Interactive Hashing: An Information Theoretic Tool (Invited Talk)

Claude Crépeau[1,⋆], Joe Kilian[2,⋆⋆], and George Savvides[3,⋆⋆⋆]

[1] McGill University, Montréal, QC, Canada
crepeau@cs.mcgill.ca
[2] Rutgers University, New Brunswick, NJ, USA
jkilian@cs.rutgers.edu
[3] European Patent Office, München, Germany
gsavvides@gmail.com

Abstract. Interactive Hashing has featured as an essential ingredient in protocols realizing a large variety of cryptographic tasks, notably Oblivious Transfer in the bounded memory model. In Interactive Hashing, a sender transfers a bit string to a receiver such that two strings are received, the original string and a second string that appears to be chosen at random among those distinct from the first.

This paper starts by formalizing the notion of Interactive Hashing as a cryptographic primitive, disentangling it from the specifics of its various implementations. To this end, we present an application-independent set of information theoretic conditions that all Interactive Hashing protocols must ideally satisfy. We then provide a standard implementation of Interactive Hashing and use it to reduce a very standard version of Oblivious Transfer to another one which appears much weaker.

1 Introduction

Interactive Hashing (IH) is a cryptographic primitive that allows a sender Alice to send a bit string w to a receiver Bob who receives two output strings, labeled w_0, w_1 according to lexicographic order. The primitive guarantees that one of the two outputs is equal to the original input. The other string is guaranteed to be effectively random, in the sense that it is chosen beyond Alice's control, even if she acts dishonestly. On the other hand, provided that from Bob's point of view w_0, w_1 are a priori equiprobable inputs for Alice, the primitive guarantees that Bob cannot guess which of the two was the original input with probability greater than $1/2$. We remark that typically both outputs are also available to Alice. See Figure 1.

In this article we provide a study of Interactive Hashing in the information theoretic setting and in isolation of any surrounding context. This modular approach

[⋆] Supported in part by NSERC, MITACS, CIFAR, and QuantumWorks.

[⋆⋆] Some of this research was done while the author worked for NEC Research.

[⋆⋆⋆] This research was done while the author was a student at McGill University.

R. Safavi-Naini (Ed.): ICITS 2008, LNCS 5155, pp. 14–28, 2008.

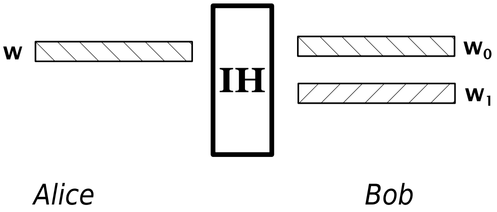

Fig. 1. Interactive Hashing: the sender Alice sends string w to Bob, who receives two strings w_0, w_1, labeled according to lexicographic order. One of the two (in our example, w_0) is equal to the input string while the other is effectively randomly chosen. Bob cannot distinguish which of the two was the original input.

allows specific implementations (protocols) of Interactive Hashing to be analyzed independently of any applications in which they appear as sub-protocols. It thus leads to a better appreciation of the power of Interactive Hashing as a *cryptographic primitive* in its own right.

To demonstrate the relevance of Interactive Hashing, we present an application to protocols for Oblivious Transfer (OT). Oblivious Transfer is an important primitive in modern cryptography. It was originally studied by Wiesner [Wie70] (under the name of "multiplexing"), in a paper that marked the birth of quantum cryptography and was later independently introduced to cryptography in several variations by Rabin [Rab81] and by Even, Goldreich and Lempel [EGL85]. Oblivious transfer has since become the basis for realizing a broad class of cryptographic protocols, such as bit commitment, zero-knowledge proofs, and general secure multiparty computation [Yao86, GMW87, Kil88, Gol04].

In a one-out-of-two Oblivious Transfer, denoted $\binom{2}{1}$-OT, a sender owns two secret bits b_0 and b_1, and a receiver wants to learn b_c for a secret bit c of his choice. The sender will only collaborate if the receiver can obtain information about exclusively one of b_0 or b_1. Likewise, the receiver will only participate provided that the sender cannot obtain any information about c.

1.1 Organization of the Paper

We present the previous work on Interactive Hashing in Section 2. In Section 3 we identify and formalize the information theoretic security properties of Interactive Hashing. Then, in Section 3.1 we turn our attention to the Interactive Hashing implementation that appeared as a sub-protocol in [OVY93] and refer the reader to recent work [Sav07, CCMS09] demonstrating that despite its simplicity, it meets all security properties set forth in Section 3. This new proof of security is an important improvement over the proof that appeared in [CCM98], where the authors demonstrate that a slight variant of the IH protocol of [OVY93] could be securely used in their specific scenario. The new proof is more general, as it is based on the security properties stated in Section 3. Moreover, the proof is significantly simpler and more intuitive. Lastly, it provides an easier to use and much tighter upper bound on the probability that the protocol fails to ensure that one of the two strings is sufficiently random. Section 4 defines our example problem: reducing $\binom{2}{1}$-OT to a very weak version of Oblivious Transfer. Section 5 exhibits the solution to our example problem using Interactive Hashing. Finally, we conclude in Section 6 and introduce a few open problems.

2 Previous Work

Various implementations of Interactive Hashing have appeared as sub-protocols in the cryptographic literature, first in computational contexts where at least one of the participants is polynomially bounded and later also in contexts where security is unconditional (information theoretic).

While reviewing the previous work, the reader should bear in mind that so far, Interactive Hashing has never been presented as an independent primitive. Instead, it only appears within the context of larger protocols achieving a variety of different cryptographic tasks. Not surprisingly, the properties it is expected to have can vary significantly from one application to the next, and thus the proof of security in each case depends on the specific setting.

2.1 Uses of Interactive Hashing in Computational Contexts

Interactive Hashing first appeared as a sub-protocol within a protocol achieving Oblivious Transfer from an unbounded sender to a polynomial-time bounded receiver [OVY93]. Soon thereafter, Interactive Hashing was deployed in various other scenarios, such as zero-knowledge proofs [OVY94] and bit commitment schemes [OVY92, NOVY98], where at least one of the participants was computationally bounded. For more recent applications of Interactive Hashing in this setting consult [HHK$^+$05, NOV06, NV06, HR07].

2.2 Uses of Interactive Hashing in Information Theoretic Contexts

Beside the computational scenarios in which it was originally used, Interactive Hashing proved to be an important tool in information theoretic contexts as well. Its first such use was in protocols for Oblivious Transfer which are information-theoretically secure under the sole assumption that the receiver's memory is bounded [CCM98, Din01, DHRS07]. Interactive Hashing was later used to optimize reductions between Oblivious Transfer variants [CS06].

We remark that while some of the security properties required of Interactive Hashing in information theoretic settings bear a very close resemblance to their counterparts in computational settings, some other properties are substantially different. Moreover, the transition from computational to information theoretic settings requires a re-evaluation of *all* security properties of any protocol. For this reason, starting with [CCM98], the security properties of the underlying Interactive Hashing sub-protocol have been re-evaluated in the light of the specific, information theoretic context where it was used.

3 Information-Theoretic Secure Interactive Hashing

We now formalize the security properties that Interactive Hashing is expected to satisfy in information theoretic contexts. As these properties do not depend on any specific application, they allow us to define Interactive Hashing as an independent cryptographic primitive.

Definition 1. Interactive Hashing *is a cryptographic primitive between two players, the sender and the receiver. It takes as input a string* $w \in \{0,1\}^t$ *from the sender, and produces as output two* t*–bit strings one of which is* w *and the other* $w' \neq w$*. The output strings are available to both the sender and the receiver, and satisfy the following properties:*

1. The receiver cannot tell which of the two output strings was the original input. *Let the two output strings be* w_0, w_1*, labeled according to lexicographic order. Then if both strings were a priori equally likely to have been the sender's input* w*, then they are a posteriori equally likely as well*[1]*.*
2. When both participants are honest, the input is equally likely to be paired with any of the other strings. *Let* w *be the sender's input and let* w' *be the second output of interactive hashing. Then provided that both participants follow the protocol,* w' *will be uniformly distributed among all* $2^t - 1$ *strings different from* w*.*
3. The sender cannot force both outputs to have a rare property*. Let* $\mathcal{G}$ *be a subset of* $\{0,1\}^t$ *representing the sender's "good set". Let* G *be the cardinality of* $\mathcal{G}$ *and let* $T = 2^t$*. Then if* ${}^G/_T$ *is "small", the probability that a dishonest sender will succeed in having both outputs* w_0, w_1 *be in* $\mathcal{G}$ *is comparably "small".*

Remark 1. In the computational contexts of Section 2.1, similar properties to Properties 1 and 2 were also required. On the other hand, the computational counterpart to Property 3 is usually stated quite differently, as there is no predetermined good set $\mathcal{G}$. For instance, in [NOVY98] where the inputs and outputs of Interactive Hashing are interpreted as images under a one-way permutation π, one of the two outputs is required to be sufficiently random so that any polynomial-time algorithm that can compute pre-images to both outputs a significant fraction of the time can be used to efficiently invert π on a randomly chosen string with non-negligible probability.

We shall also point out that Property 3 is easy to satisfy when $G \in o(\sqrt{T})$ because of the so called Birthday paradox. If the receiver picks a random hash function h from $\{0,1\}^t \rightarrow \{0,1\}^{t-1}$ and announces it to the sender, only with very small probability will there exist a pair $w_0, w_1 \in \mathcal{G}$ such that $h(w_0) = h(w_1)$. The real challenge, met by Interactive Hashing, is to obtain Property 3 for sets $\mathcal{G}$ such that $G \in \Omega(\sqrt{T})$.

[1] Note that if we want this property to hold for all possible outputs, then w must be uniformly chosen. Otherwise, this property will only hold whenever w happens to be paired with a string w' having the same a priori probability as w.

3.1 A Secure Protocol for Interactive Hashing

We will be examining the implementation of Interactive Hashing given in Protocol 1. This standard implementation was originally introduced in a computational context by Ostrovsky, Venkatesan, and Yung [OVY93]. In Section 3.1 we will see that this very simple protocol actually meets all our information theoretic security requirements as well.

Protocol 1. Interactive Hashing

Let w be a t-bit string that the sender wishes to send to the receiver. All operations below take place in the binary field $\mathcal{F}_2$.

1. The receiver chooses a $(t-1) \times t$ matrix Q uniformly at random among all binary matrices of rank $t-1$. Let q_i be the i^{th} query, consisting of the i^{th} row of Q.
2. For $1 \leq i \leq t-1$ do:
   (a) The receiver sends query q_i to the sender.
   (b) The sender responds with $c_i = q_i \cdot w$.
3. Given Q and c (the vector of Bob's responses), both parties compute the two values of w consistent with the linear system $Q \cdot w = c$. These solutions are labeled w_0, w_1 according to lexicographic order.

Remark 2. One way of choosing the matrix Q is to choose a $(t-1) \times t$ binary matrix uniformly at random and test whether it has rank $t-1$, repeating the process if necessary. Note that a later variation of the protocol [NOVY98] chose Q in a canonical way to guarantee that it has rank $t-1$, which results in a somewhat more practical implementation. However, this appears to complicate the proof of security.

Theorem 1 establishes the security of Protocol 1.

Theorem 1. *[Sav07, CCMS09] Protocol 1 satisfies all three information theoretic security properties of Definition 1. Specifically, for Property 3, it ensures that a dishonest sender can succeed in causing both outputs to be in the "good set" $\mathcal{G}$ with probability at most* $15.6805 \cdot {}^{G}/_{T}$.

3.2 Proofs of Information Theoretic Security

Cachin, Crépeau, and Marcil [CCM98] proved a similar property to Property 3 for a slight variant of Protocol 1 in the context of memory-bounded Oblivious Transfer where again, the goal of a dishonest sender is to force both outputs of the protocol to be from a subset $\mathcal{G}$ of cardinality G (out of a total $T = 2^t$). While their approach relies on upper-bounding the number of the sender's remaining good strings during the various rounds of the protocol, the new proof of [Sav07, CCMS09] focuses instead on following the evolution of the number of *pairs* of good strings remaining after each round. This seems to be a more natural choice for this scenario, as there is exactly one such pair remaining at the end of the protocol if the sender succeeds in cheating and none otherwise (as opposed to two strings versus zero or one). Consequently, the probability of cheating is simply equal to the expected number of remaining pairs. Thanks to the nature of the protocol, it is relatively easy to establish an upper bound on the expected number of remaining pairs after each incoming query, and to keep track of its evolution through the protocol.

The new approach of [Sav07, CCMS09] not only leads to a simpler and more robust proof of security, but more importantly, it also allows to establish a more general and much tighter upper bound on a dishonest sender's probability of cheating. Specifically, it allows to show that any strategy a dishonest sender might employ can succeed with probability no larger than $15.6805 \cdot {}^{G}\!/_{T}$, for all fractions G/T of good strings. The corresponding upper bound in [CCM98] is $\sqrt{2} \cdot \sqrt[8]{{}^{G}\!/_{T}}$ and is only valid provided that ${}^{G}\!/_{T} < \left(16t^8\right)^{-1}$. It should be noted that the new upper bound is in fact tight up to a small constant. Indeed, the probability of succeeding in cheating using an optimal strategy is lower-bounded by the probability of getting two good output strings when the sender chooses $w \in \mathcal{G}$ as input and then acts honestly. By Property 2 of Interactive Hashing, w is equally likely to be paired with any of the remaining strings. It follows that the probability of w being paired with one of the other $G-1$ good strings is exactly ${}^{G-1}\!/_{T-1}$. Assuming that $G \geq 50$, the new upper bound is larger than this lower bound by a factor of at most $15.6805 \cdot \left(\frac{G}{T}\right)\left(\frac{T-1}{G-1}\right) < 15.6805\left(\frac{G}{G-1}\right) \leq 16$. This establishes that the new upper bound is tight up to a small constant in all cases where the possibility of cheating exists.

3.3 An Alternative Implementation

Ding *et al.* [DHRS07] make use of a new, constant-round Interactive Hashing protocol to achieve Oblivious Transfer with a memory-bounded receiver. The main idea behind their protocol, which requires only four rounds of interaction (compared to $t-1$ rounds in Protocol 1), is that if the receiver sends a random permutation π to the sender (Round 1) who then applies it to his input string w and announces a certain number of bits of $\pi(w)$ (Round 2), then two more rounds suffice to transmit the remaining part of $\pi(w)$ so that only 1 bit remains undetermined: in Round 3, the receiver chooses a function g uniformly at random from a family of 2–wise independent 2–1 hash functions, and in Round 4 the sender announces the value of the function applied to the remaining bits of $\pi(w)$. The output of the Interactive Hashing protocol consists of the two possible inputs to the permutation π consistent with the values transmitted at rounds 2 and 4. The security of this scheme is based on the observation that the permutation π in the first round divides the (dishonest) sender's good set $\mathcal{G}$ into buckets (indexed by the bits transmitted at Round 2), so that with high probability, in each bucket the fraction of good strings is below the Birthday Paradox threshold. This allows regular 2–1 hashing to be used in Rounds 3 and 4 to complete the protocol.

It should be noted that since a random permutation would need exponential space to describe, the construction resorts to *almost t-wise independent permutations*, which can be efficiently constructed and compactly described.

Unfortunately, the protocol of [DHRS07] is less general than Protocol 1 for a variety of reasons: first, its implementation requires that the two parties know a priori an upper bound on the cardinality of the dishonest receiver's good set $\mathcal{G}$, as this will determine the number of bits of $\pi(w)$ announced in Round 2. Secondly, the upper bound for the probability that Property 3 is not met is, according to the authors' analysis, $\Omega\left(t \cdot {}^{G}\!/_{T}\right)$ and only applies when $G \geq 4t$. Moreover, the protocol does not fully satisfy Property 2, but only a slight relaxation[2] of it. Lastly, the protocol is very involved, and probably prohibitively complicated to implement in practice. We leave it as an open problem to improve upon this construction.

4 Reducing OT to a Very Weak OT

We illustrate the power of Interactive Hashing in information theoretic contexts by considering the following straightforward scenario, originally suggested by the second author: suppose that a sender Alice and a receiver Bob wish to implement 1-out-of-k Bit Oblivious Transfer, which we will denote as $\binom{k}{1}$–Bit OT. For the purposes of our example, suffice it to say that Alice would like to make available k randomly chosen bits to Bob, who must be able to choose to learn any one of them, with all choices being equally likely from Alice's point of view. Alice is only willing to participate provided that (dishonest) Bob learns information about exclusively one bit, while Bob must receive the assurance that (dishonest) Alice cannot obtain any information about his choice. Suppose that all that is available to Alice and Bob is an insecure version of $\binom{k}{1}$–Bit OT, denoted $(k-1)$–faulty $\binom{k}{1}$–Bit OT, which allows honest Bob to receive (only) one bit of his choice but might allow a dishonest Bob to learn up to $k-1$ bits of his choice. The rest of this section focuses on the early work of the first two authors who had made repeated but unsuccessful attempts to find a satisfactory reduction of $\binom{k}{1}$–Bit OT to $(k-1)$–faulty $\binom{k}{1}$–Bit OT, whereas Protocol 4 shows how Interactive Hashing makes such a reduction almost trivial.

Remark 3. For simplicity, Protocol 2 and Protocol 4 reduce $\binom{2}{1}$–Bit OT to weaker versions of OT without any loss of generality since $\binom{k}{1}$–Bit OT can in turn be reduced to $\binom{2}{1}$–Bit OT using the well-known reduction in [BCR86]. We shall denote "$x{+_k}y$" to be "$x{+}y \bmod k$" except if $x{+}y \equiv 0 \pmod{k}$ in which case "$x{+_k}y = k$". More formally, $x +_k y = (x + y - 1 \bmod k) + 1$.

4.1 Reduction of $\binom{2}{1}$–Bit OT to $O(\sqrt{k})$–Faulty $\binom{k}{1}$–Bit OT

As a warm up exercise we exhibit a simple reduction of $\binom{2}{1}$–Bit OT to $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT, a faulty primitive, allowing a dishonest Bob to get at most $O(\sqrt{k})$ bits of Alice's input at his choosing.

[2] It approximates the uniform distribution over the remaining strings within some $\eta < 2^{-t}$.

Protocol 2. Reduction of $\binom{2}{1}$–Bit OT to $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT

Let $\mathring{b}_0, \mathring{b}_1$ and $\mathring{c}$ be the inputs of Alice and Bob, respectively, for $\binom{2}{1}$–Bit OT.

1. Alice and Bob agree on a security parameter n.
2. For $1 \leq i \leq n$ do:
   (a) Alice selects at random bits $r_{i1}, r_{i2}, \ldots, r_{ik}$ while Bob selects at random $c_i \in_{\mathrm{R}} \{1, \ldots, k\}$.
   (b) Alice uses $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT to send her k bits to Bob, who chooses to learn r_{ic_i}.
   (c) Alice picks a random distance $\Delta_i \in_{\mathrm{R}} \{1, \ldots, k/2\}$ and announces it to Bob.
   (d) Bob announces σ_i such that $c_i = \sigma_i +_k \mathring{c}\Delta_i$ to Alice.
3. Alice computes $R_0 = \bigoplus_{i=1}^{n} r_{i\sigma_i}$ and $R_1 = \bigoplus_{i=1}^{n} r_{i(\sigma_i +_k \Delta_i)}$.
4. Alice sends $e_0 = \mathring{b}_0 \oplus R_0$ and $e_1 = \mathring{b}_1 \oplus R_1$ to Bob.
5. Bob obtains $\mathring{b}_{\mathring{c}} = e_{\mathring{c}} \oplus R_{\mathring{c}} = e_{\mathring{c}} \oplus \bigoplus_{i=1}^{n} r_{ic_i}$.

It is relatively straightforward to see that when both participants are honest, Protocol 2 allows Bob to obtain the bit of his choice since he knows $R_{\mathring{c}} = \bigoplus_{i=1}^{n} r_{ic_i}$ and can thus decrypt $e_{\mathring{c}}$. In case Alice is dishonest, Bob's choice $\mathring{c}$ is perfectly hidden from her when she obtains σ_i at Step 2d. This is because at the beginning of the protocol, Bob is equally likely to make the choices σ_i or $\sigma_i +_k \Delta_i$.

Now consider what a dishonest Bob can do. At round i, upon learning Δ_i in Step 2c, the probability that there exists a pair of indices at distance Δ_i where Bob knows both bits is less than $\frac{\ell_i(\ell_i-1)/2}{k/2}$ when Bob knows ℓ_i bits out of k. This is because the maximum number of distances possible between ℓ_i positions is $\ell_i(\ell_i-1)/2$, while the total number of distances is $k/2$. Thus, for an appropriate choice of the hidden constant in the $O()$ notation we have $\frac{O\left(\sqrt{k}(\sqrt{k}-1)/2\right)}{k/2} < 1/2$. In consequence, the probability that in Step 2d Bob is able to claim a σ_i such that he knows both $r_{i\sigma_i}$ and $r_{i(\sigma_i +_k \Delta_i)}$ is less than $1/2$. See Figure 2 for an example. Therefore, the probability that after n rounds Bob may compute both R_0 and R_1 is less than $1/2^n$.

4.2 Reduction of $O(\sqrt{k})$–Faulty $\binom{k}{1}$–Bit OT to $(k/2)$–Faulty $\binom{k}{1}$–Bit OT

As a continuation of the previous exercise we reduce $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT to $(k/2)$–faulty $\binom{k}{1}$–Bit OT, a faulty primitive allowing a dishonest Bob to get at most $k/2$ bits of Alice's input at his choosing.

It is again relatively straightforward to see that when both participants are honest, Protocol 3 allows Bob to obtain the bit of his choice since he knows

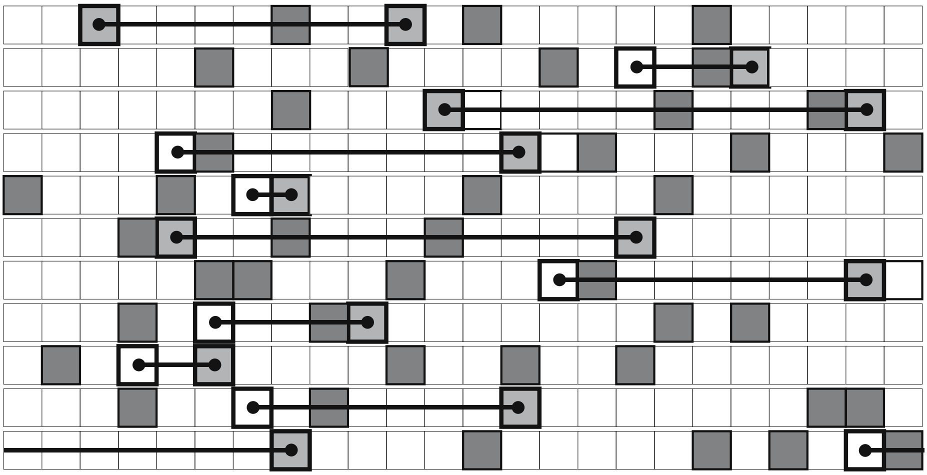

Fig. 2. $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT: Each row i corresponds to a round and in each row $O(\sqrt{k})$ grey squares indicate the positions obtained by a dishonest Bob. The bold lines indicate the distance Δ_i chosen by Alice. Bob can obtain both bits in the end if a pair of grey squares exists at the right distance in each row. We see that a few rows have such a pair but many don't.

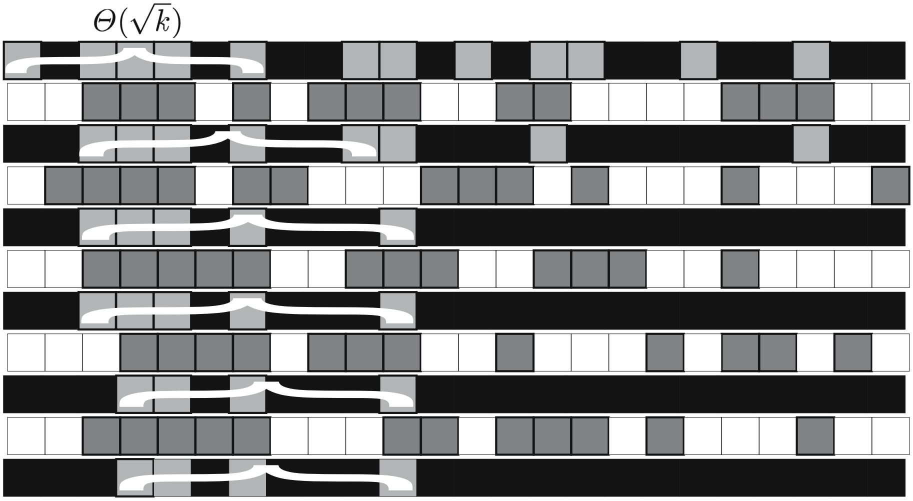

Fig. 3. $(k/2)$–faulty $\binom{k}{1}$–Bit OT: Each two rows $2i-1, 2i$ correspond to round i. Row $2i-1$ shows the number of bits known to dishonest Bob (in light grey). Each row $2i$, shows an execution of $(k/2)$–faulty $\binom{k}{1}$–Bit OT after mixing via π_i, and shifting via σ_i to align as many known bits (in darker grey) as possible in the first $\Theta(\sqrt{k})$ positions. Most of the times, it is not possible to save all the $\Theta(\sqrt{k})$ known bits.

Protocol 3. Reduction of $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT to $(k/2)$–faulty $\binom{k}{1}$–Bit OT

1. Alice and Bob agree on a security parameter n.
2. Bob selects at random $c \in_{\mathrm{R}} \{1, \ldots, k\}$.
3. For $1 \leq i \leq 2n$ do:
   (a) Alice selects at random bits $r_{i1}, r_{i2}, \ldots, r_{ik}$ while Bob selects at random $c_i \in_{\mathrm{R}} \{1, \ldots, k\}$.
   (b) Alice uses $(k/2)$–faulty $\binom{k}{1}$–Bit OT to send her k bits to Bob, who chooses to learn r_{ic_i}.
   (c) Alice picks a random permutation $\pi_i \in_{\mathrm{R}} \{1, \ldots, k\} \to \{1, \ldots, k\}$ and announces it to Bob.
   (d) Bob computes a shift σ_i such that $\pi_i(c_i) = \sigma_i +_k c$ and announces it to Alice.
4. Alice computes For $1 \leq j \leq k$

$$R_j = \bigoplus_{i=1}^{2n} r_{i\pi_i^{-1}(\sigma_i +_k j)}.$$

5. Bob outputs c and $R_c = \bigoplus_{i=1}^{2n} r_{ic_i}$.
6. Alice outputs $R_1, \ldots, R_k$.

$R_c = \bigoplus_{i=1}^{2n} r_{ic_i}$. In case Alice is dishonest, Bob's choice c is perfectly hidden from her when she obtains σ_i at Step 3d.

The rest of the reasoning is a bit more subtle. See Figure 3 for an example. Consider the first $\Theta(\sqrt{k})$ bits known by Bob. The number of sequences containing $k/2$ known bits that will have exactly those $\Theta(\sqrt{k})$ bits in the correct position is given by

$$\binom{k - \Theta(\sqrt{k})}{k/2} < \binom{k - \Theta(\sqrt{k})}{(k - \Theta(\sqrt{k}))/2} \approx \sqrt{\frac{2}{\pi}} \frac{2^{k-\Theta(\sqrt{k})}}{\sqrt{k - \Theta(\sqrt{k})}}.$$

All k shifts of these sequences are also successful for Bob because he can shift them to align them with the first $\Theta(\sqrt{k})$ bits known, thus a grand total of at most k times more or $\sqrt{\frac{2}{\pi}}\sqrt{k + \Theta(\sqrt{k})}2^{k-\Theta(\sqrt{k})}$. However, any new execution of $(k/2)$–faulty $\binom{k}{1}$–Bit OT combined with a random permutation π_i yields a completely random sequence with an equal number of bits known and unknown, or one out of $\binom{k}{k/2} \approx \sqrt{\frac{2}{\pi}} \frac{2^k}{\sqrt{k}}$. So the probability that a random sequence can be shifted to have the first $\Theta(\sqrt{k})$ known bits in the correct positions is at most the ratio of the two expressions:

$$\frac{k\binom{k-\Theta(\sqrt{k})}{k/2}}{\binom{k}{k/2}} < \frac{\sqrt{k + \Theta(\sqrt{k})}2^{k-\Theta(\sqrt{k})}}{2^k/\sqrt{k}} < O(k)2^{-\Theta(\sqrt{k})} \ll 1/2.$$

We assume that the number of bits known to Bob after the first i rounds is in $\Omega(\sqrt{k})$ (a position j is *known* to Bob if so far he obtained all the bits necessary to later compute R_j), otherwise we have already achieved our goal. For $n > k$, starting from $k/2$ known bits, and repeating the protocol $2n$ times, one of the following two options must hold:

1. At some round, Bob is left with less than $O(\sqrt{k})$ known bits
2. At all rounds, Bob has $\Omega(\sqrt{k})$ bits left, and has thus lost fewer than $k/2$ bits overall (unlikely since under these conditions, the expected number of bits lost is $n > k$)

This guarantees that the total number of bits still valid at the end of the protocol is definitely $O(\sqrt{k})$ except with exponentially small probability. Thus, this reduction can be used as a substitute for $O(\sqrt{k})$–faulty $\binom{k}{1}$–Bit OT in Protocol 2.

The combination of Protocol 2 and Protocol 3 is a $\Theta(n^2)$ time reduction from $\binom{2}{1}$–Bit OT to $(k/2)$–faulty $\binom{k}{1}$–Bit OT. However, it is easy to see that it will fail completely if we start with $(k-1)$–faulty $\binom{k}{1}$–Bit OT instead of $(k/2)$–faulty $\binom{k}{1}$–Bit OT. This is because in each execution of step 3c the resulting sequence will be a run of $k-1$ known bits. In this situation Bob is able to choose a shift σ_i such that he *never* loses a single bit through the operations of Step 4.

We finally note that indeed for any $\epsilon < 1$, if dishonest Bob obtains ϵk bits per transfer, xoring two transfers, after permuting and shifting as in Protocol 3, transfers on average $\epsilon^2 k$ instead of ϵk. We may thus claim that the combined transfer produces at most $\epsilon' k$ known bits, for $\epsilon' = \frac{\epsilon^2+\epsilon}{2} < \epsilon$, except with exponentially small probability. Repeating this idea at most a constant number of times produces a resulting $\epsilon' < 1/2$. Since the sequence $\epsilon > \epsilon' > \epsilon'' > ...$ converges to zero, using a constant extra amount of work we can extend the result established for $\epsilon = 1/2$ to any $\epsilon < 1$. This was the state of affairs until information theoretic Interactive Hashing was considered as a tool to solve this problem.

5 Reducing to $(k-1)$–Faulty $\binom{k}{1}$–Bit OT Using Interactive Hashing

Finally, we reduce $\binom{2}{1}$–Bit OT to $(k-1)$–faulty $\binom{k}{1}$–Bit OT, a faulty primitive allowing a dishonest Bob to get at most $k-1$ bits of Alice's input at his choosing. For simplicity, we will also assume that k is a power of 2.

It is relatively straightforward to see that when both participants are honest, Protocol 4 allows Bob to obtain the bit of his choice since he knows $R_d = \bigoplus_{i=1}^{n} r_{ic_i}$ and can thus decrypt $e_{\mathring{c}}$. In case Alice is dishonest, Bob's choice $\mathring{c}$ is perfectly hidden from her when she obtains f at Step 6. This is because at the beginning of the protocol, Bob is equally likely to make the choices encoded by w_0 as those encoded by w_1. Consequently, by Property 1 of Interactive Hashing, given the specific outputs, the probability of either of them having been the original input is exactly $1/2$. Hence d is uniformly distributed from Alice's point of view and so $f = d \oplus \mathring{c}$ carries no information about $\mathring{c}$.

Protocol 4. Reduction of $\binom{2}{1}$–Bit OT to $(k-1)$–faulty $\binom{k}{1}$–Bit OT

Let $\mathring{b}_0, \mathring{b}_1$ and $\mathring{c}$ be the inputs of Alice and Bob, respectively, for $\binom{2}{1}$–Bit OT.

1. Alice and Bob agree on a security parameter n.
2. For $1 \leq i \leq n$ do:
   (a) Alice selects at random bits $r_{i1}, r_{i2}, \ldots, r_{ik}$.
   (b) Alice uses $(k-1)$–faulty $\binom{k}{1}$–Bit OT to send her k bits to Bob, who chooses to learn r_{ic_i} for a randomly selected $c_i \in_{\mathrm{R}} \{1, \ldots, k\}$. .
3. Bob encodes his choices during the n rounds of 2b as a bit string w of length $n \cdot \log(k)$ by concatenating the binary representations of $c_1, c_2, \ldots, c_n$.
4. Bob sends w to Alice using Interactive Hashing. Let w_0, w_1 be the output strings labeled according to lexicographic order, and let $d \in \{0, 1\}$ be such that $w = w_d$.
5. Let $p_1, p_2, \ldots, p_n$ be the positions encoded in w_0 and let $q_1, q_2, \ldots, q_n$ be the positions encoded in w_1. Alice computes $R_0 = \bigoplus_{i=1}^{n} r_{ip_i}$ and $R_1 = \bigoplus_{i=1}^{n} r_{iq_i}$.
6. Bob sends $f = d \oplus \mathring{c}$ to Alice.
7. Alice sends $e_0 = \mathring{b}_0 \oplus R_f$ and $e_1 = \mathring{b}_1 \oplus R_{\bar{f}}$ to Bob.
8. Bob decodes $\mathring{b}_{\mathring{c}} = e_{\mathring{c}} \oplus R_{f \oplus \mathring{c}} = e_{\mathring{c}} \oplus R_d$.

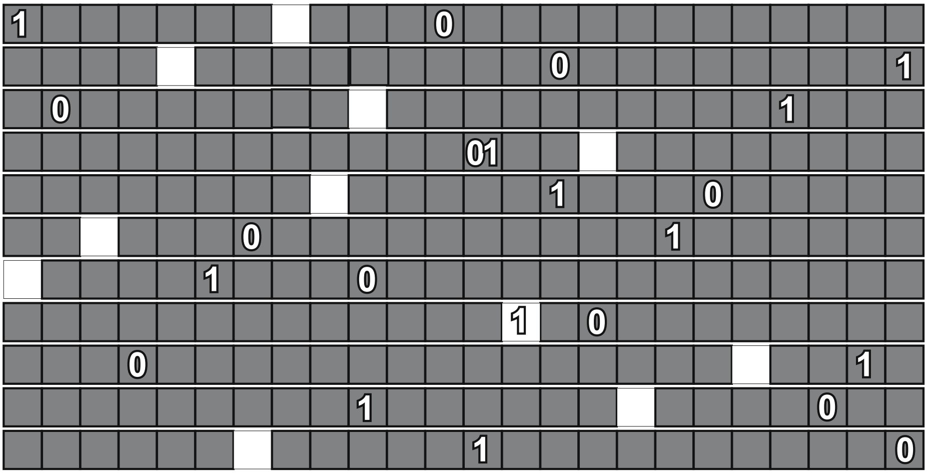

Fig. 4. $(k-1)$–faulty $\binom{k}{1}$–Bit OT: using Interactive Hashing Bob chooses two sequences of indices labelled with "zeros" and "ones". One of them corresponds to the sequence he knows (in the case where he is honest) while the second is the result of Interactive Hashing. Except with exponentially small probability, even if Bob is dishonest, one of the sequences will contain a missing (white) bit (a "one" in this example). Note that both "zero" and "one" may end up in the same location, once in a while, which is not a problem.

As for the case where Bob is dishonest, we can assume that he always avails himself of the possibility of cheating afforded by $(k-1)$–faulty $\binom{k}{1}$–Bit OT, and obtains $k-1$ out of k bits every time. Even so, though, by the end of Step 2, it is always the case that the fraction of all good encodings among all k^n possible encodings of positions is no larger than $f = \left(\frac{k-1}{k}\right)^n < e^{-n/k}$ (an encoding is "good" if all positions it encodes are known to Bob). Note that while f can be made arbitrarily small by an appropriate choice of n, the number of good strings $f * k^n$ always remains above the Birthday Paradox threshold. By Property 3 of Interactive Hashing, Bob cannot force both w_0 and w_1 to be among these "good" encodings except with probability no larger than $15.6805 \cdot e^{-n/k}$. This probability can be made arbitrarily small by an appropriate choice of the security parameter n. See Figure 4 for an example.

6 Conclusion and Open Problems

We have presented a rigorous definition of Interactive Hashing by distilling and formalizing its security properties in an information theoretic context, independently of any specific application. This opens the way to recognizing Interactive Hashing as a cryptographic primitive in its own right, and not simply as a subprotocol whose security properties, as well as their proof, depend on the specifics of the surrounding application. We have also demonstrated that there exists a simple implementation of Interactive Hashing (Protocol 1) that fully meets the above-mentioned security requirements, and cited a proof of correctness that significantly improves upon previous results in the literature.

Open problems. The interested reader is encouraged to consider the following open problems:

1. Devise a more appropriate name for Interactive Hashing which better captures its properties as a cryptographic primitive rather than the mechanics of its known implementations.
2. Investigate how much interaction, if any, is really necessary in principle to implement Interactive Hashing.
3. Explore ways to implement Interactive Hashing more efficiently.To this end, the constant-round Interactive Hashing protocol of [DHRS07] briefly described in Section 3.3 is an important step in the right direction. Improve on this construction so that it meets all the security requirements.

Acknowledgments

Claude thanks Simon Pierre Desrosiers for helping him clarify his mind while revising Section 4.

References

[BCR86] Brassard, G., Crépeau, C., Robert, J.: Information theoretic reductions among disclosure problems. In: 27th Symp. of Found. of Computer Sci., pp. 168–173. IEEE, Los Alamitos (1986)

[CCM98] Cachin, C., Crépeau, C., Marcil, J.: Oblivious transfer with a memory-bounded receiver. In: Proc. 39th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 493–502 (1998)

[CCMS09] Cachin, C., Crépeau, C., Marcil, J., Savvides, G.: Information-theoretic interactive hashing and oblivious transfer to a memory-bounded receiver. Journal of Cryptology (2009) (submitted for publication) (August 2007)

[CS06] Crépeau, C., Savvides, G.: Optimal reductions between oblivious transfers using interactive hashing. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 201–221. Springer, Heidelberg (2006)

[DHRS07] Ding, Y.Z., Harnik, D., Rosen, A., Shaltiel, R.: Constant-round oblivious transfer in the bounded storage model. Journal of Cryptology 20(2), 165–202 (2007)

[Din01] Ding, Y.Z.: Oblivious transfer in the bounded storage model. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 155–170. Springer, Heidelberg (2001)

[EGL85] Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Communications of the ACM 28, 637–647 (1985)

[GMW87] Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proc. 19th Annual ACM Symposium on Theory of Computing (STOC), pp. 218–229 (1987)

[Gol04] Goldreich, O.: Foundations of cryptography, vol. I & II. Cambridge University Press, Cambridge (2001–2004)

[HHK$^+$05] Haitner, I., Horvitz, O., Katz, J., Koo, C., Morselli, R., Shaltiel, R.: Reducing complexity assumptions for statistically-hiding commitment. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 58–77. Springer, Heidelberg (2005)

[HR07] Haitner, I., Reingold, O.: A new interactive hashing theorem, Computational Complexity. In: Twenty-Second Annual IEEE Conference on CCC 2007, June 2007, pp. 319–332 (2007)

[Kil88] Kilian, J.: Founding cryptography on oblivious transfer. In: Proc. 20th Annual ACM Symposium on Theory of Computing (STOC), pp. 20–31 (1988)

[NOV06] Nguyen, M.-H., Ong, S.J., Vadhan, S.: Statistical zero-knowledge arguments for np from any one-way function, Foundations of Computer Science. In: 47th Annual IEEE Symposium on FOCS 2006, October 2006, pp. 3–14 (2006)

[NOVY98] Naor, M., Ostrovsky, R., Venkatesan, R., Yung, M.: Perfect zero-knowledge arguments for NP using any one-way permutation. Journal of Cryptology 11(2), 87–108 (1998)

[NV06] Nguyen, M.-H., Vadhan, S.: Zero knowledge with efficient provers. In: STOC 2006: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing, pp. 287–295. ACM, New York (2006)

[OVY92] Ostrovsky, R., Venkatesan, R., Yung, M.: Secure commitment against a powerful adversary. In: Finkel, A., Jantzen, M. (eds.) STACS 1992. LNCS, vol. 577, pp. 439–448. Springer, Heidelberg (1992)

[OVY93] Ostrovsky, R., Venkatesan, R., Yung, M.: Fair games against an all-powerful adversary. In: Advances in Computational Complexity Theory. AMS, 1993, Initially presented at DIMACS workshop, vol. 13 (1990); Extended abstract in the proceedings of Sequences 1991, June 1991, Positano, Italy, pp. 155–169 (1991)

[OVY94] Ostrovsky, R., Venkatesan, R., Yung, M.: Interactive hashing simplifies zero-knowledge protocol design. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 267–273. Springer, Heidelberg (1994)

[Rab81] Rabin, M.O.: How to exchange secrets by oblivious transfer, Tech. Report TR-81, Harvard (1981)

[Sav07] Savvides, G.: Interactive hashing and reductions between oblivious transfer variants, Ph.D. thesis, McGill University (2007)

[Wie70] Wiesner, S.: Conjugate coding, Reprinted in SIGACT News, vol. 15(1), original manuscript written ca. 1970 (1983)

[Yao86] Yao, A.C.-C.: How to generate and exchange secrets. In: Proc. 27th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 162–167 (1986)