A Quantitative Approach to Reductions in Secure Computation

Amos Beimel * Tal Malkin †

December 1, 2003

Abstract

Secure computation is one of the most fundamental cryptographic tasks. It is known that all functions can be computed securely in the information theoretic setting, given access to a black box for some complete function such as AND. However, without such a black box, not all functions can be securely computed. This gives rise to two types of functions, those that can be computed without a black box ("easy") and those that cannot ("hard"). However, no further distinction among the hard functions is made.

In this paper, we take a quantitative approach, associating with each function f the *minimal number* of calls to the black box that are required for securely computing f. Such an approach was taken before, mostly in an ad-hoc manner, for specific functions f of interest. We propose a *systematic* study, towards a general characterization of the hierarchy according to the number of black-box calls. This approach leads to a better understanding of the inherent complexity for securely computing a given function f. Furthermore, minimizing the number of calls to the black box can lead to more efficient protocols when the calls to the black box are replaced by a secure protocol.

We take a first step in this study, by considering the two-party, honest-but-curious, information-theoretic case. For this setting, we provide a complete characterization for deterministic protocols. We explore the hierarchy for randomized protocols as well, giving upper and lower bounds, and comparing it to the deterministic hierarchy. We show that for every Boolean function the largest gap between randomized and deterministic protocols is at most exponential, and there are functions which exhibit such a gap.

*Dept. of Computer Science, Ben-Gurion University. Beer-Sheva 84105, Israel. E-mail: beimel@cs.bgu.ac.il.

†Dept. of Computer Science, Columbia University. New York, NY 10027-7003, USA. E-mail: tal@cs.columbia.edu. Part of this work was done while at AT&T Labs–Research

1 Introduction

The ability to compute functions securely is one of the most fundamental cryptographic tasks. Very roughly, two-party secure computation (on which we focus in this paper) involves two parties, Alice and Bob, who want to perform some computation on their inputs without leaking any additional information which does not follow from the intended output.

It is known (c.f. [4, 9, 10, 22]) that not all functions can be computed securely in the information-theoretic setting. However, Goldreich and Vainish [16] and Kilian [18] proved that every function can be computed securely in the information theoretic setting, given a black box that computes some *complete* function, such as Oblivious Transfer or the AND function. This type of a reduction is useful, because the security of the protocol is automatically maintained (computationally) when the black box is replaced by any computationally secure implementation of the function (such implementations exist under computational assumptions).[1] Moreover, such reductions provide a qualitative separation between "easy" functions that can be securely computed without calling the black box, and the "hard" functions which are the rest. Indeed, the notion of a reduction plays a central role at the heart of cryptographic foundations research (similarly to its central role in complexity theory). For example, black-box reductions between different cryptographic primitives were given in [6, 11, 12, 19, 7, 5, 13, 21].

A long line of research has focused on studying, in various settings, which functions belong to the "easy" category above, and which are "hard", as well as studying which functions are complete (which in some cases turned out to be the same as all hard functions). In particular, these questions have been answered (with full characterization) for Boolean functions [10], in the two-party model [22, 1, 3], and completeness results appear in [19, 21, 3, 20]. However, these works do not give rise to a hierarchy of different degrees of hardness, as they do not distinguish among the different functions that can be computed with a specific complete (say AND) black box.

Such a hierarchy exists (for the information-theoretic reduction setting), by a result of Beaver [2], showing that for all k, there are functions that can be securely computed with k executions of the AND black box but cannot be computed with $k-1$ executions of the black box. We explore the hierarchy in this work.

OUR GOALS. In this paper, we propose to take a *quantitative* approach, classifying functions by *how many* calls to the black box are required to compute them securely. Minimizing the number of calls to the black box is especially desired as it can lead to more efficient protocols when the calls to the black box are replaced by a secure protocol. This problem was previously investigated in an ad-hoc manner, for specific functions of interest (e.g., different forms of OT). In most cases, only upper bounds on the number of calls were given. Two exceptions are Beaver [2] who proved that securely computing n outputs of $\binom{2}{1}$OT with unrelated inputs requires at least n calls to $\binom{2}{1}$OT, and Dodis and Micali [14] who proved that securely computing $\binom{n}{1}$OT requires at least $n-1$ calls to $\binom{2}{1}$OT (see also [25]).

We propose a systematic study of the quantitative approach to reductions in secure computation, towards a deeper understanding of the inherent complexity of securely computing functions. In particular, focusing for the sake of presentation on the AND black box, we ask the following questions:

- Is there a well-defined rich hierarchy of functions based on how many ANDs are required to securely compute them?
- Given a function, can we give upper bounds on how many ANDs suffice to securely compute it? Can we give lower bounds?

[1] In this paper we consider the honest-but-curious model where modular composition is fairly straightforward. In the malicious model modular composition holds as well. See [8] for definitions and results on modular composition in the malicious model.

- Can we give a combinatorial *characterization* of the functions with a certain minimal number of ANDs?

These problems are interesting in several settings. For the first problem, Beaver [2] provided a negative answer (the hierarchy collapses) in the computational setting, and a positive answer in the information theoretic setting, for randomized protocols (and for randomized functions, as well). Recently, Ishai et al. [17] proved that the hierarchy collapses in the random oracle model as well.

We note that by results of [16], lower bounds on the number of ANDs imply circuit lower bounds, meaning that it would be very hard to prove super-linear lower bounds in n for functions of the form $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. However, it would be very interesting to prove such linear lower bounds and to try to explore tighter connections with circuit complexity and communication complexity[2] of the functions.

OUR RESULTS. We start the investigation by studying the information-theoretic, two-party, honest-but-curious setting, where the output of the AND black box is received only by Alice. Unless otherwise noted, we also consider protocols with perfect correctness and security. For this setting we prove the following results:

DETERMINISTIC PROTOCOLS. For deterministic protocols are we show:

- A complete combinatorial characterization of the minimal number of ANDs required to securely compute f (the characterization is a recursive one, based on the truth-table of f).

In particular, we establish that every function can be computed securely by a deterministic protocol with enough ANDs. This should be contrasted to the malicious model where it is known that randomization is required [14]. For finite functions one can find the optimal protocol using our characterization. However, in general, our characterization does not lead to an efficient algorithm that determines how many ANDs are required to compute a function securely. This motivates the following results:

- A simple, explicit upper bound on the number of ANDs required for f. This upper bound may be exponential in the size of the input.
- For *Boolean* functions f we prove that the above upper bound is tight by showing a matching lower bound. This implies that for some functions, an exponential number of ANDs is necessary.

RANDOMIZED PROTOCOLS. For randomized protocols are we show:

- An exponential gap using randomization: There are functions for which the number of ANDs required in a randomized protocol is exponentially smaller than the number of ANDs required in a deterministic protocol. We further exhibit a tradeoff between the number of random bits used and the number of ANDs required for one such example (Inner Product) where there is an exponential gap.
- A lower bound: We prove a lower bound, depending on the function truth-table, on the number of ANDs required by any secure randomized protocol. Using this lower bound, we prove that for Boolean functions the gap cannot be super exponential: For any randomized protocol with q ANDs, there is a deterministic protocol for the same function with at most 2^q ANDs.
- Gap already with 4 ANDS: There is a function that can be securely computed by a randomized protocol with 4 ANDs, however, every deterministic protocol securely computing it requires at least 6 ANDs.

[2]Naor and Nissim [26] give some connections between the communication complexity of a function and the communication complexity for securely computing the function. However, translating them into our model, the number of ANDs is exponential in the communication complexity.

- No gap with 1 AND: The functions that can be securely computed with one call to the AND black box are the same as in the deterministic case with one AND (for which an explicit characterization is given).
- Gap between perfect and non-perfect protocols: There are functions that require at least a linear (in the input length) number of ANDs for any perfect (randomized) protocol, but can be computed with k ANDs (for any k), achieving a protocol with $1/2^k$ probability of error and statistical distance.
- Lower bound for non-perfect protocols: We show that the one-way randomized communication complexity in the shared-randomness model is a lower bound for the number of ANDs required by non-perfect protocols.

EXTENSIONS TO OTHER MODELS AND COMPLETE FUNCTIONS. As explained earlier, we choose the simplest model of secure computation to consider our quantitative approach. Some of our results carry over directly to other models, and some questions still remain open in the other models. We hope that our paper would be a starting point for further research which will clarify the situation in more complex models as multi-party protocols, and the protocols that are secure against malicious parties.

Specifically, in this paper only Alice gets the output of the function while Bob should not learn any information on the input of Alice. This one-sided model is the correct model when considering *malicious* two-party secure computation where the first party to get the output can quit the protocol preventing the other party from getting the output. In the honest-but-curious model, the one-sidedness of the output is not the only possibility; we choose it since we want the simplest model. Some results on the two-sided model, where Alice gets an output f^{Alice} and Bob gets an output f^{Bob}, appear in Appendix B.

Furthermore, we state all our results counting the number of ANDs needed. However, every finite function (a function with a constant number of inputs) can be computed securely using a constant number of ANDs, and the AND function can be computed with one call to any complete function (this is implied by results of [3]). So, the results of this paper carry to every finite complete function, up to a constant factor. For example, the $\binom{2}{1}$OT function can be computed securely with two ANDs. Thus, all lower bounds on the number of ANDs translate into the same lower bounds on the number of $\binom{2}{1}$OT up-to a factor of 2.

CIRCUIT COMPLEXITY VS. NUMBER OF ANDS IN SECURE COMPUTATION. As explained above the circuit complexity of a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ provides an upper bound on the number of ANDs required for secure computation of f by a randomized protocol. It might seem tempting to think that the circuit complexity characterizes the number of ANDs. However, this is not true. There are functions with high circuit complexity which require few or no ANDs. For example, f can be a function only of Alice's input with high circuit complexity which Alice can compute securely without any communication or calls to the AND black box. Furthermore, our results show that circuit complexity does not characterizes the number of ANDs required to securely compute a function by a deterministic protocol (this number of ANDs can be larger or smaller than the circuit complexity).

Organization. In Section 2 we provide some necessary definitions and technical claims. In Section 3 we deal with deterministic secure protocols, and characterize which functions they can compute with q ANDs. In Section 4 we deal with randomized secure protocols, and examine how many ANDs they can save compared to deterministic protocols.

2 Preliminaries

In this section we define one-sided information-theoretic secure two-party computation in the honest-but-curios model. In our definition we allow the parties to execute a black box to a pre-defined function.

Protocols. We consider a two-party protocol with a pair of parties (Interactive Turing Machines), Alice and Bob. They have an access to a black box BB which computes some function BB : $D_1 \times D_2 \rightarrow D_3$. Briefly, on *inputs* (x, y), where x is a private input for Alice and y a private input for Bob, and *random inputs* (r_A, r_B), where r_A is a private random tape for Alice and r_B is a private random tape for Bob, protocol (Alice, Bob) computes its output in a sequence of rounds of three types: Alice's rounds, Bob's rounds, and black-box rounds. In an Alice's round (respectively, Bob's round) only Alice (respectively, only Bob) is active and sends a message (i.e., a string) that will become an available input to Bob (respectively, to Alice) in the next round. In a black-box round Alice puts a value $a \in D_1$ to a register and Bob puts a value $b \in D_2$ to a register. In the end of this round Alice gets the value BB(a, b) in a third register, and Bob gets no information. A computation of Alice and Bob ends in a round in which Alice computes a private *output*. In this paper we focus on an AND black box, where AND: $\{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$ and AND$(a, b) = a \wedge b$.

Transcripts, Views, and Outputs. Letting E be an execution of protocol (Alice, Bob) on inputs (x, y) and random inputs (r_A, r_B), we make the following definitions:

- The *transcript* of E consists of the sequence of messages exchanged by Alice and Bob, and is denoted by TRANS(x, r_A, y, r_B);
- The *black-box outputs* of E consists of the outputs of the black box during the execution of the protocol, and is denoted by BLACK-BOX(x, r_A, y, r_B);
- The *view of Alice* consists of the quadruplet

$$(x, r_A, \text{TRANS}(x, r_A, y, r_B), \text{BLACK-BOX}(x, r_A, y, r_B)),$$

  and is denoted by VIEW$_{\text{Alice}}(x, r_A, y, r_B)$;
- The *view of Bob* consists of $(y, r_B, \text{TRANS}(x, r_A, y, r_B))$, and is denoted by VIEW$_{\text{Bob}}(x, r_A, y, r_B)$.

We consider the random variables TRANS$(x, \cdot, y, r_B)$, TRANS$(x, r_A, y, \cdot)$, and TRANS$(x, \cdot, y, \cdot)$, respectively obtained by randomly selecting r_A, r_B, or both, and then outputting TRANS(x, r_A, y, r_B). We also consider the similarly defined random variables for VIEW$_{\text{Alice}}$ and VIEW$_{\text{Bob}}$.

In the model we consider, the two-party honest-but-curious model, each party is curious, that is, it may try to deduce as much information possible from its own view of an execution about the other's private input. However, each party is honest, that is, it scrupulously follows the instructions of the protocol. In such conditions, it is easy to enforce the correctness condition (for securely computing a function f), but not necessarily the privacy conditions. Note that, unlike secure computation in the malicious model, in the honest-but-curious model we can separate the security requirement into two separate requirements: correctness and privacy.

In the following definition we consider partial functions $f : A \times B \rightarrow C \cup \{*\}$, where A, B and C are some finite sets and $* \notin C$. If $f(x, y) = *$ then we say that f is undefined on x, y. The reason that we consider partial functions is that in Section 3 we use them to characterize the number of ANDs required to securely compute fully-defined functions. To define the privacy in a protocol we consider the *statistical distance* between two distributions Y_0, Y_1, denoted by DIST(Y_0, Y_1), which is defined by DIST$(Y_0, Y_1) = \frac{1}{2}\sum_y |\Pr[Y_0 = y] - \Pr[Y_1 = y]|$.[3]

Definition 2.1 (Secure Computation) *Let $f : A \times B \rightarrow C \cup \{*\}$ be a function, and $0 \leq \epsilon, \delta \leq 1$. A protocol* (Alice, Bob) (ϵ, δ)-securely computes f*, if the following conditions hold:*

[3]Equivalently, the statistical distance between Y_0 and Y_1 may be defined as the maximum, over all functions A, of the *distinguishing advantage* $|\Pr[A(Y_0) = 1] - \Pr[A(Y_1) = 1]|$.

Correctness. *For every $x \in A$ and every $y \in B$, if $f(x,y) \neq *$, then the probability that the output of Alice with* $\text{VIEW}_{\text{Alice}}(x,\cdot,y,\cdot)$ *is $f(x,y)$ is at least $1-\epsilon$, where the probability is taken over r_A and r_B.*

Bob's Privacy. *$\forall x \in A$, $\forall y_0, y_1 \in B$, $\forall r_A$, if $f(x,y_0) = f(x,y_1) \neq *$ then*

$$\text{DIST}(\text{VIEW}_{\text{Alice}}(x, r_A, y_0, \cdot), \text{VIEW}_{\text{Alice}}(x, r_A, y_1, \cdot)) \leq \delta.$$

Alice's Privacy. *$\forall x_0, x_1 \in A$, $\forall y \in B$, $\forall r_B$, if $f(x_0,y) \neq *$ and $f(x_1,y) \neq *$, then*

$$\text{DIST}(\text{VIEW}_{\text{Bob}}(x_0, \cdot, y, r_B), \text{VIEW}_{\text{Bob}}(x_1, \cdot, y, r_B)) \leq \delta.$$

A protocol securely computes *f if it $(0,0)$-securely computes f. In this case, we also say that the protocol computes f with perfect security. A protocol is* deterministic *if Alice's and Bob's moves in the protocol do not depend on their random inputs.*

Notice that the requirements in Alice's privacy and in Bob's privacy are not symmetric. We require that Alice's privacy is protected for all inputs where f is defined. As Alice learns the output of f, we require that Bob's privacy is protected only when $f(x,y_0) = f(x,y_1) \neq *$.

The main measure we consider is the number of calls to the black box during a protocol.

Definition 2.2 (Number of ANDs) *The number of calls to the AND black box in a protocol is the maximum over the inputs x and y and random inputs r_A and r_B of the number of black-box rounds in the execution with x, y, r_A, and r_B.*

Beimel, Micali, and Malkin [3], following Kushilevitz [22], characterize which functions can be computed securely without any calls to the AND black box. Their characterization uses the following notation and definitions. We represent a function $f : A \times B \to C \cup \{*\}$ by a matrix M_f whose rows are labeled by the elements of A, columns are labeled by the elements of B, and $M_f(x,y) = f(x,y)$.

Definition 2.3 (Insecure Minor) *A matrix contains an insecure minor if there are x_0, x_1, y_0, y_1 such that $M(x_0,y_0) = M(x_0,y_1) \neq *$, $M(x_1,y_0), M(x_0,y_1) \neq *$, and $M(x_1,y_0) \neq M(x_0,y_1)$. That is, there are $a,b,c \in C$ such that $b \neq c$ and the matrix can be described as follows:*

M	$\mathbf{y_0}$	$\mathbf{y_1}$
$\mathbf{x_0}$	a	a
$\mathbf{x_1}$	b	c

The following theorem of [3] states that a function can be computed securely without ANDs iff it does not contain an insecure minor.

Theorem 2.4 ([3]) *The function f can be computed by a perfectly-secure randomized protocol with 0 ANDs if and only if the function f can be computed by a deterministic protocol with 0 ANDs if and only if M_f does not contain an insecure-minor.*

The next definition is helpful for characterizing the number of required ANDs, by defining a relation on the columns of the matrix M_f.

Definition 2.5 ([22]) *The relation $\sim_c$ on the columns of a matrix M is defined as follows: $y, y' \in B$ satisfy $y \sim_c y'$ if there exists some $x \in A$ such that $M(x,y) = M(x,y') \neq *$. The equivalence relation $\equiv_c$ on the columns of M is defined as the transitive closure of the relation $\sim_c$. That is, $y \equiv_c y'$, for $y, y' \in B$, if there are $y_1, \ldots, y_\ell$ such that $y \sim_c y_1 \sim_c y_2 \sim_c \cdots \sim_c y_\ell \sim_c y'$.*

In the rest of this section we prove various properties of secure protocols used throughout the paper. We next relate the number of ANDs required to securely compute a function, to the number of ANDs required to securely compute the functions restricted to each equivalence class.

Claim 2.6 *Let $f : A \times B \to C \cup \{*\}$ be a function, let $B_1, \ldots, B_k$ the equivalence classes of the relation $\equiv_c$, and define $f_i : A \times B_i \to C \cup \{*\}$ as the restriction of f to B_i, that is $f_i(x, y) = f(x, y)$ for every $x \in A$ and every $y \in B_i$. The function f can be computed securely by a randomized protocol (respectively, deterministic protocol) with q ANDs if and only if each function f_i can be computed securely by a randomized protocol (respectively, deterministic protocol) with q ANDs.*

Proof: Clearly, if f can be computed securely with q ANDs then any restriction of f can be computed securely with q ANDs. For the other direction consider the following protocol: Bob sends to Alice the index i such that $y \in B_i$ and Alice and Bob execute the secure protocol for f_i. The correctness is immediate. As for the security, notice that Alice can deduce the equivalence class of y from x and $f(x, y)$, thus, she does not gain extra information from the message that Bob sends her, and does not gain any extra information from the secure protocol for f_i. □

For the results in this paper, we need the following standard result; for completeness we include its proof. Informally, the claim asserts that if the columns of M_f are equivalent then in perfectly-secure protocols no information is disclosed by the communication, and all the information that Alice needs to compute the function is passed through the outputs of the black box alone.

Claim 2.7 *Let $f : A \times B \to C$ be a function such that all columns of M_f are equivalent according to $\equiv_c$ and let c be any communication transcript that can be exchanged between Alice and Bob in a protocol with perfect privacy. Then for every $x, x' \in A$ and every $y, y' \in B$*

$$\Pr[c = \text{TRANS}(x, \cdot, y, \cdot)] \quad = \quad \Pr[c = \text{TRANS}(x', \cdot, y', \cdot)],$$

where the probability is taken over the random inputs of Alice and Bob.

Proof: First we claim that for every $x, x' \in A$, every $y \in B$, and every random input r_B of Bob

$$\Pr[c = \text{TRANS}(x, \cdot, y, r_B)] \quad = \quad \Pr[c = \text{TRANS}(x', \cdot, y, r_B)], \tag{1}$$

where the probability is taken over the random input of Alice. This is implied by Alice's privacy; otherwise Bob holding the input y and random input r_B can learn information weather Alice's input is x or x'. In particular, for every $x, x' \in A$ and any $y \in B$

$$\Pr[c = \text{TRANS}(x, \cdot, y, \cdot)] \quad = \quad \Pr[c = \text{TRANS}(x', \cdot, y, \cdot)], \tag{2}$$

where this time the probability is taken over the random inputs of Alice and Bob.

Next, we claim that for every $x \in A$ and every $y, y' \in B$

$$\Pr[c = \text{TRANS}(x, \cdot, y, \cdot)] \quad = \quad \Pr[c = \text{TRANS}(x, \cdot, y', \cdot)]. \tag{3}$$

In this case we need to be somewhat more careful. Assume towards contradiction that Eq. (3) does not hold for some $x \in A$ and $y, y' \in B$. Recall that all the columns of M_f are equivalent, thus, there is a sequence $y_1, y_2, \ldots, y_m$, such that $y_{i-1} \sim_c y_i$ for $i = 1, \ldots, m + 1$ where $y_0 = y$ and $y_{m+1} = y'$. Let y_i be the first in the sequence such that

$$\Pr[c = \text{TRANS}(x, \cdot, y_{i-1}, \cdot)] \quad \neq \quad \Pr[c = \text{TRANS}(x, \cdot, y_i, \cdot)].$$

Since $y_{i-1} \sim_c y_i$ there is some x_i such that $f(x_i, y_{i-1}) = f(x_i, y_i)$. By Eq. (2),

$$\Pr[c = \text{TRANS}(x, \cdot, y_{i-1}, \cdot)] \quad = \quad \Pr[c = \text{TRANS}(x_i, \cdot, y_{i-1}, \cdot)]$$

and

$$\Pr[c = \text{TRANS}(x, \cdot, y_i, \cdot)] \quad = \quad \Pr[c = \text{TRANS}(x_i, \cdot, y_i, \cdot)].$$

Thus,

$$\Pr[c = \text{TRANS}(x_i, \cdot, y_{i-1}, \cdot)] \neq \Pr[c = \text{TRANS}(x_i, \cdot, y_i, \cdot)]$$

although $f(x_i, y_{i-1}) = f(x_i, y_i)$, contradiction. The claim now follows Eq. (2) and Eq. (3). □

Recall that in any deterministic protocol, for every x, y there is one possible communication transcript. Thus, by Claim 2.7, if all the columns of M_f are equivalent, then the same transcript will be exchanged for every pair of inputs. Thus, in deterministic protocols Alice and Bob can discard the communication and only execute the AND black boxes.

Claim 2.8 *Let $f : A \times B \to C$ be a function such that all columns of M_f are equivalent according to $\equiv_c$. Then, in every deterministic secure protocol there is exactly one communication transcript that is exchanged between Alice and Bob for all inputs x, y.*

3 Deterministic Protocols

In this section we examine how many ANDs are needed to compute a function securely by a deterministic protocol. We start by giving an exact characterization of the functions that can be securely computed by deterministic protocols with q ANDs. This characterization proves that there is a complete hierarchy of functions according to the number of ANDs. In particular, we establish that every function can be computed securely by a deterministic protocol provided that enough ANDs are executed. This should be contrasted to the malicious model where it is known that randomization is required [14].

For finite functions one can find the optimal protocol using our characterization. However, in general, our characterization does not lead to an efficient algorithm that determines how many ANDs are required to compute a function securely. Therefore, in Theorem 3.2 we give a simple and explicit upper bound on the number of ANDs that are required. Finally, we show in Theorem 3.3 that this upper bound is tight for Boolean functions. We note that our upper bound seems to be impractical since the number of ANDs can be exponential in the length of the input. However, at least for Boolean functions, our lower bound proves that this is unavoidable if we consider deterministic protocols.

To characterize what can be done with q ANDs by a deterministic protocol, we note that first Alice and Bob call the AND black box once, and then execute a protocol with $q-1$ ANDs to compute a related function described in Figure 1. For the first execution there are sets $A_1 \subseteq A$ and $B_1 \subseteq B$ such that Alice gets output one from the AND black box if and only if $x, y \in A_1 \times B_1$. We have two requirements: (1) Alice does not learn any extra information from the output of the first AND black box, and (2) Alice and Bob can compute the following function f_{A_1,B_1} using $q-1$ ANDs. Formally, given a function $f : A \times B \to C \cup \{*\}$ and two sets $A_1 \subseteq A$ and $B_1 \subseteq B$ we define a function $f_{A_1,B_1} : (A \cup (A_1 \times \{1\})) \times B \to C \cup \{*\}$, described in Figure 1, as follows:

1. $f_{A_1,B_1}(x, y) = f(x, y)$ for every $x \in A \setminus A_1$ and every $y \in B$.
2. $f_{A_1,B_1}(x, y) = f(x, y)$ for every $x \in A_1$ and every $y \in B \setminus B_1$.

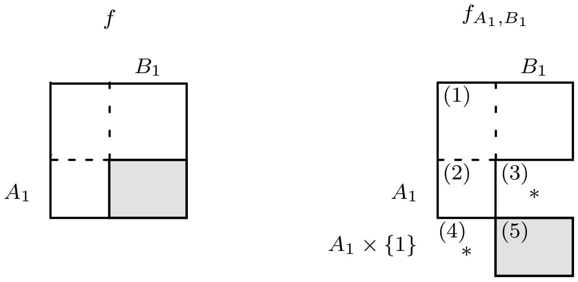

Figure 1: The matrices of the functions f and f_{A_1,B_1}. The numbers in the description of f_{A_1,B_1} refer to the different cases in its definition.

3. $f_{A_1,B_1}(x, y) = *$ for every $x \in A_1$ and every $y \in B_1$.
4. $f_{A_1,B_1}(\langle x, 1\rangle, y) = *$ for every $x \in A_1$ and every $y \in B \setminus B_1$.
5. $f_{A_1,B_1}(\langle x, 1\rangle, y) = f(x, y)$ for every $x \in A_1$ and every $y \in B_1$.

Theorem 3.1 *Let $f : A \times B \to C \cup \{*\}$ be a function such that all columns of M_f are equivalent according to $\equiv_c$. The function f can be computed securely with q calls to the AND black box if and only if there are sets $A_1 \subseteq A$ and $B_1 \subseteq B$ such that the following two requirements hold:*

1. *For every $x \in A_1$, every $y_0 \notin B_1$, and every $y_1 \in B_1$ such that $f(x, y_0), f(x, y_1) \neq *$ it holds that $f(x, y_0) \neq f(x, y_1)$, and*
2. *The function f_{A_1,B_1} can be computed securely with $q - 1$ calls to the AND black box.*

Proof: We first prove that the above conditions are sufficient. Assume the conditions hold. The secure protocol for computing f proceeds as follows:

- Alice and Bob call the AND black box where Alice puts 1 iff $x \in A_1$ and Bob put 1 iff $y \in B_1$.
- Alice and Bob execute the secure protocol for f_{A_1,B_1} with $q - 1$ calls to the AND black box, where Bob's input is y and Alice's input is $\langle x, 1\rangle$ if the AND output is 1 and x otherwise.
- Alice's output is the output of the protocol for f_{A_1,B_1}.

We first argue that the protocol is correct. On one hand, if the output of the AND black box is 1, then $x \in A_1$ and $y \in B_1$. Thus, by the definition of f_{A_1,B_1} it holds that $f_{A_1,B_1}(\langle x, 1\rangle, y) = f(x, y)$, and the output of the protocol is correct. On the other hand, if the output of the AND black box is 0, then either $x \notin A_1$ or $y \notin B_1$. Thus, $f_{A_1,B_1}(x, y) = f(x, y)$, and the output of the protocol is correct. Note that the protocol never tries to evaluate f_{A_1,B_1} on inputs where it is not defined.

To argue that the protocol is perfectly-secure, first note that Bob gets no messages during the first step of the protocol, and he does not get any information from the black box. This guarantees Alice's Privacy. To argue about Bob's privacy, note that Alice learns information about y from the first call to the AND black box only if $x \in A_1$. In this case, by Condition 1, Alice learns if $y \in B_1$ from the output of the function f itself. Thus, this step is secure, and Alice is allowed to know the output of the black box and the output of f_{A_1,B_1} which as argued is equal to the desired output of f. Finally, as the protocol for f_{A_1,B_1} is secure, the entire protocol for f is secure.

We next prove that the conditions of the theorem are necessary. Assume that f can be computed securely with q ANDs. By Claim 2.8, we can assume without loss of generality that Alice and Bob do not exchange any messages, and all information Alice gets is through the outputs of the calls to the AND black boxes. Let A_1 and B_1 be the sets of inputs of Alice and Bob respectively for which they put 1 to the first call to the AND black box. Condition 1 must hold or otherwise Alice learns extra information from the answer of the first AND. As for Condition 2, we can use the following protocol to compute f_{A_1,B_1} securely with $q-1$ ANDs: Alice and Bob execute the protocol for f with the following two changes: (1) If Alice's "real" input is $\langle x, 1\rangle$ for $x \in A_1$ then she replaces it by the input x, and (2) the first call to the AND black box is not executed. Instead, Alice simulates it by considering its output as 1 if her input is $\langle x, 1\rangle$ and 0 otherwise. The rest of the protocol is executed without any changes. As the protocol for f uses q ANDs, and Alice and Bob do not use the first AND, the resulting protocol for f_{A_1,B_1} uses $q-1$ ANDs as required. □

Our next theorem gives a simple upper bound on the number of ANDs required to compute a function securely. The proof of this upper-bound gives a simple secure protocol for computing the function.

Theorem 3.2 *Let $f : A \times B \rightarrow C \cup \{*\}$ be a function. The function f can be computed securely by a deterministic protocol with $|A| \lceil \log |C| \rceil$ ANDs.*

Proof: First assume that f is Boolean, i.e., $C = \{0, 1\}$. We next describe a protocol which uses $|A|$ ANDs. Assume the input of Alice is x and the input of Bob is y. For every $z \in A$, Alice and Bob execute the AND black box, where Alice puts 1 to the AND if $x = z$ and 0 otherwise, and Bob puts $f(z, y)$ to the AND. Alice outputs the output of the AND corresponding to x, that is, $\text{AND}(1, f(x, y)) = f(x, y)$ as required. Bob does not gain any information during this protocol (since there is no communication and only Alice gets the output of the black box) and Alice only gains $f(x, y)$.

If $|C| > 2$, then we consider the binary representation of $f(x, y)$ (of length exactly $\lceil \log |C| \rceil$), and execute the above protocol for every bit of $f(x, y)$. □

The following theorem shows that the upper bound of Theorem 3.2 is tight for every Boolean function. In the theorem we assume that there is some y_0 such that $f(x, y_0) = 0$ for every $x \in A$. This assumption is without loss of generality since Alice learns the output of the protocol and knows x, thus she can use any renaming of the outputs in every row.[4]

Theorem 3.3 *Let $f : A \times B \rightarrow \{0, 1\}$ be a* Boolean *function such that all the rows of M_f are distinct and non-constant, there is some $y_0 \in B$ such that $f(x, y_0) = 0$ for every $x \in A$, and all of its columns are equivalent according to $\equiv_c$. Then, every deterministic protocol computing f securely must use at least $|A|$ ANDs.*

Proof: Fix any deterministic protocol that computes f securely. By Claim 2.8, we can assume, without loss of generality, that Alice and Bob do not exchange any messages and the view of Alice includes her input and the outputs of the black box. Consider any $x \in A$. Since f is Boolean there are exactly two views Alice should see given x: one view for every y such that $f(x, y) = 0$ and another view for every y such that $f(x, y) = 1$. For every x, consider the first black-box call where Alice can get two different answers. As argued above one output corresponds to the case where $f(x, y) = 0$ and the other output corresponds to the case where $f(x, y) = 1$. Thus, Alice can deduce the output of the function $f(x, y)$ from this black-box answer and, therefore, we say that this is the significant call to the AND black box for x.

[4]Given a Boolean function f, we fix any $y_0 \in B$ and define f' as follows: For every $x \in A$ and $y \in B$ if $f(x, y_0) = 0$ then $f'(x, y) = f(x, y)$ and if $f(x, y_0) = 1$ then $f'(x, y) = 1 - f(x, y)$. Clearly, $f'(x, y_0) = 0$ for every $x \in A$. We claim that the number of ANDs required to compute f and f' is the same: Holding x, Alice can locally compute $f(x, y_0)$, thus, she can compute the output of $f(x, y)$ from $f'(x, y)$ and vise-versa.

$\mathbf{f}$	$\mathbf{0}$	$\mathbf{1}$	$\mathbf{2}$	$\mathbf{3}$
$\mathbf{0}$	0	1	0	1
$\mathbf{1}$	0	1	2	2
$\mathbf{2}$	0	0	2	3

The function f

$\mathbf{f_{A_1,B_1}}$	$\mathbf{0}$	$\mathbf{1}$	$\mathbf{2}$	$\mathbf{3}$
$\mathbf{0}$	0	1	0	1
$\mathbf{1}$	0	1	$*$	$*$
$\mathbf{2}$	0	0	$*$	$*$
$\langle \mathbf{1}, \mathbf{1} \rangle$	$*$	$*$	2	2
$\langle \mathbf{2}, \mathbf{1} \rangle$	$*$	$*$	2	3

The function f_{A_1,B_1}

Figure 2: The functions f and f_{A_1,B_1}.

Assume, towards contradiction, that for two different $x_0, x_1 \in A$ the significant call is the same. Recall that $f(x_0, y_0) = f(x_1, y_0) = 0$, and since the rows corresponding to x_0 and x_1 are not the same, there is some y_1 such that, without loss of generality, $f(x_0, y_1) = 0$ while $f(x_1, y_1) = 1$. Bob has to put the same value to this significant call when he holds y_0 and y_1 or Alice would learn information when she holds x_0. This means that Alice cannot compute the correct value of $f(x_1, y_0)$ or $f(x_1, y_1)$ since in both cases she gets the same information, contradiction.

To conclude, for every $x \in A$ there is a unique significant call to the AND black box, thus, there are at least $|A|$ calls to the AND black box. $\square$

In the protocol implied by Theorem 3.2, Alice is non-adaptive as her inputs to the AND black box depend only on her input and not on the outputs of previous AND black boxes. In Theorem 3.3 we prove that for Boolean functions this is optimal. However, the protocol implied by Theorem 3.1 is adaptive, and for non-Boolean functions adaptively does help as shown in the following example. Consider the function $f : \{0, 1, 2\} \times \{0, 1, 2, 3\} \to \{0, 1, 2\}$ described in Figure 2. We next describe a secure protocol for f which uses two ANDs. For the first AND, Alice puts 1 if $x \in A_1 = \{1, 2\}$ and Bob puts 1 if $y \in B_1 = \{2, 3\}$. After this AND Alice and Bob need to securely compute the function f_{A_1,B_1} described in Figure 2. Computing f_{A_1,B_1} is done using a second AND where Alice puts 1 if $x \in A_2 = \{0, 1, \langle 2, 1 \rangle\}$ and Bob puts 1 if $y \in B_2 = \{1, 3\}$. After this AND, Alice can deduce the output of f from her input and the outputs of the ANDs. In this protocol Alice is adaptive; with input 1, for example, she puts 1 to the second AND if the output of the first AND was 0 and she puts 0 otherwise. This example shows that the lower bound of Theorem 3.2 is not tight for general (non-Boolean) functions.

4 Randomized Protocols

In this section we investigate the power of randomization in our setting. We show that, in general, randomization helps: the gap between the number of ANDs required by a randomized protocol and a deterministic one may be exponential. We also quantify *how much* randomization can help, and study its limits. Finally, we show that allowing a statistically secure protocol with some error probability may significantly reduce the number of ANDs compared to the number required by a perfect randomized protocol.

4.1 Randomization Helps

The following theorem, adapted from [16], establishes an upper bound on the number of ANDs needed to securely compute a function, in terms of the number of gates in its circuit. Together with our characterization for deterministic protocols in the previous section, the theorem proves that randomization helps, as we elaborate below.

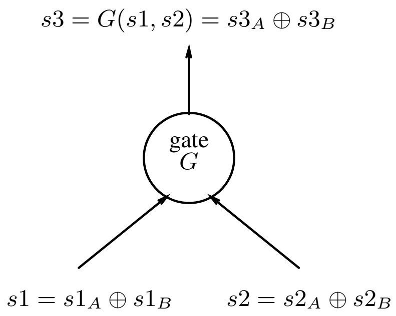

Figure 3: A secure evaluation of a gate G.

$(a1, a2)$	$s3_A$
$(0, 0)$	$s3_B \oplus (s1_B \wedge s2_B)$
$(0, 1)$	$s3_B \oplus (s1_B \wedge \overline{s2_B})$
$(1, 0)$	$s3_B \oplus (\overline{s1_B} \wedge s2_B)$
$(1, 1)$	$s3_B \oplus (\overline{s1_B} \wedge \overline{s2_B})$

Figure 4: The values of $s3_A$ for the 4 possible values of Alice's inputs for an $\wedge$ gate.

Theorem 4.1 ([16]) *If f can be computed by a Boolean circuit with fan-in 2 whose size is s, then there is a perfectly-secure randomized protocol computing f which uses $4s$ AND calls.*

Proof: Theorem 4.1 is proved in [16] by having each of the parties additively secret-share their inputs, and then processing the shares through each of the gates in the circuit. Depending on the gate, the parties may need to use the primitive of *1-out-of-4 Oblivious Transfer*, which can be implemented using four ANDs.

We next describe the protocol in our context. Alice and Bob compute the function f one gate at a time, such that for each wire in the circuit, Alice and Bob hold two random bits whose exclusive-or is the correct value for that wire in a non-secure computation of the circuit (see Figure 3). For initialization, for every variable x_i held by Alice, the bits held by Alice and Bob respectively are (s_A, s_B) where Alice holds the bit $s_A = x_i$ and Bob holds the bit $s_B = 0$. The variables held by Bob are dealt symmetrically. We next explain how to compute a Boolean gate G where the correct values of its inputs computed by the circuit are $s1$ and $s2$ and the correct value of the output of the gate is $s3 = G(s1, s2)$. Before the computation of the gate Alice holds $(s1_A, s2_A)$ and Bob holds $(s1_B, s2_B)$ such that $s1 = s1_A \oplus s1_B$ and $s2 = s2_A \oplus s2_B$. At the end of the computation Alice and Bob should hold random bits $(s3_A, s3_B)$ such that $s3 = s3_A \oplus s3_B$. To compute the gate, Bob chooses a random bit $s3_B$, and computes the value of $s3_A$ for the 4 possible values $(0, 0)$, $(0, 1)$, $(1, 0)$, and $(1, 1)$ of Alice's inputs $(s1_A, s2_A)$. That is, Bob computes for every $a1, a2 \in \{0, 1\}$ the value $s3_A = s3_B \oplus G(a1 \oplus s1_B, a2 \oplus s2_B)$. In Figure 4 we demonstrate the values of $s3_A$ for an AND gate. Thereafter, Alice and Bob perform four ANDs, corresponding to the possible values of Alice's inputs, where Alice puts 1 to the AND execution corresponding to her true inputs $(s1_A, s2_A)$, and 0 to the other three, and Bob puts the values of $s3_A$ he computed. For the final gate application, Bob chooses $s3_B = 0$, so that Alice's output for that gate (in the appropriate AND execution) is the output of the function.

The correctness of the protocol is immediate. Alice's privacy follows from the fact that Bob gets no messages and does not get the outputs of the black-box calls, thus learns no information. To argue that Bob's privacy holds, first observe that from the simulation of a gate G Alice only learns the value SG_A as specified in the protocol since in the other three AND calls Alice puts 0. We claim that these values SG_A do not convey information about Bob's input: Fix any x, y_0 and y_1 such that $f(x, y_0) = f(x, y_1)$. For $i = 0, 1$, given x and y_i let SG^i be the output of gate G. Given Alice's view and the values SG^i, there is exactly one value for SG^i_B – the random choices of Bob. That is, for both y_0 and y_1 each view of Alice has the same probability (namely, the probability of each view is 2^{s-1} where s is the number of gates in the Boolean circuit). □

The above theorem applies for circuits with gates which are arbitrary Boolean functions with fan-in 2. Depending on the circuit, the theorem can be optimized to achieve a smaller number of ANDs, as some of the gates may require only 2 ANDs (when one of the incoming wires is from Bob's initial inputs) or no ANDs (when the gate computes exclusive-or). Such optimizations are used in the following examples to obtain slightly better parameters than guaranteed by a direct application of the theorem as stated.

We conclude that randomization helps for functions where the upper bound promised by Theorem 4.1 for randomized protocols is smaller than the lower bound established in Theorem 3.1 for deterministic protocols. We next provide a few concrete examples, which exhibit when and how much randomization helps. Specifically, we show:

- A function (Inner Product) with an exponential gap between the number of ANDs required by a randomized protocol and any deterministic one.
- For the same function, a trade-off between the number of random bits used and the number of ANDs required (spanning the exponential gap mentioned above).
- A gap between deterministic and randomized protocols using four ANDs (this is the smallest number for which we show a gap). This is achieved by showing that a variation of the Inner Product function can be securely computed with four ANDs in a randomized protocol, but requires six in any deterministic one.
- Another function (Equality) exhibiting an exponential gap between randomized and deterministic protocols, which in addition exhibits a gap between randomized (perfect) protocols, and randomized protocols with small error and statistical security. For the latter, the number of ANDs depends only on the error and distance parameter (instead of linear or exponential in the input length, as required for perfect randomized and perfect deterministic protocols, respectively).
- On the negative side, we show in Section 4.2 that the gap between the number of ANDs in a perfect randomized protocol and a deterministic protocol can never be super-exponential. In Section 4.3 we show that in any protocol with a single AND, randomization does not help (similarly to protocols with no ANDs).

Example 4.2 (Inner Product $\mathrm{IP_n}$) Let $\mathrm{IP}_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be the inner-product modulo 2 function, that is, $\mathrm{IP}_n(x, y) = \oplus_{i=1}^n x_i y_i$. We show that the function IP_n can be computed with $2n$ ANDs using a perfect randomized protocol, but requires at least $2^n - 1$ ANDs in any deterministic protocol.

Claim 4.3 *The function IP_n can be securely computed with $2n$ ANDs using a randomized protocol. Any deterministic protocol for securely computing IP_n requires at least $2^n - 1$ ANDs (and there is a deterministic protocol using this number of ANDs).*

Proof: Consider the following protocol on input $x = x_1, \ldots, x_n$ for Alice and $y = y_1, \ldots, y_n$ for Bob. Bob chooses $r_1, \ldots, r_{n-1} \in \{0,1\}^n$ uniformly at random and sets $r_n \leftarrow \oplus_{i=1}^{n-1} r_i$. Then, for each

$i = 1, \ldots, n$, Alice and Bob run two ANDs, as follows: $a_i^0 \leftarrow \wedge(1 - x_i, r_i)$ and $a_i^1 \leftarrow \wedge(x_i, y_i \oplus r_i)$. Alice outputs $\oplus_{i=1}^n a_i^{x_i} = \text{IP}_n(x, y)$.

The correctness of the protocol follows from noting that $a_i^{x_i} = x_i y_i \oplus r_i$ for $i = 1, \ldots, n$. Thus,

$$\oplus_{i=1}^n a_i^{x_i} \;=\; \oplus_{i=1}^n (x_i y_i \oplus r_i) \;=\; (\oplus_{i=1}^n x_i y_i) \oplus (\oplus_{i=1}^n r_i) \;=\; \text{IP}(x, y) \oplus 0.$$

The privacy follows by observing that $a_1^{x_1}, \ldots, a_n^{x_n}$ is a uniformly distributed vector subject to the requirement $\oplus_{i=1}^n a_i^{x_i} = \text{IP}_n(x, y)$ and that $a_i^{1-x_i} = 0$ for all i. □

We note that, as we will explain in Example 4.17, the number of ANDs in this protocol is tight up to a constant, since every randomized protocol for IP_n requires at least $n/2$ ANDs even if we allow errors and statistical privacy. We next show a tradeoff between the number of random bits and the number of ANDs.

Claim 4.4 *The function* IP_n *can be securely computed using* $R - 1$ *random bits and* $R2^{\lceil n/R \rceil}$ *ANDs, for all* $1 \leq R \leq n$.

Proof: The protocol is a generalization of the protocol described in the proof of Claim 4.3. Denote $n' = \lceil n/R \rceil$. Bob chooses $R - 1$ random bits $r_1, \ldots, r_{R-1}$ and sets $r_R \leftarrow \oplus_{i=1}^{R-1} r_i$. Then, for $i = 0$ to $R - 1$ Alice and Bob compute the function $a_i \leftarrow \text{IP}(\langle x_{in'+1}, \ldots, x_{(i+1)n'} \rangle, \langle y_{in'+1}, \ldots, y_{(i+1)n'} \rangle) \oplus r_i$ using the secure deterministic protocol of Theorem 3.2, which uses $2^{\lceil n/R \rceil}$ ANDs, where Alice's input is $\langle x_{in'+1}, \ldots, x_{(i+1)n'} \rangle$ and Bob's input is $\langle y_{in'+1}, \ldots, y_{(i+1)n'} \rangle, r_i$. Alice outputs the value $\oplus_{i=0}^{R-1} a_i$ which by the properties of IP and the choice of the r_i's is the correct value. Since the first $R - 1$ random bits are chosen independently and the deterministic IP protocol is secure, the protocol we construct is secure. □

The next claim follows from Theorem 3.3 and Theorem 3.2.

Claim 4.5 *Any deterministic protocol for securely computing* IP_n *requires at least* $2^n - 1$ *ANDs (and there is a deterministic protocol using this number of ANDs).*

Example 4.6 (Restricted IP_3) Consider the restricted-domain inner product function IP_3, where Alice's input cannot be $x = (1, 1, 1)$, and denote it by $\text{IP}_3{\downarrow}_{\overline{(1,1,1)}}$. We show below that this function can be computed with 4 ANDs in a randomized protocol with perfect privacy and correctness, but requires 6 ANDs in any deterministic protocol. We note that 4 is the smallest number of ANDs for which we can prove that randomization helps (in Section 4.3 we will see that for one AND we can prove randomization does not help). We leave as an open problem whether randomization helps or not for the case of 2 or 3 ANDs.

Claim 4.7 *The function* $\text{IP}_3{\downarrow}_{\overline{(1,1,1)}}$ *can be securely computed with 4 ANDs in a randomized protocol, but the minimal number of ANDs required by a deterministic protocol for this function is 6.*

Proof: The lower bound of 6 ANDs in any deterministic protocol follows from Theorem 3.3 (with the matching upper bound from Theorem 3.2) similarly to the proof of the previous claim , since the matrix for this function has 6 rows excluding the all zeros row. For the upper bound, we describe in Figure 5 a randomized protocol for $\text{IP}_3{\downarrow}_{\overline{(1,1,1)}}$ with 4 ANDs. correctness and privacy of this protocol can be verified using the following observations. If all of Alice's input bits are zeros, the ANDs give Alice no information, and her output is 0. If exactly one of Alice's input bits x_i has value 1 then Alice knows $a_i = y_i \oplus r$ and $a_4 = r$ and computes y_i as the output of the function (no other information is leaked). If exactly two of Alice's input bits x_i and x_j have value 1 then Alice knows $a_i = y_i \oplus r$ and $a_j = y_j \oplus r$ and computes $y_i \oplus y_j$ as the output of the function (no other information is leaked since $a_4 = a = 0$). □

A randomized protocol for $\mathrm{IP}_3\!\downarrow_{\overline{(1,1,1)}}$ with 4 ANDs
Alice's input: x_1, x_2, x_3 where the number of variables with value 1 is ≤ 2. Bob's input: y_1, y_2, y_3 Alice's desired output: $x_1y_1 \oplus x_2y_2 \oplus x_3y_3$ Bob chooses r at random from $\{0,1\}$. Alice sets $a = 1$ iff exactly one of her inputs has value 1. Alice and Bob execute the following 4 ANDs: $a_1 \leftarrow \wedge(x_1, y_1 \oplus r),\ a_2 \leftarrow \wedge(x_2, y_2 \oplus r),$ $a_3 \leftarrow \wedge(x_3, y_3 \oplus r),\ a_4 \leftarrow \wedge(a, r).$ Alice's ouput: $a_1 \oplus a_2 \oplus a_3 \oplus a_4$.

Figure 5: A randomized protocol with 4 ANDs for a function requiring 6 ANDs in any deterministic protocol.

Example 4.8 (Equality EQ_n) Let $\mathrm{EQ}_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be the equality function, that is, $\mathrm{EQ}_n(x,y) = 1$ iff $x = y$. We show below that the number of ANDs required to compute the function EQ_n using a perfect deterministic protocol is exponential in n, while using a perfect randomized protocol this number is linear in n, and using a randomized protocol with small error probability and statistical privacy, the number of ANDs is independent of n and depends only on the allowed error and distance (which are exponentially small in the number of ANDs). The specific claims are stated below.

Claim 4.9 *Any deterministic protocol computing EQ_n must use at least 2^n ANDs, and there is a deterministic protocl with this number of ANDs. Any perfectly-secure randomized protocol computing EQ_n must use at least n ANDs, and there exits such a protocol using $O(n)$ ANDs.*

Proof: Noting that the matrix for EQ_n is the identity matrix, the upper bound for deterministic protocols follows directly from Theorem 3.2. The lower bound for deterministic protocols follows from Theorem 3.3, since the matrix satisfies all the conditions of the theorem (including the all-zero column, if we exchange the roles of 0 and 1 outputs in one of the rows). The upper bound for randomized protocols follows from Theorem 4.1, by noting that there is a Boolean circuit with fan-in 2 and with $O(n)$ gates that computes EQ_n.[5] The lower bound for randomized protocols follows from Theorem 4.11 below. □

Claim 4.10 *For every k, the function EQ_n can be computed with $O(k)$ ANDs by a randimized protocol with $1/2^k$ error and at most $1/2^k$ statistical distance.*

Proof: The protocl securely-computing EQ_n is described in Figure 6. The idea of the protocol is to approximately compare the initial n-bit inputs by (exactly) comparing k inner products of the inputs with random strings, which as we saw can be done using $O(k)$ ANDs. It is clear that if $\mathrm{EQ}_n(x,y) = 1$ (i.e., the inputs are equal) the protocol does not err. On the other hand, $\Pr[a^j \neq b^j | \mathrm{EQ}_n(x,y) = 0] = 1/2$ for every j, and since the vectors r^j are chosen independently at random,

$$\Pr[\mathrm{EQ}_k(a,b) = 1 | \mathrm{EQ}_n(x,y) = 0] = 1/2^k, \tag{4}$$

which establishes the error.

[5] Specifically, if the gates may be arbitrary Boolean functions, there is a circuit computing EQ_n with at most $n + n = 2n$ gates. Using some optimizations to the proof of Theorem 4.11, these can be computed securely with $4n$ ANDs as the first layer is composed of exclusive-or gates. Further optimizations may be possible.

A randomized (imperfect) protocol with for EQ_n

Alice's input: $x = x_1, \ldots, x_n \in \{0,1\}^n$
Bob's input: $y = y_1, \ldots, y_n \in \{0,1\}^n$
Alice's desired output: $\mathrm{EQ}_n(x, y)$

Bob chooses k vectors $\vec{r}^1, \ldots, \vec{r}^k \in \{0,1\}^n$ uniformly at random,
Bob sends $\vec{r}^1, \ldots, \vec{r}^k \in \{0,1\}^n$ to Alice
Alice computes $a_j = \mathrm{IP}_n(x, \vec{r}^j)$ for $j = 1, \ldots, k$, and sets $a = a_1, \ldots, a_k$
Bob computes $b_j = \mathrm{IP}_n(y, \vec{r}^j)$ for $j = 1, \ldots, k$, and sets $b = b_1, \ldots, b_k$
Alice and Bob use the randomized prot. of Claim 4.9 to compute $\mathrm{EQ}_k(a, b)$.
Alice's ouput: the output of the protocol for $\mathrm{EQ}_k(a, b)$.

Figure 6: A randomized protocol with $O(k)$ ANDs, $1/2^k$ error and $2/2^k$ distance for EQ_n.

We next prove that the protocol has statistical privacy. Intuitively, Alice learns information only when she gets an incorrect output. Formally, fix any $x, y, y' \in \{0,1\}^n$ such that $y \neq y'$ and $\mathrm{EQ}_n(x, y) = \mathrm{EQ}_n(x, y')$, and compute the statistical distance between the view seen by Alice holding input x, when executing the protocol with Bob's input set to y or to y' (we will denote the corresponding vectors computed in the protocol by (a, b) and (a', b') respectively). Observe that if $\mathrm{EQ}_n(x, y) = 1$, y and y' must be identical. Thus, we only need to consider the case where $\mathrm{EQ}_n(x, y) = 0$, namely x, y, y' are three different vectors. The only information that Alice gets in the protocol which depends on Bob's input, is the output of the perfectly secure protocol for $\mathrm{EQ}_k(a, b)$ (or $\mathrm{EQ}_k(a', b')$). This implies that given this output is the same, the views are distributed identically. On the other hand, we can bound the probability that this output is not the same, as follows.

$$
\begin{aligned}
&\Pr[\mathrm{EQ}_k(a, b) \neq \mathrm{EQ}_k(a', b') | \mathrm{EQ}_n(x, y) = 0] \\
&\quad \leq \; \Pr[\mathrm{EQ}_k(a, b) = 1 | \mathrm{EQ}_n(x, y) = 0] + [\mathrm{EQ}_k(a', b') = 1 | \mathrm{EQ}_n(x, y) = 0] \\
&\quad = \; 2/2^k,
\end{aligned}
$$

where the last equality follows from (4). We may therefore conclude that the statistical distance between Alice's views for input (x, y) vs. (x, y') is at most $1/2^k$. Finally, note that Bob does not receive any messages in this protocol, so Alice's perfect privacy follows immediately. □

The number of ANDs used in the last claim is independent of n, exhibiting an inherent gap between perfect and imperfect protocols. In order to get exponentially small statistical distance and error in this protocol, the number of ANDs should still be set to be linear in n, though it may be smaller than n. Setting the number of ANDs to be polylogarithmic in n will already give a negligible statistical distance and error. This should be contrasted with the lower bounds of n (or 2^n) ANDs for perfect randomized (or deterministic, resp.) protocols for this function.

4.2 How Much does Randomization Help?

In the previous section we showed that randomization can help significantly compared to deterministic protocols. In this section we consider the limitations of randomized protocols. We first show lower bounds on the number of ANDs required in randomized protocols. For a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ our lower bound is at most n. Notice, that by Theorem 4.1 we cannot prove super-linear lower-bounds on the number of calls to the AND black box for explicit functions unless we prove super-linear lower-bounds for circuit complexity of explicit functions which is a long-standing open problem. We use our lower bounds to show that for Boolean functions the gap in the number of calls to the AND black box between deterministic protocols and randomized protocols with perfect security can be at most exponential.

We start by giving two lower bounds on the number of ANDs in perfectly-secure protocols.

Theorem 4.11 *Let $f : A \times B \to C$ be a function such that all columns of M_f are equivalent according to $\equiv_c$ and no two columns of M_f are the same. The number of AND black box calls in any perfectly-secure randomized protocol computing f is at least $\lceil \log |B| \rceil$.*

Proof: Consider any protocol that computes f securely with q ANDs. Using this protocol, we construct a (non-secure) deterministic protocol that computes f in which Bob sends a q-bit message to Alice, and Alice computes the output of f from this message: First, fix any communication string c that has positive probability for some x_0, y_0. Denote the inputs of Alice and Bob are x and y respectively. Bob finds the lexicographically first random string r_B that is consistent with y and c, and sends to Alice the bits that he puts to the AND black boxes given y, c, and r_B. By Claim 2.7, the communication c has positive probability given x, y, thus, such r_B exists. Alice finds the lexicographically first random string r_A that is consistent with x, c, and the answers of the AND black-box calls (which she can compute since she knows what Bob puts into the AND black boxes). Such r_A exists by Eq. (1). Alice outputs the output of the secure protocol where her input is x, her random input is r_A, the communication is c, and the answers of the black box calls are the outputs when Bob puts the bits that he sends her. By the perfect correctness, Alice's output is correct.

Since the columns of M_f are not the same, then for every $y \in B$ Bob in the protocol we constructed has to send a different message. Otherwise, for some $y, y' \in B$ Bob sends the same message, and for a $x \in A$ such that $f(x, y) \neq f(x, y')$ the protocols errs for y or y'. That is, Bob has to send at least $q \geq \lceil \log |B| \rceil$ bits. □

Example 4.12 **($\binom{n}{1}$OT)** Consider the well-known function $\binom{n}{1}\text{OT} : \{1, \ldots, n\} \times \{0,1\}^n \to \{0,1\}$ defined as $\binom{n}{1}\text{OT}(i, \langle y_1, \ldots, y_n \rangle) = y_i$. Theorem 4.11 proves that in any perfectly-secure protocol for $\binom{n}{1}$OT the number of ANDs is at least n since all the 2^n columns of its matrix are equivalent and distinct.[6] This implies that in any perfectly-secure protocol for $\binom{n}{1}$OT using an $\binom{2}{1}$OT black box the number of $\binom{2}{1}$OT is at least $n/2$. This reproves the result of Dodis and Micali [14] up to a factor of 2 (our proof does not use information theory).

Theorem 4.13 *Let $f : A \times B \to \{0,1\}$ be a* Boolean *function such that all columns of M_f are equivalent according to $\equiv_c$, no two rows of M_f are the same, and there is some $y_0 \in B$ such that $f(x, y_0) = 0$ for every $x \in A$. Then, the number of calls to the AND black box in any perfectly-secure randomized protocol computing f is at least $\lceil \log |A| \rceil$.*

Proof: Consider any protocol that computes f securely with q ANDs. We claim that the number of ANDs in this protocol is at least $\lceil \log |A| \rceil$, that is, $q \geq \lceil \log |A| \rceil$.

Fix any communication string c that has positive probability for some fixed inputs. By Claim 2.7, c has positive probability for every inputs x, y. Furthermore, fix the lexicographically first random input for Bob r_B that is consistent with y_0 and c (where y_0 is as guaranteed in theorem). Finally, for every x fix a random input $r_{A,x}$ for Alice that is consistent with x, c, and the answers of the AND black boxes when Bob puts bits according to y_0 and r_B. As argued in the proof of Theorem 4.11 such r_B and $r_{A,x}$ exist.

Assume that Bob holds the input y_0 and the random input r_B, and Alice holds an input x and the random input $r_{A,x}$. There are calls to the AND black box in which Alice puts 0 and there are calls in which Alice puts 1. We denote this sequence by $a_x \in \{0,1\}^q$. Assume, towards contradiction, that for two inputs $x_0, x_1 \in A$

[6]Using Theorem 4.15 below, one can prove that even in statistically-secure protocols for $\binom{n}{1}$OT the number of ANDs is $\Omega(n)$.

it holds that $a_{x_0} = a_{x_1}$. Recall that $f(x_0, y_0) = f(x_1, y_0) = 0$, and since the rows corresponding to x_0 and x_1 are not the same, there is some y_1 such that, without loss of generality, $f(x_0, y_1) = 0$ while $f(x_1, y_1) = 1$. Since $f(x_0, y_0) = f(x_0, y_1) = 0$, there must be some random string r'_B such that the view of Alice in the executions when Alice holds x_0 and r_{A,x_0} and Bob holds either y_0 and r_B or y_1 and r'_B are the same.

Consider the execution where Alice holds x_1 and r_{A,x_1} and Bob holds y_1 and r'_B. We claim that in this execution Alice sees exactly the same information as in the execution in which she holds x_1 and r_{A,x_1}, and Bob holds y_0 and r_B. This is true since Alice has the same inputs in both cases, and Bob behaves the same in both executions since he cannot distinguish these executions from the executions in which Alice holds x_0 and r_{A,x_0}. As $a_{x_0} = a_{x_1}$ and in the calls where Alice puts 1 into the AND black box, Bob puts the same values for y_0 and y_1. Thus, Alice cannot distinguish between the two cases as a result of these calls although $f(x_1, y_0) \neq f(x_1, y_1)$, contradicting the perfect correctness of the protocol. To conclude, for every $x \in A$ there is a unique $a_x \in \{0,1\}^q$, thus $q \geq \lceil \log |A| \rceil$. □

The next theorem states that for Boolean functions the gap in the number of AND black-box calls between deterministic protocols and randomized protocols with perfect security is at most exponential. This seems to resemble the simple derandomization of randomized algorithms, however this resemblance is misleading (as executing a secure protocol with all possible random coins might leek information). As an example of the difficulty, the gap can be much larger for non-perfect randomized protocols (see the EQ_n example in the previous section). Another example is the malicious model where randomization is essential (see, e.g., [14]). Thus, in the malicious model of secure computation the gap is "infinite." We prove the gap between randomized and deterministic protocols by combining the lower bounds we proved on randomized protocols and the upper bounds we proved for deterministic protocols.

Theorem 4.14 *Let f be a* Boolean *function. If there exists a perfectly-secure randomized protocol computing f using q ANDs then there is a deterministic protocol computing f with 2^q ANDs.*

Proof: By Claim 2.6, the function f can be securely computed with q ANDs if and only if every equivalence class of the columns of M_f can be computed securely with q ANDs. Thus, by Theorem 4.13, the number of distinct rows in any equivalence class is at most at most 2^q. By Theorem 3.2, there is a deterministic protocol securely computing every equivalence class of f using 2^q ANDs, and therefore, by Claim 2.6, such protocol exists for f. □

We next generalize the lower bound of Theorem 4.11 (and its proof) to protocols which might err with some probability. We first recall some definitions from communication complexity (for more information on this subject the reader might consult [24]). The one-round randomized communication complexity in the public random coin model is defined as follows: Alice and Bob each have a private input and they have a shared random input. Bob sends one message to Alice, and Alice computes the output of the protocol (there are no privacy requirements). The error of the protocol on input x, y is the probability that Alice outputs a value different than $f(x, y)$. A protocol computes f with error ϵ if for every inputs x, y its error is at most ϵ. Let $R_\epsilon^{\mathrm{pub},B\to A}(f)$ be the number of communication bits in the best such protocol computing f with error ϵ.

Theorem 4.15 *Let $f : A \times B \to \{0,1\}$ be a* Boolean *function such that all the columns of M_f are equivalent according to $\equiv_c$ and no two columns are identical. Then, in any randomized (ϵ, δ)-secure protocol computing f, the number of ANDs is at least $R_{\epsilon+7\delta}^{\mathrm{pub},B\to A}(f)$.*

Proof: First consider any $(\epsilon, 0)$-secure protocol (Alice, Bob) that computes f with q calls to the ANDs black box and with ϵ-error. (Later we deal with $\delta > 0$.) Let R_A and R_B be random variables denoting the random coins chosen by Alice and Bob respectively in protocol (Alice, Bob). Using this protocol, we construct a (non-secure) randomized protocol (Alice', Bob') with public coins that computes f in which Bob' with input y sends a q-bit message to Alice, and Alice' with input x computes the output $f(x, y)$ from this message.

We next describe Protocol (Alice', Bob'). First, Alice' and Bob' choose at random a communication string c using the public random string. That is, for some fixed $x_0 \in A, y_0 \in B$ they pick at random r'_A, r'_B (with the same distribution as Alice and Bob pick them), and set $c \leftarrow \text{TRANS}(x_0, r'_A, y_0, r'_B)$ (by Claim 2.7 the probability of picking c is independent of the choice of x_0 and y_0). Note that c is computed from the public string so both parties know c without using any communication. Next, Bob' picks a random string r_B using the public random string with probability $\Pr[r_B|\text{TRANS}(x_0, \cdot, y, \cdot) = c] = \Pr[\text{TRANS}(x_0, \cdot, y, r_B) = c] \Pr[R_B = r_B] / \Pr[\text{TRANS}(x_0, \cdot, y, \cdot) = c]$ (by Eq. (1) this probability is independent of the choice of x_0). Bob' sends to Alice the q bits that he puts in the calls to the AND black boxes given y, c, and r_B. Denote this message by M. Alice' picks at random a string r_A using the public random string with probability $\Pr[r_A|c, x, M]$. That is, given c and M, Alice finds some y_1 and r''_B such that Bob with y_1 and r''_B sends his part in c and puts the inputs M into the black-box calls, and chooses r_A with probability $\Pr[R_A = r_A|\text{TRANS}(x, \cdot, y_1, r''_B) = c]$. Note that since Alice does not know the input and random input of Bob, then the probability of choosing r_A equals $\Pr[R_A = r_A|\text{TRANS}(x, \cdot, y, r_B) = c]$. Alice outputs the output of the secure protocol where her input is x, her random input is r_A, the communication is c, and the answers of the black box are the outputs when Bob puts the bits that he sends her. The following claim asserts that the probability that Alice' errs is at most ϵ. Thus, Bob' has to send at least $R_\epsilon^{\text{pub}, B\rightarrow A}(f)$ bits.

Claim 4.16 *The above process of first choosing c, then r_B, and finally r_A, yields the same distribution as choosing r_A and r_B independently at random and then executing the secure protocol.*

Proof: First, we claim that r_B is distributed as if Bob chose it during the protocol.

$$\begin{aligned}
\Pr[\text{ Bob' picks } r_B] &= \sum_c \Pr[\text{ Bob' picks } r_B|\text{TRANS}(x_0, \cdot, y, \cdot) = c] \cdot \Pr[\text{TRANS}(x_0, \cdot, y, \cdot) = c] \\
&= \sum_c \left(\frac{\Pr[\text{TRANS}(x_0, \cdot, y, r_B) = c] \Pr[R_B = r_B]}{\Pr[\text{TRANS}(x_0, \cdot, y, \cdot) = c]} \right) \cdot \Pr[\text{TRANS}(x_0, \cdot, y, \cdot) = c] \\
&= \Pr[R_B = r_B] \sum_c \Pr[\text{TRANS}(x_0, \cdot, y, r_B) = c] \\
&= \Pr[R_B = r_B].
\end{aligned}$$

Fix any r_A and r_B. Next, we prove that for every r_B, the random choice r_A is distributed as if Alice chose it during the protocol. Let $c_0 = \text{TRANS}(x, r_A, y, r_B)$. Alice' picks r_A given that Bob' picks r_B only if Alice' and Bob' picked c_0 in the first step of the protocol. Thus,

$$\begin{aligned}
&\Pr[\text{ Alice' picks } r_A|\text{ Bob' picks } r_B] \\
&\quad = \Pr[\text{ Alice' picks } r_A|R_B = r_B] \\
&\quad = \Pr[\text{ Alice' picks } r_A|\text{TRANS}(x_0, \cdot, y, r_B) = c_0] \cdot \Pr[\text{TRANS}(x, \cdot, y, \cdot) = c_0] \\
&\quad = \Pr[R_A = r_A|\text{TRANS}(x, \cdot, y, r_B) = c_0] \cdot \Pr[\text{TRANS}(x, \cdot, y, r_B) = c_0] \\
&\quad = \Pr[R_A = r_A \text{ and } \text{TRANS}(x, \cdot, y, r_B) = c_0] \;=\; \Pr[R_A = r_a].
\end{aligned}$$

The last equality holds since $c_0 = \text{TRANS}(x, r_A, y, r_B)$. □

Thus, we proved the theorem for protocols that may err, i.e., $\epsilon > 0$, but Alice's and Bob's privacy is perfect, i.e., $\delta = 0$. Now assume the protocol only has statistical privacy, i.e., $\delta > 0$. Assume that:

- In the first step of the protocol Alice' and Bob' choose c according to their actual inputs x and y (rather than x_0 and y_0).

- In the second step of the protocol Bob' chooses r_B according to Alice's actual input x (rather than x_0).

Under these assumptions, the analysis for the case $\delta = 0$ will carry through. We next claim that we can remove theses assumptions without paying too much. By Claim A.1, the distance between choosing c according to the inputs x, y and x_0, y_0 is at most 6δ. By Alice's privacy, the distance between choosing r_B according to x_0 or x is at most δ. Thus, the error of Protocol (Alice', Bob') is at most $\epsilon + 7\delta$, which completes the proof of the theorem. $\square$

Theorem 4.11 is a special case of Theorem 4.15 since it easy to see that $R_0^{\text{pub},B\to A}(f) = \lceil \log |B| \rceil$.

Example 4.17 By [15] it holds that $R_\epsilon^{\text{pub},B\to A}(\text{IP}_n) = n/2$ for every $\epsilon < 1/2$ (in fact this lower bound holds even for protocols with many rounds). Thus, unlike EQ_n, for every ϵ, δ where $\epsilon + 7\delta < 1/2$, the inner-product function does not have an (ϵ, δ)-secure protocol which uses less than $n/2$ ANDs.

4.3 One AND: Randomization does Not Help

We have seen in previous sections that randomization can significantly reduce (up to an exponential factor) the number of required ANDs, and that already with 4 ANDs, randomized protocols compute a strictly stronger class of functions than deterministic protocols with the same number of ANDs. On the other hand, it is known (see Theorem 2.4) that for secure computation in our model without any ANDs, randomization does not help. In this section we show that with *one* AND randomization still does not help. This is done by characterizing the functions that can be computed with one AND.

Theorem 4.18 *Let $f : A \times B \to C$ be a function such that all the columns of M_f are equivalent according to $\equiv_c$. The function f can be computed securely by a randomized protocol using one call to the AND black box if and only if there are $A_1 \subseteq A$ and $B_1 \subseteq B$ such that:*

1. *For every $x \in A_1$, $y_0 \notin B_1$, and $y_1 \in B_1$ it holds that $f(x, y_0) \neq f(x, y_1)$,*
2. *For every $x \in A$ and every $y, y' \in B$ such that either $x \notin A_1$ or $y, y \notin B_1$ it holds that $f(x, y) = f(x, y')$.*
3. *For every $x \in A_1$ and every $y, y' \in B$ it holds that $f(x, y) = f(x, y')$.*

Proof: First, if Conditions (1)-(3) hold then, by Theorem 3.1, f can be computed by a secure (deterministic) protocol with 1 AND. The function f_{A_1,B_1} can be computed by Alice without any communication since each row of f_{A_1,B_1} is constant.

For the other direction, assume there is a secure protocol that computes f with 1 AND. Fix any communication string c that has positive probability for some fixed inputs; by Claim 2.7 c has positive probability given every x, y. Now, define

$$A_1 = \{x : \Pr[\text{Alice puts 1 to the black box with input } x \text{ and communication } c] > 0\},$$

and

$$B_1 = \{y : \Pr[\text{Bob puts 1 to the black box with input } y \text{ and communication } c] > 0\}.$$

We claim that A_1 and B_1 satisfy Conditions (1)-(3):

CASE 1. Assume that for some $x \in A_1$, some $y_0 \notin B_1$, and $y_1 \in B_1$ it holds that $f(x, y_0) = f(x, y_1)$. Consider executions where Alice's input is x and the communication is c. When Bob holds y_0 he puts 0 to the AND black box, thus the probability that Alice gets 1 as the output of the AND black box when Bob's input is y_0 is 0. On the other hand, the probability that Alice gets 1 as the output of the AND black box when Bob's input is y_1 is greater than 0, contradicting Bob's privacy.

CASE 2. Assume that for some $x \in A$ and $y, y' \in B$ such that either $x \notin A_1$ or $y, y' \notin B_1$ it holds that $f(x, y) \neq f(x, y')$. For both pairs of inputs (x, y) and (x, y'), the output of the black box is always 0 given that the communication is c, since either $x \notin A_1$ and Alice always puts 0 to the AND black box, or $y, y' \notin B_1$ and Bob always puts 0 to the black box. Consider executions where Alice's input is x Bob's input is either y or y', and the communication is c. Alice sees the same view in these executions although the outputs $f(x, y)$ and $f(x, y')$ are different, contradicting the correctness of the protocol.

CASE 3. If for some $x \in A_1$ and $y, y' \in B_1$ it holds that $f(x, y) \neq f(x', y)$ then again we get a contradiction to the correctness, since on both pairs of inputs (x, y) and (x, y') the communication c and the output 1 of the black box have positive probability. □

The protocol proving the sufficiency of the conditions in Theorem 4.18 is deterministic. Thus,

Corollary 4.19 *Randomized protocols with one AND can compute securely exactly the same functions as deterministic protocols with one AND.*

Acknowledgments. We thank Yuval Ishai for helpful discussions and Enav Weinreb for helpful remarks on earlier versions of this paper. We are also grateful to AT&T Labs–Research that hosted us for two weeks and for three years, respectively, during which part of this research was conducted.

References

[1] D. Beaver. Perfect privacy for two-party protocols. Technical Report TR-11-89, Computer Science, Harvard University, 1989.

[2] D. Beaver. Correlated pseudorandomness and the complexity of private computations. In *Proc. of the 28th Annu. ACM Symp. on the Theory of Computing*, pages 479–488, 1996.

[3] A. Beimel, T. Malkin, and S. Micali. The all-or-nothing nature of two-party secure computation. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 80–97. Springer-Verlag, 1999.

[4] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for noncryptographic fault-tolerant distributed computations. In *Proc. of the 20th Annu. ACM Symp. on the Theory of Computing*, pages 1–10, 1988.

[5] G. Brassard and C. Crépeau. Oblivious transfers and privacy amplification. In *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 334–347. Springer-Verlag, 1997.

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

[7] G. Brassard, C. Crépeau, and M. Sántha. Oblivious transfers and intersecting codes. *IEEE Trans. on Information Theory*, 42(6):1769–1780, 1996.

[8] R. Canetti. Security and composition of multiparty cryptographic protocols. *J. of Cryptology*, 13(1):143–202, 2000.

[9] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proc. of the 20th Annu. ACM Symp. on the Theory of Computing*, pages 11–19, 1988.

[10] B. Chor and E. Kushilevitz. A zero-one law for Boolean privacy. *SIAM J. on Discrete Mathematics*, 4(1):36–47, 1991.

[11] C. Crépeau. Equivalence between two flavors of oblivious transfers. In C. Pomerance, editor, *Advances in Cryptology – CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 350–354. Springer-Verlag, 1988.

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

[13] I. Damgard, J. Kilian, and L. Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 1999.

[14] Y. Dodis and S. Micali. Lower bounds for oblivious transfer reductions. In *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 42–55. Springer, 1999.

[15] J. Forster. A linear lower bound on the unbounded error probabilistic communication complexity. In *Proc. of the 16th Annu. IEEE Conf. on Computational Complexity*, pages 100–106, 2001.

[16] O. Goldreich and R. Vainish. How to solve any protocol problem—an efficiency improvement. In C. Pomerance, editor, *Advances in Cryptology – CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 73–86. Springer-Verlag, 1988.

[17] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. In D. Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 145 – 161 of *Lecture Notes in Computer Science*, page 2729. Springer, 2003.

[18] J. Kilian. Basing cryptography on oblivious transfer. In *Proc. of the 20th Annu. ACM Symp. on the Theory of Computing*, pages 20–31, 1988.

[19] J. Kilian. A general completeness theorem for two-party games. In *Proc. of the 23th Annu. ACM Symp. on the Theory of Computing*, pages 553–560, 1991.

[20] J. Kilian. More general completeness theorems for two-party games. In *Proc. of the 32nd Annu. ACM Symp. on the Theory of Computing*, pages 316–324, 2000.

[21] J. Kilian, E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in private computations. *SIAM J. on Computing*, 28(4):1189–1208, 2000. This is the journal version of [19, 23].

[22] E. Kushilevitz. Privacy and communication complexity. *SIAM J. on Discrete Mathematics*, 5(2):273–284, 1992.

[23] E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in multi-party private computations. In *Proc. of the 35th Annu. IEEE Symp. on Foundations of Computer Science*, pages 478–491, 1994.

[24] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[25] U. Maurer. Information-theoretic cryptography. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 47–64. Springer-Verlag, 1999.

[26] M. Naor and K. Nissim. Communication preserving protocols for secure function evaluation. In *Proc. of the 33th Annu. ACM Symp. on the Theory of Computing*, 2001. Long version: Communication complexity and secure function evaluation, *Cryptology ePrint Archive*, number 2001/076, 2001.

A Properties of Statistically-Secure Protocols

We next prove a generalization of Claim 2.7 to Boolean functions with respect to protocols with statistical privacy.

Claim A.1 *Let $f : A \times B \to \{0,1\}$ be a* Boolean *function such that all columns of M_f are equivalent according to $\equiv_c$. For every protocol that (ϵ, δ)-securely computes f it holds that for every $x, x' \in A$ and every $y, y' \in B$*

$$\text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(x',\cdot,y',\cdot)) \quad \leq \quad 6\delta.$$

Proof: First we claim that, by Alice's privacy, for every $u_0, u_1 \in A$ and every $v \in B$ it holds that

$$\text{DIST}(\text{TRANS}(u_0,\cdot,v,\cdot), \text{TRANS}(u_1,\cdot,v,\cdot)) \quad \leq \quad \delta. \tag{5}$$

Similarly, by Bob's privacy, for every $u \in A$ and every $v_0, v_1 \in B$, if $f(u, v_0) = f(u, v_1)$ then

$$\text{DIST}(\text{TRANS}(u,\cdot,v_0,\cdot), \text{TRANS}(u,\cdot,v_1,\cdot)) \quad \leq \quad \delta. \tag{6}$$

In the above equation we require that $f(u, v_0) = f(u, v_1)$. We next remove this requirement and prove that for every $x \in A$ and every $y, y' \in B$, it holds that

$$\text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(x,\cdot,y',\cdot)) \quad \leq \quad 5\delta. \tag{7}$$

Towards this goal, we claim that for every $x \in A$ and every $y, y' \in B$, there are $v \in B$ and $u_0, u_1 \in A$ such that $f(u_0, y) = f(u_0, v)$ and $f(u_1, v) = f(u_1, y')$. If there is some u_0 such that $f(u_0, y) = f(u_0, y')$ then we take $u_0 = u_1$ and $v = y'$. Otherwise, fix any $v \in B$ such that its column is not equal to the columns of y and y'. That is, there is some u_1 such that $f(u_1, y) \neq f(u_1, v)$ and there is some u_0 such that $f(u_0, y') \neq f(u_0, v)$. Since f is Boolean and the columns of y and y' disagree on each entry, it holds that $f(u_0, y) = f(u_0, v)$ and $f(u_1, v) = f(u_1, y')$. Thus, by (5) and (6),

$$\begin{aligned}
&\text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(x,\cdot,y',\cdot)) \\
&\quad \leq \quad \text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(u_0,\cdot,y,\cdot)) \; + \; \text{DIST}(\text{TRANS}(u_0,\cdot,y,\cdot), \text{TRANS}(u_0,\cdot,v,\cdot)) \\
&\qquad\quad + \text{DIST}(\text{TRANS}(u_0,\cdot,v,\cdot), \text{TRANS}(u_1,\cdot,v,\cdot)) \; + \; \text{DIST}(\text{TRANS}(u_1,\cdot,v,\cdot), \text{TRANS}(u_1,\cdot,y',\cdot)) \\
&\qquad\quad + \text{DIST}(\text{TRANS}(u_1,\cdot,y',\cdot), \text{TRANS}(x,\cdot,y',\cdot)) \\
&\quad \leq \quad 5\delta.
\end{aligned}$$

Finally, by (5) and (7),

$$\begin{aligned}
&\text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(x',\cdot,y',\cdot)) \\
&\quad \leq \quad \text{DIST}(\text{TRANS}(x,\cdot,y,\cdot), \text{TRANS}(x,\cdot,y',\cdot)) + \text{DIST}(\text{TRANS}(x,\cdot,y',\cdot), \text{TRANS}(x',\cdot,y',\cdot)) \\
&\quad \leq \quad 6\delta.
\end{aligned}$$

□

B Extensions to the Two-Sided Model

The model we considered in the paper was one-sided output, that is, only Alice gets the output of the protocol while Bob gets no information. In this section we consider a generalized model where Alice gets the output $f^{\rm Alice}(x,y)$ and Bob the gets output $f^{\rm Bob}(x,y)$. A secure protocol for $\mathbf{f} = \langle f^{\rm Alice}, f^{\rm Bob}\rangle$ guarantees that Alice does not learn any information not implied by x and $f^{\rm Alice}(x,y)$, and Bob does not learn any information not implied by y and $f^{\rm Bob}(x,y)$. The one-sided model is the special case of the two-sided model when we fix $f^{\rm Bob}$ to be a constant function.

In this section we consider deterministic protocols. The AND black box where Alice always gets the output of the function is not complete for the two-sided model. Neither is the AND black box where both Alice and Bob get the output of the AND. Thus, we consider an AND black box where only one party gets the output of the AND black box and in each execution of the black box the protocol defines if Alice gets the output or if Bob gets the output. To see that this black box is complete, first Alice and Bob use the protocol of Theorem 3.2 for computing $(f^{\rm Alice}, {\rm const})$ always giving the output of the black box to Alice, and then they independently use the protocol of Theorem 3.2 for computing $({\rm const}, f^{\rm Bob})$ always giving the output of the black box to Bob.

We represent a pair of functions $\mathbf{f} = \langle f^{\rm Alice}, f^{\rm Bob}\rangle$, where $\mathbf{f} : A \times B \to (C \times C) \cup \{*\}$ by a matrix $M_{\mathbf{f}}$ whose rows are labeled by the elements of A, columns are labeled by the elements of B, and $M_{\mathbf{f}}(x,y) = \langle f^{\rm Alice}(x,y), f^{\rm Bob}(x,y)\rangle$.

Definition B.1 ([22]) *The relation $\sim_c$ on the columns of a matrix $M_{\mathbf{f}}$ is defined as follows: $y, y' \in B$ satisfy $y \sim_c y'$ if there exists some $x \in A$ such that $f^{\rm Alice}(x,y) = f^{\rm Alice}(x,y') \neq *$. The equivalence relation $\equiv_c$ on the columns of $M_{\mathbf{f}}$ is defined as the transitive closure of the relation $\sim_c$. Similarly, the relations $\sim_r$ and $\equiv_r$ are defined on the rows of $M_{\mathbf{f}}$. That is, $x, x' \in A$ satisfy $x \sim_r x'$ if there exists some $y \in B$ such that $f^{\rm Bob}(x,y) = f^{\rm Bob}(x',y) \neq *$, and the relation $\equiv_r$ on the rows of $M_{\mathbf{f}}$ is defined as the transitive closure of the relation $\sim_r$.*

Definition B.2 (Forbidden Matrix) *A matrix $M_{\mathbf{f}}$ whose rows are labeled by A and columns are labeled by B is constant if:*

- *For every $x \in A$ there is $z_x \in C$ such that for every $y \in B$ either $f^{\rm Alice}(x,y) = z_x$ or $f^{\rm Alice}(x,y) = *$, and*
- *For every $y \in B$ there is $z_y \in C$ such that for every $x \in A$ either $f^{\rm Bob}(x,y) = z_y$ or $f^{\rm Bob}(x,y) = *$.*

A matrix $M_{\mathbf{f}}$ is a forbidden matrix iff the matrix is not constant, all the rows of $M_{\mathbf{f}}$ are equivalent according to $\equiv_r$, and all the columns of $M_{\mathbf{f}}$ are equivalent according to $\equiv_c$.

The next theorem of Kushilevitz [22] characterizes when a function can be computed securely without any ANDs.

Theorem B.3 ([22]) *The pair of functions $\mathbf{f} = \langle f^{\rm Alice}, f^{\rm Bob}\rangle$ can be computed by a perfectly-secure randomized protocol with 0 ANDs if and only if the function f can be computed by a deterministic protocol with 0 ANDs if and only if $M_{\mathbf{f}}$ does not contain a forbidden sub-matrix.*

The protocol that proves that if $M_{\mathbf{f}}$ does not contain a forbidden sub-matrix then $\mathbf{f}$ can be computed securely proceeds in rounds. In each beginning of a round, Alice and Bob both know sets $A' \subseteq A$ and $B' \subseteq B$ such that $x \in A'$ and $y \in B'$. In the beginning of the protocol $A' \leftarrow A$ and $B' \leftarrow B$. Let M' be the restriction of $M_{\mathbf{f}}$ to A' and B'. Since $M_{\mathbf{f}}$ does not contain a forbidden matrix then either

- The sub-matrix M' is constant and each party can can compute its output from its input and halt.
- Not all the rows of M' are equivalent. In this case Alice sends to Bob the equivalence class of x in M'. Alice and Bob both set A' to contain all inputs x' in this equivalence class, and B' does not change. Or,
- Not all the columns of M' are equivalent. In this case Bob sends to Alice the equivalence class of y in M'. Alice and Bob both set B' to contain all inputs y' in this equivalence class, and A' does not change.

Since in each round the size of either A' or B' decreases then this protocol must terminate after reaching a constant M'. For every A' and B' Alice can deduce the equivalence class of x from x and $f^{\mathrm{Alice}}(x, y)$. Similarly, Bob can deduce the equivalence class of y from y and $f^{\mathrm{Bob}}(x, y)$. Thus, the protocol is secure.

If we apply the above protocol to a matrix $M_{\mathbf{f}}$ that contains at least one forbidden sub-matrix then Alice and Bob get a decomposition of $M_{\mathbf{f}}$ into forbidden matrices and constant matrices. When Alice and Bob reach a forbidden sub-matrix, they can use the AND black box. Unlike the one-sided model, the output of the AND black box can result in a function where either not all the rows are equivalent or not all the columns are equivalent. Thus, Alice and Bob can exchange messages until they reach another forbidden sub-matrix, use another call to the AND black box and so on. The following two theorems characterize the number of ANDs required to securely compute a function:

Theorem B.4 *Let $\mathbf{f} = \langle f^{\mathrm{Alice}}, f^{\mathrm{Bob}} \rangle$ be a pair of function. The function $\mathbf{f}$ can be computed securely by a deterministic protocol with q ANDs if and only if each sub-matrix of $M_{\mathbf{f}}$ whose rows are equivalent according to $\equiv_r$ and columns are equivalent according to $\equiv_c$ can be computed securely by a deterministic protocol with q ANDs.*

Theorem B.5 *Let $\mathbf{f} = \langle f^{\mathrm{Alice}}, f^{\mathrm{Bob}} \rangle$ be a pair of function such that all columns of $M_{\mathbf{f}}$ are equivalent according to $\equiv_c$, and all the rows of $M_{\mathbf{f}}$ are equivalent according to $\equiv_r$. The functions $\mathbf{f}$ can be computed securely with q calls to the AND black box if and only if there are sets $A_1 \subseteq A$ and $B_1 \subseteq B$ such that at least one of the following of requirements hold:*

1. *For every $x \in A_1$, every $y_0 \notin B_1$, and every $y_1 \in B_1$ such that $\mathbf{f}(x, y_0), \mathbf{f}(x, y_1) \neq *$ it holds that $f^{\mathrm{Alice}}(x, y_0) \neq f^{\mathrm{Alice}}(x, y_1)$, and the function $\mathbf{f}_{A_1,B_1}$ can be computed securely with $q - 1$ calls to the AND black box.*
2. *For every $x_0 \notin A_1$, every $x_1 \in A_1$, and every $y \in B_1$ such that $\mathbf{f}(x_0, y), \mathbf{f}(x_1, y) \neq *$ it holds that $f^{\mathrm{Bob}}(x_0, y) \neq f^{\mathrm{Bob}}(x_1, y)$, and the function $\mathbf{f}'_{A_1,B_1}$, described in Figure 7, can be computed securely with $q - 1$ calls to the AND black box.*

Case (1) in the above theorem corresponds to the case that Alice gets the output of the AND black box and Case (2) corresponds to the case that Bob gets the output of the AND black box.

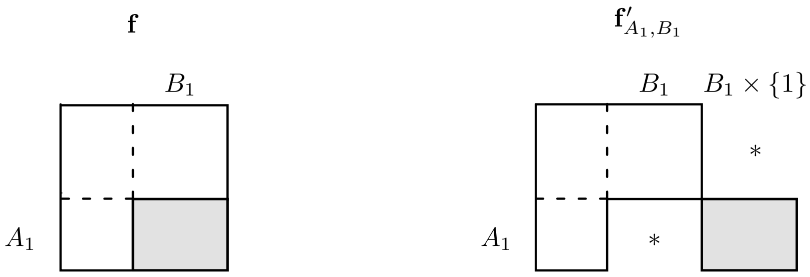

Figure 7: The matrices of the functions $\mathbf{f}$ and $\mathbf{f}'_{A_1,B_1}$.