# Optimal Reductions between Oblivious Transfers using Interactive Hashing

Claude Crépeau ⋆ and George Savvides ⋆

{crepeau,gsavvi1}@cs.mcgill.ca
McGill University, Montéral, QC, Canada.

**Abstract.** We present an asymptotically optimal reduction of one-out-of-two String Oblivious Transfer to one-out-of-two Bit Oblivious Transfer using Interactive Hashing in conjunction with Privacy Amplification. Interactive Hashing is used in an innovative way to test the receiver's adherence to the protocol. We show that $(1+\epsilon)k$ uses of Bit OT suffice to implement String OT for $k$-bit strings. Our protocol represents a two-fold improvement over the best constructions in the literature and is asymptotically optimal. We then show that our construction can also accommodate weaker versions of Bit OT, thereby obtaining a significantly lower expansion factor compared to previous constructions. Besides increasing efficiency, our constructions allow the use of any 2-universal family of Hash Functions for performing Privacy Amplification. Of independent interest, our reduction illustrates the power of Interactive Hashing as an ingredient in the design of cryptographic protocols.

*Keywords:* interactive hashing, oblivious transfer, privacy amplification.

## 1 Introduction

The notion of Oblivious Transfer was originally introduced by Rabin [12]. However, a variant of OT was first invented by Wiesner [14] but his work was only published post-facto. Its application to multi-party computation was shown by Even, Goldreich and Lempel in [8]. One-out-of-two String Oblivious Transfer, denoted $\binom{2}{1}$–String OT$^k$, is a primitive that allows a sender Alice to send one of two $k$-bit strings, $a_0, a_1$ to a receiver Bob who receives $a_c$ for a choice bit $c \in \{0, 1\}$. It is assumed that the joint probability distribution $P_{a_0 a_1 c}$ from which the inputs are generated is known to both parties. The primitive offers the following security guarantees to an honest party facing a dishonest party:

– (Dishonest) Alice does not learn any extra information about Bob's choice $c$ beyond what can be inferred from her inputs $a_0, a_1$ under distribution $P_{a_0 a_1 c}$.
– (Dishonest) Bob can learn information about only one of $a_0, a_1$. This excludes any joint information about the two strings except what can be inferred from Bob's input, (legitimate) output, and $P_{a_0 a_1 c}$.

⋆ Supported in part by NSERC, MITACS, and CIAR.

One-out-of-two Bit Oblivious Transfer, denoted $\binom{2}{1}$–Bit OT or simply Bit OT, is a simpler primitive which can be viewed as a special case of $\binom{2}{1}$–String OT$^k$ with $k = 1$. Its apparent simplicity belies its surprising power as a cryptographic primitive: it is by itself sufficient to securely implement any two-party computation [9]. It is therefore not surprising that $\binom{2}{1}$–String OT$^k$ can, in principle at least, be reduced to Bit OT. However, as such generic reductions are typically inefficient and impractical, many attempts at finding direct and efficient reductions have been made in the past. Besides increasing efficiency an orthogonal goal of some of these reductions has been to reduce $\binom{2}{1}$–String OT$^k$ to weaker variants of Bit OT such as XOR OT, Generalized OT and Universal OT.

**Contributions of this paper** The original motivation behind our work was to highlight the potential of Interactive Hashing [11, 10] as an ingredient in the design of cryptographic protocols. This paper shows how in the context of reductions between Oblivious Transfers, Interactive Hashing (both its round-unbounded and constant-round version[6]) can be used for the selection of a small subset of positions to be subsequently used for tests. This selection is sufficiently random to thwart any dishonest receiver's attempts at cheating as well as sufficiently under the honest receiver's control to protect his privacy.

We show how such tests can be embedded in the reduction of String OT to Bit OT and weaker variants given by Brassard, Crépeau and Wolf [3]. The tests ensure that the receiver cannot deviate from the protocol more than an arbitrarily small fraction of the time, leading to two important improvements over the original reduction:

1. The *expansion factor* $n/k$ (namely, the ratio of Bit OT uses to string length) is significantly reduced. Specifically:
   - In the case of Bit OT and XOR OT it decreases from $2+\epsilon$ to $1+\epsilon'$. This is in fact asymptotically optimal as the receiver has $n$ bits of entropy after the $n$ executions of Bit OT. For a formal proof that any reduction of $\binom{2}{1}$–String OT$^k$ requires at least $k$ executions of Bit OT, see [7].
   - In the case of Generalized OT it decreases from 4.8188 to $1 + \epsilon''$, which is again optimal.
   - In the case of Universal OT it is reduced by a factor of at least $8 \ln 2 = 5.545$ (its exact value is a function of the channel's characteristics).
2. The construction is more general as it allows any 2-Universal Family of Hash Functions to be used for Privacy Amplification.

## 2 Oblivious Transfer variants and their specifications

### 2.1 $\binom{2}{1}$–ROT$^k$ and its equivalence to $\binom{2}{1}$–String OT$^k$

$\binom{2}{1}$–ROT$^k$ is a randomized variant of $\binom{2}{1}$–String OT$^k$ where Alice sends to Bob two independently chosen random strings $r_0, r_1 \in_{\mathrm{R}} \{0,1\}^k$, of which Bob learns $r_c$ for $c \in_{\mathrm{R}} \{0,1\}$.

**Security requirements** Let $R_0, R_1$ be two independent random variables uniformly distributed in $\{0,1\}^k$ corresponding to the strings sent by Alice. Let $C$ be a binary random variable uniformly distributed in $\{0,1\}$ corresponding to Bob's choice. The security requirements for $\binom{2}{1}$–ROT$^k$ are captured by the following information-theoretic conditions:

1. (Dishonest) Alice does not gain any information about $C$ during the protocol. In other words $\mathbf{H}(C) = 1$.
2. (Dishonest) Bob obtains information about only one of the two random strings during the protocol. Formally, at the end of each run of the protocol, there exists some $d \in \{0,1\}$ such that $\mathbf{H}(R_d \mid R_{\bar{d}}) = k$.

**Equivalence to $\binom{2}{1}$–String OT$^k$** It is easy to see that $\binom{2}{1}$–ROT$^k$ reduces to $\binom{2}{1}$–String OT$^k$. Conversely, as Protocol 1 shows, it is also possible to reduce $\binom{2}{1}$–String OT$^k$ to $\binom{2}{1}$–ROT$^k$ in a straightforward way. As $\binom{2}{1}$–ROT$^k$ and $\binom{2}{1}$–String OT$^k$ are equivalent, in this paper we will focus on reductions of $\binom{2}{1}$–ROT$^k$ to Bit OT. This choice is motivated by the fact that the randomized nature of $\binom{2}{1}$–ROT$^k$ and the independence of the two parties' inputs yield simpler constructions with easier to prove security.

---

**Protocol 1** Reducing $\binom{2}{1}$–String OT$^k$ to $\binom{2}{1}$–ROT$^k$

---

Let the inputs to $\binom{2}{1}$–String OT$^k$ be $a_0, a_1 \in \{0,1\}^k$ for Alice and $c \in \{0,1\}$ for Bob.

1. Alice uses $\binom{2}{1}$–ROT$^k$ to send $r_0, r_1 \in_{\mathrm{R}} \{0,1\}^k$ to Bob, who receives $r_{c'}$ for some randomly chosen $c' \in \{0,1\}$.
2. Bob sends $d = c \oplus c'$ to Alice.
3. Alice sets $e_0 = a_0 \oplus r_d$ and $e_1 = a_1 \oplus r_{\bar{d}}$ and sends $e_0, e_1$ to Bob.
4. Bob decodes $a_c = e_c \oplus r_{c'}$.

---

Note that Step 1 of Protocol 1 can be performed before the two parties' inputs to $\binom{2}{1}$–String OT$^k$ have been determined and its results stored for later. In Step 2 Bob sends to Alice a "flip bit" $d$ which effectively allows him to invert the order in which Alice's strings are encrypted and thus to eventually learn the string $a_c$ of his choice regardless of his initial random choice of $c'$ in Step 1.

### 2.2 Weaker variants of Bit OT

By relaxing the security guarantees against a dishonest receiver (Bob) we obtain weaker variants of Bit OT, as described below. In all cases $b_0, b_1$ denote Alice's input bits. Whatever extra choices may be available to Bob, he can always act honestly and request $b_c$ for a choice $c \in \{0,1\}$. As in 'regular' Bit OT, dishonest Alice never obtains information about Bob's choice.

**XOR OT (XOT)** Bob can choose to learn one of $b_0, b_1, b_\oplus$ where $b_\oplus \stackrel{\text{def}}{=} b_0 \oplus b_1$.

**Generalized OT (GOT)** Bob can choose to learn $f(b_0, b_1)$ where $f$ is any of the 16 possible one-bit functions of $b_0, b_1$.

**Universal OT (UOT)** Bob can choose to learn $\Omega(b_0, b_1)$ where $\Omega$ is any arbitrary discrete memoryless channel whose input is a pair of bits and whose output satisfies the following constraint: let $B_0, B_1 \in \{0,1\}$ be uniformly distributed random variables and let $\alpha \leq 1$ be a constant. Then,

$$\mathbf{H}\left(B_0, B_1 \mid \Omega\left(B_0, B_1\right)\right) \geq \alpha.$$

Note that we disallow $\alpha > 1$ as the channel would not allow Bob to act honestly.

## 3 Tools and Mathematical Background

### 3.1 Encoding of Subsets as Bit Strings

Let $x$ be a small constant. In our protocols we will need to encode subsets of $xn$ elements out of a total of $n$ as bit strings. Let $K = \binom{n}{xn}$ be the number of such subsets. There exists a simple and efficiently computable bijection between the $K$ subsets and the integers $0, \ldots, K-1$, providing an encoding scheme with output length $m = \lceil \log(K) \rceil \leq n\mathbf{H}(x)$. See [5] (Section 3.1) for details on its implementation. Note that in this encoding scheme, the bit strings in $\{0,1\}^m$ that correspond to valid encodings, namely the binary representations of numbers $0, \ldots, K-1$, could potentially make up only slightly more than half of all strings. In order to avoid having to deal with invalid encodings, we will consider any string $w \in \{0,1\}^m$ to encode the same subset as $w \pmod{K}$. Thus in our modified encoding scheme each string in $\{0,1\}^m$ is a valid encoding of some subset, while to each of the $K$ subsets correspond either 1 or 2 bit strings in $\{0,1\}^m$. This imbalance[1] in the number of encodings per subset turns out to be of little importance in our scenario thanks to Lemma 1 below.

**Lemma 1.** *Assume the modified encoding of Section 3.1 mapping subsets to bit strings in $\{0,1\}^m$. If the fraction of subsets possessing a certain property is $f$, then the fraction $f'$ of bit strings in $\{0,1\}^m$ that map to subsets possessing that property satisfies $f' \leq 2f$.*

*Proof.* Let $P$ be the set containing all subsets possessing the property, and let $Q$ be its complement. Then $f = \frac{|P|}{|P|+|Q|}$. The maximum fraction of strings in $\{0,1\}^m$ mapping to subsets in $P$ occurs when all subsets in $P$ have two encodings each, while all subsets in $Q$ have only one. Consequently, $f' \leq \frac{2|P|}{2|P|+|Q|} \leq \frac{2|P|}{|P|+|Q|} = 2f$ □

[1] Note that this imbalance could be further reduced, if necessary, at the cost of a slight increase in the encoding length. Let $M \geq m$ and let every $w \in \{0,1\}^M$ map to the same subset as $w \pmod{K}$. Then each of the $K$ subsets will have at least $\lfloor \frac{2^M}{K} \rfloor$ and at most $\lceil \frac{2^M}{K} \rceil$ different encodings.

### 3.2 Interactive Hashing

Interactive Hashing is a primitive (first appearing in [11, 10] in the context of perfectly hiding commitments) that allows a sender to send an $m$–bit string $\boldsymbol{s}$ to a receiver, who receives both $\boldsymbol{s}$ and another, effectively random string in $\{0,1\}^m$. The security properties of this primitive that are relevant to our setting are:

1. *The receiver cannot tell which of the two output strings was the original input.* Let the two output strings be $\boldsymbol{s_0}, \boldsymbol{s_1}$ (labeled according to lexicographic order). Then if both strings were apriori equally likely to have been the sender's input $\boldsymbol{s}$, then they are aposteriori equally likely as well.
2. *When both participants are honest, the input is equally likely to be paired with any of the other strings.* Let $\boldsymbol{s}$ be the sender's input and let $\boldsymbol{s'}$ be the second output of Interactive Hashing. Then provided that both participants follow the protocol, $\boldsymbol{s'}$ will be uniformly distributed among all $2^m - 1$ strings different from $\boldsymbol{s}$.
3. *The sender cannot force both outputs to have a rare property.* Let $G$ be a subset of $\{0,1\}^m$ such that $\frac{|G|}{2^m}$ is exponentially small in $m$. Then the probability that a dishonest sender will succeed in having both outputs $\boldsymbol{s_0}, \boldsymbol{s_1}$ be in $G$ is also exponentially small in $m$.

**Implementation of Interactive Hashing** In our reductions we will use Protocol 2 to implement Interactive Hashing. All operations below take place in $\mathcal{F}_2$.

---

**Protocol 2** Interactive Hashing

---

Let $\boldsymbol{s}$ be a $m$-bit string that the sender wishes to send to the receiver.

1. The receiver chooses a $(m-1) \times m$ matrix $Q$ of rank $m-1$. Let $q_i$ be the $i$-th query, consisting of the $i$-th row of $Q$.
2. The receiver sends query $q_1$ to the sender. The sender responds with $c_1 = q_1 \cdot \boldsymbol{s}$ where $\cdot$ denotes the dot product.
3. For $2 \leq i \leq m-1$ do:
   (a) Upon receiving $c_{i-1}$ the receiver sends query $q_i$ to the sender.
   (b) The sender responds with $c_i = q_i \cdot \boldsymbol{s}$
4. Both parties compute the two solutions to the resulting system of $m-1$ equations and $m$ unknowns and label them $\boldsymbol{s_0}, \boldsymbol{s_1}$ according to lexicographic order.

---

**Security of Protocol 2** The properties of the linear system resulting from the interaction between the two parties easily establish that the first security requirement is met: that the receiver cannot guess which of the two output strings was the sender's original input to the protocol. Let $V$ be the receiver's (marginal) view at the end of the protocol and let $\boldsymbol{s_0}, \boldsymbol{s_1}$ be the corresponding output strings. Note that $V$ would be identical whether the sender's input was $\boldsymbol{s_0}$ or $\boldsymbol{s_1}$ as the responses obtained after each challenge would be the same in both cases. Consequently, if before the protocol begins the sender is equally likely to have chosen $\boldsymbol{s_0}$ and $\boldsymbol{s_1}$ as input — both with some small probability $\alpha$ — then at the end of the protocol each of these two strings has equal probability $1/2$ of having been the original input string given $V$. We remark that a dishonest receiver would gain nothing by selecting a matrix $Q$ in a non-random fashion or with rank less than $t-1$.

As for the second property, let $\boldsymbol{s}$ be the sender's input and let $\boldsymbol{s'}$ be the second output of Interactive Hashing. We first note that since the linear system has two distinct solutions, it is always the case that $\boldsymbol{s'} \neq \boldsymbol{s}$. To see that $\boldsymbol{s'}$ is uniformly distributed among all strings in $\{0,1\}^m \setminus \boldsymbol{s}$, it suffices to observe that $Q$ is randomly chosen among all rank $m-1$ matrices and that the number of such $Q$'s satisfying $Q(\boldsymbol{s}) = Q(\boldsymbol{s'}) \Leftrightarrow Q(\boldsymbol{s}-\boldsymbol{s'}) = \mathbf{0}$ is the same for any $\boldsymbol{s'} \neq \boldsymbol{s}$.

Concerning the third security requirement, it can be shown (see [5], Lemma 6) that if $G$ is an exponentially small (in $m$) subset of $\{0,1\}^m$, then whatever dishonest strategy the receiver might use with the aim of forcing both outputs $\boldsymbol{s_0}$ and $\boldsymbol{s_1}$ to be strings from $G$, he will only succeed in doing so with exponentially small probability. We remark that more recent, unpublished results by the second author of this paper establish a tight upper bound of $15.682 \cdot |G|/2^m$ for this probability and that this upper bound remains valid for all ratios $|G|/2^m$.

**More efficient implementations of Interactive Hashing** A constant-round Interactive Hashing protocol appears in [6]. The construction capitalizes on results from pseudorandomness, in particular efficient implementations of *almost t-wise independent permutations*, to significantly reduce the amount of interaction necessary. Specifically, it is shown that 4 rounds are sufficient for inputs of any size, in contrast to Protocol 2 that requires $m-1$ rounds for inputs of size $m$. The main disadvantages of this constant-round implementation are its much greater complexity as well as the fact that some parameters in the construction require prior knowledge of an upper bound on $G$. As our only efficiency concern in this paper is the number of Bit OT executions, we will not deal with this alternative construction any further even though the authors believe that it would be a suitable replacement to Protocol 2, at least in the context of our reductions.

### 3.3 Tail bounds

**Markov's Inequality** Let $X$ be a random variable assuming only positive values and let $\mu = \mathbf{E}[X]$. Then $\mathbf{Pr}[X \geq t] \leq \frac{\mu}{t}$.

**Chernoff bounds** Let $B(n,p)$ be the binomial distribution with parameters $n, p$ and mean $\mu = np$. We will use the following versions of the Chernoff bound for $0 < \delta \leq 1$:

$$\mathbf{Pr}[B(n,p) \leq (1-\delta)\mu] \leq e^{-\delta^2\mu/2} \quad (1)$$

$$\mathbf{Pr}[B(n,p) \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/3} \quad (2)$$

From (1) we can also deduce the following inequality

$$\mathbf{Pr}[B(n,p) \leq \mu - \Delta n] \leq e^{-\Delta^2 n/2} \quad (3)$$

### 3.4 Error probability and its concentration on an erasure event

**Fano's lemma** (Adapted from [3]) Let $X$ be a random variable with range $\mathcal{X}$ and let $Y$ be another, related random variable. Let $p_e$ be the (average) error probability of correctly guessing the value of $X$ with any strategy given the outcome of $Y$ and let $h(p) \stackrel{\text{def}}{=} -p \log p - (1-p)\log(1-p)$. Then $p_e$ satisfies:

$$h(p_e) + p_e \cdot \log_2(|\mathcal{X}| - 1) \geq \mathbf{H}\left(X \mid Y\right) \tag{4}$$

**Specifying an erasure event $\boldsymbol{\Delta}$** Let $X$ be a *binary* random variable and let $p_e$ be the error probability of guessing $X$ correctly using an optimal strategy (in other words, $p_e$ is the minimum average error probability). Let $p \leq p_e$. For a specific guessing strategy with average guessing error at most $1/2$, let $E$ be an indicator random variable corresponding to the event of guessing the value of $X$ incorrectly. Note that $\mathbf{Pr}\left[\bar{E}\right] \geq \mathbf{Pr}\left[E\right] \geq p_e \geq p$. Define $\Delta$ to be another indicator random variable such that

$$\mathbf{Pr}\left[\Delta \mid E\right] = \frac{p}{\mathbf{Pr}\left[E\right]} \qquad\qquad \mathbf{Pr}\left[\Delta \mid \bar{E}\right] = \frac{p}{\mathbf{Pr}\left[\bar{E}\right]} \tag{5}$$

It follows that $\mathbf{Pr}\left[\Delta\right] = 2p$ and that $\mathbf{Pr}\left[E \mid \Delta\right] = \mathbf{Pr}\left[\bar{E} \mid \Delta\right] = \frac{1}{2}$. Suppose that the value of $\Delta$ is provided as side information by an oracle. Then with probability $2p$ we have $\Delta = 1$ in which case $X$ is totally unknown We will refer to this event as an *erasure* of $X$. This leads to the following lemma:

**Lemma 2.** *Let $X$ be a binary random variable and let $p_e$ be the error probability when guessing $X$. Then $X$ can be* erased *with probability $2p \leq 2p_e$.*

### 3.5 Privacy Amplification

Privacy Amplification [2] is a technique that allows a partially known string $R$ to be shrunk into a shorter but almost uniformly distributed string $r$ that can be used effectively as a one-time pad in cryptographic applications. For our needs we will use a simplified version of the Generalized Privacy Amplification Theorem [1] (also covered in [2]) which assumes that there are always $u$ or more unknown physical bits about $R$ (as opposed to general bounds on $R$'s entropy).

**Theorem 1.** *Let $R$ be a random variable uniformly distributed in $\{0,1\}^n$. Let $V$ be a random variable corresponding to Bob's knowledge of $R$ and suppose that any value $V = v$ provides no information about $u$ or more physical bits of $R$. Let $s$ be a security parameter and let $k = u - s$. Let $\mathcal{H}$ be a 2-Universal Family of Hash functions mapping $\{0,1\}^n$ to $\{0,1\}^k$ and let $H$ be uniformly distributed in $\mathcal{H}$. Let $r = H(R)$ Then the following holds:*

$$\mathbf{H}\left(r \mid VH\right) \geq k - \log\left(1 + 2^{k-u}\right) \geq k - \frac{2^{k-u}}{\ln 2} = k - \frac{2^{-s}}{\ln 2} \tag{6}$$

*It follows from Equation (6) that $I(r; VH) \leq 2^{-s}/\ln 2$. From Markov's inequality it follows that the probability that Bob has more than $2^{-s/2}$ bits of information about $r$ is no larger than $2^{-s/2}/\ln 2$.*

# 4 Previous work

All reductions of $\binom{2}{1}$–ROT$^k$ to Bit OT fall within two major categories: reductions based on Self-Intersecting Codes (Section 4.1) and reductions based on Privacy Amplification (Section 4.2).

## 4.1 Reductions based on self-intersecting codes

These reductions use a special class of error-correcting codes called "self-intersecting codes" encoding $k$-bit input strings into $n$-bit codewords. They have the extra property that any two non-zero codewords $c_0, c_1$ must have a position $i$ such that $c_{0i} \neq 0 \neq c_{1i}$. Consult [4] for more details.

**Advantages and Disadvantages** The main advantage of this approach is that the self-intersecting code can be chosen ahead of time and embedded once and for all in the protocol. One of its main disadvantages is the rather large expansion factor $n/k$, theoretically lower-bounded by 3.5277 [13] and in practice roughly 4.8188. Another important limitation is that this approach does not lend itself to generalizations to weaker forms of Bit OT, such as XOT, GOT and UOT.

## 4.2 Reductions based on Privacy Amplification

In Protocol 3 we introduce the construction of [3] upon which our own construction (Protocol 4) builds and expands.

**Protocol 3** Reducing $\binom{2}{1}$–ROT$^k$ to Bit OT

1. Alice selects $R_0$, $R_1 \in_{\mathrm{R}} \{0,1\}^n$. Bob selects $c \in_{\mathrm{R}} \{0,1\}$.
2. Alice sends $R_0, R_1$ to Bob using $n$ executions of Bit OT, where the $i$-th round contains bits $R_0^i, R_1^i$. Bob receives $R_c$.
3. Let $k = n/2 - s$ where $s$ is a security parameter. Alice randomly chooses two $k \times n$ binary matrices $M_0, M_1$ of rank $k$ and sets $r_0 = M_0 \cdot R_0$ and $r_1 = M_1 \cdot R_1$.
4. Alice sends $M_0, M_1$ to Bob, who sets $r_c = M_c \cdot R_c$.

It is easy to see that Protocol 3 always succeeds in achieving $\binom{2}{1}$–ROT$^k$ when both parties are honest. The properties of Bit OT guarantee that (dishonest) Alice cannot obtain any information on Bob's choice bit $c$ at Step 2. On the other hand, at the end of Step 2 (dishonest) Bob is guaranteed to be missing at least $n/2$ bits of $R_d$ for some $d \in \{0,1\}$. This is exploited at Step 3 by performing Privacy Amplification with output length $k = n/2 - s$. Specifically, the 2-universal family of Hash Functions used in Protocol 3 guarantees that $r_d$ is uniformly distributed in $\{0,1\}^k$ and independent of $r_{\bar{d}}$ except with probability exponentially small in $s$. It is shown in [3] that using this family of hash functions this property can be maintained even if Bit OT is replaced with weaker variants such as XOR OT, Generalized OT and Universal OT — albeit at the cost of further reducing the size of $k$.

**Advantages and disadvantages** Besides its apparent simplicity and straightforward implementation, the reduction of Protocol 3 has two main advantages over reductions based on Self-Intersecting Codes: Using $n$ executions of Bit OT one can achieve $\binom{2}{1}$–ROT$^k$ for $k$ slightly less than $n/2$, leading to an expansion factor of $2+\epsilon$. Consequently, it achieves a lower expansion factor than any reduction based in Self-Intersecting Codes. Using the 2-universal family of Hash Functions defined at Step 3, the reduction works without any modification when Bit OT is replaced with XOT and requires only a decrease in the size of $k$ to work with GOT and UOT.

The construction suffers from two disadvantages: The proof of security relies heavily on the properties of matrices in $\mathcal{F}_2$ used for Privacy Amplification in Step 3. A general result for any universal class of hash functions was left as an open problem. In every run of the protocol a new set of matrices $M_0, M_1$ must be selected and transmitted, thereby increasing the amount of randomness needed as well as the communication complexity by $\Theta(n^2)$ bits.

## 5 The new reduction of $\binom{2}{1}$–ROT$^k$ to Bit OT

**Notation and conventions** In our reduction, two randomly chosen strings $T_0, T_1 \in_{\mathrm{R}} \{0,1\}^n$ are transmitted pairwise using $n$ executions of Bit OT. We denote by $t_0^i, t_1^i$ the bits at position $i$ of $T_0, T_1$, respectively. Let $I$ be the set of all $n$ positions. For a subset $s \subseteq I$ let $T(s)$ be the substring of $T$ consisting of the bits at all positions $i \in s$ in increasing order of position. Note that $T(I) = T$. Subsets of $I$ of cardinality $xn$ will be mapped to bit strings of length $m = \lceil \log \left( \binom{n}{xn} \right) \rceil$ using the encoding/decoding scheme of Section 3.1.

**Intuition behind Protocol 4** At Step 1, the two parties agree on the value of $x$ which will determine the proportion of bits sacrificed for tests.

At Step 2 Alice selects the two random $n$-bit strings to be transmitted to Bob using $n$ executions of Bit OT.

At Step 3 Bob randomly chooses his choice bit $c \in \{0,1\}$. He also selects a small subset $s \in I$ of cardinality $xn$. This selection is made by first choosing an encoding $w$ uniformly at random among $\{0,1\}^m$ and then mapping it to the corresponding subset $s$. This guarantees that on one hand, $s$ is sufficiently random and on the other hand, that every string in $\{0,1\}^m$ is equally likely to be Bob's initial choice. The latter fact will be crucial in preventing Alice from guessing Bob's choice bit in later steps.

At Step 4 Alice transmits $T_0, T_1$ using $n$ executions of Bit OT. Bob selects to learn $t_c^i$ at all positions except at the few positions in $s$ where his choice is reversed. As a result he knows most bits of $T_c$ and only $xn$ bits of $T_{\bar{c}}$. See Fig. 1.

The goal of the protocol at Step 5 is to select a second, effectively random subset. Bob starts by sending $w$ to Alice using Interactive Hashing, the output of which will be $w_0, w_1$. As from Alice's point of view both strings are equally likely to have been Bob's original choice at Step 3, Property 1 of Interactive Hashing (Section 3.2) guarantees to Bob that Alice cannot guess the value of $b$ such that $w_b = w$. At the same time Property 3 of Interactive Hashing provides

**Protocol 4** New reduction of $\binom{2}{1}$–ROT$^k$ to Bit OT using Interactive Hashing

1. Alice and Bob select $x$ to be a (very small) positive constant less than 1.
2. Alice chooses two random strings $T_0, T_1 \in_{\mathrm{R}} \{0,1\}^n$.
3. Bob chooses a random $c \in_R \{0,1\}$. Let $m = \lceil \log \left( \binom{n}{xn} \right) \rceil$. Bob selects $w \in_R \{0,1\}^m$ uniformly at random and decodes $w$ into a subset $s \subset I$ of cardinality $xn$ according to the encoding/decoding scheme of Section 3.1.
4. Alice transmits $T_0, T_1$ to Bob using $n$ executions of Bit OT, with round $i$ containing bits $t_0^i, t_1^i$. Bob chooses to learn $t_c^i$ if $i \notin s$ and $t_{\bar{c}}^i$ if $i \in s$.
5. Bob sends $w$ to Alice using Interactive Hashing (Protocol 2). Alice and Bob compute the two output strings, labeled $w_0$ , $w_1$ according to lexicographic order, as well as the corresponding subsets $s_0, s_1 \subset I$. Bob computes $b \in \{0,1\}$ s.t. $w_b = w$.
6. Alice checks that $|s_0 \cap s_1| \leq 2 \cdot x^2 n$ and aborts otherwise.
7. Both parties compute $s_0' = s_0 \setminus (s_0 \cap s_1)$ and $s_1' = s_1 \setminus (s_0 \cap s_1)$.
8. Bob announces $a = b \oplus c$ to Alice. He also announces $T_0(s_{1-a}')$ and $T_1(s_a')$.
9. Alice checks that the strings announced by Bob are consistent with $a$ and contain no errors. Otherwise she aborts the protocol.
10. Alice and Bob discard the Bit OT's at positions $s_0 \cup s_1$ and concentrate on the remaining positions in $J = I \setminus (s_0 \cup s_1)$. Let $j = |J|$ and $R_0 = T_0(J), R_1 = T_1(J)$.
11. Alice chooses two functions $h_0, h_1$ randomly and independently from a 2-universal family of hash functions with input length $j$ and output length $k = j - 6xn \geq n - 8xn$. She sets $r_0 = h_0(R_0)$ and $r_1 = h_1(R_1)$. She sends $h_0, h_1$ to Bob.
12. Bob sets $r_c = h_c(R_c)$.

Alice with the guarantee that the choice of one of $w_0, w_1$ was effectively random and beyond Bob's control. We will see that this implies that the corresponding subset is also random enough to ensure that a cheating Bob will fail the tests at Step 9 except with negligible probability.

At Step 6 Alice makes sure that the intersection of $s_0, s_1$ is not too large as this would interfere with the proof of security against a dishonest Bob.

At Step 7 the two parties exclude the bits in this intersection from the tests that will follow since Bob cannot be expected to know both $T_0(s_0 \cap s_1)$ and $T_1(s_0 \cap s_1)$. What remains of $s_0, s_1$ is denoted $s_0', s_1'$.

At Step 8 Bob effectively announces $T_c(s_{\bar{b}}')$ and $T_{\bar{c}}(s_b')$ in both cases. Note that the only information related to $c$ which is implied by the announced bits is the value of $a$, which is already made available to Alice at the beginning of the step. Alice can correctly guess $c = a \oplus b$ if and only if she can correctly guess $b$.

At Step 9 Alice checks that the strings were announced correctly and are consistent with the value of $a$ — see Fig. 3. If that is the case then Alice is convinced that Bob has not deviated much from the protocol at Step 4. In a nutshell the idea here is that Interactive Hashing guarantees that even if Bob behaves dishonestly, without loss of generality $s_1$ was chosen effectively at random. Therefore, if Bob can announce all bits in $T_0(s_0'), T_1(s_1')$, say, it must have

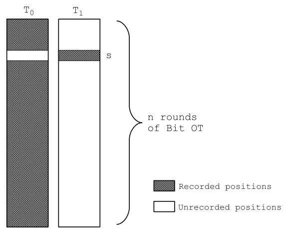


**Fig. 1.** During the $n$ Bit OT executions Bob chooses $t_c^i$ at positions $i \in I \setminus s$, and $t_{\bar{c}}^i$ at positions $i \in s$. In the Figure, $c = 0$ so in the end Bob knows $T_0(I \setminus s)$ and $T_1(s)$. Note that while $s \subset I$ is shown here as a contiguous block, in reality the positions it represents occur throughout the $n$ executions.

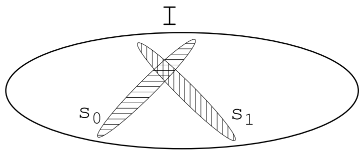


**Fig. 2.** Honest Bob sends his subset $s$ to Alice through Interactive Hashing. With overwhelming probability this procedure produces two outputs $s_0, s_1$ of which one is $s$ and the other is effectively randomly chosen. Alice does not know which of the two was Bob's original choice. The intersection of $s_0, s_1$ is later excluded to form $s_0', s_1'$.

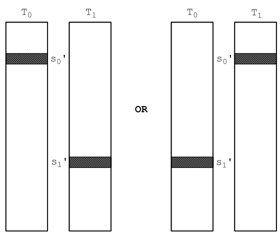


**Fig. 3.** After establishing sets $s_0', s_1'$, Alice expects Bob to announce either $T_0(s_0')$ and $T_1(s_1')$ or $T_0(s_1')$ and $T_1(s_0')$ depending on the value of $a$. If Bob's choice was $c = 0$ as in Figure 1 and $s = s_0$ after Interactive Hashing, then he would choose the latter option.

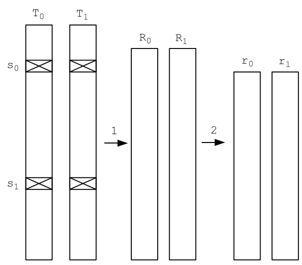


**Fig. 4.** After Bob has passed the tests, both players ignore the Bit OT executions at positions $s_0 \cup s_1$ and form strings $R_0, R_1$ from the remaining bits. Then independent applications of Privacy Amplification on $R_0, R_1$ produce $r_0, r_1 \in \{0, 1\}^k$.

been the case that he knew most bits in $T_1$ to begin with and consequently few bits in $T_0$. In fact, we prove that if (dishonest) Bob learns more than $5xn$ bits of both $T_0$ and $T_1$ during Step 4 then he gets caught with overwhelming probability.

In Step 10 the two players discard the Bit OT executions at positions $s_0 \cup s_1$ that were used for tests and concentrate on the remaining $j$ executions. Note that $j \geq n - 2xn$. As Bob passed the tests of Step 9, Alice is convinced that there is a $d \in \{0,1\}$ such that Bob knows at most $5xn$ bits in $T_d$ and thus at most $5xn$ bits in $R_d$. This implies that he is missing at least $j - 5xn$ bits of $R_d$.

In Step 11 she thus sets $k = (j - 5xn) - xn \geq n - 8xn$ and performs Privacy Amplification (with security parameter $xn$) on $R_0, R_1$ to get $r_0, r_1$. See Fig. 4.

**Gains in efficiency** As $k \geq n - 8xn$ for any small constant $x$, the expansion factor $n/k$ is $1 + \epsilon$ for some small constant $\epsilon = \frac{8x}{1-8x}$. This is asymptotically optimal (see [7]) and represents a two-fold improvement over the corresponding reduction in [3] where the expansion factor was at least $2 + \epsilon'$.

### 5.1 Proof of Security and Practicality

**Theorem 2.** *The probability of failure of Protocol 4 with honest participants is exponentially small in $n$.*

*Proof.* If both parties are honest then Protocol 4 can only fail at Step 6. We will show that for any (fixed) $w \in \{0,1\}^m$ that Bob inputs to Interactive Hashing at Step 5, the probability that the second output $w'$ is such that $|s \cap s'| > 2 \cdot x^2 n$ is exponentially small in $n$. Let $s$ be the subset corresponding to Bob's choice of $w$. We will call a subset $s'$ *bad* if $|s \cap s'| > 2 \cdot x^2 n$. Likewise, we will call a string $w' \in \{0,1\}^m$ *bad* if it maps to a bad subset.

We start by showing that the fraction of bad subsets is exponentially small in $n$. Suppose $s' \subset I$ is randomly chosen among all subsets of cardinality $xn$. One way to choose $s'$ is by sequentially selecting $xn$ positions uniformly at random without repetition among all $n$ positions in $I$. The probability $q_i$ that the $i$–th position thus chosen happens to collide with one of the $xn$ positions in $s$ satisfies

$$q_i < \frac{xn}{n - xn} = \frac{x}{1-x}$$

As a thought experiment, suppose that one were to choose $xn$ positions independently at random, so that each position collides with an element of $s$ with probability exactly $q = \frac{x}{1-x}$. This artificial way of choosing $xn$ positions can only increase the probability of ending up with more than $2x^2 n$ collisions. We can use the Chernoff bound (2) to upper bound this (larger) probability. Assuming $x < 1/2$ and setting $\delta = 1 - 2x$ we get

$$\mathbf{Pr}\left[B(xn, \frac{x}{1-x}) > 2x^2 n\right] \leq \epsilon'$$

where $\epsilon' = e^{-\frac{(1-2x)^2 x^2}{3(1-x)} n}$. This in turn guarantees that when $s'$ is selected in the appropriate way, the event $|s \cap s'| > 2 \cdot x^2 n$ occurs with probability $\epsilon < \epsilon'$. In other words, the fraction of bad subsets is upper bounded by $\epsilon < \epsilon'$.

By Lemma 1, the fraction of bad strings in $\{0,1\}^m$ is at most $2\epsilon$. As $w$ itself is bad, it follows that among all $2^m - 1$ strings other than $w$ the fraction of bad strings is no larger than $2\epsilon$. Since by Property 2 of Interactive Hashing, $w$ is paired to some uniformly chosen $w' \neq w$, the probability that the protocol aborts at Step 6 is upper bounded by $2\epsilon$ which is exponentially small in $n$. $\square$

**Theorem 3.** *Alice learns nothing about (honest) Bob's choice bit $c$.*

*Proof.* During Bob's interaction with Alice, his choice bit $c$ comes into play only during the Bit OT executions of Step 4 and later at Step 8 when Bob announces $a = b \oplus c$. As Bit OT is secure by assumption, Alice cannot obtain any information about $c$ in Step 4. As for Step 8, since (honest) Bob chooses $w$ uniformly at random in $\{0,1\}^m$, both $w_0$ and $w_1$ are apriori equally likely choices. By Property 1 of Interactive Hashing (see Section 3.2), the aposteriori probabilities of $w_0, w_1$ having been Bob's input are then equal as well. Consequently, Alice cannot guess $b$ with probability higher than $1/2$ and the same holds for $c = a \oplus b$. $\square$

**Security against a dishonest Bob** The proof of security against a dishonest Bob is considerably more involved. The main idea is that if Bob deviates from the protocol more than a small fraction of the time then he gets caught by the end of Step 9 with overwhelming probability. If, on the other hand, he deviates only a small fraction of the time, then Privacy Amplification effectively destroys any illegal information he may have obtained. We start with some definitions and lemmas that will help to prove the main theorem (Theorem 4) of this section.

**Definition 1.** *For a bit string $\sigma$, define $u_p(\sigma)$ to be the number of bits in $\sigma$ that can be guessed correctly with probability at most $p < 1$. These bits will be referred to as* unknown *bits.*

**Definition 2.** *Let $s \subset I$. Assuming Definition 1, we call $s$* good for $T_c$ *if $u_p(T_c(s)) \leq 3x^2n$. Otherwise, we call $s$* bad for $T_c$*. We say that $s$ is good for either $T_0$ or $T_1$ if at least one of $u_p(T_0(s)), u_p(T_1(s))$ is at most $3x^2n$.*

**Definition 3.** *Let $w$ be a string in $\{0,1\}^m$. We call $w$* good for $T_c$ *if the subset $s$ it encodes is good for $T_c$ according to Definition 2. Otherwise, $w$ is* bad for $T_c$*.*

**Lemma 3.** *Let $u_p(T_c) \geq 5xn$. Then among all subsets $s \subset I$ of cardinality $xn$ the fraction of good subsets for $T_c$ is less than $e^{-x^2n/8}$.*

*Proof.* We will use the Probabilistic Method to show that the probability that a randomly chosen subset $s$ is good for $T_c$ is less than $e^{-x^2n/8}$. One way of choosing $s$ would be to sequentially choose $xn$ positions in $I$ at random and without replacement. Note that regardless of previous choices, for all $1 \leq i \leq xn$ the probability $q_i$ of position $i$ being chosen among the $u_p(T_c)$ positions of unknown bits always satisfies

$$q_i > \frac{u_p(T_c) - xn}{|I|} \geq \frac{5xn - xn}{n} = 4x$$

This implies that the probability of choosing a good subset for $T_c$ would be greater if we were to choose the $xn$ positions independently at random so that each position corresponds to an unknown bit with probability $q = 4x$. In this artificial case the distribution of the number of unknown bits is binomial with parameters $xn, 4x$ and mean $\mu = 4x^2n$. Applying the Chernoff bound (Equation 1) with $\delta = 1/4$ we get

$$\mathbf{Pr}\left[B(xn, 4x) \leq 3x^2n\right] \leq e^{-x^2n/8}$$

We conclude that a subset $s$ chosen randomly in the appropriate way has probability smaller than $e^{-x^2n/8}$ of being good for $T_c$, which establishes the claim. $\square$

**Lemma 4.** *Let both $u_p(T_0), u_p(T_1) \geq 5xn$. Then the fraction of strings in $\{0,1\}^m$ that are good for either $T_0$ or $T_1$ is no larger than $4 \cdot e^{-x^2n/8}$.*

*Proof.* It follows from Lemma 3 and the Union Bound that the proportion of good subsets for either $T_0$ or $T_1$ is no larger than $2 \cdot e^{-x^2n/8}$. Lemma 1 in turn guarantees that the fraction of strings in $\{0,1\}^m$ that are good for either $T_0$ or $T_1$ in $\{0,1\}^m$ is at most $4 \cdot e^{-x^2n/8}$. $\square$

**Lemma 5.** *Let both $u_p(T_0), u_p(T_1) \geq 5xn$. Then the probability that (dishonest) Bob will clear Step 9 is exponentially small in $n$.*

*Proof.* By Lemma 4, the proportion of good strings in $\{0,1\}^m$ for either $T_0$ or $T_1$ is at most $4 \cdot e^{-x^2n/8}$. By Property 3 of Interactive Hashing, the probability that both $w_0, w_1$ will be good at Step 5 of the protocol is at most $\epsilon_1$ which is exponentially small in $m$ (and hence in $n$). Consequently, with probability at least $1 - \epsilon_1$, at least one of the two bit strings (without loss of generality, $w_1$) is bad for *both* $T_0$ and $T_1$. In other words, $w_1$ corresponds to a subset $s_1$ with both $u_p(T_0(s_1)), u_p(T_1(s_1)) \geq 3x^2n$. Moreover, as Alice did not abort at Step 6 it must be the case that $|s_0 \cap s_1| \leq 2x^2n$. It follows that both $u_p(T_0(s_1')), u_p(T_1(s_1')) \geq 3x^2n - 2x^2n = x^2n$. Therefore, however Bob decides to respond in Step 8, he must correctly guess the value of at least $x^2n$ unknown bits in one of $T_0, T_1$. As the bits were independently chosen, the probability of guessing them is $\epsilon_2 \leq p^{x^2n}$.

Bob will clear Step 9 only if he got two good strings from Interactive Hashing or got at least one bad string and then correctly guessed all the relevant bits. This probability is upper bounded by $\epsilon_1 + \epsilon_2$ (exponentially small in $n$). $\square$

**Theorem 4.** *The probability of (dishonest) Bob successfully cheating in Protocol 4 is exponentially small in $n$.*

*Proof.* Let $v_0, v_1 \subseteq I$ be the positions where (dishonest) Bob requested $t_0^i, t_1^i$ respectively during Step 4. Note that $v_0 \cap v_1 = \emptyset$. We distinguish two cases: (Case 1 and Case 2 taken together establish the claim.)

*Case 1:* Both $|v_0|, |v_1| \leq n - 5xn$.

In this case $u_{1/2}(T_0), u_{1/2}(T_1) \geq 5xn$, so by Lemma 5 (dishonest) Bob will fail to clear Step 9 except with exponentially (in $n$) small probability.

*Case 2:* One of $|v_0|, |v_1|$ is greater than $n - 5xn$.

Without loss of generality, let $|v_0| > n - 5xn$. Then Bob knows less than $5xn$ bits about $T_1$, and consequently, less than $5xn$ bits about $R_1 = T_1(J)$. Note that as $T_0, T_1$ are independently chosen, even if an oracle were to subsequently provide all the bits of $T_0$ (or $R_0$ , or $r_0$), Bob would obtain no new information about $R_1$. As $u_{1/2}(R_1) \geq j - 5xn$, Privacy Amplification with output length $k = (j - 5xn) - xn$ destroys all but an exponentially (in $n$) small amount of information about $r_1$, with probability exponentially close to 1. □

## 6 Extension to weaker variants of Bit OT

We demonstrate that Protocol 4 can accommodate weaker versions of Bit OT. Specifically, it requires no modification at all if Bit OT is replaced with XOT, while a virtually imperceptible decrease in the output length $k$ guarantees its security with GOT. Decreasing $k$ even further allows us to prove the Protocol's security when Bob has access to UOT with $\alpha \leq 1$. As in all three cases honest Bob's choices during Step 4 are identical to the case of Bit OT and remain equally well hidden from Alice's view, the proofs of Theorems 2 and 3 (establishing the Protocol's practicality and security against dishonest Alice) carry over verbatim to the new settings.

On the other hand, arguing that the Protocol remains secure against dishonest Bob is more involved and requires a separate analysis in each case. The basic idea, however, is the same as in the case of Bit OT and consists in showing that if Bob has deviated 'significantly' from the protocol then he gets caught with overwhelming probability, and if he has not, then Privacy Amplification effectively eliminates any illegal information he may have accumulated.

### 6.1 Security against a dishonest Bob using XOT

**Theorem 5.** *The probability of (dishonest) Bob successfully cheating in Protocol 4 is exponentially small in $n$ even if the Bit OT protocol is replaced with XOT.*

*Proof.* Let $v_0, v_1, v_\oplus \subseteq I$ denote the sets of positions $i$ where (dishonest) Bob requested $t_0^i, t_1^i, t_\oplus^i = t_0^i \oplus t_1^i$ respectively during Step 4. As in the proof of Theorem 3, we distinguish two cases, in both of which the probability of cheating is exponentially small in $n$, as desired.

*Case 1:* One of $|v_0|, |v_1|$ is greater than $n - 5xn$.

Without loss of generality, let $|v_0| > n - 5xn$. Then $|v_1 \cup v_\oplus| < 5xn$. Consequently, Bob knows less than $5xn$ bits about $R_1$ even if he is provided with all the bits of $T_0$ by an oracle after Step 4. We note in passing that such oracle information can only be helpful for the positions in $v_\oplus$. Since $u_{1/2}(R_1) > j - 5xn$, Privacy Amplification with output length $k = (j - 5xn) - xn$ would destroy all but an exponentially (in $n$) small amount of information about $r_1$, with probability exponentially close to 1.

*Case 2:* Both $|v_0|, |v_1| \leq n - 5xn$.

This implies that both $|v_1 \cup v_\oplus|$ and $|v_0 \cup v_\oplus|$ are at least $5xn$ and consequently, $u_{1/2}(T_0), u_{1/2}(T_1) \geq 5xn$. By Lemma 5, Bob will fail to clear Step 9 except with exponentially (in $n$) small probability. □

**Gains in efficiency** The expansion factor is identical to the case of Bit OT (and optimal). Compared to the reduction in [3], ours is again twice as efficient.

### 6.2 Security against a dishonest Bob using GOT

In the case of Generalized OT, during round $i$ of Step 4 dishonest Bob can choose to obtain $f(t_0^i, t_1^i)$ for any of the 16 functions $f : \{0,1\}^2 \mapsto \{0,1\}$. Without loss of generality, we will assume that Bob never requests the two constant functions as this would provide him with no information. It is not difficult to see that in our context the information content of each of the remaining 14 functions is equivalent to that of one of the four functions $f_0, f_1, f_\oplus, f_{\text{AND}}$ defined in Equation (7) below. We will thus assume that Bob always requests the output of one of these functions. In keeping with the notation of previous sections we let $v_0, v_1, v_\oplus, v_{\text{AND}} \subseteq I$ be the positions where Bob requested $f_0, f_1, f_\oplus, f_{\text{AND}}$ respectively.

$$f_0(t_0,t_1) = t_0, \quad f_1(t_0,t_1) = t_1, \quad f_\oplus(t_0,t_1) = t_0 \oplus t_1, \quad f_{\text{AND}}(t_0,t_1) = t_0 \wedge t_1 \quad (7)$$

**A necessary modification to Protocol 4** Our proof of security requires that $k$ be slightly shorter than in the case of Bit OT and XOR OT, that is $k = (j - 8xn) - xn \geq n - 11xn$.

The security analysis of the protocol in this setting is somewhat more complicated compared to the case of Bit OT and XOT. This is due to the fact that requesting $f_{\text{AND}}$ may or may not result in loss of information about $(t_0, t_1)$: with probability $1/4$ the output of $f_{\text{AND}}$ is 1 and so Bob learns both bits while with complementary probability $3/4$ the output is 0 in which case the input bits were $(0,0), (0,1), (1,0)$, all with equal probability. Note that in this latter case both $t_0, t_1$ are unknown as each can be guessed correctly with probability at most $2/3$.

**Complications arising from adaptive strategies** If dishonest Bob's requests could be assumed to be fixed ahead of time, our analysis would be quite straightforward since we could claim that among all requests in $v_{\text{AND}}$, with high probability a fraction $3/4 - \epsilon$ would produce an output of 0 and thus both $t_0, t_1$ would be added to the set of unknown bits in $T_0, T_1$. Our task is complicated by the fact that Bob obtains the output of the function he requested immediately after each round and can thus adapt his future strategy to past results. For example, Bob may be very risk-averse and start by asking for $f_{\text{AND}}$ in the first round. If he is lucky and the output is 1, he asks for $f_{\text{AND}}$ again, until he gets unlucky in which case he starts behaving honestly. This strategy makes it almost impossible to catch Bob cheating while it allows Bob to learn both $r_0, r_1$ with some nonzero — but admittedly quite small— probability. This example illustrates that we cannot assume that $|v_{\text{AND}}|$ is known ahead of time and remains independent of results obtained during the $n$ executions of Step 4.

**Dealing with adaptive strategies** In order to prove the security of the protocol for any conceivable strategy that dishonest Bob might use, we start by observing that at the end of Step 4 one of the following two cases always holds:

**Case 1:** One of $|v_0|, |v_1| > n - 8xn$, **Case 2:** Both $|v_0|, |v_1| \leq n - 8xn$

Note that these two cases refer only to the types of requests issued by Bob during Step 4 and do not depend in any way on the results obtained along the way. Given any (adaptive) strategy $S$ for Bob, one can construct the following two strategies: Strategy $S_1$ begins by making the same choices as $S$ but ensures that eventually the condition in Case 1 will be met: it "applies the brakes" just before this constraint becomes impossible to meet in the future and makes its own choices from that point on in order to meet its goal. Similarly, Strategy $S_2$ initially copies the choices of $S$ but if necessary, stops following them to ensure that the condition of Case 2 is met. Let $\delta, \delta_1, \delta_2$ be the probabilities of successfully cheating using Strategies $S, S_1, S_2$, respectively. We will argue that $\delta \leq \delta_1 + \delta_2$. To see this, imagine three parallel universes in which Bob is interacting with Alice using strategies $S, S_1, S_2$ respectively. Recall that by the end of Step 4 the universe of Strategy $S$ is identical either to the Universe of Strategy $S_1$ or to the Universe of Strategy $S_2$ (one of the two never had to "apply the brakes"). Therefore, Strategy $S$ succeeds only if one of $S_1, S_2$ succeeds and so $\delta \leq \delta_1 + \delta_2$.

It remains to prove that both $\delta_1, \delta_2$ are exponentially small in $n$. To do this, we let $\Sigma_1, \Sigma_2$ be *any* adaptive strategies ensuring that the conditions of Case 1 and Case 2, respectively, are met. We will show that for any such strategies (thus, for $S_0, S_1$ as well), the probabilities of success $\Delta_1, \Delta_2$ are exponentially small in $n$, and therefore so is $\delta$ (since $\delta \leq \delta_1 + \delta_2 \leq \Delta_1 + \Delta_2$).

**Theorem 6.** *The probability of (dishonest) Bob cheating in (modified) Protocol 4 is exponentially small in $n$ even if Bit OT is replaced with GOT.*

*Proof.* We will prove that $\Delta_1, \Delta_2$ are both exponentially small in $n$.

Without loss of generality, let $|v_0| > n - 8xn$ at the end of Step 4. Then Bob knows at most $8xn$ bits about $T_1$, even if he is provided with all the bits of $T_0$ by an oracle. Consequently, $u_{1/2}(R_1) > j - 8xn$ and therefore using Privacy Amplification with output length $k = (j - 8xn) - xn \geq n - 11xn$ will result in Bob having only an exponentially small amount of information about $r_1$ (even given $r_0$), except with an exponentially small probability $\Delta_1$.

As for Strategies $\Sigma_2$, we start by showing that $\mathbf{Pr}\left[u_{2/3}(T_1) \leq 5xn\right]$ is small. Since any such strategy guarantees that $|v_1| \leq n - 8xn$, it follows that $|v_0 \cup v_\oplus \cup v_{\text{AND}}| \geq 8xn$. Given this constraint, the probability that $u_{2/3}(T_1) \leq 5xn$ is maximized if $|v_{\text{AND}}| = 8xn, |v_0| = |v_\oplus| = 0$. This is because each request in $v_0$ and $v_\oplus$ results with certainty in the corresponding bit in $T_1$ being unknown, while a request in $v_{\text{AND}}$ produces an unknown bit in $T_1$ with probability $3/4$ (moreover, in this case the unknown bit can be guessed correctly with probability $2/3$ instead of $1/2$). Using the Chernoff bound (Equation 1) with $(n, p, \delta) \mapsto (8xn, 3/4, 1/6)$ gives

$$\mathbf{Pr}\left[u_{2/3}(T_1) \leq 5xn\right] \leq \mathbf{Pr}\left[B(8xn, \frac{3}{4}) \leq 5xn\right] \leq e^{-xn/12}$$

and similarily for $u_{2/3}(T_0)$. By the Union Bound, both $u_{2/3}(T_0), u_{2/3}(T_1) \geq 5xn$ except with probability at most $2 \cdot e^{-xn/12}$. In this case, Lemma 5 guarantees that Bob will manage to clear Step 9 with some probability $\epsilon$ exponentially small in $n$. We conclude that using any Strategy $\Sigma_2$, Bob can successfully cheat with probability $\Delta_2 \leq 2 \cdot e^{-xn/12} + \epsilon$ which is exponentially small in $n$.

**Probability of successfully cheating using any adaptive strategy $S$** As argued above, for any adaptive strategy $S$, the probability $\delta$ of cheating is upper bounded by $\delta_1 + \delta_2 \leq \Delta_1 + \Delta_2$ and hence exponentially small in $n$. □

**Gains in efficiency** As $k \geq n - 11xn$ for any small constant $x$, the expansion factor $n/k$ is $1 + \epsilon'$ for some (related) small constant $\epsilon'$. It is only slightly larger than the expansion factor in the case of Bit OT and XOR OT and remains asymptotically optimal. This represents an increase in efficiency by a factor of about 4.8188 over the corresponding reduction in [3].

### 6.3 Security against a dishonest Bob using Universal OT

In this case, in each round of Bit OT at Step 4 dishonest Bob can choose to obtain the output of any discrete, memoryless channel subject to the following constraint: let $B_0, B_1$ be independent, uniformly distributed random variables corresponding to Alice's inputs to Bit OT and let $\Omega = \Omega(B_0, B_1)$ be the channel's output to Bob. Then for some constant $\alpha \leq 1$ the following holds:

$$\mathbf{H}\left((B_0, B_1) \mid \Omega\right) \geq \alpha \tag{8}$$

Note that we require $\alpha$ to be at most 1 since otherwise, the channel would disallow honest behavior as well. Let $\epsilon$ to be any (very small) positive constant strictly less than $1/2$. We can then partition all possible channels satisfying the constraint of Equation 8 into the following three categories.

**$\Omega_0$:** All channels satisfying $\mathbf{H}(B_0 \mid \Omega) < \epsilon\alpha$ and $\mathbf{H}(B_1 \mid B_0\Omega) > (1-\epsilon)\alpha$.

**$\Omega_1$:** All channels satisfying $\mathbf{H}(B_1 \mid \Omega) < \epsilon\alpha$ and $\mathbf{H}(B_0 \mid B_0\Omega) > (1-\epsilon)\alpha$.

**$\Omega_b$:** All channels satisfying $\mathbf{H}(B_0 \mid \Omega), \mathbf{H}(B_1 \mid \Omega) \geq \epsilon\alpha$.

Let $\rho(\alpha)$ be the unique solution $x \in [0, 1/2]$ to the equation $h(x) = \alpha$. Let $p_0 = p_1 = \rho((1-\epsilon)\alpha)$ and $p_b = \rho(\epsilon\alpha)$. Then from Fano's inequality and Lemma 2 (Section 3.4) we can assert the following:

- $p_0$ is a lower bound on the error probability when guessing the value of $B_1$ after using a channel of type $\Omega_0$ and this is true even if the value of $B_0$ is known with certainty. There thus exists an indicator random variable $\Delta_0$ (provided as side information by an oracle) which leads to an erasure of $B_1$ with probability $2p_0$. Note: when there is no erasure ($\Delta_0 = 0$) it is not necessarily the case the corresponding bit is known with certainty.

– Likewise, $p_1$ lower bounds the error probability when guessing $B_0$ given the value of $B_1$ and the output of a channel of type $\Omega_1$. This implies the existence of side information in the form of an indicator random variable $\Delta_1$ that leads to an erasure of $B_0$ with probability $2p_1 = 2p_0$.
– When using a channel of type $\Omega_b$, the probability of guessing $B_0$ incorrectly given the channel's output is at least $p_b$, and the same holds when guessing the value of $B_1$. Thus, there exists an indicator random variable $\Delta_b^0$ (resp. $\Delta_b^1$) which, if provided by an oracle, would lead to an erasure of $B_0$ (resp. $B_1$) with probability $2p_b$. Note that this statement is true only if the oracle provides *one* of $\Delta_b^0, \Delta_b^1$ each time. To see why this is so, suppose both were provided at the same time. Since $\Delta_b^0$ along with $\Omega$ might contain more information about $B_1$ than was available in $\Omega$ alone, one can no longer assume that the event $\Delta_b^1 = 1$ would necessarily correspond to an erasure of $B_1$.

In order to simplify our analysis we will assume that after each round of UOT in Step 4, an oracle supplies Bob with the following side information, depending on the type of channel that Bob used:

**$\Omega_0$:** The exact value of $B_0$, as well as the value of $\Delta_0$. Note that this leads to $B_1$ being erased with probability $2p_0$.

**$\Omega_1$:** The exact value of $B_1$, as well as the value of $\Delta_1$. Note that this leads to $B_0$ being erased with probability $2p_1 = 2p_0$.

**$\Omega_b$:** One of $\Delta_b^0, \Delta_b^1$, chosen at random with equal probability. Note that this leads to each of $B_0, B_1$ being erased with probability $p_b$ in each round (not independently, though: $B_0$ and $B_1$ cannot be erased at the same time).

**Another modification to Protocol 4** For any very small positive constant $\epsilon$, let $p_b \stackrel{\text{def}}{=} \rho(\epsilon\alpha)$ and $p_0 \stackrel{\text{def}}{=} \rho((1-\epsilon)\alpha)$. Our proof of security will require that we reduce $k$ even further at step 11, by setting $k = 2p_0(j - 8p_b n) \geq 2p_0 n - 9p_0 p_b n$. For convenience, we will also set $x = p_b^2$ in Step 1.

**Theorem 7.** *The probability of dishonest Bob successfully cheating in (modified) Protocol 4 is exponentially small in $n$ even if the Bit OT protocol is replaced with UOT satisfying the constraint of Equation* (8)*.*

*Proof.* Let $v_0, v_1, v_b \subseteq I$ be the positions in Step 4 where Bob selected a channel of type $\Omega_0, \Omega_1, \Omega_b$, respectively. Then, at the end of Step 4 one of the following two cases always holds:

**Case 1:** One of $|v_0|, |v_1| > n - 6p_b n$

**Case 2:** Both $|v_0|, |v_1| \leq n - 6p_b n$

We proceed as in the proof of security for GOT in Section 6.2.

Without loss of generality, let $|v_0| > n - 6p_b n$ at the end of Step 4. This implies that at least $j - 6p_b n$ of the bits of $R_1$ were received over a channel of type $\Omega_0$. Let $\mu_1$ be the expected number of erasures in $R_1$, resulting from the side information $\Delta_0$ provided by the oracle in each round. Then $\mu_1 \geq 2p_0(j - 6p_b n)$. From Equation (3) we deduce that with probability exponentially close to 1 there will be at least $2p_0(j - 7p_b n)$ erasures, in which case $u_{1/2}(R_1) \geq 2p_0(j - 7p_b n)$.

Applying Privacy Amplification with output length $k = 2p_0(j - 8p_b n)$ will thus produce an almost-uniformly distributed $k$-bit string $r_1$ (independent of $r_0$), except with exponentially (in $n$) small probability. Note that as $p_b^3 < 1/2$ and $j \geq n - 2x^2 n = n - 2p_b^4 n$, the output size $k$ satisfies $k = 2p_0(j - 8p_b n) \geq 2p_0\left(n - 2p_b^4 n - 8p_b n\right) \geq 2p_0(n - 9p_b n) = 2p_0 n - 9p_0 p_b n$.

The probability of any strategy $\Sigma_1$ successfully cheating is at most equal to the probability that there are too few erasures to begin with plus the probability that Privacy Amplification failed to produce an almost-uniformly distributed string. Our choices guarantee that this probability is exponentially small in $n$.

We show that with near certainty both $u_{1/2}(T_0)$ and $u_{1/2}(T_1)$ are at least $5xn$, which by Lemma 5 guarantees that Bob will fail to clear Step 9 with probability exponentially close to 1. We start by upper bounding the probability that $u_{1/2}(T_1) \leq 5xn$. Since $|v_1| \leq n - 6p_b n$, there are at least $6p_b n$ bits that were either sent over a channel of type $\Omega_0$ or $\Omega_b$. We will assume that exactly $6p_b n$ bits were sent over a channel of type $\Omega_b$, as this choice minimizes the expected number of erasures in $T_1$ given our constraints, and hence maximizes the probability that $u_{1/2}(T_1) \leq 5xn$. Note that the expected number of erasures of $B_1$ in this case is $p_b \cdot 6p_b n = 6p_b^2 n = 6xn$. By the Chernoff bound

$$\mathbf{Pr}\left[u_{1/2}(T_1) \leq 5xn\right] \leq \mathbf{Pr}\left[B(6p_b n, p_b) \leq 5p_b^2 n\right] \leq \lambda$$

where $\lambda$ is exponentially small in $n$.

The same argument applies to $u_{1/2}(T_0)$. Therefore, both $u_{1/2}(T_0), u_{1/2}(T_1) \geq 5xn$ except with probability at most $2\lambda$. Then Lemma 5 guarantees that Bob will fail to clear Step 9 with probability $1 - \epsilon'$ for some $\epsilon'$ exponentially small in $n$. We conclude that using any Strategy $\Sigma_2$, Bob can successfully cheat with probability at most $2\lambda + \epsilon'$ which is exponentially small in $n$.

**Probability of cheating using any adaptive strategy $S$** As argued in Section 6.2, the probability of successful cheating for any adaptive strategy $S$ is upper bounded by the sum of the probabilities of success of any strategies $\Sigma_1, \Sigma_2$. We have shown that both of these are exponentially small. $\square$

**Gains in efficiency** In both our reduction and that of [3], the expansion factor is a function of $\alpha$. In our case $k \geq 2p_0 n - 9p_0 p_b n$. Since $p_b = \rho(\epsilon\alpha), p_0 = \rho((1-\epsilon)\alpha)$, for $\epsilon \to 0$ we get $p_0 \to \rho(\alpha), p_b \to 0$ and therefore $k \approx 2\rho(\alpha)n$, which translates to an expansion factor of $\frac{1}{2\rho(\alpha)} + \epsilon'$. The corresponding expansion factor in [3] is at least $\frac{4\ln 2}{p_e}$ where $p_e$ is the unique solution in $(0, 1/2]$ to the equation $h(p_e) + p_e \log_2 3 = \alpha$. It is easy to verify by means of a graph that for all $0 \leq \alpha \leq 1$, we have $\rho(\alpha) > p_e$. Consequently, our expansion factor is always at least $8\ln 2 = 5.545$ times smaller than the one in [3]. It is noteworthy that in the special case where $\alpha = 1$ we have $\rho(\alpha) = 1/2$ and therefore the expansion factor is $1 + \epsilon'$, which is optimal. Proving optimality for other values of $\alpha$ is left as an open problem.

## 7 Conclusions, and open problems

We have demonstrated how the properties of Interactive Hashing can be exploited to increase the efficiency and generality of existing String OT reductions. Specifically, we have shown that our reductions are optimal in the case of Bit OT, XOT and GOT, as well as for the special case of UOT where $\alpha = 1$. We conclude by listing some problems that our current work leaves open. **(1)** Modify Protocol 4 so that it never aborts when both participants are honest. This will require proving that Interactive Hashing would not allow a dishonest Bob to obtain strings $w_0$, $w_1$ such that the corresponding subsets $s_0$, $s_1$ have a large intersection. **(2)** Prove that our reduction is optimal for all $\alpha$ in the case of UOT, or modify it accordingly to achieve optimality. **(3)** Replace the Interactive Hashing Protocol (Protocol 2) with an appropriately adapted implementation of the constant round Protocol of [6] and prove that the ensuing reduction (Protocol 4) remains secure. **(4)** Further explore the potential of Interactive Hashing as an ingredient in cryptographic protocols design.

## References


1. C. H. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized privacy amplification. *IEEE Trans. on Info. Theory*, 41(6):1915–1923, November 1995.
2. C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, 1988.
3. G. Brassard, C. Crépeau, and S. Wolf. Oblivious transfers and privacy amplification. *IEEE Transaction on Information Theory*, 16(4):219–237, 2003.
4. G. Brassard, C. Crépeau, and M. Santha. Oblivious transfers and intersecting codes. *IEEETIT: IEEE Transactions on Information Theory*, 42, 1996.
5. C. Cachin, C. Crépeau, and J. Marcil. Oblivious transfer with a memory-bounded receiver. In *IEEE Symposium on Foundations of Computer Science*, 1998.
6. Y. Zong Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *TCC*, pages 446–472, 2004.
7. Y. Dodis and S. Micali. Lower bounds for oblivious transfer reductions. *Lecture Notes in Computer Science*, 1592, 1999.
8. S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.
9. Joe Kilian. Founding crytpography on oblivious transfer. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 20–31, New York, NY, USA, 1988. ACM Press.
10. M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *Journal of Cryptology*, 11(2), 1998.
11. R. Ostrovsky, R. Venkatesan, and M. Yung. Fair games against an all-powerful adversary. *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 13, 1993.
12. M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Memo TR–81, Aiken Computation Laboratory, Harvard University, 1981.
13. D.R. Stinson. Some results on nonlinear zigzag functions. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 29:127–138, 1999.
14. S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.