Interactive Hashing and reductions between Oblivious Transfer variants

George Savvides

School of Computer Science
McGill University, Montreal
January 2007

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Doctor of Philosophy.

Abstract

Interactive Hashing has featured as an essential ingredient in protocols realizing a large variety of cryptographic tasks. We present a study of this important cryptographic tool in the information-theoretic context. We start by presenting a security definition which is independent of any particular setting or application. We then show that a standard implementation of Interactive Hashing satisfies all the conditions of our definition. Our proof of security improves upon previous ones in several ways. Despite its generality, it is considerably simpler. Moreover, it establishes a tighter upper bound on the cheating probability of a dishonest sender. Specifically, we prove that if the fraction of good strings for a dishonest sender is f, then the probability that both outputs will be good is no larger than $15.6805 \cdot f$. This upper bound is valid for any f and is tight up to a small constant since a sender acting honestly would get two good outputs with probability very close to f.

We illustrate the potential of Interactive Hashing as a cryptographic primitive by demonstrating efficient reductions of String Oblivious Transfer with string length k to Bit Oblivious Transfer and several weaker variants. Our reductions incorporate tests based on Interactive Hashing that allow the sender to verify the receiver's adherence to the protocol without compromising the latter's privacy. This allows a much more efficient use of the available entropy without any appreciable impact on security. As a result, for Bit OT and most of its variants $n = (1 + \epsilon)k$ executions suffice, improving efficiency by a factor of two or more compared to the most efficient reductions that do not use Interactive Hashing. As it is theoretically impossible to achieve an expansion factor n/k smaller than 1, our reductions are in fact asymptotically optimal. They are also more general since they place no restrictions on the types of 2-universal hash families used for Privacy Amplification. Lastly, we present a direct reduction of String OT to Rabin OT which uses similar methods to achieve an expansion factor of $2 + \epsilon$ which is again asymptotically optimal.

Résumé

Le hachage interactif figure parmi les ingrédients essentiels de plusieurs protocoles accomplissant tout un éventail de tâches cryptographiques. Cette thèse présente, dans le contexte de la théorie de l'information, une étude de cet important outil cryptographique. Tout d'abord, nous définissons la sécurité indépendamment du cadre de toute application particulière. Ensuite, nous démontrons qu'un protocole standard réalisant le hachage interactif satisfait à toutes les conditions de notre définition. Notre preuve de sécurité constitue une amélioration significative par rapport aux preuves antérieures. Malgré sa généralité, elle est considérablement plus simple. De plus, elle établit une borne supérieure plus serrée sur la probabilité de succès d'un expéditeur malhonnête. Plus précisément, nous prouvons que si l'expéditeur commence avec un ensemble de bonnes chaînes de bits représentant une fraction f du total, la probabilité que les deux valeurs de sortie soient bonnes ne dépasse pas $15.6805 \cdot f$. Cette borne supérieure vaut pour toute fraction f et est juste à une petite constante près puisqu'un expéditeur suivant le protocole obtiendrait deux bonnes valeurs de sortie avec probabilité presque f.

À titre d'exemple du potentiel, en tant que primitive cryptographique, du hachage interactif, nous démontrons des réductions efficaces de String OT avec longueur k à Bit OT et quelques unes de ses variantes plus faibles. Nos réductions font appel à des tests dérivés du hachage interactif pour permettre à l'expéditeur de vérifier l'adhésion du receveur au protocole, tout en respectant la confidentialité de la valeur d'entrée de ce dernier. Les réductions qui en résultent font un usage sécuritaire bien plus efficace de l'entropie disponible du côte du receveur. Pour Bit OT et la plupart de ses variantes, $n = (1 + \epsilon)k$ exécutions suffisent, ce qui double l'efficacité de nos réductions par rapport aux meilleures réductions qui n'utilisent pas le hachage interactif. Comme il est théoriquement impossible d'avoir un facteur d'expansion n/k plus petit que 1, nos réductions sont en fait asymptotiquement optimales. Elles sont aussi plus générales puisqu'elles permettent l'utilisation de toute famille universelle–2 de fonctions de hachage pour la phase de Privacy Amplification. Enfin, nous présentons une réduction directe de String OT à Rabin OT utilisant des méthodes semblables pour obtenir un facteur d'expansion de $2 + \epsilon$ qui est, lui aussi, optimal.

Acknowledgments

Contents

List of Protocols

List of Tables

List of Figures

1
Introduction

An Interactive Hashing protocol allows a sender Alice to input a string w which, in the course of the protocol, will be transmitted to a receiver Bob along with a second output string $w' \neq w$. In a nutshell, the protocol should guarantee that (any dishonest) Bob cannot guess which of w, w' was the original input, while at the same time, it should ensure that at least one of the two output strings must be chosen effectively at random, and beyond (any dishonest) Alice's control.

Interactive Hashing has found many applications in computational as well as information-theoretic contexts. Various implementations of Interactive Hashing appear in protocols achieving a multitude of cryptographic tasks, ranging from zero-knowledge proofs to bit commitment and oblivious transfer [OVY93, OVY94, NOVY98, OVY92, CCM98, DHRS04]. The versatility and wide applicability of Interactive Hashing suggest that a more thorough investigation of this crypto-

graphic tool is in order. This thesis sets out to present a study of Interactive Hashing in the information-theoretic context, namely under the assumption that any (dishonest) party may be computationally unbounded. The properties that are typically required of Interactive Hashing protocols in this context are distilled and formalized independently of any particular application. This application independence sets the stage for viewing Interactive Hashing as a cryptographic primitive in its own right rather than simply as a class of sub-protocols within a larger application, with security properties defined on an ad-hoc basis according to the specific needs of the given setting. It is our hope and belief that this encapsulation of Interactive Hashing as a stand-alone primitive with well-defined properties will lead to a greater appreciation of its potential as a cryptographic tool. At the same time, it will render Interactive Hashing more accessible to designers of cryptographic protocols, who will be able to incorporate it in their constructions as a self-contained building block with several implementations to choose from, each with security properties that have (ideally) been independently and rigorously scrutinized.

Regarding practicality, we demonstrate that Interactive Hashing as we defined it can be efficiently implemented in practice. Specifically, we prove that one of the Interactive Hashing protocols that appeared in the literature [OVY93] in a computational context actually satisfies all our information-theoretic security requirements as well. Our proof of security is one of the major contributions of this thesis, as it improves in several important ways upon a previous proof for a slight variant of this protocol [CCM98]: besides its application-independence,

our proof takes a different, more natural approach to establishing that the probability of successfully cheating for any dishonest receiver is small. The resulting upper bound on this probability is much tighter and the proof is considerably less complicated overall.

Another major goal of this thesis is to illustrate the power of Interactive Hashing as a cryptographic protocol. To this end, we demonstrate its applicability to reductions between Oblivious Transfer variants [Rab81, EGL85]. In short, a protocol for String Oblivious Transfer allows a sender Alice to send to a receiver Bob one of two strings x_0, x_1. The protocol should guarantee that (honest) Bob can receive the string of his choice x_c without (dishonest) Alice being able to obtain information about Bob's choice bit c. On the other hand, (honest) Alice is assured that (dishonest) Bob can receive information about exclusively one of the two strings. Bit Oblivious Transfer can be seen as a special case of String OT, with both strings having length 1. We show that Interactive Hashing can lead to efficient reductions of String Oblivious Transfer to Bit Oblivious Transfer and several of its variants. The novelty of our reductions arises from tests based on Interactive Hashing that are incorporated into well-known reductions [BCW03, Cré87] based on Privacy Amplification [BBR88]. These tests allow the sender (in String OT) to query the receiver on a small subset of the bits he received. Without compromising the honest receiver's privacy concerning his choice bit, these tests ensure that a dishonest receiver cannot deviate much from the protocol without getting caught. Consequently, as our reductions need only allow for a small potential deviation in the case of a dishonest receiver, they

make much more efficient use of the receiver's entropy about the transmitted bits. Compared to the best known reductions that do not use Interactive Hashing, our reductions are at least twice as efficient, and in most cases provably asymptotically optimal. Moreover, they are more general since they can use any 2-universal family of hash functions to perform Privacy Amplification.

Remark: The nature of our subject matter calls for a modular presentation of the material covered in this thesis. We thus defer a more detailed technical treatment of all the notions and results presented above, as well as a thorough review of prior work, to the introductory sections of Chapters 2 through 5.

1.1 Structure of this thesis

Each of the following chapters is as self-contained as possible. Chapter 2 presents our study of Interactive Hashing in the information-theoretic context. Chapter 3 introduces the notion of Oblivious Transfer in detail and defines the variants we will be encountering in the rest of the thesis. The reductions of String Oblivious Transfer to Bit Oblivious Transfer and several weaker variants are the subject of Chapter 4. Chapter 5 shows how the techniques behind the reductions of Chapter 4 can be adapted to provide a direct reduction of String Oblivious Transfer to Rabin Oblivious Transfer. The conclusion, along with a brief summary of our results, is given in Chapter 6. Finally, a brief Appendix lists some useful mathematical tools and notions.

2

Interactive Hashing

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 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 2.1.

In this Chapter we provide a study of Interactive Hashing in the information theoretic setting. We follow a modular approach, whereby we study Interactive

$$w \quad \boxed{\text{IH}} \quad w_0 \\ w_1$$

Alice   Bob
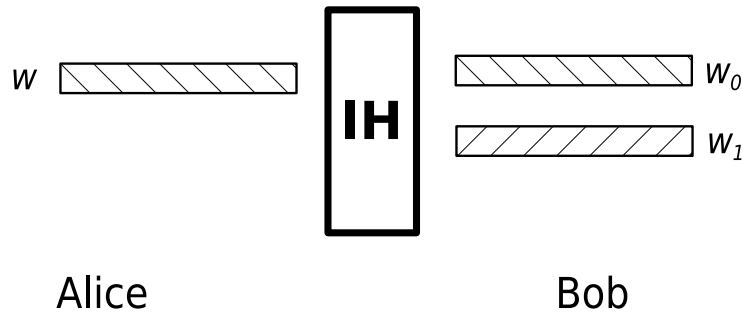
Figure 2.1: Interactive Hashing: the sender Alice sends string w to Bob, who receives two strings w_0, w_1, labeled according to lexicographic order. One of the two (in our example, w_0) is equal to the input string while the other is effectively randomly chosen. Bob cannot distinguish which of the two was the original input.

Hashing independently of the context of any specific application where it may be used as a sub-protocol. Our application-independent analysis opens the way to a better appreciation of the power of Interactive Hashing as a *cryptographic primitive* in its own right.

We start by identifying and formalizing the information theoretic security properties of Interactive Hashing in Section 2.2. Then, in Section 2.3 we turn our attention to the Interactive Hashing implementation that appeared as a sub-protocol in [OVY93] and demonstrate that despite its simplicity, it meets all security properties set forth in Section 2.2. Our 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.

Since it does not rely on the specific context of any application, our new proof is more general. Moreover, it is significantly simpler and more intuitive. Lastly, our proof establishes 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.

2.1 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.

2.1.1 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 compu-

tationally bounded.

An illustrative example of its applications in such computational contexts is the bit commitment scheme of Naor *et al.* [NOVY98]. We briefly remind the reader that a bit commitment scheme allows a player, Alice, to send a commitment to a bit b of her choice to some other player Bob. The scheme should guarantee that, on one hand, the value of b remains hidden from (dishonest) Bob until the decommitment phase, when Alice opens the commitment and reveals the value of b she had committed to. On the other hand, the scheme should also guarantee that after the commitment phase, (dishonest) Alice is only able to decommit to one value. In the bit commitment scheme of [NOVY98], Alice commits to a bit b by choosing uniformly at random a string $m \in \{0,1\}^t$, computing $w = \pi(m)$ where π is a one-way permutation[1], and sending the image w to Bob using Interactive Hashing. Alice then announces a labeling of the two outputs w_0, w_1 such that $w_b = w$ (this labeling allows her to later decommit to the right value). Note that b remains perfectly hidden even from a computationally unbounded[2] Bob since, by the properties of Interactive Hashing, Bob cannot tell which of w_0, w_1 was Alice's original input since they are both equally likely to be the image of Alice's uniformly chosen string m (recall that π is a permutation). At a later time, Alice decommits to b by announcing m, namely the pre-image under π of one of w_0, w_1. As Interactive Hashing guarantees that one of the two outputs is chosen effectively at random, cheating would imply having to invert π on an

---

[1] In short, a *one-way permutation* π has the property that it can be efficiently evaluated on any input x, yet given an image y chosen uniformly at random, it is computationally infeasible to compute the pre-image x such that $\pi(x) = y$, except with negligible probability.

[2] A *computationally unbounded* player can be thought of as having infinite computational power at his disposal.

effectively random string in $\{0,1\}^t$. Therefore, if π is one-way, Alice can only cheat a negligible fraction of the time. Indeed, the security of this commitment scheme is formally proved via a reduction showing that if (polynomially-bounded) Alice can decommit both ways a non-negligible fraction of the time, then there exist efficient algorithms that invert π, thereby contradicting its one-wayness.

2.1.2   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, ADR02, Din01, DHRS04]. Interactive Hashing was later used to optimize reductions between oblivious transfer variants [CS06] (see also Chapter 4).

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 in which it was used.

2.2 Information theoretic security definition of Interactive Hashing

We now formalize the security properties that Interactive Hashing is expected to satisfy in information theoretic contexts[3]. As these properties do not depend on any specific application, they allow us to define Interactive Hashing as an independent cryptographic primitive.

Definition 2.1. *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[4].

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

---

[3]In some specific applications, one or more of the security properties may actually be relaxed.

[4]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.

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: In the computational contexts of Section 2.1.1, 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 pre-determined good set $\mathcal{G}$. For instance, in [NOVY98] (see Section 2.1.1) 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.

2.3   A Protocol for Interactive Hashing and a new proof of its security

We will be examining the implementation of Interactive Hashing given in Protocol 2.1. This standard implementation was originally introduced in a computational context by Ostrovsky, Venkatesan, and Yung [OVY93]. In Section 2.3 we will see that this very simple protocol actually meets all our information theoretic

security requirements as well.

Protocol 2.1 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: 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 2.1 establishes the security of Protocol 2.1.

Theorem 2.1. Protocol 2.1 satisfies all three information theoretic security properties of Definition 2.1. 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 $15.6805 \cdot \mathcal{G}/T$.

Theorem 2.1 follows from Lemmas 2.1 and 2.2, which we prove in Sections 2.3.1 and 2.3.2, respectively.

Lemma 2.1. Protocol 2.1 satisfies Properties 1 and 2 of Definition 2.1.

Lemma 2.2. Protocol 2.1 ensures that a dishonest sender can succeed in causing both outputs to be in the "good set" $\mathcal{G}$ with probability at most $15.6805 \cdot {}^{G}/_{T}$.

2.3.1 Satisfying Properties 1 and 2

Lemma 2.1 is rather straightforward to prove. Protocol 2.1 essentially builds the linear system of equation (2.1) in a row-by-row manner.

$$\underbrace{\begin{pmatrix} \cdots & q_1 & \cdots \\ \cdots & q_2 & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & q_{t-1} & \cdots \end{pmatrix}}_{Q} \cdot \begin{pmatrix} \vdots \\ w \\ \vdots \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{t-1} \end{pmatrix}. \tag{2.1}$$

The properties of the linear system easily establish that Property 1 of Definition 2.1 is met, in other words, 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 w_0, w_1 be the corresponding output strings. Note that V would be identical whether the sender's input was w_0 or w_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 w_0 and w_1 as input — both with some typically very small probability α — then at the end of the protocol, given the view V, each of these two strings has equal probability $^1/_2$ of having been the original input string. We

27

observe that a dishonest receiver would have nothing to gain by selecting a matrix Q of queries in a non-random fashion or by selecting a matrix with rank less than $t-1$.

As for Property 2, let w be the sender's input and let w' 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 $w' \neq w$. To see that w' is uniformly distributed among all strings in $\{0,1\}^t \setminus w$, it suffices to observe that Q is chosen uniformly at random among all rank $t-1$ matrices and that the number of such Q's satisfying $Q(w) = Q(w') \Leftrightarrow Q(w - w') = 0$ is the same[5] for any $w' \neq w$.

2.3.2   Satisfying Property 3

The bulk of the proof is devoted to the considerably more ambitious undertaking of proving Lemma 2.2 establishing that Property 3 is also met. Note that Property 3 would be rather easy to satisfy when $G \in o(\sqrt{T})$ as in this case, the probability that a matrix Q selected uniformly at random will lead to collisions[6] within G is negligible. Consequently, in this scenario there would not even be a need for interaction since the sender could simply send the whole of Q in one round. Interaction only becomes necessary for larger sets $\mathcal{G}$ for which the probability of collision becomes significant because of the Birthday Paradox[7]. What

---

[5]To be more specific, to each such pair w, w' correspond exactly $\prod_{i=0}^{t-2} \left(2^{t-1} - 2^i\right)$ matrices Q. To see this, let $v = w - w'$. As $v \neq \vec{0}$, the equation $q \cdot v = 0$ has 2^{t-1} solutions. A matrix Q of rank $t-1$ satisfying $Q(v) = 0$ must have rows $q_1, \ldots, q_{t-1}$ that are non-zero and linearly independent of all previous rows.

[6]A collision occurs when there exist strings $w_0, w_1 \in \mathcal{G}$ such that $Q \cdot w_0 = Q \cdot w_1$. In other words, when $w_0 - w_1 = w$ where w is the unique non-zero solution to the equation $Q(w) = \vec{0}$.

[7]According to the Birthday Paradox, it is almost certain that there will be collisions if $G \in \omega(\sqrt{T})$.

we will show is that interaction in effect "beats" the Birthday Paradox, in the sense that a dishonest sender can only produce a collision in G with probability $O(G/T)$.

Notation

Table 2.1 presents the notation we will be using for the rest of the proof.

Remark: at the beginning of Protocol 2.1, G_i, P_i, etc. are *random variables* whose exact value will only be determined at the beginning of round i. The intended interpretation of statements such as $P_i = \frac{G_i(G_i-1)}{2}$ or $P_i < \frac{G_i^2}{2}$ is that the relation holds in all executions of the protocol once the corresponding values are fully determined. When we say that we condition on a given value of G_i, say, the intended interpretation is that we are setting G_i to a specific value g_i. All associated variables determined at the same round (such as P_i) are then also implicitly set to the specific values corresponding to g_i, but variables such as G_{i+1}, whose exact value will depend on future queries, still remain undetermined.

Alice's cheating strategy

At the beginning of round i, Alice has G_i good strings which are then split into G_i^0, G_i^1 by query q_i. It is tempting to assume that the optimal cheating strategy for Alice is to always choose the value of c_i that allows her to carry the larger of the two sets into the next round. This would simplify our analysis since it would allow us to establish both an upper and a lower bound on the expected size of G_i for all i. However intuitively obvious it may be, though, proving that

29

Table 2.1 Notation

$T = 2^t$	The number of all strings in $\{0,1\}^t$.
$\mathcal{G}$	The set of Alice's good strings at the beginning of Protocol 2.1. We will denote its cardinality $\lvert \mathcal{G} \rvert$ by G.
2^{-u}	The fraction of Alice's good strings at the beginning, namely G/T.
G_i	The number of Alice's (remaining) good strings at the beginning of round i, right before query i is sent. Note that there are $G_1 = G$ good strings at the beginning of Protocol 2.1 and G_t good strings at the end (after $t-1$ rounds). We say that a dishonest sender succeeds in cheating if and only if $G_t = 2$ (i.e., if both output strings are good).
P_i	The number of pairs of good strings remaining at the beginning of round i. Note that $P_i = \frac{G_i(G_i - 1)}{2}$ and that a dishonest sender succeeds in cheating if and only if $P_t = 1$.
G_i^0, G_i^1	The number of strings in G_i that are mapped to $0, 1$ respectively, by query i. Note that $G_i^0 + G_i^1 = G_i$.
G_i^m, G_i^n	Respectively, $\max\left(G_i^0, G_i^1\right)$ and $\min\left(G_i^0, G_i^1\right)$. Note that $G_i^m + G_i^n = G_i$ and that $G_i^m G_i^n = G_i^0 G_i^1$.

choosing the maximum subset is indeed the optimal strategy for Alice is not that straightforward[8]. To avoid this difficulty, we will consider an imaginary Alice to whom we grant extra powers compared to real Alice, subject to the following condition: after query q_i splits the G_i remaining strings into two sets of cardinality G_i^0 and G_i^1, imaginary Alice can choose and announce either value for c_i, and must then construct a new set of good strings to be carried forward into the next round. This new set must be of cardinality $G_{i+1} = \max(G_i^0, G_i^1)$ and its contents can be chosen arbitrarily among all strings that satisfy Equation (2.1) up to and including the row containing q_i. We remark that intuitively, it would be in imaginary Alice's best interest to choose the value of c_i and the contents of the set so as to maximize the probability that two good strings will remain at the end of the protocol. However, for the purposes of our proof, we do not need to assume anything about imaginary Alice's actual strategy; it suffices to argue that imaginary Alice is no less powerful than real Alice. This is true because whatever strategy real Alice uses, imaginary Alice can always copy it by choosing the same value for c_i in each round, and by defining the set she carries into the next round to contain all the strings that real Alice would carry, plus some arbitrarily chosen strings to reach the size imposed by our condition. Thus, if we were to run the two Alices in parallel, with the same queries, then for all i, imaginary Alice's G_i would be a superset of real Alice's G_i. It is easy to see that real Alice cheats only if imaginary Alice copying real Alice's strategy cheats.

---

[8]The structure of the subsets may have an impact on the future probability of cheating. For example, sets consisting of linear subspaces are probably undesirable to dishonest Alice, as each incoming query would break them into two subsets of equal cardinality. It is thus conceivable that in some cases, the smaller subset might be preferable to the larger one.

Therefore, real Alice is no more likely to cheat than imaginary Alice, so if we show that any strategy followed by imaginary Alice can succeed with probability no larger than p, the same bound will apply to any strategy followed by real Alice as well. From this point on, we will assume that we are dealing with imaginary Alice.

Some preliminary results

Lemma 2.3. Alice's strategy implies the following relations for all i:

$$G_i \geq 1 \tag{2.2}$$

$$G_i \geq \frac{G}{2^{i-1}} \tag{2.3}$$

$$G_{i+1} = G_i^m = \frac{G_i}{2} + \sqrt{\frac{(G_i)^2}{4} - G_i^0 G_i^1} \tag{2.4}$$

$$P_{i+1} = \frac{G_i^m(G_i - 1) - G_i^0 G_i^1}{2} \tag{2.5}$$

$$P_{i+1} \leq P_i - G_i^0 G_i^1. \tag{2.6}$$

Proof.

(2.2) For some i, let $G_i \geq 1$. Then, as query q_i separates Alice's good strings into those that evaluate to 0 and those that evaluate to 1 and Alice's strategy is to carry into the next round a subset of cardinality equal to that of the larger set, it will necessarily be the case that $G_{i+1} = G_i^m \geq 1$. Consequently, provided that $G_1 \geq 1$, it follows by induction that for all i we have $G_i \geq 1$.

(2.3) It follows from Alice's strategy that for all i, $G_{i+1} \geq G_i/2$. We then apply this bound $i - 1$ times to get from $G_1 = G$ to G_i.

(2.4) We can write $(G_i^m)^2 = G_i^m(G_i - G_i^n) = G_i^m G_i - G_i^m G_i^n$. As $G_i^m G_i^n = G_i^0 G_i^1$, this implies

$$(G_i^m)^2 - G_i G_i^m + G_i^0 G_i^1 = 0.$$

Equation (2.4) gives the larger of the two solutions to this quadratic equation (the smaller one would be $G_i^n = \frac{G_i}{2} - \sqrt{\frac{(G_i)^2}{4} - G_i^0 G_i^1}$).

(2.5) We have: $2P_{i+1} = G_i^m(G_i^m - 1) = (G_i^m)^2 - G_i^m$. Substituting $(G_i^m)^2 = G_i G_i^m - G_i^0 G_i^1$ we get

$$2P_{i+1} = \left(G_i G_i^m - G_i^0 G_i^1\right) - G_i^m = G_i^m(G_i - 1) - G_i^0 G_i^1.$$

(2.6) Note that $G_i^0 G_i^1$ can be interpreted as the number of pairs in P_i that are separated by query q_i (one element of the pair is mapped to 0 and the other to 1). These pairs will be separated no matter what value Alice announces for c_i. It is thus intuitively obvious that P_{i+1} cannot be any larger than $P_i - G_i^0 G_i^1$. For a more rigorous proof, we will prove that

33

$P_i - G_i^0 G_i^1 - P_{i+1} \geq 0$. We have

$$
\begin{aligned}
2P_i - 2G_i^0 G_i^1 - 2P_{i+1} &= G_i(G_i - 1) - 2G_i^m G_i^n - G_i^m(G_i^m - 1) \\
&= (G_i^m + G_i^n)(G_i - 1) - G_i^m(G_i^m - 1) - 2G_i^m G_i^n \\
&= G_i^m(G_i - 1 - G_i^m + 1 - 2G_i^n) + G_i^n(G_i - 1) \\
&= -G_i^m G_i^n + G_i^n(G_i - 1) \\
&= G_i^n(G_i - G_i^m - 1) \\
&= G_i^n(G_i^n - 1) \\
&\geq 0 \qquad \text{since } G_i^n \text{ is always a non-negative integer.}
\end{aligned}
$$

$\square$

Lemma 2.4. Conditioning on a given (specific) value of G_i the expected value of $G_i^0 G_i^1$ satisfies

$$
\mathsf{E}\left[G_i^0 G_i^1 \mid G_i\right] = \left(\frac{2^{t-1}}{2^t - 2^{i-1}}\right) P_i \tag{2.7}
$$

$$
\geq \frac{1}{2} P_i. \tag{2.8}
$$

Proof. Let $\mathcal{G}_i$ be any set having cardinality G_i. We can arbitrarily enumerate all its strings and write

$$
G_i^0 = \sum_{j=1}^{G_i} z_j \quad \text{and} \quad G_i^1 = \sum_{j=1}^{G_i} n_j
$$

where z_j (resp. n_j) is an indicator random variable taking on the value of 1

whenever the corresponding string in $\mathcal{G}_i$ is mapped to 0 (resp. 1) by query q_i (which has not been sent yet). Then we have

$$G_i^0 G_i^1 = \left(\sum_{j=1}^{G_i} z_j \right) \cdot \left(\sum_{j=1}^{G_i} n_j \right)$$

$$= \underbrace{z_1 n_1 + z_2 n_2 + \cdots + z_{G_i} n_{G_i}}_{G_i \text{ terms (A)}} + \underbrace{z_1 n_2 + z_1 n_3 + \cdots + z_{G_i} n_{G_i-1}}_{G_i^2 - G_i \text{ terms (B)}}$$

$$= \underbrace{\sum_{j=1}^{G_i} \sum_{k=j+1}^{G_i} \left(z_j n_k + z_k n_j \right)}_{(G_i^2 - G_i)/2 = P_i \text{ terms}}. \tag{2.9}$$

The last step follows by observing that part A vanishes since $\forall j, \ z_j n_j = 0$ and by grouping the terms in B into the sum of P_i terms of type $z_j n_k + z_k n_j$ with $k > j$.

We will now show that $\forall j, k \quad (j \neq k) \quad \mathbb{E}\left[z_j n_k + z_k n_j \right] = \frac{2^{t-1}}{2^t - 2^{i-1}}$. Note that we are still conditioning on the same G_i. Fix j, k such that $1 \leq j, k \leq G_i$ and $j \neq k$. Let w_j, w_k be the corresponding strings in $\mathcal{G}_i$. There are two cases to consider:

Case 1: $w_j, w_k \neq \vec{0}$. Let's count the queries that result in $z_j n_k = 1$. These queries must satisfy

$$\begin{pmatrix} \cdots & w_j & \cdots \\ \cdots & w_k & \cdots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ q_i \\ \vdots \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{2.10}$$

As w_j, w_k are different and non-zero, they are linearly independent. Consequently, there are exactly 2^{t-2} solutions for q_i. Note that all such solutions are linearly independent of all previous queries. This is because both w_j, w_k satisfy the linear system of Equation (2.1) up to the row containing q_{i-1}, which makes it impossible

for a linearly dependent query q_i to map them to different values.

At this round there are $2^t - 2^{i-1}$ valid queries q_i for Bob to choose from (since the 2^{i-1} queries that are linearly dependent on $q_1, \ldots, q_{i-1}$ are excluded). Therefore $\mathsf{E}\left[z_j n_k\right] = \frac{2^{t-2}}{2^t - 2^{i-1}}$ and by symmetry the same holds for $\mathsf{E}\left[z_k n_j\right]$. Consequently by linearity of expectation we have:

$$\mathsf{E}\left[z_j n_k + z_k n_j\right] = \frac{2^{t-1}}{2^t - 2^{i-1}}.$$

Case 2: one of w_j, w_k is $\vec{0}$. Without loss of generality, suppose it is w_j. Then no query can result in $z_k n_j = 1$. Let's count the queries that produce $z_j n_k = 1$. These must satisfy the system

$$\begin{pmatrix} \cdots & w_j & \cdots \\ \cdots & w_k & \cdots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ q_i \\ \vdots \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{2.11}$$

This system has (effectively) one equation with t unknowns and hence 2^{t-1} solutions for q_i, all of which are linearly independent of all previous queries[9].

As in Case 1, Bob has $2^t - 2^{i-1}$ possible values for q_i to choose from, from which it follows that $\mathsf{E}\left[z_j n_k\right] = \frac{2^{t-1}}{2^t - 2^{i-1}}$. Since $\mathsf{E}\left[z_k n_j\right] = 0$, we have that again,

$$\mathsf{E}\left[z_j n_k + z_k n_j\right] = \frac{2^{t-1}}{2^t - 2^{i-1}}.$$

---

[9]This is because as $w_j = \vec{0}$, we must have had $c_1 = c_2 = \cdots = c_{i-1} = 0$ and so a linearly dependent query would necessarily map w_k to 0. For the special case of the first round, observe that $q_1 = \vec{0}$ (the only disallowed query) would map w_k to 0.

Combining the two cases, we see that all P_i terms in (2.9) satisfy

$$\mathbb{E}\left[z_j n_k + z_k n_j\right] = \frac{2^{t-1}}{2^t - 2^{i-1}} \geq \frac{1}{2}$$

and so, by linearity of expectation

$$\mathbb{E}\left[G_i^0 G_i^1 \mid G_i\right] = \left(\frac{2^{t-1}}{2^t - 2^{i-1}}\right) P_i \geq \left(\frac{1}{2}\right) P_i. \tag{2.12}$$

$\square$

Lemma 2.5. Using Lemma 2.4 we can establish the following bounds:

$$\mathbb{E}\left[G_{i+1} \mid G_i\right] \leq \frac{G_i + \sqrt{G_i}}{2} \tag{2.13}$$

$$\mathbb{E}\left[P_{i+1} \mid P_i\right] \leq \frac{1}{4}\left(1 + \frac{2^{\frac{i+1}{2}}}{\sqrt{G}}\right) P_i \tag{2.14}$$

$$\mathbb{E}\left[P_{i+1} \mid P_i\right] \leq \frac{1}{2}\left(\frac{2^t - 2^i}{2^t - 2^{i-1}}\right) P_i. \tag{2.15}$$

Proof. We first remark that conditioning on a specific value of P_i (i.e. setting $P_i = p$) is equivalent to conditioning on the corresponding value of G_i and vice versa as the value of one uniquely[10] determines the other.

---

[10] G_i uniquely determines $P_i = \frac{G_i(G_i - 1)}{2}$. On the other hand, given P_i there are two solutions to the corresponding quadratic equation for G_i of which only the larger is valid since the smaller one is either negative or zero, in violation of (2.2).

(2.13) From (2.4) we have

$$E\left[G_i^m \mid G_i\right] = \frac{G_i}{2} + E\left[\sqrt{\frac{(G_i)^2}{4} - G_i^0 G_i^1} \mid G_i\right]$$

$$\leq \frac{G_i}{2} + \sqrt{E\left[\frac{(G_i)^2}{4} - G_i^0 G_i^1 \mid G_i\right]} \qquad \text{by Jensen's Inequality}$$

$$= \frac{G_i}{2} + \sqrt{\frac{(G_i)^2}{4} - E\left[G_i^0 G_i^1 \mid G_i\right]} \qquad \text{by linearity of expectation}$$

$$\leq \frac{G_i}{2} + \sqrt{\frac{(G_i)^2}{4} - \frac{G_i(G_i - 1)}{2}\frac{1}{2}} \qquad \text{using (2.8)}$$

$$= \frac{G_i + \sqrt{G_i}}{2}.$$

(2.14) Using (2.5), we get the following:

$$E\left[P_{i+1} \mid G_i\right] = \frac{1}{2} E\left[G_i^m(G_i - 1) - G_i^0 G_i^1 \mid G_i\right] \tag{2.16}$$

$$= \frac{1}{2}(G_i - 1) E\left[G_i^m \mid G_i\right] - \frac{1}{2} E\left[G_i^0 G_i^1 \mid G_i\right] \tag{2.17}$$

$$\leq \frac{1}{2}(G_i - 1)\frac{G_i + \sqrt{G_i}}{2} - \frac{1}{2}\frac{G_i(G_i - 1)}{4} \tag{2.18}$$

$$= \frac{G_i(G_i - 1)}{8}\left(1 + \frac{2}{\sqrt{G_i}}\right) \tag{2.19}$$

$$= \frac{1}{4}\left(1 + \frac{2}{\sqrt{G_i}}\right) P_i \tag{2.20}$$

$$\leq \frac{1}{4}\left(1 + \frac{2^{\frac{i+1}{2}}}{\sqrt{G}}\right) P_i. \tag{2.21}$$

Note that (2.17) follows by linearity of expectation, (2.18) follows from bounds (2.2), (2.8), (2.13) while (2.21) is obtained by applying Inequality (2.3).

(2.15) From (2.6) we have

$$\mathsf{E}\left[P_{i+1} \mid P_i\right] \leq P_i - \mathsf{E}\left[G_i^0 G_i^1 \mid P_i\right]$$
$$= P_i - P_i \left(\frac{2^{t-1}}{2^t - 2^{i-1}}\right) \qquad \text{using (2.7)}$$
$$= \frac{1}{2}\left(\frac{2^t - 2^i}{2^t - 2^{i-1}}\right) P_i.$$

$\square$

Lemma 2.6. Let $1 \leq b \leq t - 1$ be a positive integer. Let $R \stackrel{\text{def}}{=} 2^{\frac{u-b}{2}}$. Then the expected number of pairs at the end of Protocol 2.1 satisfies

$$\mathsf{E}\left[P_t\right] \leq \frac{G}{T} \cdot \underbrace{\frac{2}{2 - 2^{-b}} \cdot R^{-2} \cdot \prod_{j=0}^{\infty}\left(1 + \frac{R}{2^{j/2}}\right)}_{Y}. \qquad (2.22)$$

Proof. Taking expectations on both sides[11] of (2.14) we get

$$\mathsf{E}\left[P_{i+1}\right] \leq \frac{1}{4}\left(1 + \frac{2^{\frac{i+1}{2}}}{\sqrt{G}}\right) \mathsf{E}\left[P_i\right]$$
$$= \frac{1}{4}\left(1 + 2^{(i+1-t+u)/2}\right) \mathsf{E}\left[P_i\right] \quad \text{replacing } G = 2^{t-u}$$
$$= \frac{1}{4}\left(1 + R_i\right) \mathsf{E}\left[P_i\right] \qquad \text{writing } R_i \stackrel{\text{def}}{=} 2^{(i+1-t+u)/2}. \qquad (2.23)$$

---

[11]This can alternatively be seen as follows:

$$\sum_{p=0}^{\infty} \mathsf{E}\left[P_{i+1} \mid P_i = p\right] \cdot \Pr\left[P_i = p\right] \leq \sum_{p=0}^{\infty} \frac{1}{4}\left(1 + \frac{2^{\frac{i+1}{2}}}{\sqrt{G}}\right) \cdot p \cdot \Pr\left[P_i = p\right] \Rightarrow$$
$$\mathsf{E}\left[P_{i+1}\right] \leq \frac{1}{4}\left(1 + \frac{2^{\frac{i+1}{2}}}{\sqrt{G}}\right) \mathsf{E}\left[P_i\right].$$

Similarly, taking expectations on both sides of (2.15), we get

$$E[P_{i+1}] \leq \frac{1}{2}\left(\frac{2^t - 2^i}{2^t - 2^{i-1}}\right)E[P_i].$$
(2.24)

Note that (2.23) and (2.24) are both valid upper bounds on $E[P_{i+1}]$ for any i. Note also that these two inequalities remain valid if $E[P_i]$ on the right hand side is replaced by any upper bound for $E[P_i]$.

Let $a = (t - 1) - b$. Suppose that we start by sequentially applying (2.23) a times. Then,

$$
\begin{aligned}
E[P_{a+1}] &\leq E[P_1]\cdot\frac{1}{4^a}\prod_{i=1}^{a}(1 + R_i) \\[2mm]
&= P_1\frac{1}{4^a}\prod_{i=1}^{a}(1 + R_i) && \text{since } E[P_1] = P_1 \\[2mm]
&= P_1\cdot\frac{1}{4^a}\prod_{i=0}^{a-1}\left(1 + \frac{R_a}{2^{i/2}}\right) && \text{using } {}^{R_i}/{}_{R_{i-1}} = \sqrt{2} \\[2mm]
&= P_1\cdot\frac{1}{4^a}\prod_{i=0}^{a-1}\left(1 + \frac{R}{2^{i/2}}\right) && \text{since } R_a = R = 2^{\frac{u-b}{2}} \\[2mm]
&\leq P_1\cdot\frac{1}{4^a}\prod_{i=0}^{\infty}\left(1 + \frac{R}{2^{i/2}}\right) && \text{since all terms are at least 1.} \quad (2.25)
\end{aligned}
$$

Now suppose we sequentially apply (2.24) for the last b rounds to get an upper

40

bound for $E[P_t]$ in terms of $E[P_{t-b}]$. Then

$$E[P_t] \leq E[P_{t-b}] \cdot \frac{1}{2^b} \prod_{i=t-b}^{t-1} \left(\frac{2^t - 2^i}{2^t - 2^{i-1}} \right)$$

$$= E[P_{t-b}] \cdot \frac{1}{2^b} \left(\frac{2^t - 2^{t-b}}{2^t - 2^{t-b-1}} \right) \left(\frac{2^t - 2^{t-b+1}}{2^t - 2^{t-b}} \right) \cdots$$

$$\cdots \left(\frac{2^t - 2^{t-2}}{2^t - 2^{t-3}} \right) \left(\frac{2^t - 2^{t-1}}{2^t - 2^{t-2}} \right)$$

$$= E[P_{t-b}] \cdot \frac{1}{2^b} \left(\frac{2^t - 2^{t-1}}{2^t - 2^{t-b-1}} \right)$$

$$= E[P_{t-b}] \cdot \frac{1}{2^b} \left(\frac{1}{2 - 2^{-b}} \right)$$

$$= E[P_{a+1}] \cdot \frac{1}{2^b} \left(\frac{1}{2 - 2^{-b}} \right) \tag{2.26}$$

where Equation (2.26) follows from the fact that $t - b = a + 1$. Combining (2.25) and (2.26), we have

$$E[P_t] \leq P_1 \cdot \frac{1}{4^a} \frac{1}{2^b} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right)$$

$$\leq \frac{G^2}{2} 2^{-2a-b} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right) \tag{2.27}$$

$$= 2^{2t-2u-1-2a-b} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right) \tag{2.28}$$

$$= 2^{b+1-2u} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right) \tag{2.29}$$

$$= 2^{-u} \cdot 2 \cdot 2^{b-u} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right)$$

$$= \frac{G}{T} \cdot 2 \cdot R^{-2} \left(\frac{1}{2 - 2^{-b}} \right) \prod_{i=0}^{\infty} \left(1 + \frac{R}{2^{i/2}} \right). \tag{2.30}$$

Note that for Equation (2.27) we used the fact that $P_1 = \frac{G(G-1)}{2}$. For Equation (2.28) recall that $G = 2^{t-u}$. Equation (2.29) uses $2t - 2a = 2b + 2$ while Equation (2.30) follows from $G/T = 2^{-u}$ and $R = 2^{\frac{u-b}{2}}$. □

We are now ready to prove Lemma 2.2, which, along with Lemma 2.1 establishes Theorem 2.1.

Proof of Lemma 2.2. Since $P_t = 1$ if the two output strings are both good and $P_t = 0$ otherwise, it follows that the probability that Alice cheats satisfies $\Pr[P_t = 1] = \mathsf{E}[P_t]$. Similarly, the upper bound on $\mathsf{E}[P_t]$ established by (2.22) is also an upper bound on the probability of successful cheating. Note that any integer value of b in (2.22) establishes a valid upper bound. We can fix[12] $b = \lceil u + 0.03 \rceil$ in which case part Y becomes a function of u only. The probability of cheating is thus upper bounded by $\min(G/T \cdot Y(u), 1) = G/T \cdot \min(Y(u), T/G)$. Recalling that $G/T = 2^{-u}$, we set

$$Z(u) = \min\left(Y(u), 2^u\right).$$

A graph of $Z(u)$ (see Figure 2.2) shows that it never exceeds 15.6805. It therefore holds that for all ratios G/T the probability that Alice cheats is upper bounded by $15.6805 \cdot G/T$. □

Remark: The maximum of $Z(u)$ cannot occur beyond the first few peaks depicted in Figure 2.2. To see this, recall that $R = 2^{\frac{u-b}{2}}$ where $b = \lceil u + 0.03 \rceil$,

---

[12]This value of b was chosen with the help of at a 3-D graph of part Y of Equation (2.22). We remind the reader that however b is chosen, it establishes a valid upper bound.
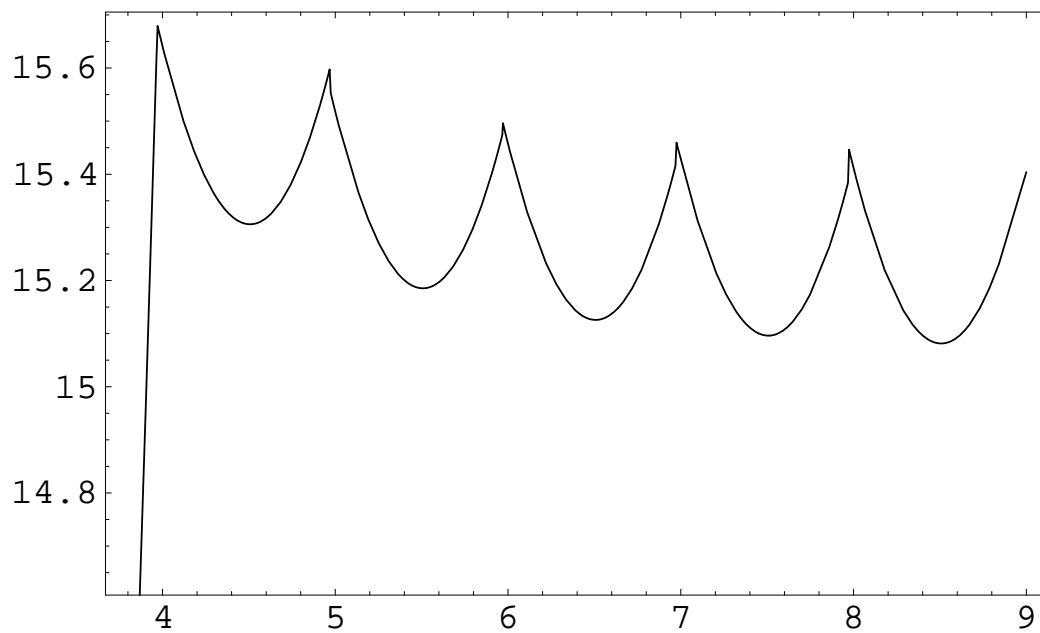
Figure 2.2: Graph of $Z(u)$ vs u, showing that the maximum value of $Z(u) = \min(Y(u), 2^u)$ does not exceed 15.6805. This maximum is attained at the intersection of $Y(u)$ with 2^u, occurring near $u_1 = 3.9709$. Note that $Z(u) = 2^u$ for $u \leq u_1$ while $Z(u) = Y(u)$ for $u > u_1$.

while $Y(u)$ from Equation (2.22) can be expressed as the product of two factors, A and B, as follows:

$$Y(u) = \underbrace{\frac{2}{2 - 2^{-b}}}_{A} \cdot R^{-2} \cdot \underbrace{\prod_{j=0}^{\infty} \left(1 + \frac{R}{2^{j/2}}\right)}_{B}.$$ (2.31)

Note that factor B of Equation (2.31) depends entirely on R, which, within any interval of length 1 of u, takes on all the values in $(R_0, R_1]$ where $R_0 = 2^{-1.03/2}$ and $R_1 = 2^{-0.03/2}$. This explains the oscillations, whose peaks are decreasing due to the fact that $A(u)$ is decreasing, converging to 1 as $u \to \infty$. For any u, u' such that $u' = u + 1$, we have $R' = R$ while $b' = b + 1$. As $A(u') < A(u)$ and $B(u') = B(u)$, we have $Y(u') < Y(u)$. This shows that the maximum cannot occur after the first two[13] peaks of the graph.

2.3.3 Contributions of our new proof

Cachin, Crépeau, and Marcil [CCM98] proved a similar property to Property 3 for a slight variant of Protocol 2.1 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, ours focuses instead on

---

[13]Our reasoning does not allow us to claim that the global maximum cannot occur after the *first* peak of $Z(u)$, as is in fact the case. This is because the first peak does not necessarily correspond to a peak of $Y(u)$, but might be determined by the intersection of $Y(u)$ with 2^u. It is thus conceivable that this first intersection is lower than the peak of $Y(u)$ immediately following it.

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.

Our approach not only leads to a simpler and more robust proof of security, but more importantly, it also allows us to establish a more general and much tighter upper bound on a dishonest sender's probability of cheating. Specifically, it allows us 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 our 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 $G \geq 2$, our upper bound is larger than this lower bound by a factor of at most $15.6805 \cdot \left(\frac{G}{T}\right) \left(\frac{T-1}{G-1}\right) < 15.6805 \left(\frac{G}{G-1}\right) \leq 2 \cdot 15.6805$. This establishes that

our upper bound is tight up to a small constant in all cases where the possibility of cheating exists (cheating is impossible when $G < 2$).

2.3.4 An alternative implementation

Ding *et al.* [DHRS04] 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 2.1), 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 permu-*

tations, which can be efficiently constructed and compactly described.

Unfortunately, the protocol of [DHRS04] is less general than Protocol 2.1 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, $\Omega\left(t \cdot G/\tau\right)$ and only applies when $G \geq 4t$. Moreover, the protocol does not fully satisfy Property 2, but only a slight relaxation[14] 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.

2.4   A sample application

In order to illustrate the power of Interactive Hashing in information theoretic contexts, we will consider its application to the following problem: suppose that a sender Alice and a receiver Bob wish to implement 1-out-of-k Bit Oblivious Transfer (more on Oblivious Transfer in later chapters), 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

---

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

(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. Over the past few years, Crépeau and Kilian [CK] have 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. Protocol 2.2 shows how Interactive Hashing makes such a reduction almost trivial.

Remark: For simplicity, Protocol 2.2 reduces $\binom{2}{1}$–Bit OT to $(k-1)$–faulty $\binom{k}{1}$–Bit 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]. 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 2.2 allows Bob to obtain the bit of his choice since he knows $R_d = \bigoplus_{i=1}^{n} r_{i c_i}$ and can thus decrypt $e_{\check{c}}$. In case Alice is dishonest, Bob's choice $\check{c}$ 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 $f = d \oplus \check{c}$ carries no information about $\check{c}$. 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

Protocol 2.2 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 $1 \le i \le n$ do:

   (a) Alice selects at random bits $r_{i1}, r_{i2}, \ldots, r_{ik}$ while Bob selects at random $c_i \in_R \{1, \ldots, k\}$.

   (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}.

3. Bob encodes his choices during the n rounds of $(k-1)$–faulty $\binom{k}{1}$–Bit OT 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

$$R_0 = \bigoplus_{i=1}^{n} r_{ip_i} \qquad\qquad R_1 = \bigoplus_{i=1}^{n} r_{iq_i}.$$

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 $\mathring{b}_{\mathring{c}} = e_{\mathring{c}} \oplus R_{f \oplus \mathring{c}} = e_{\mathring{c}} \oplus R_d$.

bits every time. Then, by the end of Step 2, it is always the case that among all encodings of positions, only $\left(\frac{k-1}{k}\right)^n < e^{-n/k}$ are "good", in the sense that they represent positions that are all known to him (along with their exclusive OR). 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.

2.5   Conclusion and open problems

We have provided 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 sub-protocol 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 that fully meets the above-mentioned security requirements, and gave a proof of correctness that significantly improves upon previous results in the literature. We have also provided a simple example that offers a glimpse into the power of Interactive Hashing as a cryptographic primitive, as a preview to the more elaborate applications that we will be encountering in Chapters 4 and 5.

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, especially regarding the amount of interaction. To this end, the constant-round Interactive Hashing protocol of [DHRS04] we briefly described in Section 2.3.4 is an important step in the right direction. We invite the interested reader to improve on this construction so that it meets all the security requirements.

3

Oblivious Transfer

Oblivious Transfer (OT) is a cryptographic primitive of paramount importance, especially in the context of multi-party computation. One of its early variants had been studied by Wiesner [Wie70] under the name of "multiplexing" but his work was only published post-facto. The notion of Oblivious Transfer was introduced to cryptography by Rabin [Rab81]. Rabin OT is a primitive that allows a sender Alice to send a bit b to a receiver Bob who receives either b or Δ (the erasure symbol), each with probability $1/2$. The primitive guarantees that Alice does not learn which of the two events occurred.

Another, more frequently encountered variant of Oblivious Transfer is *one out of two Bit Oblivious Transfer* [EGL85], denoted $\binom{2}{1}$–Bit OT or simply Bit OT. Here, the sender Alice sends two bits b_0, b_1 to Bob, who can choose to learn the bit of his choice c, namely b_c. This primitive guarantees that on one hand,

Bob learns nothing about the other bit while on the other hand Alice doesn't find out what c was. Despite the differences in appearance between Bit OT and Rabin OT, the two variants are in fact equivalent cryptographic primitives, as was demonstrated by Crépeau [Cré87].

The apparent simplicity of Oblivious Transfer belies its surprising power as a cryptographic primitive. Its applicability to multi-party computation was first studied by Even, Goldreich and Lempel [EGL85]. Ever since, Oblivious Transfer has featured as a main ingredient in an array of protocols implementing a large variety of cryptographic tasks, such as bit commitment, zero-knowledge proofs, and general secure multi-party computation [Yao86, GMW87, Gol04]. Kilian [Kil88] demonstrated that this primitive is in and of itself sufficient to securely implement any two-party computation.

String OT is a generalization of Bit OT that allows Alice to send one of two k–bit strings to Bob. In the next two Chapters, we will see how Interactive Hashing, which was presented in Chapter 2, enables String OT to be *efficiently* reduced to Bit OT and other related but weaker primitives (Chapter 4), as well as to the original Rabin OT (Chapter 5). The rest of the present chapter is devoted to introducing the various types of Oblivious Transfers that we will later encounter.

3.1   String OT and Bit OT

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 bit strings, $x_0, x_1 \in \{0,1\}^k$ to a

receiver Bob who receives x_c for a choice bit $c \in \{0, 1\}$. It is assumed that the joint probability distribution $P_{x_0 x_1 c}$ from which the inputs are generated is known to both parties. The primitive offers the following guarantees:

1. *(Correctness)* When both parties are honest, Bob obtains x_c while Alice obtains nothing.

2. *(Security for Bob)* Any (dishonest) Alice cannot learn any extra information about Bob's choice c beyond what can be inferred from her inputs x_0, x_1 and the distribution $P_{x_0 x_1 c}$.

3. *(Security for Alice)* Any (dishonest) Bob can learn information concerning exclusively one of x_0, x_1. This excludes any joint information about the two strings except what can be inferred from Bob's input, (legitimate) output, and $P_{x_0 x_1 c}$.

Bit OT can then simply be viewed as a special case of $\binom{2}{1}$–String OT^k with $k = 1$.

3.2   Weaker variants of Bit OT

XOR OT[1], Generalized OT and Universal OT are weaker variants of $\binom{2}{1}$–Bit OT obtained by relaxing the security guarantees against a dishonest receiver (Bob), as

---

[1]As a brief historical aside, we mention that XOR OT was originally studied in the context of reversing the direction of Oblivious Transfer. Crépeau and Sántha [CS91] showed that it is very easy to obtain XOR OT in one direction if $\binom{2}{1}$–Bit OT in the reverse direction is available. Using their approach, obtaining $\binom{2}{1}$–Bit OT itself required a more elaborate construction involving several executions of $\binom{2}{1}$–Bit OT in the reverse direction. These results were obviated by a more recent approach [WW06] that fully reverses $\binom{2}{1}$–Bit OT using just one execution.

described below. Note that in all cases, b_0, b_1 denote Alice's input bits. Whatever extra choices may be available to Bob, he always has the option of acting honestly to obtain b_c for a choice $c \in \{0, 1\}$. As in "regular" $\binom{2}{1}$–Bit OT, Alice never obtains information about Bob's choice c, or learns whether Bob actually made use of his expanded choices.

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

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 Ω is any arbitrary discrete memoryless channel whose input is a pair of bits and whose output satisfies the following information theoretic constraint: let $B_0, B_1 \in \{0, 1\}$ be uniformly distributed random variables and let $\alpha \leq 1$ be a constant. Then,

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

Note that we do not consider channels with $\alpha > 1$ as this would disallow Bob to act honestly.

3.3   Rabin OT

In this incarnation of Oblivious Transfer, which, as already noted, was the first one to appear in the cryptographic literature [Rab81], the sender Alice sends a bit b to the receiver Bob over an *erasure channel* with erasure probability $^1/_2$ and is

then oblivious to what transpired during the transmission. In Chapter 5 where we deal with this variant of OT, we will only be concerned with the case where the bits sent by Alice are chosen independently and uniformly at random. In other words, as the bits are uncorrelated with any information that Bob might have in his possession, whenever he receives the erasure symbol Δ, he cannot guess thee value of b with probability greater than $1/2$.

3.4   Randomized OT

$\binom{2}{1}$–ROT^k is a randomized variant of $\binom{2}{1}$–String OT^k where Alice makes available to Bob two strings $r_0, r_1 \in \{0,1\}^k$ chosen uniformly at random and independently. Bob learns r_c for a randomly chosen $c \in_R \{0,1\}$. The fact that the inputs are random and uncorrelated greatly simplifies the security requirements of $\binom{2}{1}$–ROT^k. Specifically, this primitive offers the following guarantees:

1. *(Correctness)* When both parties are honest, Bob obtains r_c while Alice obtains nothing.

2. *(Security for Bob)* Bob's choice bit c is uniformly distributed in (dishonest) Alice's view.

3. *(Security for Alice)* Any (dishonest) Bob can learn information concerning exclusively one of r_0, r_1. Specifically, at the end of every execution there must exist some $c' \in \{0,1\}$ such that, given (dishonest) Bob's view as well as $r_{c'}$ (provided by an oracle), $r_{\bar{c}'}$ is uniformly distributed in $\{0,1\}^k$.

Despite its simplicity, $\binom{2}{1}$–ROT^k is in fact equivalent to $\binom{2}{1}$–String OT^k. Intuitively, it is easy to see that $\binom{2}{1}$–ROT^k reduces to $\binom{2}{1}$–String OT^k. Protocol 3.1 shows that there exists a straightforward reduction in the reverse direction as well. For a more formal proof of the equivalence of these two variants, see Section 3.5.3.

Protocol 3.1 Reduction of String OT to Randomized OT

Let the inputs to $\binom{2}{1}$–String OT^k be $x_0, x_1 \in \{0, 1\}^k$ for the sender and $c \in \{0, 1\}$ for the receiver.

1. The sender uses $\binom{2}{1}$–ROT^k to send $\mathring{r}_0, \mathring{r}_1 \in_R \{0, 1\}^k$ to the receiver, who receives $\mathring{r}_{\mathring{c}}$ for some randomly chosen $\mathring{c} \in \{0, 1\}$.

2. The receiver sends $d = c \oplus \mathring{c}$ to the sender.

3. The sender sets $e_0 = x_0 \oplus \mathring{r}_d$ and $e_1 = x_1 \oplus \mathring{r}_{\bar{d}}$ and sends e_0, e_1 to the receiver.

4. The receiver decodes $x_c = e_c \oplus \mathring{r}_{\mathring{c}}$.

Remark: Step 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 use.

The simplicity of $\binom{2}{1}$–ROT^k compared to $\binom{2}{1}$–String OT^k makes it considerably easier to work with. For this reason, in Chapters 4 and 5 where our goal is to provide efficient reductions of $\binom{2}{1}$–String OT^k to other Oblivious Transfer variants, we actually resort to reductions of $\binom{2}{1}$–ROT^k, without any loss of generality.

3.5   Information-theoretic definitions of OT

Although the formulation of the properties of $\binom{2}{1}$–String OT^k is quite intuitive and rather straightforward even for the general case where the inputs of the two parties may be correlated, a corresponding formal definition in the language of information theory is rather elusive. Indeed, over the past decades, several attempts have been made to capture the security properties of $\binom{2}{1}$–String OT^k in such an information theoretic definition. Most of the resulting definitions are either too restrictive in scope and thus applicable to only a few specialized scenarios, or suffer from subtle (and sometimes not so subtle) flaws. An overview of some of these definitions and their shortcomings appears in [CSSW06], along with a new information theoretic definition of $\binom{2}{1}$–String OT^k which is shown to be equivalent to a widely accepted security definition of general two-party computation in the real/ideal model paradigm and will thus hopefully stand the test of time.

We present this new definition of $\binom{2}{1}$–String OT^k in Section 3.5.1 and its counterpart for $\binom{2}{1}$–ROT^k in Section 3.5.2. Finally, in Section 3.5.3 we provide a formal proof establishing that $\binom{2}{1}$–String OT^k and $\binom{2}{1}$–ROT^k are indeed equivalent using these definitions.

3.5.1   Definition of $\binom{2}{1}$–String OT^k

In what follows, $X = X_0 X_1$ is a random variable denoting the sender's input, C is a random variable denoting the receiver's choice bit and Z is a random variable denoting the environment. U, V are random variables denoting the outputs of the

sender and receiver, respectively.

Theorem 3.1 ([CSSW06]). A protocol between Player 1 and Player 2 securely computes $\binom{2}{1}$–String OT^k perfectly if and only if for every pair of algorithms $A = (A_1, A_2)$ such that at least one of A_1, A_2 follows the protocol, and for all inputs (X, C) and auxiliary input Z, A produces outputs[2] (U, V) such that the following conditions are satisfied:

1. *(Correctness)* If both players are honest, then $(U, V) = (\bot, X_C)$.

2. *(Security for Player 1)* If Player 1 (the sender) is honest, then we have $U = \bot$ and there exists a random variable C', such that

$$\mathrm{I}(X; C' \mid ZC) = 0 \qquad \text{and} \qquad \mathrm{I}(X; V \mid ZCC'X_{C'}) = 0.$$

3. *(Security for Player 2)* If Player 2 (the receiver) is honest, then we have

$$\mathrm{I}(C; U \mid ZX) = 0.$$

3.5.2 Definition of $\binom{2}{1}$–$\mathbf{ROT}^k$

We provide an information theoretic definition of $\binom{2}{1}$–ROT^k along the lines of Theorem 3.1. Let $R_0, R_1 \in \{0, 1\}^k$ be two uniformly distributed, independently chosen random variables corresponding to Alice's input and let $R = R_0 R_1$. Let $C \in \{0, 1\}$ be a binary, uniformly distributed random variable corresponding

---

[2]We remark that the output of a dishonest party can, without loss of generality, be assumed to contain the party's marginal view of the protocol's execution.

to Bob's choice bit. Theorem 3.2 captures the information theoretic security requirements for $\binom{2}{1}$–ROT^k.

Theorem 3.2. A protocol between Player 1 and Player 2 securely computes $\binom{2}{1}$–ROT^k perfectly if and only if, for every pair of algorithms $A = (A_1, A_2)$ such that at least one of A_1, A_2 follows the protocol, and for randomly and independently chosen inputs (R, C) and auxiliary input Z, A produces outputs (U, V) such that the following conditions are satisfied:

1. *(Correctness)* If both players are honest, then $(U, V) = (\bot, R_C)$.

2. *(Security for Player 1)* If Player 1 (the sender) is honest, then we have $U = \bot$ and there exists a random variable C', such that

$$\mathsf{I}(R; C' \mid ZC) = 0 \qquad \text{and} \qquad \mathsf{H}(R_{\bar{C}'} \mid ZCC'R_{C'}V) = k.$$

3. *(Security for Player 2)* If Player 2 (the receiver) is honest, then we have

$$\mathsf{H}(C \mid ZRU) = 1 .$$

Intuitively, Theorem 3.2 guarantees that from the point of view of any dishonest receiver, one of R_0, R_1 is uniformly distributed, even if the other string is provided to the receiver by an oracle (this ensures that no joint information is available). In other words, there exists some C', which must not depend on R, such that $\mathsf{H}(R_{\bar{C}'}) = k$ after conditioning on $R_{C'}$ and all information available to the receiver. Likewise, from the point of view of a dishonest sender, C is

uniformly distributed, namely $H(C) = 1$, given all available information.

3.5.3 Equivalence of $\binom{2}{1}$–String OT^k and $\binom{2}{1}$–ROT^k

In this section we show that $\binom{2}{1}$–ROT^k can be implemented using $\binom{2}{1}$–String OT^k and vice versa.

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

Lemma 3.1. If a $\binom{2}{1}$–String OT^k Protocol satisfying the security conditions of Theorem 3.1 is used with uniformly and independently chosen inputs ($R = R_0 R_1$ and C for the sender and receiver, respectively), then the security conditions of Theorem 3.2 will also be met.

Proof. It is easy to see that if both players are honest and the protocol for $\binom{2}{1}$–String OT^k satisfies Condition 1 of Theorem 3.1, then it also satisfies Condition 1 of Theorem 3.2. As for Condition 2 of Theorem 3.2, we first observe that there exists C' such that $I(R; C' \mid ZC) = 0$ since the corresponding condition in Theorem 3.1 guarantees it. Moreover, since it holds that $I(R; V \mid ZCC'R_{C'}) = 0$, we have

$$H(R \mid ZCC'R_{C'}V) = H(R \mid ZCC'R_{C'}) \tag{3.1}$$

which implies that

$$H(R_{\bar{C}'} \mid ZCC'R_{C'}V) + \overbrace{H(R_{C'} \mid ZCC'R_{C'}R_{\bar{C}'}V)}^{=0} = \tag{3.2}$$
$$H(RR_{C'} \mid ZCC') - H(R_{C'} \mid ZCC').$$

62

From (3.2) it follows that

$$\mathsf{H}\left(R_{\bar{C}'} \mid ZCC'R_{C'}V\right) = \mathsf{H}\left(R \mid ZCC'\right) - \mathsf{H}\left(R_{C'} \mid ZCC'\right)$$

$$= \mathsf{H}\left(R \mid ZC\right) - \mathsf{H}\left(R_{C'} \mid ZC\right) \qquad (3.3)$$

$$= 2k - k \qquad (3.4)$$

$$= k.$$

Note that 3.3 follows from $\mathsf{I}(R; C' \mid ZC) = 0$ while 3.4 is follows from the fact that $R_0, R_1 \in \{0, 1\}^k$ are chosen uniformly at random and independently of ZC.

Finally, for Condition 3 (protecting the honest receiver from a dishonest sender), observe that the corresponding condition for $\binom{2}{1}$–String OT^k guarantees that $\mathsf{I}(C; U \mid ZR) = 0 \implies \mathsf{H}(C \mid ZRU) = \mathsf{H}(C \mid ZR) = 1$ since C is chosen uniformly at random and independently of ZR. $\qquad\square$

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

Lemma 3.2 proves the security of the reduction of $\binom{2}{1}$–String OT^k to $\binom{2}{1}$–ROT^k presented in Protocol 3.1.

Note on notation: In order to make the distinction between the variables for $\binom{2}{1}$–String OT^k and $\binom{2}{1}$–ROT^k, we will place a small circle above all the latter ones, whether they have a similarly-named counterpart in $\binom{2}{1}$–String OT^k or not.

Lemma 3.2. If the $\binom{2}{1}$–ROT^k subprotocol used in Protocol 3.1 satisfies the conditions of Theorem 3.2, then the conditions of Theorem 3.1 for $\binom{2}{1}$–String OT^k

are also met.

Proof. Let $\mathring{R} = \mathring{R}_0\mathring{R}_1$ and $\mathring{C}$ be random variables corresponding to the two parties' random inputs for the $\binom{2}{1}$–ROT^k subprotocol. Let $\mathring{U}, \mathring{V}$ be the two parties' outputs, and let $\mathring{Z}$ be the auxiliary string (which honest parties always ignore). Similarly, let $X = X_0X_1$ and C be the two parties' respective inputs to the String OT protocol and let U, V be the corresponding outputs. Let $D = C\oplus\mathring{C}$ and let $E_0 = X_0 \oplus \mathring{R}_D$, $E_1 = X_1 \oplus \mathring{R}_{\bar{D}}$. Let Z be the auxiliary string denoting the environment. Since no new auxiliary information is made available during Protocol 3.1, we will assume that $Z = \mathring{Z}$.

Condition 1 If the $\binom{2}{1}$–ROT^k subprotocol satisfies Condition 1 then it is easy to see that Condition 1 for $\binom{2}{1}$–String OT^k will also be met. Indeed, in Step 2 the receiver sends to the sender a "flip bit" d which effectively allows him to invert the order in which the input strings of $\binom{2}{1}$–String OT^k are encrypted and thus to eventually output the string x_c of his choice regardless of his initial random choice of $\mathring{c}$ in Step 1. Clearly, the honest sender will not output anything. Therefore, $(U, V) = (\perp, X_C)$.

Condition 2 We first note that indeed, if the sender is honest then $U = \mathring{U} = \perp$.

For the subprotocol, there must exist a random variable $\mathring{C}'$ such that $\mathsf{I}(\mathring{R}; \mathring{C}' \mid \mathring{Z}\mathring{C}) = 0$. Let $C' = \mathring{C}' \oplus D$. Note that by Step 2 when $\mathring{C}, D$ have been determined, the (honest) sender has not made any use of his input X. As only the value of C, which is known to the dishonest receiver at the beginning of the protocol, could have influenced Steps 1 and 2, it

must be the case that C' contains no information about X beyond what is already included in C and thus $I(X; C' \mid ZC) = 0$.

We now need to show that $I(X; V \mid ZCC'X_{C'}) = 0$ or equivalently, that

$$H\left(X_{\bar{C}'} \mid ZCC'X_{C'}\right) = H\left(X_{\bar{C}'} \mid ZCC'X_{C'}V\right).$$

Notice that the dishonest receiver's output (or view) V can, without loss of generality, be assumed to be $V = \mathring{V}E_0E_1 = \mathring{V}E_{C'}E_{\bar{C}'}$. It is clear that $I(\mathring{V}; X \mid C) = 0$ since $\mathring{V}$ is unrelated to X given C. Since we condition on $X_{C'}$, $E_{C'}$ adds no further information on X. As for $E_{\bar{C}'}$, it corresponds to $X_{\bar{C}'}$ after encryption using $\mathring{R}_{D \oplus \bar{C}'} = \mathring{R}_{\bar{C}'}$ as a one-time pad. Since it holds that $H\left(\mathring{R}_{\bar{C}'} \mid \mathring{Z}\mathring{C}\mathring{C}'\mathring{R}_{\hat{C}'}\mathring{V}\right) = k$, by the properties of the one-time pad no information about $X_{\bar{C}'}$ is made available through $E_{\bar{C}'}$. It follows that $H\left(X_{\bar{C}'} \mid ZCC'X_{C'}V\right) = H\left(X_{\bar{C}'} \mid ZCC'X_{C'}\right)$.

Condition 3 The only information made available to the dishonest sender after Step 1 is the value of D. We can thus assume that $U = \mathring{U}D$. Since $D = C \oplus \mathring{C}$ and $H\left(\mathring{C} \mid \mathring{Z}\mathring{R}\mathring{U}\right) = 1$, D contains no information about C. Since X is available to the dishonest sender at the beginning of Protocol 3.1, it is conceivable that the subprotocol was influenced by X. However, $\mathring{U}$ cannot carry any information beyond X since the honest receiver never made any use of C during the $\binom{2}{1}$–ROT^k subprotocol. It follows that $I(C; U \mid ZX) = 0$.

$\square$

4

Reducing String OT to Bit OT variants

As mentioned in Chapter 3, $\binom{2}{1}$–Bit OT is by itself sufficient to securely implement any two-party computation [Kil88]. It should thus not come as a surprise that $\binom{2}{1}$–String OT^k can be reduced to $\binom{2}{1}$–Bit OT, at least in principle. 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. As we shall see in this Chapter, Interactive Hashing can supplement some of the techniques used in reductions of $\binom{2}{1}$–String OT^k to the above-mentioned variants. This gives rise to enhanced reductions that are both more efficient — in fact, most of them can be proved to be asymptotically optimal — and more general than reductions that do not make use of Interac-

tive Hashing [CS06]. Note that as $\binom{2}{1}$–ROT^k and $\binom{2}{1}$–String OT^k are equivalent (see Section 3.5.3), without loss of generality, our goal in this Chapter and the next will be to present reductions of $\binom{2}{1}$–ROT^k to Bit OT and its variants. 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.

4.1 Previous work

All reductions of $\binom{2}{1}$–ROT^k to Bit OT fall within two major categories: reductions based on Self-intersecting Codes [BCS96] (Section 4.1.1) and reductions based on Privacy Amplification [BBR88] (Section 4.1.2).

4.1.1 Reductions based on Self-intersecting Codes

Self-intersecting Codes are a special class of error-correcting 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 always have at least one non-zero position in common. In other words, there exist some position i such that $c_0^i c_1^i \neq 0$. This property turns out to be relevant to Oblivious Transfer, since it can be shown that it guarantees that if I is the set of n positions and v_0, v_1 are any two disjoint subsets of I, then there exists $d \in \{0, 1\}$ such that the following always holds about v_d: if $R \in \{0, 1\}^n$ is randomly chosen among all encodings that decode to any specific $r \in \{0, 1\}^k$, then announcing the bits of R at positions v_d provides no information at all about r.

Consequently, to achieve $\binom{2}{1}$–ROT^k, the sender Alice first selects two random strings R_0, R_1 from $\{0,1\}^n$, which are decoded into $r_0, r_1 \in \{0,1\}^k$ by the code. Alice sends R_0, R_1 pairwise through n executions of Bit OT to the receiver Bob. If Bob is honest, he will receive R_c for some choice bit c, which he can then easily decode into r_c. On the other hand, if Bob is dishonest he will receive the bits of R_0 at positions v_0 and the bits of R_1 at positions v_1 with $v_0 \cap v_1 = \emptyset$. By the properties of the code, then, he learns nothing about r_d for some $d \in \{0,1\}$. Note that this would remain true even if Bob were given $r_{\bar{d}}$ by an oracle.

Advantages and Disadvantages

The main advantage of these reductions is that the self-intersecting code can be chosen ahead of time and embedded once and for all in the protocol for future use. One of the main disadvantages is the rather large expansion factor n/k, theoretically lower-bounded by 3.5277 [Sti99] and in practice ranging from roughly 4.8188 to 18 depending on the type of code. 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.

For more information on Self-intersecting Codes and their use in String OT reductions we refer the reader to [BCS96].

4.1.2 Reductions based on Privacy Amplification

Privacy Amplification [BBR88] is a technique that allows a partially known string to be hashed to a shorter string about which almost nothing is known. This

shorter string can then be used in cryptographic contexts where guaranteeing an (almost) uniform distribution from the point of view of an eavesdropper or adversary is crucial as, for example, in the case where the string is to be used as a one-time pad. For more information on Privacy Amplification, see Section A.3.

In Protocol 4.1 we introduce the construction of [BCW03] upon which our own construction (Protocol 4.2) builds and expands using Interactive Hashing.

Protocol 4.1 Reduction of $\binom{2}{1}$–ROT^k to Bit OT

---

1. Alice selects R_0, $R_1 \in_R \{0, 1\}^n$. Bob selects $c \in_R \{0, 1\}$.

2. Alice sends R_0, R_1 to Bob using n executions of Bit OT, where the ith round contains bits R_0^i, R_1^i. Bob receives R_c^i.

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 when both parties are honest, Protocol 4.1 always succeeds in achieving $\binom{2}{1}$–ROT^k. 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 using matrices M_0, M_1 as hash functions to perform Privacy Amplification with output length $k = n/2 - s$. This guarantees that r_d is uniformly distributed in $\{0, 1\}^k$ and independent of $r_{\bar{d}}$ except with probability exponentially small in the security parameter s. Quite importantly, this property remains true even if Bit OT is

replaced with weaker variants such as XOR OT, Generalized OT and Universal OT — albeit at the cost of having to further reduce the size of k in the last two cases.

Advantages and disadvantages

Besides its apparent simplicity and straightforward implementation, the reduction of Protocol 4.1 has two main advantages over reductions based on Self-intersecting Codes:

1. Using n executions of Bit OT one can achieve $\binom{2}{1}$–ROT^k for k slightly less than $n/2$. This translates into an expansion factor n/k of $2 + \epsilon$, which is smaller than that of any reduction based on Self-intersecting Codes.

2. 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:

1. The proof of security relies heavily on the properties of matrices in $\mathcal{F}_2$ that are used as hash functions for Privacy Amplification in Step 3. A general result for any universal class of hash functions was left as an open problem.

2. 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.

4.2 Reduction of $\binom{2}{1}$–ROT^k to $\binom{2}{1}$–Bit OT using Interactive Hashing

We now demonstrate how Interactive Hashing allows us to augment Protocol 4.1 of [BCW03] with tests that check the receiver's adherence to the protocol. As we shall see, these tests limit a dishonest receiver's ability to deviate from the protocol, thus allowing our reduction to be about twice as efficient in terms of the expansion factor n/k, without any appreciable sacrifice of security.

4.2.1 Preliminaries

Encoding of Subsets as Bit Strings

Let x be a very small positive constant. In our reductions 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 $m = \lceil \log(K) \rceil \leq n\,\mathrm{H}(x)$. See [CCM98] (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 (which could cause the parties to abort the protocol), we modify the encoding of [CCM98] so that any string $w \in \{0, 1\}^m$ encodes the same subset

as $w \pmod{K}$, which is always a valid encoding in the original scheme[1]. 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[2] in the number of encodings per subset will turn out to be of little importance in our scenario thanks to Lemma 4.1 below.

Lemma 4.1. Assume the modified encoding of Section 4.2.1 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, $f' \leq \frac{2|P|}{2|P|+|Q|} \leq \frac{2|P|}{|P|+|Q|} = 2f$. $\qquad \square$

Notation and conventions

In the reduction of Protocol 4.2, two randomly chosen strings $T_0, T_1 \in_R \{0, 1\}^n$ are transmitted pairwise using n executions of Bit OT. We denote by t_0^i, t_1^i the

---

[1]An alternative would be to reduce the fraction of invalid encodings to an arbitrarily small fraction by adding redundancy to the encoding. Indeed, as was shown in [DHRS04], at the modest cost of increasing the encoding length from m to $m + \ell$, one can guarantee that the proportion invalid encodings is no larger than $2^{-\ell}$. While this scheme does not completely eliminate invalid encodings, it has the advantage of assigning an equal number of encodings to each subset.

[2]We remark that the 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.

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 $m = \left\lceil \log\left(\binom{n}{xn}\right) \right\rceil$ using the encoding/decoding scheme of Section 4.2.1. Let $w \in \{0,1\}^m$ be such a bit string, encoding a subset s. We will let $T(w)$ denote the same substring as $T(s)$.

4.2.2 The reduction

Protocol 4.2 presents our reduction of $\binom{2}{1}$–ROT^k to $\binom{2}{1}$–Bit OT.

Intuition behind Protocol 4.2

At Step **1**, the two parties agree on the value of x, namely the proportion among the n bit positions that will be sacrificed for tests. This also determines the encoding length m for subsets of xn positions. At Step **2**, Alice selects the two random n-bit strings that are 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 $s \subset I$ of cardinality xn. The selection is made by first choosing an encoding w uniformly at random in $\{0,1\}^m$ and then mapping it to the corresponding subset s. This ensures that all strings in $\{0,1\}^m$ are equally likely to be Bob's initial choice w, a fact which will become important at Step 5 when w is sent to Alice using Interactive Hashing. Note that s is not uniformly chosen, as some subsets might have two encodings in $\{0,1\}^m$ while others only have one. Nonetheless, as we shall see, it is random enough for our needs. At

Protocol 4.2 Reduction of $\binom{2}{1}$–ROT^k to Bit OT using IH

1. Alice and Bob select x to be a (typically very small) positive constant less than 1. They let $m = \left\lceil \log\left(\binom{n}{xn}\right) \right\rceil$ be the encoding length for the encoding scheme of Section 4.2.1.

2. Alice chooses two random strings $T_0, T_1 \in_R \{0,1\}^n$.

3. Bob chooses a random $c \in_R \{0,1\}$. Bob selects $w \in_R \{0,1\}^m$ uniformly at random and decodes w into a subset $s \subset I$ of cardinality xn.

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 2.1). 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 $|s_0 \cap s_1| \leq 2x^2 n$ 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 bits 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 sends h_0, h_1 to Bob and sets $r_0 = h_0(R_0)$ and $r_1 = h_1(R_1)$.

12. Bob sets $r_c = h_c(R_c)$.

Step **4**, Alice transmits T_0, T_1 in pairs, 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 $T_{\bar{c}}$. This is depicted in Figure 4.1.

The goal of the protocol at Step **5** (see Figure 4.2) 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 with $w_b = w$. As both strings are equally likely to have been Bob's original choice at Step 3, Property 1 of Interactive Hashing guarantees that from (dishonest) Alice's point of view, the value of b is uniformly distributed. At the same time, Property 3 guarantees that the choice of one of w_0, w_1 was effectively random and beyond (dishonest) Bob's control. We will see that this implies that among the corresponding subsets, s_0, s_1, which will be used for tests at Step 9, one is random enough to ensure that a dishonest Bob who deviates "too much" from the protocol will get caught with overwhelming 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 bit positions contained in the 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**, (honest) Bob announces $T_c(s_{\bar{b}}')$ and $T_{\bar{c}}(s_b')$. Note that he can do so since $s_{\bar{b}}' \cap s = \emptyset$ and so he knows all of $T_c(s_{\bar{b}}')$. As for $T_{\bar{c}}(s_b')$, it is also known to him since $s_b' \subseteq s$. Observe that the only information related to c which is implied by the choice of which

substrings to announce is the value of a, which is already made available to Alice at the beginning of this step. Alice can thus correctly guess $c = a \oplus b$ if and only if she can correctly guess b which, as mentioned above, is uniformly distributed given her view. At Step **9**, Alice checks that the strings were announced correctly and are consistent with the value of a — see Figure 4.3. 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, at least one of s_0, s_1 — say, without loss of generality, s_1 — was chosen effectively at random. Therefore, if Bob can announce all bits in $T_0(s_0'), T_1(s_1')$, it must have 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 at this Step with overwhelming probability during these tests. At Step **10**, the two players discard the bits at positions $s_0 \cup s_1$ that were used for tests and concentrate on the remaining j positions. Note that $j \geq n - 2xn$. As Bob passed the tests of Step 9, Alice is convinced that for some $d \in \{0, 1\}$, Bob knows at most $5xn$ bits of T_d and thus at most $5xn$ bits of R_d. This implies that he is missing at least $j - 5xn$ bits of R_d. At 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 Figure 4.4. At Step **12**, honest Bob obtains the string of his choice by applying the appropriate hash function to R_c, which is known to him entirely.
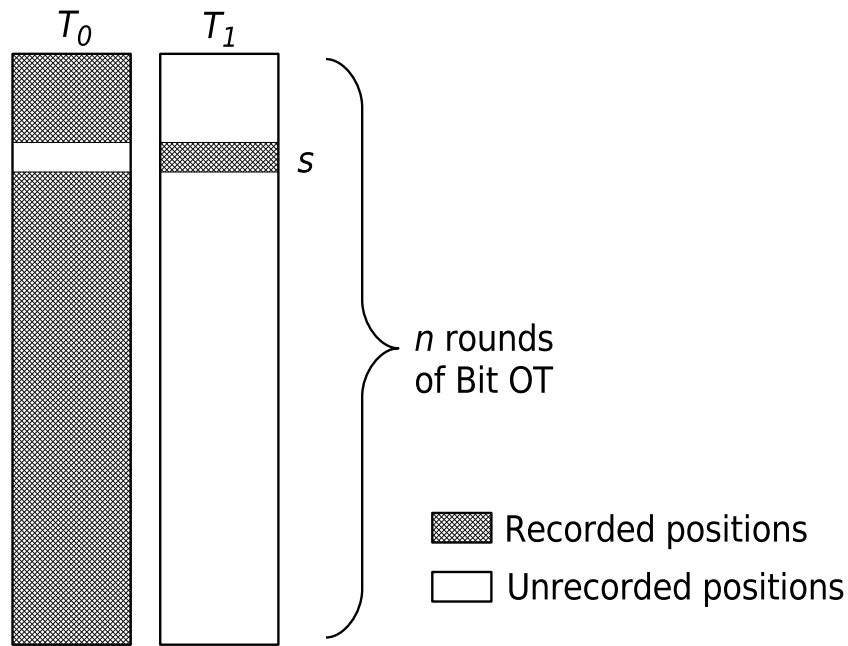
Figure 4.1: During the n Bit OT executions Bob chooses t_c^i at positions $i \in I \setminus s$, and $t_{\bar{c}}^i$ at positions $i \in s$. In the Figure, $c = 0$ so in the end Bob knows $T_0(I \setminus s)$ and $T_1(s)$. Note that while $s \subset I$ is shown here as a contiguous block, in reality the positions it represents occur throughout the n executions.
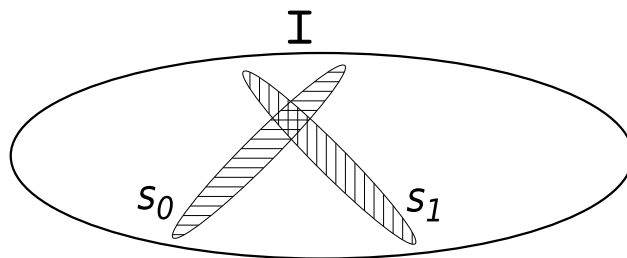
Figure 4.2: Honest Bob uses Interactive Hashing to send the encoding w of his subset s to Alice. Alice does not know which of the two outputs was Bob's input w. These two outputs correspond to subsets s_0, s_1 of which one is s and the other is effectively randomly chosen. The intersection of s_0, s_1 is later excluded to form s'_0, s'_1.
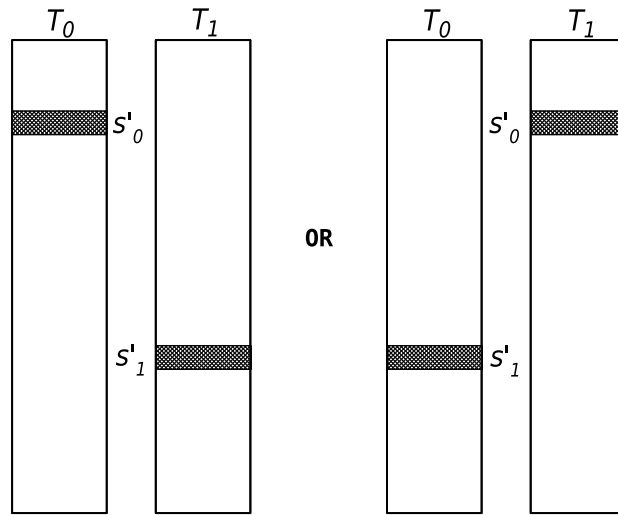
Figure 4.3: After establishing sets s'_0, s'_1, Alice expects Bob to announce either $T_0(s'_0)$ and $T_1(s'_1)$ or $T_0(s'_1)$ and $T_1(s'_0)$, depending on the value of a. If, for example, Bob's choice is $c = 0$ as in Figure 4.1 and $s = s_0$ after Interactive Hashing, then he would choose the latter option.
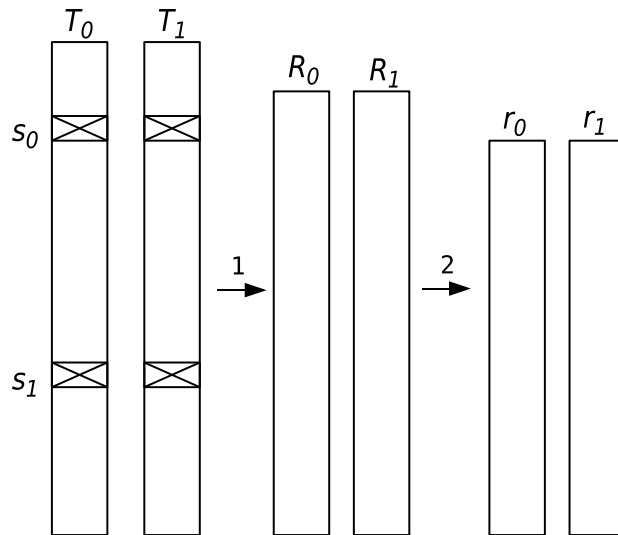
Figure 4.4: After Bob has passed the tests in Step 9, both players ignore the bits at positions $s_0 \cup s_1$ and form strings R_0, R_1 from the remaining bits. Then independent applications of Privacy Amplification on R_0, R_1 give rise to $r_0, r_1 \in \{0, 1\}^k$.

Gains in efficiency

As $k \geq n - 8xn$ where x is a very small constant less than 1, the expansion factor n/k is $1 + \epsilon$ for $\epsilon = \frac{8x}{1-8x} \approx 8x$. As one cannot do better than $n/k = 1$ (see [DM99] for a formal proof of this fact), our expansion factor is asymptotically optimal and represents a two-fold improvement over the corresponding reduction in [BCW03] where the expansion factor is at least $2 + \epsilon'$.

4.2.3   Proof of Security and Practicality

Theorem 4.1 establishes that Protocol 4.2 rarely needs to be aborted when both participants are honest while Theorems 4.2 and 4.3 establish the protocol's security against a dishonest sender and a dishonest receiver, respectively.

Theorem 4.1. The probability of failure of Protocol 4.2 when both participants are honest is exponentially small in n.

Proof. If both parties are honest then Protocol 4.2 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 $|s \cap s'| > 2x^2 n$ 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 $|s \cap s'| > 2x^2 n$. 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 ith 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 $q = \frac{x}{1-x}$. Since $\forall i, \quad q_i < q$, this artificial way of choosing xn positions can only increase the probability of ending up with more than $2x^2n$ collisions. We can use the Chernoff bound (Equation (A.2)) to upper bound this (larger) probability. Assuming $x < 1/2$ and setting $\delta = 1 - 2x$ we get

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

where $\epsilon' = e^{-\frac{(1-2x)^2 x^2}{4(1-x)} n}$. This in turn guarantees that when s' is selected in the appropriate way, the event $|s \cap s'| > 2x^2 n$ occurs with probability $\epsilon < \epsilon'$. In other words, the fraction of bad subsets is upper bounded by $\epsilon < \epsilon'$.

By Lemma 4.1, the fraction of bad strings in $\{0,1\}^m$ is at most 2ϵ. 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 2ϵ. Since by Property 2 of Interactive Hashing, w is paired with some uniformly chosen $w' \ne w$, the probability that the protocol aborts at Step 6 is upper bounded by $2\epsilon < 2 \cdot e^{-\frac{(1-2x)^2 x^2}{4(1-x)} n}$, which is exponentially small in n. $\qquad\square$

Remark: Theorem 4.1 establishes that Condition 1 (Correctness) of Theorem 3.2 defining the information theoretic properties of a perfect protocol for Randomized Oblivious Transfer is met except with exponentially small probability. Indeed, unless the protocol aborts, the honest sender does not output anything and the honest receiver always succeeds in recovering one of the two strings.

Theorem 4.2. 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 a priori equally likely choices. By Property 1 of Interactive Hashing (see Section 2.2), the a posteriori 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 $c = a \oplus b$. □

Remark: Theorem 4.2 establishes that Condition 3 (Security for Player 2) of Theorem 3.2 is perfectly met in all cases since given all available information, the sender's entropy about the receiver's choice bit c is 1 bit.

Security against a dishonest Bob

The proof of Theorem 4.3 establishing the protocol's 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 must be missing "too many" bits of both T_0 and T_1 and will thus fail to pass the tests at Step

8 with overwhelming probability. If, on the other hand, he deviates only a small fraction of the time, then Privacy Amplification at Step 11 will effectively destroy the illegal information he may have obtained. We start with some definitions and lemmas that will help to prove Theorem 4.3.

Definition 4.1. For a bit string T, define $u_p(T)$ to be the number of bits in T that can be guessed correctly with probability at most $p < 1$. These bits will be referred to as *unknown* bits.

Definition 4.2. Let $s \subset I$. Assuming Definition 4.1, we call s *good* for $T \in \{0, 1\}^n$ if $u_p(T(s)) \leq 3x^2 n$, namely if $T(s)$ does not contain more than $3x^2 n$ unknown bits. Otherwise, we call s *bad* for T. We say that s is good for either T_0 or T_1 if at least one of $u_p(T_0(s))$, $u_p(T_1(s))$ does not exceed $3x^2 n$.

Definition 4.3. Let w be a string in $\{0, 1\}^m$. We call w *good* for T if the subset s it encodes is good for T according to Definition 4.2. Otherwise, w is *bad* for T.

Lemma 4.2. Let $u_p(T) \geq 5xn$. Then among all subsets $s \subset I$ of cardinality xn the fraction of good subsets for T is less than $e^{-x^2 n/8}$.

Proof. We will use the Probabilistic Method to show that the probability that a randomly chosen subset s is good for T is less than $e^{-x^2 n/8}$. 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 $u_p(T)$ positions of unknown

bits always satisfies

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

This implies that the probability of choosing a good subset for T 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 $\mu = 4x^2 n$. Applying the Chernoff bound (Equation A.1) with $\delta = 1/4$ we get

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

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

Lemma 4.3. 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 $4 \cdot e^{-x^2 n/8}$.

Proof. It follows from Lemma 4.2 and the Union Bound that the proportion of good subsets for either T_0 or T_1 is no larger than $2 \cdot e^{-x^2 n/8}$. Lemma 4.1 in turn guarantees that the fraction of strings in $\{0,1\}^m$ that are good for either T_0 or T_1 is at most $4 \cdot e^{-x^2 n/8}$. $\square$

Lemma 4.4. 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 4.3, the proportion of good strings in $\{0,1\}^m$ for either T_0 or T_1 is at most $4 \cdot e^{-x^2 n/8}$. By Theorem 2.1, Interactive Hashing guarantees

that the probability that both w_0, w_1 will be good at Step 5 of the protocol is no larger than

$$\epsilon_1 = 15.6805 \cdot 4 \cdot e^{-x^2 n / 8}.$$

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 $u_p(T_0(s_1)), u_p(T_1(s_1)) \geq 3x^2 n$. Moreover, as Alice did not abort at Step 6 it must be the case that $|s_0 \cap s_1| \leq 2x^2 n$. It follows that both $u_p(T_0(s_1')), u_p(T_1(s_1')) \geq 3x^2 n - 2x^2 n = x^2 n$. Therefore, however Bob decides to respond in Step 8, he must correctly guess the value of at least $x^2 n$ unknown bits in one of T_0, T_1. As the bits were independently chosen, the probability of guessing them all correctly is no larger than $\epsilon_2 = p^{x^2 n}$.

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 $\epsilon_1 + \epsilon_2$, which is exponentially small in n.

$\square$

Theorem 4.3. The probability of (dishonest) Bob successfully cheating in Protocol 4.2 is exponentially small in n.

Proof. Let $v_0 \subseteq I$ be the subset of all positions i where (dishonest) Bob obtained t_0^i during Step 4. Let v_1 be defined analogously. Note that $v_0 \cap v_1 = \emptyset$. We distinguish two cases, which 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 4.4 (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 provide to Bob all the bits of T_0 (or R_0, or r_0), he 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. $\qquad\square$

Remark: Theorem 4.3 shows that Protocol 4.2 comes arbitrarily close to satisfying Condition 2 (Security for Player 1) of Theorem 3.2 defining the information theoretic properties of a perfect protocol for Randomized Oblivious Transfer. Recall that in Case 1, dishonest Bob is caught except with exponentially small probability while in Case 2, there always exists some "effective" c' such that $|v_{c'}| > n - 5xn$, determined by the end of Step 4. Since Alice's random strings r_0, r_1 are only determined after applying the hash functions h_0, h_1 chosen at Step 11, c' is independent of r_0, r_1. Moreover, as we have seen, Privacy Amplification guarantees that the entropy of $r_{\bar{c}'}$ given all available information is exponentially close to k.

4.3 Extension to weaker variants of Bit OT

We demonstrate that Protocol 4.2 can accommodate certain weaker versions of Bit OT, specifically XOT, GOT and UOT as described in Section 3.2. We show that the Protocol 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 4.1 and 4.2 (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 becomes even more involved and requires a separate analysis in each case. However, the basic idea remains 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.

4.3.1 Security against a dishonest Bob using XOT

Theorem 4.4. The probability of (dishonest) Bob successfully cheating in Protocol 4.2 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 4.2, 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 $k = (j - 5xn) - xn$ destroys 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 both $u_{1/2}(T_0)$ and $u_{1/2}(T_1)$ are at least $5xn$. By Lemma 4.4, 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). The reduction of String OT to XOT using Protocol 4.2 is thus again twice as efficient compared to the one in [BCW03].

4.3.2 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 $f_0, f_1, f_\oplus, f_{AND}$ defined in Equation (4.1) 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 $v_0, v_1, v_\oplus, v_{AND} \subseteq I$ be the positions where Bob requested $f_0, f_1, f_\oplus, f_{AND}$, respectively.

$$f_0(t_0, t_1) = t_0 \qquad f_\oplus(t_0, t_1) = t_0 \oplus t_1$$
$$f_1(t_0, t_1) = t_1 \quad f_{AND}(t_0, t_1) = t_0 \wedge t_1.$$
(4.1)

A necessary modification to Protocol 4.2

Our proof of security requires that the output length of the hash functions used for Privacy Amplification be slightly shorter than in the case of Bit OT and XOT. Specifically, in Step 11 we let $k = (j - 8xn) - xn \geq n - 11xn$.

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_{AND} may or may not result in loss of information about (t_0, t_1): with probability $1/4$ the output of f_{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_{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_{AND} in the first round. If he is lucky and the output is 1, he asks for f_{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 $|v_{AND}|$ 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 $|v_0|, |v_1| > n - 8 \varkappa n$.

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 "steps on 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. Note 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 S_1, S_2 never had to "brake"). Therefore, Strategy S succeeds only if one of S_1, S_2 succeeds and so $\delta \leq \delta_1 + \delta_2$.

Remark: this upper bound is not unreasonably large: S_1 and S_2 might be successful in disjoint events. As S has more flexibility than either of them during Step 4, it is conceivable that $\delta > \max(\delta_1, \delta_2)$.

It remains to prove that both δ_1, δ_2 are exponentially small in n. To do this, we let Σ_1, Σ_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 Δ_1, Δ_2 are exponentially

small in n, and therefore so is δ (since $\delta \leq \delta_1 + \delta_2 \leq \Delta_1 + \Delta_2$).

Theorem 4.5. The probability of (dishonest) Bob cheating in (modified) Protocol 4.2 is exponentially small in n even if Bit OT is replaced with GOT.

Proof. We will prove that Δ_1, Δ_2 are both exponentially small in n.

Probability of cheating using any Strategy Σ_1   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 Δ_1.

Probability of cheating using any Strategy Σ_2   We start by showing that $\Pr\left[u_{2/3}(T_1) \leq 5xn\right]$ is small. Since any such strategy guarantees that $|v_1| \leq n - 8xn$, it follows that $|v_0 \cup v_\oplus \cup v_{\mathrm{AND}}| \geq 8xn$. Given this constraint, the probability that $u_{2/3}(T_1) \leq 5xn$ is maximized if $|v_{\mathrm{AND}}| = 8xn, |v_0| = |v_\oplus| = 0$. 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_{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 A.1) with $(n, p, \delta) \mapsto (8xn, 3/4, 1/6)$ gives

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

By the same reasoning, this also holds for $u_{2/3}(T_0)$ and so, by the Union Bound, $u_{2/3}(T_0)$ and $u_{2/3}(T_1)$ are both at least $5xn$ except with probability at most $2 \cdot e^{-xn/12}$. In this case, Lemma 4.4 guarantees that Bob will only manage to clear Step 9 with some exponentially (in n) small probability ϵ. We conclude that using any Strategy Σ_2, Bob can successfully cheat with probability $\Delta_2 \leq 2 \cdot e^{-xn/12} + \epsilon$ which is also exponentially small in n.

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

Gains in efficiency

As $k \geq n - 11xn$ where x is a very small positive constant, the expansion factor n/k is $1 + \epsilon'$ for $\epsilon' = \frac{11x}{1-11x} \approx 11x$. This factor is only slightly larger than the one for the case of Bit OT and XOT and remains asymptotically optimal. Compared to the corresponding reduction in [BCW03], ours improves efficiency by a factor of about 4.8188.

4.3.3   Security against a dishonest Bob using Universal OT

In the case where Bit OT is replaced with UOT, at each execution during 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 input bits 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:

$$H\left((B_0, B_1) \mid \Omega\right) \geq \alpha. \tag{4.2}$$

Note that we require α to be at most 1, since otherwise the channel would disallow honest behavior as well. Let $\epsilon < 1/2$ be a (very small) positive constant. We can partition all possible channels satisfying the constraint of Equation 4.2 into the following three categories.

Ω_0: All channels satisfying $H\left(B_0 \mid \Omega\right) < \epsilon\alpha$ and $H\left(B_1 \mid B_0\Omega\right) > (1 - \epsilon)\alpha$.

Ω_1: All channels satisfying $H\left(B_1 \mid \Omega\right) < \epsilon\alpha$ and $H\left(B_0 \mid B_1\Omega\right) > (1 - \epsilon)\alpha$.

Ω_b: All channels satisfying $H\left(B_0 \mid \Omega\right), H\left(B_1 \mid \Omega\right) \geq \epsilon\alpha$.

Let $\rho(\alpha)$ be the unique solution to the equation $h(x) = \alpha$ for $x \in [0, 1/2]$. Let $p_0 = p_1 = \rho\left((1 - \epsilon)\alpha\right)$ and $p_b = \rho\left(\epsilon\alpha\right)$. Then from Fano's inequality and Lemma A.1 (Section A.2) 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 Ω_0 and this is true even if the value of B_0 is known with certainty (via an oracle, say). There thus exists an indicator random variable Δ_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 that B_1 is known with certainty.

- Likewise, p_1 lower bounds the error probability when guessing B_0 given the output of a channel of type Ω_1 and the value of B_1. This implies the

94

existence of side information in the form of an indicator random variable Δ_1 that leads to an erasure of B_0 with probability $2p_1 = 2p_0$.

- When using a channel of type Ω_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 Δ_b^0 (resp. Δ_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 Δ_b^0, Δ_b^1 each time. To see why this is so, suppose both were provided at the same time, with $\Delta_b^0 = 1$. Since the value of Δ_b^1 along with that of Ω might convey more information about B_0 than would otherwise be available in Ω alone, one can no longer assume that this event corresponds to an erasure of B_0.

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:

Ω_0: The exact value of B_0, as well as the value of Δ_0. Note that this leads to B_1 being erased with probability $2p_0$.

Ω_1: The exact value of B_1, as well as the value of Δ_1. Note that this leads to B_0 being erased with probability $2p_1 = 2p_0$.

Ω_b: One of Δ_b^0, Δ_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 4.2

Our proof of security will require that we reduce k even further at Step 11, by setting $k = 2p_0(j - 8p_b n)$. For convenience, we will also set $x = p_b^2$ at Step 1.

Theorem 4.6. The probability of dishonest Bob successfully cheating in (modified) Protocol 4.2 is exponentially small in n even if the Bit OT protocol is replaced with UOT satisfying the constraint of Equation (4.2).

Proof. Let v_0, v_1, $v_b \subseteq I$ be the positions in Step 4 where Bob selected a channel of type Ω_0, Ω_1, Ω_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 $|v_0|, |v_1| \leq n - 6p_b n$.

As in the proof of security for GOT in Section 4.3.2, we will assume the existence of two strategies S_1, S_2 initially following the choices of Bob's strategy S, but ensuring that Case 1 and Case 2 respectively always holds. We will show that the probabilities of successfully cheating of *any* adaptive strategies Σ_1, Σ_2 satisfying the constraints of Case 1 and Case 2, respectively, are both exponentially small in n and thus so is their sum, which in turn upper bounds the probability that *any* adaptive strategy S that dishonest Bob may use will successfully cheat.

Probability of successfully cheating using any Strategy Σ_1   Without loss of generality, let $|v_0| > n - 6p_b n$ at the end of Step 4. This implies that at

96

least $j - 6p_b n$ of the bits of R_1 were received over a channel of type Ω_0. Let μ_1 be the expected number of erasures in R_1, resulting from the side information Δ_0 provided by the oracle in each round. Then $\mu_1 \geq 2p_0 (j - 6p_b n)$. From the Chernoff bound (Equation A.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.

The probability of any strategy Σ_1 successfully cheating is at most equal to the probability that there are too few erasures to begin with (fewer than $2p_0 (j - 7p_b n)$) plus the probability that there are enough erasures but Privacy Amplification fails to produce an almost uniformly distributed string. As both probabilities are exponentially small in n, so is their sum.

Probability of successfully cheating using any Strategy Σ_2 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 4.4 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 Ω_0 or Ω_b. We will assume that exactly $6p_b n$ bits were sent over a channel of type Ω_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 (Equation A.1)

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

where λ is exponentially small in n.

The same argument applies to $u_{1/2}\left(T_0\right)$. Therefore, except with probability at most 2λ, both $u_{1/2}\left(T_0\right), u_{1/2}\left(T_1\right) \ge 5xn$ in which case, by Lemma 4.4 Bob fails to clear Step 9 with probability $1 - \epsilon'$ where ϵ' is exponentially small in n. We conclude that using any Strategy Σ_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 4.3.2, the probability of successful cheating for any adaptive strategy S is upper bounded by the sum of the largest possible probabilities of success of strategies of type Σ_1, Σ_2. We have shown that both of these are exponentially small. □

Gains in efficiency

In both our reduction and that of [BCW03], the expansion factor n/k is a function of α. In our case

$$k = 2p_0\left(j - 8p_b n\right)$$
$$\ge 2p_0\left(n - 2xn - 8p_b n\right)$$
$$= 2p_0\left(n - 2p_b^2 n - 8p_b n\right).$$

Since $p_0 = \rho((1-\epsilon)\alpha), p_b = \rho(\epsilon\alpha)$, 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 [BCW03] is at least $\frac{4\ln 2}{p_e}$ where p_e is the unique solution in $(0, 1/2]$ to the equation $h(p_e) + p_e \log_2 3 = \alpha$. Thus our expansion factor is about $\frac{4\ln 2}{p_e} \cdot 2\rho(\alpha) > 8\ln 2 = 5.545$ times smaller than the one in [BCW03]. Note that the inequality follows from the fact that $\rho(\alpha) > p_e$. This can be seen by observing that $h(\rho(\alpha)) = \alpha = h(p_e) + p_e \log_2 3$ and thus $h(\rho(\alpha)) > h(p_e)$. Since both $\rho(\alpha)$ and p_e are at most $1/2$ and the entropy function h is strictly increasing in the range $[0, 1/2]$, it follows that $\rho(\alpha) > p_e$.

Remark: 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 α is left as an open problem.

4.4 Conclusion, open problems and possible avenues of further research

In this Chapter, we have demonstrated how tests based on Interactive Hashing can be embedded in reductions of String OT to Bit OT and various weaker primitives in order to ensure the receiver's adherence to the protocol. By severely limiting a dishonest receiver's ability to deviate from the protocol without getting caught, these tests allow our reductions to be much more efficient than others in the literature, without any appreciable impact on security. Our reductions are provably asymptotically optimal for the case of Bit OT, XOT and GOT, as well

as for the special case of UOT where $\alpha = 1$. Moreover, our reductions are more general since they can use any 2-universal family of hash functions to perform Privacy Amplification.

We end this chapter by listing some problems that our current work leaves open, as well as some suggestions for further research.

- Modify Protocol 4.2 so that it never aborts when both participants are honest. One possibility to go about this is to abolish Step 6 and show that Interactive Hashing at Step 5 would be effective in preventing dishonest Bob from obtaining subsets s_0, s_1 that have too large an intersection.

- Prove that our reduction is optimal for all α in the case of UOT, or modify it accordingly to achieve optimality.

- Replace the Interactive Hashing Protocol (Protocol 2.1) with an appropriately adapted implementation of the constant round Protocol of [DHRS04] (see Section 2.3.4) and prove that the ensuing reduction (Protocol 4.2) remains secure.

5

Reducing String OT to Rabin OT

As we have seen in Chapter 4, $\binom{2}{1}$–String OT^k can be efficiently reduced, via $\binom{2}{1}$–ROT^k, to Bit OT and other weaker variants. As Bit OT is in turn equivalent to Rabin OT (see [Cré87]), it follows that a reduction of $\binom{2}{1}$–String OT^k to Rabin OT is also possible. In fact, the main technique behind the reduction of Bit OT to Rabin OT in [Cré87] can be used in conjunction with later results on Privacy Amplification [BBR88] to provide a direct reduction of $\binom{2}{1}$–ROT^k requiring $n = (4 + \epsilon)k$ executions of Rabin OT.

In the present Chapter we will be concerned with providing an *optimal* reduction of $\binom{2}{1}$–ROT^k to Rabin OT. Our reduction (Protocol 5.1) employs tests based on Interactive Hashing similar to those used in the reduction of $\binom{2}{1}$–ROT^k to $\binom{2}{1}$–Bit OT (Protocol 4.2) in order to prevent a dishonest receiver from deviating "too much" from the prescribed behavior. This allows us to implement

$\binom{2}{1}$–ROT^k with only $n = (2 + \epsilon')k$ executions of Rabin OT. We observe that after n executions of Rabin OT, the expected entropy of the receiver about the transmitted bits is $n/2$. This precludes reductions with an expansion factor n/k smaller than 2 (see [DM99] for a formal proof) and suggests that this reduction, just like the one in Protocol 4.2, is asymptotically optimal.

Remark: As in Chapter 4, we will be focusing on reducing $\binom{2}{1}$–ROT^k rather than $\binom{2}{1}$–String OT^k since the former is easier to work with even though the two are in fact equivalent (see Section 3.5.3).

5.1 Optimally reducing $\binom{2}{1}$–ROT^k to Rabin OT using Interactive Hashing

Notation and conventions

Unless otherwise noted, our notation is consistent with that of Chapter 4, in particular Section 4.2.1. In our reduction (Protocol 5.1), Alice transmits n randomly chosen bits to Bob using n executions of Rabin Bit OT. We will call the positions of bits received by Bob "good" while the "bad positions" will be those of bits that were erased during the transfer. Let G and B be the set of all good and bad positions, respectively, with each element being an integer in the interval $[1, n]$. Alice and Bob choose x to be a (very small) positive constant that will determine the fraction of bits that will be sacrificed for tests. Let $y = 1/2 - 2x$ and let I denote the set of all positions $1, \ldots, yn$. Let R be a bit string of length yn. For any subset $s \subseteq I$, we will let $R(s)$ be the substring consisting of the bits of R at

the positions in s, in increasing order of position. Using this notation, note that $R(I) = R$.

Encoding of Subsets as Bit Strings

Alice and Bob will use the modified encoding scheme of Section 4.2.1 to encode subsets $s \subset I$ of cardinality xn (out of yn) as bit strings of length $m = \left\lceil \log \left(\binom{yn}{xn} \right) \right\rceil$.

5.1.1 The reduction

Protocol 5.1 presents our reduction of $\binom{2}{1}$–ROT^k to Rabin OT.

Intuition behind Protocol 5.1

At Step 1 Alice and Bob agree on the value of x which determines the number of bits that will eventually be used for tests. At Step 2 Alice uses n executions of Rabin OT to transmit n randomly chosen bits to Bob. As erasures occur independently with probability $1/2$ at every execution, with overwhelming probability Bob will receive no more than $(1/2 + x)n$ bits and no less than $(1/2 - x)n$ by the end this step. At Step 3, Bob ensures that enough bits have been received for the needs of the rest of the protocol, and aborts otherwise. The good and bad positions are collected in sets G and B, respectively. At Step 4, Bob chooses $c \in_R \{0, 1\}$ at random and defines two bit strings R_0, R_1 of length yn each so that R_c is composed exclusively of good positions. As for $R_{\bar{c}}$, the protocol requires that it has at least xn good positions spread randomly throughout $R_{\bar{c}}$

Protocol 5.1 Reduction of $\binom{2}{1}$–ROT^k to Rabin Bit OT using IH

1. Alice and Bob select a (typically very small) positive constant $x < 1/4$ and set $y = 1/2 - 2x$.

2. Alice transmits n random bits using n executions of Rabin OT.

3. Bob collects the good and bad positions in sets G and B, respectively. He aborts if $|G| < (1/2 - x)n = yn + xn$.

4. Bob chooses at random $c \in_R \{0, 1\}$ as well as $w \in_R \{0, 1\}^m$, where $m = \left\lceil \log\left(\binom{yn}{xn}\right) \right\rceil$. He decodes w into a subset s of cardinality xn (out of yn) using the encoding scheme of Section 4.2.1. He then defines two yn–bit strings R_c and $R_{\bar{c}}$ as follows: yn positions from G are chosen at random and without repetition. The corresponding bits, in the order chosen, make up R_c. For $R_{\bar{c}}$, xn (new) positions are chosen at random from G, and define substring $R_{\bar{c}}(s)$. For the remainder, $yn - xn$ positions are randomly chosen from $G \cup B$.

5. Bob announces the bit positions making up R_0 and R_1. Alice checks that no bit position appears more than once.

6. Bob sends w to Alice using Interactive Hashing (Protocol 2.1). Let w_0, w_1 be the output strings, let $s_0, s_1 \subset I$ be the corresponding subsets of cardinality xn and let $b \in \{0, 1\}$ be such that $w_b = w$.

7. Bob announces $a = b \oplus c$ as well as $R_0(s_{1-a})$ and $R_1(s_a)$.

8. Alice checks that the substrings announced contain no errors.

9. Alice announces h_0, h_1, chosen randomly and independently from a 2-universal family of hash functions with input length yn and output length $k = yn - 6xn$. She sets $r_0 = h_0(R_0)$ and $r_1 = h_1(R_1)$.

10. Bob sets $r_c = h_c(R_c)$.

— see Figure 5.1. To this effect, Bob chooses a substring s of cardinality xn. This choice is done by first choosing an m–bit string w uniformly at random and then using the encoding scheme of Section 4.2.1 to map it into s. The fact that the choice of w was uniform will be crucial in ensuring that Bob's choice bit c remains hidden from Alice at later steps. We observe that s is not entirely uniformly selected since the encoding scheme maps some subsets to two bit strings and others to only one. This, however, will turn out not to be important in our scenario.

At Step 5, Bob provides a description of R_0, R_1 to Alice, who makes routine checks to ensure that both strings are properly constructed. As from the point of view of (dishonest) Alice both R_0 and R_1 are equally likely to consist entirely of good positions for Bob, she cannot guess which of the two is R_c with probability greater than $1/2$ and thus (honest) Bob's choice bit c is perfectly hidden.

Steps 6 through 8 introduce tests based on Interactive Hashing that are designed to catch a dishonest Bob who deviates from the prescribed behavior at Step 5 by putting "too many" good positions in both R_0 and R_1. The tests allow Alice to verify that Bob has indeed used almost exclusively good bits to construct one of R_0, R_1. If so, then Bob has very little information about the other string (a fact that will be used later at Step 9 to set the parameters for Privacy Amplification). More specifically, at Step 6 Bob uses Interactive Hashing to send to Alice string w encoding $s \subset I$ (recall that, if Bob is honest, then he knows all of $R_{\bar{c}}(s)$). The output of Interactive Hashing consists of two strings w_0, w_1 encoding subsets s_0, s_1 — see Figure 5.2 — and there must exist $b \in \{0, 1\}$ such

that $w_b = w$. While the value of b is known to Bob, it is completely hidden from (any dishonest) Alice thanks to Property 1 of Interactive Hashing and the fact that both w_0 and w_1 are equally likely to have been chosen by Bob at Step 4. At Step 7, (honest) Bob effectively announces substrings $R_c(s_{\bar{b}})$ and $R_{\bar{c}}(s_b)$, both of which consist entirely of good positions — see Figure 5.3. Note that Bob's choice of which substrings to announce does not carry any information related to c beyond the value of $a = b \oplus c$ which was announced at the beginning of the step. Consequently, as long as Alice cannot guess b, Bob's choice bit c remains perfectly hidden from her.

While these tests are easy to pass for an honest Bob, a dishonest Bob who has deviated "significantly" from the protocol will get caught with overwhelming probability and cause the protocol to abort, while a dishonest Bob who has deviated only slightly will obtain no advantage in the end. The reasoning is as follows: by the properties of Rabin OT and the Chernoff bound, Bob receives no more than $(1/2 + x)n$ bits during Step 4 except with some probability exponentially small in n. Assuming that this is the case, we observe that since R_0 and R_1 are made up of distinct bit positions and have length $yn = (1/2 - 2x)n$ each, at least one of the two — say, without loss of generality, R_0 — will consist mostly of good positions (Case 1) or else both R_0, R_1 will have a significant fraction of bad positions (Case 2). In Case 1, R_1 will necessarily consist mostly of bad positions and so at Step 9, Privacy Amplification with output length k slightly less than yn will result in an almost uniformly distributed[1] string r_1 — see Figure 5.4. As for Case 2, recall that Property 3 of Interactive Hashing guarantees that

---

[1]Note that r_1 would be almost uniform even if Bob were given R_0 (or r_0) by an oracle.

the choice of at least one of w_0, w_1 (say, without loss of generality, w_1) was effectively out of Bob's control. This also holds for the corresponding subset s_1 which, with overwhelming probability, will be such that both $R_0(s_1)$ and $R_1(s_1)$ contain several bad positions each. Consequently, whatever value of a (dishonest) Bob announces at Step 7, in order to pass the tests he will have to correctly guess the value of a large number of unknown bits, which can only happen with negligible probability. It follows that, except with negligible probability, either Bob has not deviated much from the prescribed behavior and so Privacy Amplification destroys any illegal information he has gathered, or else Bob gets caught cheating at Step 8 in which case Alice aborts the protocol.

Lastly, at Step 10 (honest) Bob can easily obtain r_c by applying the appropriate hash function to R_c, which he knows completely.

Gains in efficiency

We first observe that a reduction of $\binom{2}{1}$–Bit OT to Rabin OT was provided in [Cré87]. It is not hard to see that the approach of [Cré87] can be combined with Privacy Amplification [BBR88] to provide a reduction of $\binom{2}{1}$–ROT^k to Rabin OT. The main difference between this approach and ours is that without the tests ensuring that one of R_0, R_1 is made almost exclusively of good positions, one has to make the assumption that in the worst case, (dishonest) Bob could have divided the good positions evenly between the two strings. To protect against this possibility, the output of Privacy Amplification must be reduced to less than half of the length of each string, yielding an overall expansion factor n/k of at
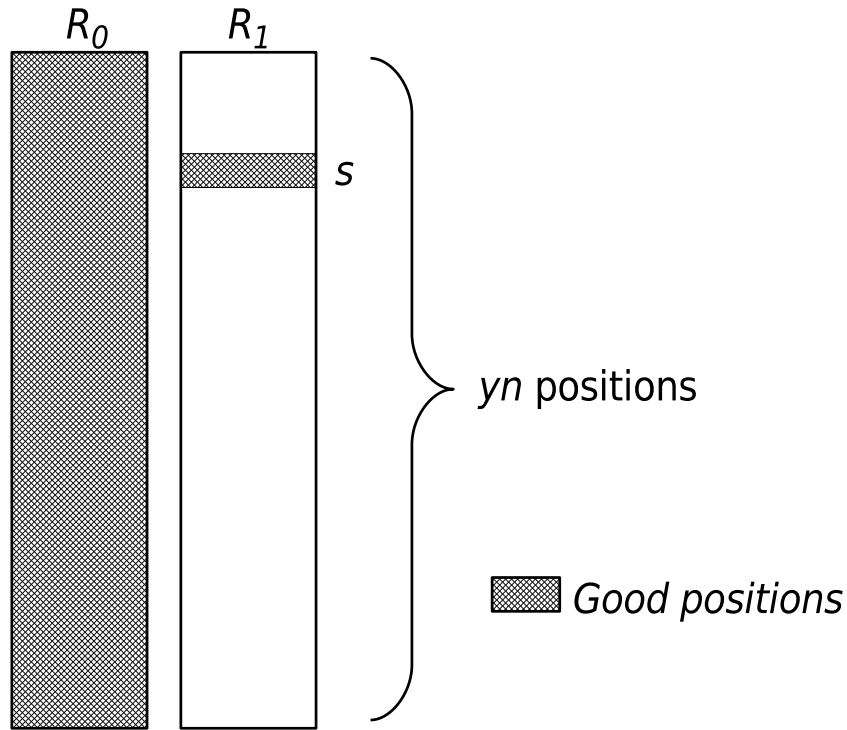
Figure 5.1: Bob constructs two strings, R_0, R_1 by selecting only good positions for the whole of R_c as well as for a small, random subset $s \subset I$ of all positions in $R_{\bar{c}}$. The rest of $R_{\bar{c}}$ consists of leftover, mostly bad positions. In the Figure, $c = 0$. Note that while s is shown here as a contiguous block, in reality the positions it represents are generally distributed throughout $R_{\bar{c}}$.
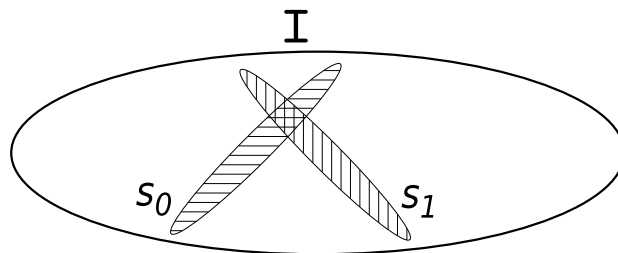

Figure 5.2: Honest Bob sends the string w encoding subset s to Alice through Interactive Hashing. This procedure produces two outputs w_0, w_1, encoding two subsets s_0, s_1. Alice does not know which of the two outputs was Bob's input.
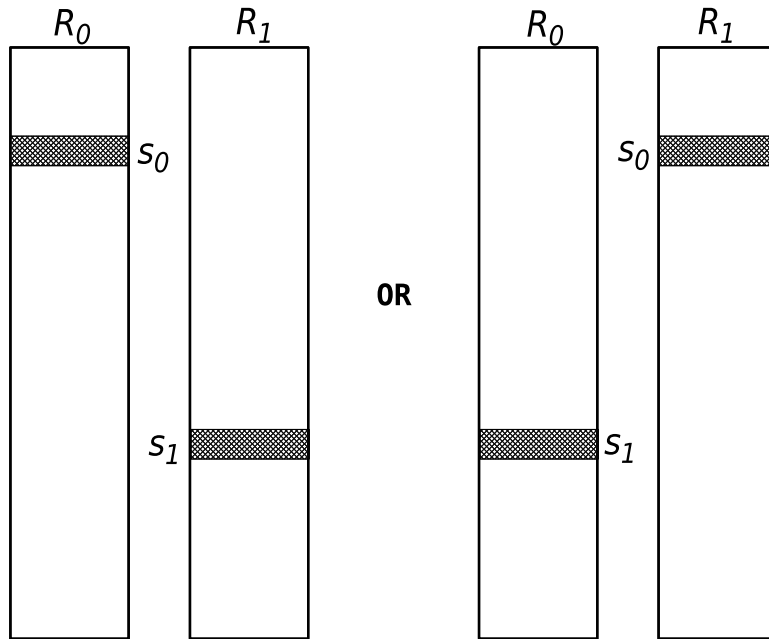
Figure 5.3: Alice expects Bob to announce either $R_0(s_0)$ and $R_1(s_1)$ or $R_0(s_1)$ and $R_1(s_0)$, depending on the value of a. If Bob's choice was $c = 0$ as in Figure 5.1 and $b = 0$ at Step 6, then Bob would choose the latter option.
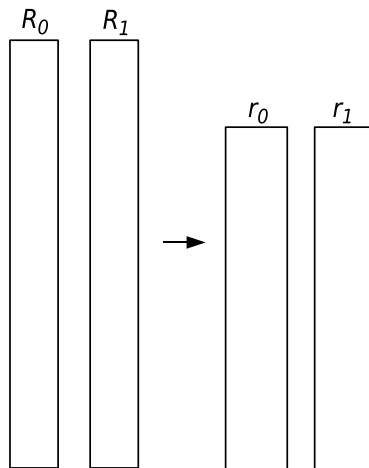
Figure 5.4: Alice performs Privacy Amplification on R_0, R_1 independently, by randomly choosing two functions h_0, h_1 from a 2-universal family. This results in two shorter strings $r_0, r_1 \in \{0, 1\}^k$. If Bob's choice was $c = 0$ then he would know r_0 and have practically no information about r_1, except with negligible probability.

least $4 + \epsilon$.

In our case, as $k = yn - 6xn = (1/2 - 8x)n$ where x is a very small positive constant, the expansion factor n/k is $2 + \epsilon'$ for $\epsilon' = \frac{16x}{1/2 - 8x} \approx 32x$. This represents a two-fold improvement over the method described above. Moreover, since the n bits transmitted at Step 2 give rise to $n/2$ bits of entropy on average and in any reduction of $\binom{2}{1}$–ROT^k the length of the strings k cannot exceed the amount of available entropy [DM99], our reduction is in fact asymptotically optimal.

5.1.2   Proof of Security and Practicality

Theorem 5.1 establishes that Protocol 5.1 rarely needs to be aborted when both participants are honest while Theorems 5.2 and 5.3 establish the protocol's security against a dishonest sender Alice and a dishonest receiver Bob, respectively.

Theorem 5.1. The probability of failure of Protocol 5.1 with honest participants is exponentially small in n.

Proof. If both parties are honest, Protocol 5.1 can only fail at Step 3, namely if the n executions of Rabin OT at Step 2 have not produced enough good positions. Recalling that erasures occur independently with probability $1/2$, we can use the Chernoff bound (Equation A.1) with $(n, p, \delta, \mu) \mapsto (n, 1/2, 2x, n/2)$ to establish that

$$\Pr\left[|G| < (1/2 - x)n)\right] \leq e^{-x^2 n}.$$

Note that in the unlikely event that the protocol is aborted, no private information of either party has been compromised. □

Theorem 5.2. Alice learns nothing about (honest) Bob's choice bit c.

Proof. During Bob's interaction with Alice, c comes into play only at Steps 5 and 7. By the properties of Rabin OT, after each execution at Step 2, Alice cannot guess with probability greater than $1/2$ whether Bob received the bit or an erasure. Consequently, when Bob announces R_0, R_1 at Step 5, from Alice's point of view both strings have equal probability of corresponding to R_c (namely, of being made up entirely of good positions). At Step 7 Bob announces $a = b \oplus c$, so Alice can correctly guess c if and only if she can correctly guess the value of b such that $w = w_b$ after Interactive Hashing at Step 6. As the input w was chosen uniformly at random, Property 1 of Interactive Hashing establishes that from Alice's point of view w_0, w_1 both have probability exactly $1/2$ of having been Bob's input w and thus b is uniformly distributed and perfectly hides the value of c. ∎

Security against a dishonest Bob

Theorem 5.3. The probability of (dishonest) Bob successfully cheating in Protocol 5.1 is exponentially small in n.

The proof of Theorem 5.3 has many similarities to the proof of Theorem 4.3. In a nutshell, the main idea is that Bob has only about $n/2$ good positions at the end of Step 2, except with some probability exponentially small in n. There are two cases to consider. In Case 1, Bob uses "many" good positions in both R_0 and R_1, which means that both strings will necessarily have several bad positions as well. Consequently, with overwhelming probability the subsets produced at

Step 6 will contain many positions corresponding to bits that are unknown to Bob. As a result, except in the unlikely event that Bob guesses all the unknown bits correctly at Step 7, he will fail to clear Alice's checks at Step 8. In Case 2, Bob uses only a few good positions in one of R_0, R_1 (perhaps a few more than the protocol prescribes). In this case, Bob may well be able to pass the tests at Step 8 but Privacy Amplification at Step 9 will then effectively destroy the partial information Bob has about one of the strings, by virtue of having included those extra good positions in its construction.

We start with some definitions and lemmas that will help us prove Theorem 5.3.

Definition 5.1. For a bit string R, define $u_{1/2}(R)$ to be the number of bits in R whose value can only be guessed correctly with probability $1/2$.

Definition 5.2. Let $s \subset I$. Assuming Definition 5.1, we call s *good* for R if $u_{1/2}(R(s)) < x^2 n$. Otherwise, we call s *bad* for R. Similarly, a string $w \in \{0,1\}^m$ is *good* for R if and only if the subset s it encodes is good for R. We say that s is good for either R_0 or R_1 if at least one of $u_{1/2}(R_0(s)), u_{1/2}(R_1(s))$ does not exceed $x^2 n$.

Lemma 5.1. Let $R \in \{0,1\}^{yn}$ be such that $u_{1/2}(R) \geq 2xn$. Then the fraction f of subsets s of cardinality xn that are good for R satisfies $f < e^{-x^2 n/4}$.

Proof. We will use the Probabilistic Method to show that a subset s chosen uniformly at random would be good for R with probability less than $e^{-x^2 n/4}$. One way of choosing s would be to sequentially choose at random and without

replacement xn positions among all yn positions of R. Then for all $1 \leq i \leq xn$, the probability q_i that position i is chosen among the $u_{1/2}(R)$ bad positions always satisfies

$$q_i > \frac{u_{1/2}(R) - xn}{yn} > \frac{2xn - xn}{n/2} = 2x.$$

This implies that the probability of choosing a good subset for R would be strictly greater if we were to choose the xn positions independently at random so that each position is bad with probability $q = 2x$. In this artificial case, the distribution of the number of unknown bits is binomial and we can apply the Chernoff bound (Equation A.1) with $(n, p, \delta, \mu) \mapsto (xn, 2x, 1/2, 2x^2 n)$ to get

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

We conclude that a subset s chosen uniformly at random in the appropriate way has probability strictly smaller than $e^{-x^2 n/4}$ of being good for R, which establishes the claim. $\square$

Lemma 5.2. Let $R_0, R_1 \in \{0, 1\}^{yn}$ and let both $u_{1/2}(R_0)$ and $u_{1/2}(R_1)$ be at least $2xn$. Then the fraction of strings in $\{0, 1\}^m$ that are good for either R_0 or R_1 is no larger than $4 \cdot e^{-x^2 n/4}$.

Proof. It follows directly Lemma 5.1 and the Union Bound that the fraction of subsets s that possess this property is no larger than $2 \cdot e^{-x^2 n/4}$. By Lemma 4.1, the fraction in $\{0, 1\}^m$ of strings that are mapped to such subsets by the encoding scheme of Section 4.2.1 must be no larger than $4 \cdot e^{-x^2 n/4}$. $\square$

Lemma 5.3. In Protocol 5.1 let both $u_{1/2}(R_0), u_{1/2}(R_1) \geq 2xn$. Then the probability that (dishonest) Bob will clear Step 8 is exponentially small in n.

Proof. By Lemma 5.2, the proportion of strings in $\{0,1\}^m$ that are good for either R_0 or R_1 is at most $4 \cdot e^{-x^2n/4}$. Consequently, at Step 6 by Property 3 of Interactive Hashing the probability that (dishonest) Bob can get both w_0, w_1 to be good for either R_0 or R_1 is no larger than $\epsilon_1 = 15.6805 \cdot 4 \cdot e^{-x^2n/4}$. It follows that with probability at least $1 - \epsilon_1$, at least one of the two bit strings — say, without loss of generality, w_1 — is bad for *both* R_0 and R_1. In other words, w_1 corresponds to a subset s_1 with both $u_{1/2}(R_0(s_1))$ and $u_{1/2}(R_1(s_1))$ being at least x^2n. Recalling that at Step 7 Bob must announce either $R_0(s_1)$ or $R_1(s_1)$, we see that he can clear the checks of Step 8 only if he correctly guesses the values of at least x^2n unknown bits. As the bits were independently chosen, the probability of guessing them all correctly is $\epsilon_2 < 2^{-x^2n}$.

Bob clears Step 8 only if he either gets two good strings as outputs from Interactive Hashing or else, if he gets at least one bad string and then correctly guesses all the relevant bits at Step 7. This probability is upper bounded by $\epsilon_1 + \epsilon_2$ which is exponentially small in n. $\qquad\square$

We are now ready to prove Theorem 5.3.

Proof of Theorem 5.3. During each execution at Step 2, Bob receives an erasure independently with probability $1/2$. From the Chernoff bound (Equation A.1) with $(n, p, \delta, \mu) \mapsto (n, 1/2, 2x, n/2)$ we obtain $|B| \geq \frac{n}{2} - xn$ except with probability $\delta_1 \leq e^{-x^2n}$. We condition on $|B| \geq \frac{n}{2} - xn$ and distinguish two cases:

Case 1: One of $u_{1/2}(R_0)$, $u_{1/2}(R_1)$ is smaller than $2xn$.

Without loss of generality, let $u_{1/2}(R_0) < 2xn$. We will show that R_1 is then almost entirely composed of bad positions. Recall that $n - 2yn = 4xn$ bit positions were never used when defining R_0, R_1. Since $u_{1/2}(R_0) + u_{1/2}(R_1) + 4xn \geq |B| \geq \frac{n}{2} - xn$, we have $u_{1/2}(R_1) \geq \frac{n}{2} - 4xn - 2xn - xn = yn - 5xn$. Consequently, at Step 9 Privacy Amplification with output length $k = yn - 6xn$, will produce a string r_1 which, from Bob's point of view, is almost uniformly distributed in $\{0,1\}^k$ except with some exponentially small (in n) probability δ_2. Note that this property holds even if we condition on R_0 (or r_0) since Alice chose her bits independently at random at Step 2.

Case 2: Both $u_{1/2}(R_0)$, $u_{1/2}(R_1) \geq 2xn$.

Then by Lemma 5.3 Bob will fail to clear Step 8 except with some probability δ_3 exponentially small (in n).

The probability that either too few erasures occurred or else that Bob has successfully cheated either in Case 1 or in Case 2 is no larger than $\delta_1 + \max(\delta_2, \delta_3)$. As $\delta_1, \delta_2, \delta_3$ are all exponentially small in n, this establishes the claim. $\square$

5.2 Conclusion

We have demonstrated a direct reduction of $\binom{2}{1}$–ROT^k to Rabin OT. As in Chapter 4, our reduction relies on tests based on Interactive Hashing to verify the receiver's adherence to the protocol. The assurance that a dishonest receiver cannot pass the tests unless he has deviated little from the protocol allows our

reduction to be twice as efficient as a a similar reduction without such tests, achieving an expansion factor n/k of only $2 + \epsilon$. Since the expected entropy of the receiver about the n bits transmitted using Rabin OT is no larger than $n/2$, our reduction is in fact optimal. Our reduction provides yet another example of the applicability of Interactive Hashing to reductions between Oblivious Transfer variants.

6

Summary and Conclusion

The goal of this thesis has been two-fold. First, to provide a study of Interactive Hashing in the information theoretic context. To this end, we have given a definition formalizing its security properties in isolation of any specific application setting. This abstraction enables Interactive Hashing to be treated as a cryptographic primitive in its own right rather than as a class of sub-protocols whose implementation, properties and proof of security all depend, to various extents, on the surrounding application.

In order to demonstrate the practicality of this primitive, we have shown that there exists at least one protocol implementing Interactive Hashing which fully satisfies our security requirements. The corresponding proof of security has been one of the major contributions of this thesis. As in other proofs in the literature establishing the security of similar protocols, its most challenging aspect has been,

by far, showing that if a dishonest sender starts with a small fraction of good strings, he cannot force both outputs of the protocol to be good except with a comparably small probability. Unlike the other proofs that focus on bounding the number of strings remaining after each round, our proof follows the evolution of the number of *pairs* of good strings instead. This is a more natural choice for our setting as the dishonest sender succeeds if and only if exactly one pair remains at the end. Consequently, the probability of success is simply equal to the expected number of such pairs remaining when the protocol finishes. This observation leads to a significantly less complicated proof and results in a simpler, tighter and more general upper bound on the dishonest sender's probability of success. Specifically, it establishes that if the fraction of good strings at the beginning of the protocol is f, then no dishonest sender can succeed in obtaining two good strings with probability greater than $15.6805 \cdot f$. This upper bound is tight up to a small constant since a dishonest sender who uses one of the good strings as input and then acts honestly will succeed with probability slightly less than f.

The second goal has been to highlight the potential of Interactive Hashing as a cryptographic primitive by demonstrating its applicability to reductions of $\binom{2}{1}$–ROT^k to simpler primitives such as $\binom{2}{1}$–Bit OT. In our reduction we use tests based on Interactive Hashing to allow the sender in $\binom{2}{1}$–ROT^k to query the receiver on a small subset of the bits he obtained during the executions of $\binom{2}{1}$–Bit OT. We have shown that the properties of Interactive Hashing guarantee that on one hand, these tests do not compromise the receiver's privacy while on the other hand, they effectively prevent a dishonest receiver from deviating

significantly from the prescribed behavior without getting caught. This extra guarantee allows our reduction to take almost full advantage of the receiver's entropy of the transmitted bits. Specifically, we show that $n = (1+\epsilon)k$ executions of $\binom{2}{1}$–Bit OT suffice to securely implement $\binom{2}{1}$–ROT^k, making our reduction at least twice as efficient as the best known constructions in the literature. As a reduction with an expansion factor n/k smaller than 1 would be theoretically impossible, our reduction is asymptotically optimal.

As far as weaker variants of $\binom{2}{1}$–Bit OT are concerned, we have shown that the reduction works without any modification if $\binom{2}{1}$–Bit OT is replaced with XOR OT, while an imperceptible increase in the expansion factor allows it to accommodate Generalized OT as well. Further modifications allow the reduction to also cover the case of Universal OT. In the case of XOR OT and Generalized OT our reductions remain asymptotically optimal and improve efficiency by a factor of 2 and 4.8188, respectively. As for Universal OT, the reduction is more efficient than previous ones by a factor of at least 5.545, but we prove it to be optimal only for the special case where $\alpha = 1$, leaving the general case as an open problem. It should be noted that our reductions can use any 2-universal family of hash functions in the Privacy Amplification phase. Consequently, besides being more efficient, they are also more general than previous ones which require special classes of hash functions.

Lastly, we have shown that our techniques can be adapted to provide a direct reduction of $\binom{2}{1}$–ROT^k to Rabin OT with an expansion factor of $2 + \epsilon'$. This reduction is again asymptotically optimal, and twice as efficient compared to the

case where Interactive Hashing is not used.

A

Tools and Mathematical Background

A.1 Tail 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 [Che52] (as they appear in [Vaz04], p.354) for $0 < \delta \leq 1$:

$$\Pr\left[B(n, p) \leq (1 - \delta)\mu\right] \leq e^{-\delta^2 \mu/2} \tag{A.1}$$

$$\Pr\left[B(n, p) \geq (1 + \delta)\mu\right] \leq e^{-\delta^2 \mu/4}. \tag{A.2}$$

From (A.1) we can also deduce the following inequality

$$\Pr\left[B(n, p) \leq \mu - \Delta n\right] \leq e^{-\Delta^2 n/2}. \tag{A.3}$$

A.2 Error probability and its concentration on an erasure event

A.2.1 Fano's lemma

(Adapted from [BCW03]) Let X be a random variable with range $\mathcal{X}$ and let Y be another 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. Let $\mathsf{H}\left(X \mid Y\right)$ denote the conditional entropy of X given Y, and let $h(p) \stackrel{\text{def}}{=} -p \log p - (1-p) \log(1-p)$. Then p_e satisfies:

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

A.2.2 Specifying an erasure event Δ

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 $\Pr\left[\bar{E}\right] \geq \Pr\left[E\right] \geq p_e \geq p$. Define Δ to be another indicator random variable such that

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

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

Lemma A.1. 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 $2p \leq 2p_e$.

A.3 Privacy Amplification

Privacy Amplification [BBR88] 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 [BBCM95] (also covered in [BBR88]) which assumes that there are always u or more unknown physical bits in R (as opposed to general bounds on R's entropy).

Theorem A.1. Let R be a random variable with uniform distribution 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 $r = H(R)$ (note that H, r, R are random variables). Then the following

holds:

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

It follows from Equation (A.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 $2^{-s/2}/\ln 2$. In other words, except with exponentially (in s) small probability, Bob's information about r is no more than an exponentially small fraction of a bit.

Bibliography

[ADR02]   Yonatan Aumann, Yan Zong Ding, and Michael O. Rabin. Everlasting security in the bounded storage model. *IEEE Transactions on Information Theory*, 48(6):1668–1680, June 2002.

[BBCM95] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli Maurer. Generalized privacy amplification. *IEEE Transaction on Information Theory*, 41(6):1915–1923, November 1995.

[BBR88]   Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, 1988.

[BCR86]   Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. All-or-nothing disclosure of secrets. In *Advances in Cryptology: CRYPTO*, pages 234–238, 1986.

[BCS96]   G. Brassard, C. Crépeau, and M. Sántha. Oblivious transfers and intersecting codes, 1996.

[BCW03]   Gilles Brassard, Claude Crépeau, and Stefan Wolf. Oblivious transfers and privacy amplification. *IEEE Transaction on Information Theory*, 16(4):219–237, 2003.

[CCM98]   Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. In *Proc. 39th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 493–502, 1998.

[Che52]   H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on a sum of observations. *The Annals of Mathematical Statistics*, 23:493–507, 1952.

[CK]   Claude Crépeau and Joe Kilian. Private communication.

[Cré87]   Claude Crépeau. Equivalence between two flavours of oblivious transfers. In *CRYPTO*, pages 350–354, 1987.

[CS91]   Claude Crépeau and Miklós Sántha. On the reversibility of oblivious transfer. *Lecture Notes in Computer Science*, 547:106–113, 1991.

[CS06]   Claude Crépeau and George Savvides. Optimal reductions between oblivious transfers using interactive hashing. In Serge Vaudenay, editor, *Advances in Cryptology: EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 201–221. Springer, 2006.

[CSSW06]   Claude Crépeau, George Savvides, Christian Schaffner, and Jürg Wullschleger. Information-theoretic conditions for two-party secure function evaluation. In *Advances in Cryptology: EUROCRYPT '06*, Lecture Notes in Computer Science, pages 538–554. Springer-Verlag, 2006.

[DHRS04]   Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In Moni Naor, editor, *Proc. 1st Theory of Cryptography Conference (TCC 2004)*, volume 2951 of *Lecture Notes in Computer Science*, pages 446–472. Springer, 2004.

[Din01]   Yan Zong Ding. Oblivious transfer in the bounded storage model. In Joe Kilian, editor, *Advances in Cryptology: CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 155–170. Springer, 2001.

[DM99]   Yevgeniy Dodis and Silvio Micali. Lower bounds for oblivious transfer reductions. *Lecture Notes in Computer Science*, 1592:42–54, 1999.

[EGL85]   Shimon Even, Oded Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28:637–647, 1985. A preliminary version was presented at CRYPTO '82.

[GMW87]   Oded Goldreich, Silvio Micali, and Avi Wigderson. 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)*, pages 218–229, 1987.

[Gol04]   Oded Goldreich. *Foundations of Cryptography*, volume I & II. Cambridge University Press, 2001–2004.

[Kil88]   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.

[NOVY98]   Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, March 1998.

[OVY92]   Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Secure commitment against a powerful adversary. In Alain Finkel and Matthias Jantzen, editors, *STACS '92: Proc. 9th Annual Symposium on Theoretical Aspects of Computer Science*, volume 577 of *Lecture Notes in Computer Science*, pages 439–448. Springer, 1992.

[OVY93]   R. Ostrovsky, R. Venkatesan, and M. Yung. Fair games against an all-powerful adversary. In *Advances in Computational Complexity Theory*, volume 13, pages 155–169. AMS, 1993. Initially presented at DIMACS workshop, 1990. Extended abstract in the proceedings of Sequences '91, June 1991, Positano, Italy.

[OVY94]   Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Interactive hashing simplifies zero-knowledge protocol design. In Tor Helleseth, editor, *Advances in Cryptology: EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 267–273. Springer, 1994.

[Rab81]   M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Memo TR–81, Aiken Computation Laboratory, Harvard University, 1981.

[Sti99]   D.R. Stinson. Some results on nonlinear zigzag functions. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 29:127–138, 1999.

[Vaz04]   Vijay V. Vazirani. *Approximation Algorithms*. Springer, 1 edition, 2004.

[Wie70]   Stephen Wiesner. Conjugate coding. Reprinted in SIGACT News, vol. 15, no. 1, 1983, original manuscript written ca. 1970.

[WW06]   Stefan Wolf and Jürg Wullschleger. Oblivious transfer is symmetric. In *Advances in Cryptology: EUROCRYPT '06*, pages 222–232, 2006.

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