecting codes," *Discr. Math.*, vol. 56, pp. 35–43, 1985.

[9] G. Cohen and G. Zémor, "Intersecting codes and independent families," *IEEE Trans. Inform. Theory*, vol. 40, no. 6, pp. 1872–1881, Nov. 1994.

[10] C. Crépeau, "A zero-knowledge poker protocol that achieves confidentiality of the players' strategy or how to achieve an electronic poker face," in *Advances in Cryptology: Crypto '86 Proc.* Berlin: Springer-Verlag, 1987, pp. 239–247.

[11] ———, "Equivalence between two flavors of oblivious transfers (abstract)," in *Advances in Cryptology: Crypto '87 Proc.* Berlin, Germany: Springer-Verlag, 1988, pp. 350–354.

[12] ———, "Verifiable disclosure of secrets and applications," in *Advances in Cryptology: Eurocrypt '89 Proc.* Berlin: Springer-Verlag, 1990, pp. 150–154.

[13] ———, "Correct and private reductions among oblivious transfers," Ph.D. dissertation, Dept. Elec. Engi. and Comput. Sci., MIT, Cambridge, MA, 1990, supervised by S. Micali.

[14] ———, "Quantum oblivious transfer," *J. Mod. Opt.*, vol. 41, no. 12, pp. 2445–2454, Dec. 1994.

[15] C. Crépeau, J. van de Graaf, and A. Tapp, "Committed oblivious transfer and private multi-party computations," in *Advances in Cryptology: Crypto '95 Proc.* Berlin: Springer-Verlag, 1995, pp. 110–123.

[16] C. Crépeau and J. Kilian, "Achieving oblivious transfer using weakened security assumptions," in *Proc. 29th Annu. IEEE Symp. on Foundations of Computer Science* 1988, pp. 42–52.

[17] C. Crépeau and L. Salvail, "Oblivious verification of common string," *CWI Quart.*, vol. 8, no. 2, pp. 97–109, June 1995.

[18] C. Crépeau and M. Sántha, "On the reversibility of oblivious transfer," in *Advances in Cryptology: Eurocrypt '91 Proc.* Berlin: Springer-Verlag, 1992, pp. 106–113.

[19] C. Crépeau and M. Sántha, "Efficient reductions among oblivious transfer protocols based on new self-intersecting codes," in *Proc. Sequences II: Methods in Communications, Security, and Computer Science.* Berlin: Springer-Verlag, 1993, pp. 360–368.

[20] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," in *Advances in Cryptology: Proc. Crypto '82.* New York: Plenum, 1983, pp. 205–210.

[21] R. Fagin, M. Naor, and P. Winkler, "Comparing common secret information without leaking it," *Commun. Assoc. Comput. Mach.*, vol. 39, no. 5, pp. 77–85, May 1996.

[22] G. D. Forney, *Concatenated Codes.* Cambridge, MA: MIT Press, 1966.

[23] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game, or: A completeness theorem for protocols with honest majority," in *Proc. 19th Annual ACM Symp. on Theory of Computing*, 1987, pp. 218–229.

[24] O. Goldreich and R. Vainish, "How to solve any protocol problem—An efficiency improvement," in *Advances in Cryptology: Crypto '87 Proc.* Berlin: Springer-Verlag, 1988, pp. 73–86.

[25] S. Goldwasser and L. Levin, "Fair computation of general functions in presence of immoral majority," in *Advances in Cryptology: Crypto '90 Proc.* Berlin: Springer-Verlag, 1991, pp. 77–93.

[26] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof-systems," *SIAM J. Comput.*, vol. 18, pp. 186–208, 1989.

[27] V. D. Goppa, "Codes on algebraic curves," *Sov. Math.–Dokl.*, vol. 24, no. 1, pp. 170–172, 1981.

[28] G. Katona and J. Srivastava, "Minimal 2-coverings of finite affine spaces based on GF (2)," *J. Statist. Plan. Inference*, vol. 8, pp. 375–388, 1983.

[29] G. L. Katsman, M. A. Tsfasman, and S. G. Vlăduţ, "Modular curves and codes with a polynomial construction," *IEEE Trans. Inform. Theory*, vol. IT-30, no. 2, pp. 353–355, Mar. 1984.

[30] J. Kilian, "Founding cryptography on oblivious transfer," in *Proc. 20th Annual ACM Symp. on Theory Comp.*, 1988, pp. 20–31.

[31] ———, "A general completeness theorem for two-party games," in *Proc. 23rd Ann. ACM Symp. on Theory Comp.*, 1991, pp. 553–560.

[32] ———, private communication.

[33] J. Kilian, S. Micali, and R. Ostrovsky, "Minimum resource zero-knowledge proofs," in *Proc. 30th Ann. IEEE Symp. on Foundations of Computer Science*, 1989, pp. 474–479.

[34] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam, The Netherlands: North-Holland, 1977.

[35] Y. I. Manin and S. G. Vlăduţ, "Linear codes and modular curves" *Sovr. Prob. Mat.*, VINITI 1984, in Russian.

[36] S. Micali and P. Rogaway, "Secure computation," in *Advances in Cryptology: Crypto '91 Proc.* Berlin: Springer-Verlag, 1992, pp. 392–404.

[37] D. Miklós, "Linear binary codes with intersecting properties," *Discr. Appl. Math.*, vol. 9, no. 2, pp. 187–196, 1984.

[38] H. Nurmi, A. Salomaa, and L. Santean, "Secret ballot elections in computer networks," *Computers & Security*, vol. 10, no. 6, pp. 553–560, 1991.

[39] M. O. Rabin, "How to exchange secrets by oblivious transfer," Aiken Computation Laboratory, Harvard Univ., Tech. Memo TR-81, 1981.

[40] C. T. Retter, "Intersecting Goppa codes," *IEEE Trans. Inform. Theory*, vol. 35, pp. 822–829, 1989.

[41] A. De Santis, G. Di Crescenzo, and G. Persiano, "Zero-knowledge arguments and public-key cryptography," *Inform. Comput.*, vol. 121, no. 1, pp. 23–40, 1995.

[42] N. J. A. Sloane, "Covering arrays and intersecting codes," *J. Comb. Des.*, vol. 1, pp. 51–63, 1993.

[43] S. Wiesner, "Conjugate coding," *Sigact News*, vol. 15, no. 1, pp. 78–88, 1983. Original manuscript written *circa* 1970.