Towards Characterizing Securely Computable Two-Party Randomized Functions

Deepesh Data
School of Technology & Computer Science
Tata Institute of Fundamental Research
deepeshd@tifr.res.in

Manoj Prabhakaran
Dept. of Computer Science & Engineering
Indian Institute of Technology Bombay
mp@cse.iitb.ac.in

Abstract

A basic question of cryptographic complexity is to combinatorially characterize all randomized functions which have information-theoretic semi-honest secure 2-party computation protocols. The corresponding question for deterministic functions was answered almost three decades back, by Kushilevitz [Kus89]. In this work, we make progress towards understanding securely computable *randomized* functions. We bring tools developed in the study of completeness to bear on this problem. In particular, our characterizations are obtained by considering only symmetric functions with a combinatorial property called *simplicity* [MPR12].

Our main result is a complete combinatorial characterization of randomized functions *with ternary output* kernels, that have information-theoretic semi-honest secure 2-party computation protocols. In particular, we show that there exist simple randomized functions with ternary output that do not have secure computation protocols. (For deterministic functions, the smallest output alphabet size of such a function is 5, due to an example given by Beaver [Bea89].)

Also, we give a complete combinatorial characterization of randomized functions that have *2-round* information-theoretic semi-honest secure 2-party computation protocols.

We also give a counter-example to a natural conjecture for the full characterization, namely, that all securely computable simple functions have secure protocols with a unique transcript for each output value. This conjecture is in fact true for deterministic functions, and – as our results above show – for ternary functions and for functions with 2-round secure protocols.

1 Introduction

Understanding the nature of secure multiparty computation has been a key problem in modern cryptography, ever since the notion was introduced. While this has been a heavily researched area, some basic problems have remained open. In this work we explore the following fundamental question:

> Which randomized functions have information-theoretic semi-honest secure 2-party computation protocols?

The corresponding question for deterministic functions was answered almost three decades back, by Kushilevitz [Kus89] (originally, restricted to symmetric functions, in which both parties get the same output). The dual question of which functions are complete, initiated by Kilian [Kil88], has also been fully resolved, for semi-honest [MPR12] and even active security [KMPS14]. However, the above question itself has seen little progress since 1989.

In this work, we make progress towards understanding securely computable *randomized* functions. (Through out this paper, security will refer to information-theoretic semi-honest security.) We bring tools developed in the study of completeness to bear on this problem. In particular, our characterizations are obtained by considering only symmetric functions with a combinatorial property called *simplicity* [MPR12]. (As shown in [MPR12], a function is semi-honest securely computable if and only if it is simple, and a related function called its "kernel" – which is always a simple function – is securely computable.)

One may start off by attempting to generalize the result of Kushilevitz [Kus89] so that it applies to randomized functions as well. This characterization showed that any securely computable deterministic function has a secure protocol in which the two parties take turns to progressively reveal more and more information about their respective inputs – by restricting each input to smaller and smaller subsets – until there is exactly enough information to evaluate the function. However, a naïve generalization of this result fails for randomized functions, as it is possible for a securely computable function to have every output value in the support of every input combination; thus the input spaces cannot be shrunk at all during a secure protocol.

A more fruitful approach would be to consider the protocol for deterministic functions as partitioning the *output space* at each step, and choosing one of the parts. This is indeed true when considering deterministic functions which are *simple*. Such a protocol results in a unique transcript for each output value. An *a priori* promising conjecture would be that every securely computable simple function – deterministic or randomized – has such a unique-transcript protocol. Unfortunately, this conjecture turns out to be false.

However, for small output alphabets, we can prove that this conjecture holds. Indeed, we show that the exact threshold on the alphabet size where this conjecture breaks down is 4. When the output alphabet size of a simple function is at most 3, we give a combinatorial characterization of secure computability; our characterization implies that such functions do have unique-transcript secure protocols. We also characterize simple functions which have two-round secure protocols, which again turn out to all have unique-transcript protocols.

We leave the full characterization as an important open problem.

Our Results

- Our main result is a complete combinatorial characterization of randomized functions *with ternary output* kernels, that have information-theoretic semi-honest secure 2-party computation protocols. In particular, we show that there exist simple randomized functions with ternary output that do not have secure computation protocols. (For deterministic functions, the smallest output alphabet size of such a function is 5, due to an example given by Beaver [Bea89].)

- We also give a complete combinatorial characterization of randomized functions that have *2-round* information-theoretic semi-honest secure 2-party computation protocols.

- We also give a counter-example to a natural conjecture for the full characterization, namely, that all securely computable simple functions have secure protocols with a unique transcript for each output value. This conjecture is in fact true for deterministic functions, and – as our results above show – for ternary functions and for functions with 2-round secure protocols.

1.1 Technical Overview

Prior work [MPR12, MPR13] lets us focus on symmetric functions: A randomized function F is securely realizable if and only if it is "isomorphic" – i.e., essentially equivalent, upto sampling additional local outputs – to a symmetric function G called its kernel, and G itself has a secure protocol. Whether F is isomorphic to its kernel or not is given by an easy to check combinatorial property, called simplicity: a randomized function F is said to be simple if the probability of the outputs to Alice and Bob being z_A and z_B respectively, when their inputs are x and y respectively, can be factorized as $\Pr[F(x,y) = (z_A, z_B)] = \rho(x, z_A) \cdot \sigma(y, z_B)$, for some fixed functions $\rho : \mathcal{X} \times \mathcal{Z}_A \to \mathbb{R}_+$ and $\sigma : \mathcal{Y} \times \mathcal{Z}_B \to \mathbb{R}_+$, where $\mathcal{X}, \mathcal{Y}$ are Alice's and Bob's input domains and $\mathcal{Z}_A, \mathcal{Z}_B$ their output domains. Further the kernel of F is explicitly defined and easy to find.

Hence the problem of characterization secure computability reduces to the problem of characterizing secure computability of functions which are kernels. Such functions are symmetric and simple. As such, we work with symmetric and simple functions.

Characterizing Ternary-Kernel Functions. Our main result could be stated as follows:

Theorem 1. *If a randomized function F has a kernel G with an output alphabet of size at most 3, then F is securely computable if and only if F is simple and there is some ordering of G's output alphabet $\mathcal{Z}$ as (z_1, z_2, z_3) and two functions $q : \mathcal{X} \to [0,1]$ and $r : \mathcal{Y} \to [0,1]$, such that one of the following holds:*

$$\begin{aligned} &\Pr[G(x,y) = z_1] = q(x) \\ &\Pr[G(x,y) = z_2] = (1 - q(x)) \cdot r(y) \\ &\Pr[G(x,y) = z_3] = (1 - q(x)) \cdot (1 - r(y)) \end{aligned} \qquad \textit{or} \qquad \begin{aligned} &\Pr[G(x,y) = z_1] = r(y) \\ &\Pr[G(x,y) = z_2] = (1 - r(y)) \cdot q(x) \\ &\Pr[G(x,y) = z_3] = (1 - r(y)) \cdot (1 - q(x)) \end{aligned}$$

Note that if the first set of conditions holds, there is a secure protocol in which Alice either sends z_1 as the output to Bob (with probability $q(x)$) or asks Bob to pick the output; if Bob is asked to pick the output he sends back either z_2 as the output (with probability $r(y)$) or z_3 otherwise. If the second condition holds there is a symmetric protocol with Bob making the first move.

Surprisingly, these are the only two possibilities for G to have a secure protocol. To prove this, however, takes a careful analysis of the linear-algebraic constraints put on a protocol by the security definition and the fact that the function is simple. We start by observing that a secure protocol for a symmetric simple function must have a simulator that can simulate the transcript of an execution merely from the output (rather than the corrupt party's input and the output).[1] Then, supposing that the first message in the protocol is a single bit sent by Alice, we identify that there is a quantity independent of either party's input, denoted by $\phi(z)$, that gives the probability of the first message being 0, conditioned on the output being z. Specifying these quantities, $\phi(z)$ at each round fully specifies the protocol. We show that $\sum_{z \in \mathcal{Z}} \Pr[G(x,y) = z] \cdot \phi(z) = q(x)$, a quantity independent of y. By carefully analyzing the constraints arising from these equations, we prove Theorem 1.

Characterizing 2-Round Protocols. Our second result is as follows:

Theorem 2. *A function F has a 2-round secure protocol iff its kernel has a 2-round unique-transcript protocol.*

Observe that F has a 2-round secure protocol iff its kernel has one too, as F is isomorphic to its kernel and a secure protocol for a function can be transformed to one for an isomorphic function without changing the communication involved. What needs to be proven is that if the kernel (or any symmetric simple function) has a 2-round secure protocol, then it has a 2-round unique-transcript protocol. We do this by identifying an equivalence relation among the outputs, such that any 2-round protocol with (say) Alice making the first will have Alice's input only influencing which equivalence class of the output is chosen, and then Bob's input influences which output is chosen, given its equivalence class.

[1] This is not true for every symmetric function. For instance, in a semi-honest secure protocol for XOR, the transcript must necessarily reveal both parties' inputs, but this cannot be simulated from the output without knowing one party's input. A function like XOR is not simple, though it is isomorphic to one.

1.2 Related Work

There has been a large body of work regarding the complexity of secure multi-party and 2-party computation. [MPR13] surveys many of the important results in the area. Kushilevitz [Kus89] gave a combinatorial characterization of securely computable two-party *deterministic* functions (with perfect security), along with a generic round-optimal secure protocol for functions that satisfy the characterization condition. This was later extended to statistical security and also to security against active corruption [MPR09, KMQR09].

Among randomized functions in which only Bob gets any output, simple functions have a deterministic kernel corresponding to a function of Alice's input alone, and hence are always securely computable. This observation was already made by Kilian [Kil00]. Recently, Data [Dat16] considered the same problem with a probability distribution on the inputs, and gave communication-optimal protocols in different security settings.

2 Preliminaries

A general randomized function is denoted by a conditional probability distribution $p_{Z_AZ_B|XY}$, where X, Y, Z_A, Z_B take values in finite alphabets $\mathcal{X}, \mathcal{Y}, \mathcal{Z}_A, \mathcal{Z}_B$, respectively. In a protocol for computing this function, when Alice and Bob are given inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, respectively, they should output $z_A \in \mathcal{Z}_A$ and $z_B \in \mathcal{Z}_B$ respectively, such that (z_A, z_B) is distribute according to $p_{Z_AZ_B|X=x,Y=y}$.

Notation. We shall consider various discrete random variables (inputs, outputs, protocol messages). We denote the probability mass function of a random variable U by p_U. For random variables (U, V), we denote the conditional probability mass function of U conditioned on V by $p_{U|V}$. $[n]$ denotes the set $\{1, \cdots, n\}$.

Protocols. We consider computationally unbounded two-party protocols, without any setup. Such a protocol Π is fully defined by the input and output domains, the *next message functions* and the *output functions* for Alice and Bob. Let $(\mathcal{X}, \mathcal{Z}_A, \mathrm{next}_\Pi^A, \mathrm{out}_\Pi^A)$ and $(\mathcal{Y}, \mathcal{Z}_B, \mathrm{next}_\Pi^B, \mathrm{out}_\Pi^B)$ denote the input domain, output domain, the next message function and the output function, for Alice and Bob, respectively. The functions are all potentially randomized. next_Π^A and next_Π^B output the next message given the transcript so far and the local input. (Note that in the information-theoretic setting, protocols need not maintain local state.) Similarly, out_Π^A and out_Π^B map the transcript and local input to a local output in $\mathcal{Z}$.

In running a protocol, Alice and Bob are first given inputs $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, respectively. Then they take turns sending messages to each other according to their next message functions, and in the end (recognizable by both parties) each party produces an output according to its output function. We assume that the protocol terminates with probability one after a finite number rounds.

Running a protocol induces a distribution over the set of all possible (complete) transcripts, $\mathcal{M}$. The protocol Π induces a conditional probability distribution $\mathrm{Pr}_\Pi[m|x, y]$, for every input $x \in \mathcal{X}, y \in \mathcal{Y}$ and every $m \in \mathcal{M}$. Suppose that $m = (m_1, m_2, \dots,)$ is the transcript generated by Π when parties have inputs x and y, where m_i's, for odd i, are sent by Alice, and m_i's, for even i, are sent by Bob. Note that during the execution of a protocol, a message sent by any party is determined by its input and all the messages it has exchanged so far; and conditioned on these two, the message is independent of other party's input. Using this we can write $\mathrm{Pr}_\Pi[m|x, y] = \alpha(m, x)\beta(m, y)$, where $\alpha(m, x) = \Pi_{i:i \text{ is odd}}\mathrm{Pr}_\Pi[m_i|m_{<i}, x]$, and $\beta(m, y) = \Pi_{i:i \text{ is even}}\mathrm{Pr}_\Pi[m_i|m_{<i}, y]$.

Secure Protocols Through out this paper security refers to *information-theoretic semi-honest security*. We restrict ourselves to finite functions and perfect security. We remark that all our results can be extended to statistical security, as we shall show in the full version.

Definition 1. *A protocol Π for computing a function $p_{Z_AZ_B|XY}$, with transcript space $\mathcal{M}$, is said to be (perfectly semi-honest) secure iff there exist functions $S_1 : \mathcal{M} \times \mathcal{X} \times \mathcal{Z}_A \to [0, 1]$ and $S_2 : \mathcal{M} \times \mathcal{Y} \times \mathcal{Z}_B \to$*

$[0,1]$ *such that* $\Pr_\Pi[m|x,y,z_A,z_B] = S_1(m,x,z_A) = S_2(m,y,z_B)$ *for all* $m \in \mathcal{M}$ *and* x, y, z_A, z_B *such that* $p_{Z_AZ_B|XY}(z_A, z_B|x, y) > 0$.

Kernel and Simple Functions. In order to formally define the kernel of a function, we define the *characteristic bipartite graph* of a randomized function $p_{Z_AZ_B|XY}$ as $\mathcal{G}(p_{Z_AZ_B|XY}) = (V, E)$, where $V = (\mathcal{X} \times \mathcal{Z}_A) \cup (\mathcal{Y} \times \mathcal{Z}_B)$ and $E = \{((x, z_A), (y, z_B)) : p_{Z_AZ_B|XY}(z_A, z_B|x, y) > 0\}$.

Definition 2 (Kernel – Common-information in a randomized function [MPR12])**.** *The kernel of a randomized function* $p_{Z_AZ_B|XY}$ *is a symmetric randomized function, which takes* x *and* y *from the parties and samples* (z_A, z_B) *according to* $p_{Z_AZ_B|X=x,Y=y}$*. Then it outputs to both parties the connected component of* $\mathcal{G}(p_{Z_AZ_B|XY})$ *which contains the edge* $((x, z_A), (y, z_B))$*.*

Note that the kernel of $p_{Z_AZ_B|XY}$ is a symmetric randomized function.

Definition 3. *A randomized function* $p_{Z_AZ_B|XY}$ *is said to be* simple *if there exist two functions* $\rho : \mathcal{X} \times \mathcal{Z}_A \to \mathbb{R}_+$ *and* $\sigma : \mathcal{Y} \times \mathcal{Z}_B \to \mathbb{R}_+$ *such that for all* $x \in \mathcal{X}, y \in \mathcal{Y}, z_A \in \mathcal{Z}_A$*, and* $z_B \in \mathcal{Z}_B$*,* $p_{Z_AZ_B|XY} = \rho(x, z_A) \cdot \sigma(y, z_B)$*.*

Here $\mathbb{R}_+$ denotes the set of non-negative real numbers. For deterministic functions, instead of $\mathbb{R}_+$, one can take $\{0, 1\}$ in the above definition.

As shown in [MPR12], a function is simple iff it is "isomorphic" to its kernel: i.e., a secure protocol for the kernel can be transformed to one for the function itself, and vice versa, without changing the communication involved. A function is securely computably iff it is simple and its kernel is securely computable. Thus we shall focus on characterizing secure computability for kernel functions, which are all symmetric, simple functions.

The combinatorial definition of simplicity above will be crucially used in our analysis. Indeed, the factorization property clarifies otherwise obscure connections and elusive constraints.

Remark 1. *Our definition of a simple function is slightly different from the one in [MPR12]. Specifically, the original definition, which is in terms of isomorphism between a function and its kernel, allows the functions outputs to be relabeled as long as, combined with either input, it corresponds to the output of the kernel. As an example, the bit XOR function has only two outputs, 0 and 1, but its kernel has 4 outputs (corresponding to the* exchange function *where the output is the pair of inputs); two of the outputs of the kernel (say* $(0, 0)$ *and* $(1, 1)$*) are mapped to the same output (say, 0) of the XOR function. We call this "symbol overloading," wherein two (publicly) distinguishable outcomes are assigned the same output symbol. The definition above precludes symbol overloading. (Thus, XOR in itself will not meet our definition, but after relabeling the outputs – so that it corresponds to the exchange function – it does.)*

For protocols for symmetric functions, the output can be written as a function merely of the transcript and we can simulate the transcript just based on the (common) output, without needing either party's input. We prove the following lemma in Appendix A.

Lemma 1. *If* Π *is a perfectly semi-honest secure protocol for a simple function* $p_{Z|XY}$*, then there are (deterministic) functions* $\mathrm{out}_\Pi : \mathcal{M} \to \mathcal{Z}$ *and* $S : \mathcal{M} \to [0, 1]$ *such that for all* $x \in \mathcal{X}$*,* $y \in \mathcal{Y}$*,* $z \in \mathcal{Z}$ *and* $m \in \mathcal{M}$*,*

$$\begin{aligned} \mathrm{out}_\Pi^A(m, x) = \mathrm{out}_\Pi^B(m, y) &= \mathrm{out}_\Pi(m) && \text{if } \Pr_\Pi[m|x, y] > 0, \\ \Pr_\Pi[m|x, y, z] &= S(m, z) && \text{if } p_{Z|XY}(z|x, y) > 0. \end{aligned}$$

Note that above, if $z \neq \mathrm{out}_\Pi(m)$, $S(m,z) = \Pr_\Pi[m|x,y,z] = 0$. By writing $\mu(m) = S(m, \mathrm{out}_\Pi(m))$ we have the following: for all $x \in \mathcal{X}$, $y \in \mathcal{Y}$, $z \in \mathcal{Z}$ s.t. $p_{Z|XY}(z|x,y) > 0$, and all $m \in \mathcal{M}$, we have

$$\Pr_\Pi[m|x,y,z] = \begin{cases} \mu(m) & \text{if } \mathrm{out}_\Pi(m) = z \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Thus, for each $z \in \mathcal{Z}$ (such that for some (x,y), $p_{Z|XY}(z|x,y) > 0$), μ defines a probability distribution over $\{m \in \mathcal{M} : \mathrm{out}_\Pi(m) = z\}$. Also, since $\Pr_\Pi[m|x,y] = \Pr_\Pi[m,z|x,y]$, for $z = \mathrm{out}_\Pi(m)$, and $\Pr_\Pi[m,z|x,y] = \Pr_\Pi[m|x,y,z] \cdot \Pr_\Pi[z|x,y] = \mu(m) \cdot p_{Z|XY}(z|x,y)$ we have, for all $x \in \mathcal{X}, y \in \mathcal{Y}, m \in \mathcal{M}$,

$$\Pr_\Pi[m|x,y] = \mu(m) \cdot p_{Z|XY}(\mathrm{out}_\Pi(m)|x,y). \tag{2}$$

A Normal Form for $p_{Z|XY}$. For a symmetric randomized functionality $p_{Z|XY}$, we define the relation $x \equiv x'$ for $x, x' \in \mathcal{X}$ to hold, if $\forall y \in \mathcal{Y}, z \in \mathcal{Z}$, $p(z|x,y) = p(z|x',y)$; similarly we define $y \equiv y'$ for $y, y' \in \mathcal{Y}$. We define $z \equiv z'$ for $z, z' \in \mathcal{Z}$, if there exists a constant $c \geq 0$ such that $\forall x \in \mathcal{X}, y \in \mathcal{Y}$, $p(z|x,y) = c \cdot p(z'|x,y)$. We say that $p_{Z|XY}$ is in normal form if $x \equiv x' \Rightarrow x = x'$, $y \equiv y' \Rightarrow y = y'$, and $z \equiv z' \Rightarrow z = z'$.

It is easy to see that any $p_{Z|XY}$ can be transformed into one in normal form $p_{Z^*|X^*Y^*}$ with possibly smaller alphabets, so that $p_{Z|XY}$ is securely computable if and only if $p_{Z^*|X^*Y^*}$ is securely computable. We will assume in this paper that $p_{Z|XY}$ is in normal form.

For the ease of notation, in this paper we often denote a randomized function $p_{Z|XY}$ by an equivalent function f, such that $f(x,y,z) = p_{Z|XY}(z|x,y)$, for every $x \in \mathcal{X}, y \in \mathcal{Y}, z \in \mathcal{Z}$. We may use f and $p_{Z|XY}$ interchangeably.

Unique-Transcript Protocols. A unique transcript protocol Π for a symmetric function is one in which each output $z \in \mathcal{Z}$ has a unique transcript that can result in it: i.e., for every $z \in \mathcal{Z}$, there is at most one $m \in \mathcal{M}$ such that $\mathrm{out}_\Pi(m) = z$. Such a protocol is always a secure protocol for the function it computes (and has a deterministic simulator, which when given an output z assigns probability 1 to the unique transcript m such that $\mathrm{out}_\Pi(m) = z$).

It follows from [Kus89] that every securely computable simple *deterministic* function has a unique-transcript secure protocol. The subset of securely computable simple randomized functions that we characterize also have unique-transcript protocols. However, we shall show an example of a securely computable function (just outside the sets we characterize) which does not have any unique-transcript protocol. Understanding such functions remains the next step in fully characterizing securely computable randomized functions.

3 Characterization of Functions up to Ternary Output Alphabet

As mentioned earlier, we shall represent a randomized function $p_{Z|XY}$ by an equivalent function $f : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to [0,1]$, such that $f(x,y,z) = p_{Z|XY}(z|x,y)$. For a simple function f, we shall write $f = (\rho, \sigma)$ where $f(x,y,z) = \rho(x,z) \cdot \sigma(y,z)$. Note that for every $x \in \mathcal{X}, y \in \mathcal{Y}$ we have $\sum_{z \in \mathcal{Z}} f(x,y,z) = 1$.

3.1 Compact Representation of Secure Protocols

We assume, w.l.o.g., that both Alice and Bob send only binary messages to each other in every round, and only one party sends a message in one round. Suppose Π is a protocol that securely computes a simple function $f = (\rho, \sigma)$. In this section we consider protocols of arbitrary number of rounds, and so, w.l.o.g., we assume that in Π Alice and Bob take turns exchanging single bits and that Alice sends the first message in Π.

Let $q : \mathcal{X} \to [0,1]$ be such that, Alice, given an input x chooses 0 as her first message with probability $q(x)$ (and 1 with probability $1 - q(x)$). We define $\Pi^{(0)}$ to be the protocol, with input spaces $\mathcal{X}^{(0)} = \{x \in \mathcal{X} : q(x) > 0\}$ and $\mathcal{Y}$, in which the Alice's first message is redefined to be 0 with probability 1 for every input $x \in \mathcal{X}^{(0)}$; otherwise $\Pi^{(0)}$ has identical next message and output functions as Π. Let $f^{(0)}$ be the function computed by $\Pi^{(0)}$: i.e., $f^{(0)}(x,y,z) = \Pr_{\Pi^{(0)}}[z|x,y]$ for all $x \in \mathcal{X}^{(0)}, y \in \mathcal{Y}, z \in \mathcal{Z}$. $f^{(1)}$ is defined symmetrically.

Claim 1. *There exists a function $\phi : \mathcal{Z} \to [0,1]$ such that for all $x \in \mathcal{X}, y \in \mathcal{Y}$,*

$$q(x) = \sum_{z \in \mathcal{Z}} \phi(z) f(x,y,z). \tag{3}$$

Further, $f^{(0)}(x,y,z) = \frac{\phi(z)}{q(x)} \cdot f(x,y,z)$, for all $x \in \mathcal{X}^{(0)}, y \in \mathcal{Y}, z \in \mathcal{Z}$.

Proof. We define

$$\phi(z) := \sum_{\substack{m : m_1 = 0, \\ \mathrm{out}_\Pi(m) = z}} \mu(m), \tag{4}$$

where $\mu(m)$ is as defined in (1). Here m_1 denotes the first bit in the transcript m. Note that we have $\phi(z) \in [0,1]$ because for each z, μ defines a probability distribution over $\mathcal{M}_z := \{m : \mathrm{out}_\Pi(m) = z\}$ and $\phi(z)$ sums up the probabilities for a subset of $\mathcal{M}_z$.

To see that ϕ satisfies the claim, note that $\sum_{z \in \mathcal{Z}} \phi(z) f(x,y,z) = \sum_{m : m_1 = 0} \mu(m) \cdot f(x,y,\mathrm{out}_\Pi(m))$. Now, from (2), $\mu(m) \cdot f(x,y,\mathrm{out}_\Pi(m)) = \Pr_\Pi[m|x,y]$. Hence,

$$\sum_{z \in \mathcal{Z}} \phi(z) f(x,y,z) = \sum_{m : m_1 = 0} \Pr_\Pi[m|x,y] = q(x).$$

Also, for all $x \in \mathcal{X}^{(0)}, y \in \mathcal{Y}, z \in \mathcal{Z}$,

$$f^{(0)}(x,y,z) = \sum_{\substack{m: \\ \mathrm{out}_\Pi(m) = z}} \Pr_{\Pi^{(0)}}[m|x,y] = \sum_{\substack{m : m_1 = 0, \\ \mathrm{out}_\Pi(m) = z}} \frac{\Pr_\Pi[m|x,y]}{q(x)} = \sum_{\substack{m : m_1 = 0, \\ \mathrm{out}_\Pi(m) = z}} \mu(m) \frac{f(x,y,z)}{q(x)}$$

$$= \frac{f(x,y,z)}{q(x)} \phi(z).$$

□

Note that since $\mu(m)$ is the probability of the transcript being m given that the output is $\mathrm{out}_\Pi(m)$, (4) gives that for every $z \in \mathcal{Z}$, $\phi(z) = \Pr[m_1 = 0 | \mathrm{out}_\Pi(m) = z]$, i.e., the probability of the first message being 0, conditioned on the output being z. It follows from Claim 1 that a secure protocol is completely and compactly described by the values of $(\phi(z))_{z \in \mathcal{Z}}$ similarly defined in every round.

Remark 2. *If $\phi(z) = c$ for all $z \in \mathcal{Z}$, then $q(x) = c$ for all $x \in \mathcal{X}$ (because $\sum_{z \in \mathcal{Z}} f(x,y,z) = 1$ for all (x,y)). Further, if $c > 0$, $f^{(0)} = f$, and if $c < 1$, $f^{(1)} = f$. This corresponds to a protocol in which Alice sends an inconsequential first message, which neither depends on her input, nor influences the output. Hence, if Π is round-optimal, it cannot be the case that $\phi(z) = c$ for all $z \in \mathcal{Z}$.*

Note that (3) holds for every $y \in \mathcal{Y}$. Hence, we obtain

$$\sum_{z \in \mathcal{Z}} \big(f(x,y,z) - f(x,y',z)\big)\phi(z) = 0 \qquad \forall x \in \mathcal{X}, y, y' \in \mathcal{Y}. \tag{5}$$

For each $x \in \mathcal{X}$, this gives a system of $|\mathcal{Y}| - 1$ equations, by choosing a fixed $y \in \mathcal{Y}$ and all $y' \in \mathcal{Y} \setminus \{y\}$ (one may write the equations resulting from other choices of y, y' as linear combinations of these $|\mathcal{Y}| - 1$ equations).

3.2 Binary Output Alphabet

Theorem 3. *If $f : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to [0,1]$ is a simple function with $|\mathcal{Z}| = 2$, then f is securely computable.*

Proof. In fact, we can prove a stronger statement: if f is as above, then $f(x,y,z)$ is either independent of x or independent of y (or both). In that case, clearly f is securely computable by a protocol in which one party computes the output and sends it to the other party.

Since f is simple, we have $f(x,y,z) = \rho(x,z) \cdot \sigma(y,z)$, for all $x \in \mathcal{X}, y \in \mathcal{Y}, z \in \mathcal{Z}$. Since f is binary, we let $\mathcal{Z} = \{0,1\}$, and abbreviate $\rho(x,z)$ as $\rho_z(x)$ and $\sigma(y,z)$ as $\sigma_z(y)$ (for $z \in \{0,1\}$). Note that we have the following system of equations:

$$\rho_0(x)\sigma_0(y) + \rho_1(x)\sigma_1(y) = 1 \qquad \forall (x,y) \in \mathcal{X} \times \mathcal{Y} \tag{6}$$

We consider 3 cases:

Case 1: $\underline{\exists x, x' \in \mathcal{X}, x \neq x', \rho_0(x)\rho_1(x') \neq \rho_0(x')\rho_1(x).}$ In this case, one can solve the system in (6) to get two values s_0, s_1 such that $\sigma_0(y) = s_0$ and $\sigma_1(y) = s_1$ for all $y \in \mathcal{Y}$. (Concretely, $s_0 = \frac{\rho_1(x') - \rho_1(x)}{\Delta}$ and $s_1 = \frac{\rho_0(x) - \rho_0(x')}{\Delta}$, where $\Delta = \rho_0(x)\rho_1(x') - \rho_0(x')\rho_1(x) \neq 0$.) Hence $f(x,y,z) = \rho(x,z) \cdot s_z$, depends only on x.

Case 2: $\underline{\forall x, x' \in \mathcal{X}, \rho_0(x)\rho_1(x') = \rho_0(x')\rho_1(x), \text{ and } \exists x \in \mathcal{X}, \rho_0(x) = 0.}$ In this case we shall show that, $\forall x' \in \mathcal{X}$, $\rho_0(x') = 0$. Hence the function is the constant, deterministic function, with $f(x,y,0) = 0$ for all (x,y).

Let x be such that $\rho_0(x) = 0$. Since $\rho_0(x)\sigma_0(y) + \rho_1(x)\sigma_1(y) = 1$ (for any $y \in \mathcal{Y}$), we have $\rho_1(x) \neq 0$. Then, since $\forall x' \in \mathcal{X}$, $\rho_0(x)\rho_1(x') = \rho_0(x')\rho_1(x)$, we have $0 = \rho_0(x')\rho_1(x)$ which implies $\rho_0(x') = 0$, as claimed.

Case 3: $\underline{\forall x, x' \in \mathcal{X}, \rho_0(x)\rho_1(x') = \rho_0(x')\rho_1(x), \text{ and } \forall x \in \mathcal{X}, \rho_0(x) \neq 0.}$ In this case, $\forall x, x' \in \mathcal{X}$, $\frac{\rho_1(x)}{\rho_0(x)} = \frac{\rho_1(x')}{\rho_0(x')} = \theta$, say. Then, from (6), we have that $\forall x \in \mathcal{X}, y \in \mathcal{Y}$, $\rho_0(x) = \frac{1}{\sigma_0(y) + \theta\sigma_1(y)}$ and $\rho_1(x) = \frac{\theta}{\sigma_0(y) + \theta\sigma_1(y)}$. Since the RHS in these two expressions do not depend on x, there are constants r_0, r_1 such that for all $x \in \mathcal{X}$, $\rho_0(x) = r_0$ and $\rho_1(x) = r_1$. Hence $f(x,y,z) = r_z \cdot \sigma(y,z)$, depends only on y. □

3.3 Ternary Output Alphabet

To prove Theorem 1, we can focus on kernel functions, or simple symmetric functions. Let $\mathcal{Z} = \{z_1, z_2, z_3\}$. For a given symmetric function f with $\mathcal{Z}$ as its output alphabet, we define two *binary* functions $\hat{f}_i$ and f_i, for any $i \in [3]$, as follows:

1. Output alphabet of $\hat{f}_i$ is $\{z_i, z_*\}$. For every $x \in \mathcal{X}, y \in \mathcal{Y}$, we define $\hat{f}_i(x,y,z_i) := f(x,y,z_i)$ and $\hat{f}_i(x,y,z_*) := \sum_{j \neq i} f(x,y,z_j)$.

2. Output alphabet of f_i is $\mathcal{Z} \setminus \{z_i\}$. For every $x \in \mathcal{X}, y \in \mathcal{Y}$ for which $f(x,y,z_i) < 1$, we define, $f_i(x,y,z_j) := f(x,y,z_j)/(1 - f(x,y,z_i))$, for $j \in [3] \setminus \{i\}$. If $f(x,y,z_i) = 1$ for some $x \in \mathcal{X}, y \in \mathcal{Y}$, we leave $f_i(x,y,z_i)$ undefined.

Note that if for some $\tilde{x} \in \mathcal{X}$, $f(\tilde{x},y,z_i) = 1, \forall y \in \mathcal{Y}$, then we need not define f_i for this particular $\tilde{x}$, because $f(\tilde{x},y,z_j) = 0, \forall j \neq i$ and for all $y \in \mathcal{Y}$. Similarly, if for some $\tilde{y} \in \mathcal{Y}$, $f(x,\tilde{y},z_i) = 1, \forall x \in \mathcal{X}$, then we need not define f_i for this particular $\tilde{y}$, because $f(x,\tilde{y},z_j) = 0, \forall j \neq i$ and for all $x \in \mathcal{X}$. In the following, when we say that f_i is securely computable, it must be defined for all inputs.

Theorem 4. *Suppose f is simple and in normal form. Then, f is securely computable if and only if there exists an $i \in [3]$ such that both $\hat{f}_i$ and f_i are simple.*

Remark 3. *Since $\hat{f}_i$ and f_i are binary functions, being simple implies that they are functions of only one party's input (see proof of Theorem 3). Theorem 1 follows by considering the different possibilities of which party's input they can each depend on.*

Remark 4. *In the case of functions with a binary output alphabet, we proved in Subsection 3.2 that if a function is simple, it is securely computable. However, Theorem 4 lets us show that this is not true in general (see in Section 5.1).*

Proof of Theorem 4. If $|\mathcal{X}| = 1$ or $|\mathcal{Y}| = 1$, then the theorem is trivially true. So, in the following we assume that $|\mathcal{X}|, |\mathcal{Y}| \geq 2$. First we show the only if (⇒) part, and then the if (⇐) part.

⇒: Suppose f is securely computable. Since f is simple, we can write $f(x, y, z) = \rho(x, z)\sigma(y, z)$. Fix a round-optimal secure protocol Π for f. Below, we assume that Alice sends the first message in Π (the case when Bob sends the first message being symmetric). Let ϕ be as in Claim 1. Since Π is round-optimal, as discussed in Remark 2,

$$\exists i, j \in [3] \text{ s.t. } \phi(z_i) \neq \phi(z_j). \tag{7}$$

For $i \in [3]$ and $x \in \mathcal{X}$, $y, y' \in \mathcal{Y}$, we define

$$\nabla f_i^{x,y,y'} := f(x, y, z_i) - f(x, y', z_i). \tag{8}$$

It follows from (5) and (8) that, $\forall x \in \mathcal{X}, y, y' \in \mathcal{Y}$:

$$\nabla f_1^{x,y,y'} \phi(z_1) + \nabla f_2^{x,y,y'} \phi(z_2) + \nabla f_3^{x,y,y'} \phi(z_3) = 0. \tag{9}$$

Since $\sum_{z \in \mathcal{Z}} f(x, y, z) = 1$ holds for every $x \in \mathcal{X}, y \in \mathcal{Y}$, we have that $\sum_{i=1}^{3} \nabla f_i^{x,y,y'} = 0$, for every $x \in \mathcal{X}$ and $y, y' \in \mathcal{Y}$. Using this to replace $\nabla f_3^{x,y,y'}$ in (9), we can write, $\forall x \in \mathcal{X}, y, y' \in \mathcal{Y}$:

$$\nabla f_1^{x,y,y'} (\phi(z_1) - \phi(z_3)) + \nabla f_2^{x,y,y'} (\phi(z_2) - \phi(z_3)) = 0 \tag{10}$$

(and two similar equations, replacing $\nabla f_1^{x,y,y'}$ and $\nabla f_2^{x,y,y'}$, respectively).

We define a function $\mathsf{type}_f : \mathcal{X} \times \mathcal{Y} \times \mathcal{Y} \to \mathbb{T}$, which classifies (x, y, y') into one of 5 possible *types* in the set $\mathbb{T} = \{\mathsf{T}_1, \mathsf{T}_{2:1}, \mathsf{T}_{2:2}, \mathsf{T}_{2:3}, \mathsf{T}_3\}$, depending on for which all i, $\nabla f_i^{x,y,y'} = 0$:

1. $\mathsf{type}_f(x, y, y') = \mathsf{T}_1$ if $\nabla f_1^{x,y,y'} = \nabla f_2^{x,y,y'} = \nabla f_3^{x,y,y'} = 0$.

2. $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:i}$ if $\nabla f_i^{x,y,y'} = 0$ and $\forall j \in [3] \setminus \{i\}$, $\nabla f_j^{x,y,y'} \neq 0$.

3. $\mathsf{type}_f(x, y, y') = \mathsf{T}_3$ if $\forall i \in [3]$, $\nabla f_i^{x,y,y'} \neq 0$.

Note that since $\sum_{i=1}^{3} \nabla f_i^{x,y,y'} = 0$, it cannot be the case that exactly one of $\nabla f_i^{x,y,y'} \neq 0$.

We define the type of f itself as the set:

$$T(f) := \{\tau \in \mathbb{T} : \exists (x, y, y') \text{ s.t. } \mathsf{type}_f(x, y, y') = \tau\}. \tag{11}$$

We prove several claims regarding $T(f)$ before showing that $\hat{f}_i$ and f_i are simple. In proving these claims, we shall use the fact that f is simple, is in normal-form, and $|\mathcal{X}|, |\mathcal{Y}| > 1$.

Claim 2. $\mathsf{T}_3 \notin T(f)$.

Proof. For the sake of contradiction, suppose there exists (x, y, y') such that $\mathsf{type}_f(x, y, y') = \mathsf{T}_3$. Consider any $x' \in \mathcal{X} \setminus \{x\}$. From (10), we have

$$\nabla f_1^{x,y,y'}(\phi(z_1) - \phi(z_3)) + \nabla f_2^{x,y,y'}(\phi(z_2) - \phi(z_3)) = 0$$
$$\nabla f_1^{x',y,y'}(\phi(z_1) - \phi(z_3)) + \nabla f_2^{x',y,y'}(\phi(z_2) - \phi(z_3)) = 0$$

We shall show that $\nabla f_1^{x,y,y'} \nabla f_2^{x',y,y'} \neq \nabla f_2^{x,y,y'} \nabla f_1^{x',y,y'}$. Then, the above system can be uniquely solved for $\phi(z_1) - \phi(z_3)$ and $\phi(z_2) - \phi(z_3)$, to yield 0 as the solution in each case. That is, $\phi(z_1) = \phi(z_2) = \phi(z_3)$, contradicting (7).

To complete the proof, we argue that $\nabla f_1^{x,y,y'} \nabla f_2^{x',y,y'} \neq \nabla f_2^{x,y,y'} \nabla f_1^{x',y,y'}$. Suppose not. Then, $\frac{\nabla f_1^{x',y,y'}}{\nabla f_1^{x,y,y'}} = \frac{\nabla f_2^{x',y,y'}}{\nabla f_2^{x,y,y'}} = \theta$, say (the denominators being non-zero, since $\mathsf{type}_f(x, y, y') = \mathsf{T}_3$). Then, $\frac{\nabla f_3^{x',y,y'}}{\nabla f_3^{x,y,y'}} = \frac{-\nabla f_1^{x',y,y'} - \nabla f_2^{x',y,y'}}{-\nabla f_1^{x,y,y'} - \nabla f_2^{x,y,y'}} = \theta$. Invoking the simplicity of f, we get that $\forall j \in [3]$, $\frac{\rho(x,z_j)}{\rho(x',z_j)} = \theta$. However, since for any y we have $\sum_j \rho(x, z_j)\sigma(y, z_j) = \sum_j \rho(x', z_j)\sigma(y, z_j)$, we get $\theta = 1$. Then, $x \equiv x'$, contradicting the normal form of f. This completes the proof. □

Claim 3. *There can be at most one $i \in [3]$ such that $\mathsf{T}_{2:i} \in T(f)$.*

Proof. For the sake of contradiction, suppose $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:j}$, and $\mathsf{type}_f(\tilde{x}, \tilde{y}, \tilde{y}') = \mathsf{T}_{2:k}$, for $j \neq k$. We consider the case $j = 1, k = 2$, as the other cases are symmetric. Now, $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:1}$, implies that $\nabla f_i^{x,y,y'} = 0$ only for $i = 1$ and hence, by (10), we have $\nabla f_2^{x,y,y'}(\phi(z_2) - \phi(z_3)) = 0$, and so $\phi(z_2) = \phi(z_3)$. Similarly, $\mathsf{type}_f(\tilde{x}, \tilde{y}, \tilde{y}') = \mathsf{T}_{2:2}$ implies that $\phi(z_1) = \phi(z_3)$. Thus we have $\phi(z_1) = \phi(z_2) = \phi(z_3)$, contradicting (7). □

Claim 4. *If from some $x \in \mathcal{X}$ and distinct $y, y' \in \mathcal{Y}$, $\mathsf{type}_f(x, y, y') = \mathsf{T}_1$, then there exists $x' \in \mathcal{X}$ such that $\mathsf{type}_f(x', y, y') \neq \mathsf{T}_1$.*

Proof. Since $\mathsf{type}_f(x, y, y') = \mathsf{T}_1$, we have that for all $j \in [3]$, $f(x, y, z_j) = f(x, y', z_j)$. Since f is in normal form, $y \not\equiv y'$, and hence there exists $x' \in \mathcal{X}$ such that for some $j \in [3]$, $f(x', y, z_j) \neq f(x', y', z_j)$. Hence $\mathsf{type}_f(x', y, y') \neq \mathsf{T}_1$. □

Claim 5. *Suppose $T(f) = \{\mathsf{T}_1, \mathsf{T}_{2:i}\}$ for some $i \in [3]$. Then, if for some $x \in \mathcal{X}$, and distinct $y, y' \in \mathcal{Y}$, $\mathsf{type}_f(x, y, y') = \mathsf{T}_1$, then for all $\tilde{y} \in \mathcal{Y}$, $f(x, \tilde{y}, z_i) = 1$.*

Proof. By Claim 4, we have x' such that $\mathsf{type}_f(x', y, y') \neq \mathsf{T}_1$; since $T(f) = \{\mathsf{T}_1, \mathsf{T}_{2:i}\}$, we have $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:i}$. Then, for both values of $j \neq i$ in $[3]$, we have $f(x', y, z_j) \neq f(x', y', z_j)$.

Invoking the simplicity of f, we have that for $j \neq i$, $\rho(x', z_j)(\sigma(y, z_j) - \sigma(y', z_j)) \neq 0$, but $\rho(x, z_j)(\sigma(y, z_j) - \sigma(y', z_j)) = 0$. Hence $\rho(x, z_j) = 0$ for $j \neq i$. That is, for all $\tilde{y} \in \mathcal{Y}$, $f(x, \tilde{y}, z_j) = 0$ for $j \neq i$ and hence $f(x, \tilde{y}, z_i) = 1$. □

From the above claims, we have two possibilities for $T(f)$: either $\{\mathsf{T}_{2:i}\}$ or $\{\mathsf{T}_1, \mathsf{T}_{2:i}\}$ for some $i \in [3]$. Then we shall prove that $\hat{f}_i$ and f_i are simple. Note that both these functions are binary functions. For our case where f has a minimal-round protocol with Alice sending the first bit, and with $|\mathcal{X}|, |\mathcal{Y}| > 1$, we show that this means that $\hat{f}_i$ is a function of only Alice's input, and f_i is a function of only Bob's input.

Claim 6. *If $T(f) \subseteq \{\mathsf{T}_1, \mathsf{T}_{2:i}\}$, then $\hat{f}_i$ is simple.*

Proof. We have that for all (x, y, y'), $\nabla f_i^{x,y,y'} = 0$, i.e., $\rho(x, z_i)(\sigma(y, z_i) - \sigma(y', z_i)) = 0$. Now, if $\rho(x, z_i) = 0$ for all x, then $\hat{f}_i$ is a constant function with $\hat{f}_i(x, y, z_i) = 0$ and $\hat{f}_i(x, y, z_*) = 1$. Otherwise, there is some $x \in \mathcal{X}$ such that $\rho(x, z_i) \neq 0$. Hence for all y, y' we have $\sigma(y, z_i) = \sigma(y', z_i) = s$, say. Then $\hat{f}_i(x, y, z_i) = \rho(x, z_i) \cdot s$, which is independent of y. Thus, in either case, $\hat{f}_i$ is simple. □

Claim 7. *If $T(f) \subseteq \{\mathsf{T}_1, \mathsf{T}_{2:i}\}$, then $f_i(x, y, z)$ is simple.*

Proof. Recall that f_i has input spaces $\mathcal{X}_i$ and $\mathcal{Y}$, where $\mathcal{X}_i = \{x \in \mathcal{X} : \exists y \in \mathcal{Y} \text{ s.t. } f(x, y, z_i) < 1\}$. We shall prove that for all $x, x' \in \mathcal{X}_i$ and $y \in \mathcal{Y}$, $f_i(x, y, z_j) = f_i(x', y, z_j)$ for $j \neq i$.

By Claim 5, for all $x \in \mathcal{X}_i$ and distinct $y, y' \in \mathcal{Y}$, $\mathsf{type}_f(x, y, y') \neq \mathsf{T}_1$, and hence $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:i}$. Therefore, for all $x \in \mathcal{X}_i$ and $j \neq i$, $\rho(x, z_j) \neq 0$.

Let $\{j, \bar{j}\} = [3] \setminus \{i\}$. Now,

$$f_i(x, y, z_j) = \frac{f(x, y, z_j)}{1 - f(x, y, z_i)} = \frac{\rho(x, z_j)\sigma(y, z_j)}{\rho(x, z_j)\sigma(y, z_j) + \rho(x, z_{\bar{j}})\sigma(y, z_{\bar{j}})} = \frac{\sigma(y, z_j)}{\sigma(y, z_j) + \gamma(x)\sigma(y, z_{\bar{j}})},$$

where $\gamma(x) := \frac{\rho(x, z_{\bar{j}})}{\rho(x, z_j)}$. Here we have used the fact that $\rho(x, z_j) \neq 0$ for all $x \in \mathcal{X}_i$. Thus to prove the claim, it is enough to show that $\gamma(x) = \gamma(x')$ for all $x, x' \in \mathcal{X}_i$.

For any $x \in \mathcal{X}_i$, as mentioned above, for any distinct $y, y' \in \mathcal{Y}$, we have $\mathsf{type}_f(x, y, y') = \mathsf{T}_{2:i}$, which implies that $\nabla f_j^{x,y,y'} + \nabla f_{\bar{j}}^{x,y,y'} = -\nabla f_i^{x,y,y'} = 0$. That is,

$$\rho(x, z_j)(\sigma(y, z_j) - \sigma(y', z_j)) + \rho(x, z_{\bar{j}})(\sigma(y, z_{\bar{j}}) - \sigma(y', z_{\bar{j}})) = 0.$$

Writing the above equation for any $x, x' \in \mathcal{X}_i$, if $\rho(x, z_j)\rho(x', z_{\bar{j}}) \neq \rho(x, z_{\bar{j}})\rho(x', z_j)$, we will be able to solve that $(\sigma(y, z_j) - \sigma(y', z_j)) = (\sigma(y, z_{\bar{j}}) - \sigma(y', z_{\bar{j}})) = 0$. But this implies that $\nabla f_j^{x,y,y'} = \nabla f_{\bar{j}}^{x,y,y'} = 0$ (for any $x \in \mathcal{X}_i$), contradicting the fact that $\mathsf{type}_f(x, y, y') \neq \mathsf{T}_1$ for all $x \in \mathcal{X}_i$ and distinct $y, y' \in \mathcal{Y}$. Thus we should have $\rho(x, z_j)\rho(x', z_{\bar{j}}) = \rho(x, z_{\bar{j}})\rho(x', z_j)$, or (dividing by $\rho(x, z_j) \cdot \rho(x', z_j) \neq 0$), $\gamma(x) = \gamma(x')$. □

Taken together, the above claims prove the only if ($\Rightarrow$) part of Theorem 4.

$\Leftarrow$: Let $i \in [3]$ be such that $\hat{f}_i$ and f_i are simple, and therefore, securely computable. A 2-round secure protocol for f is as follows. If both $\hat{f}_i$ and f_i are independent of x or y (or both), then clearly f is securely computable by a protocol in which one party computes the output and sends it to the other party. The only interesting case is when $\hat{f}_i$ is independent of y and f_i is independent of x (the other case when $\hat{f}_i$ is independent of x and f_i is independent of y being symmetric): Alice picks $j \in \{i, *\}$ with probability $\hat{f}_i(x, y_1, z_j)$ and sends j to Bob. If $j = i$, both Alice and Bob output z_i with probability 1; otherwise, Bob picks $k \in [3] \setminus \{i\}$ with probability $f_i(x_1, y, z_k)$ and sends k to Alice. Now both Alice and Bob output z_k with probability 1. It is clear from the definitions of $\hat{f}_i$ and f_i that the output from this protocol is correctly distributed. It is easy to see that this is a unique-transcript protocol, and therefore, is perfectly private. □

4 Functions with 2-Round Secure Protocols

We prove the following theorem which, when applied to the kernel of a given function, implies Theorem 2.

Theorem 5. *A simple symmetric function $p_{Z|XY}$ with X, Y, Z over alphabets $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$, has a two-round protocol with Alice making the first move iff there exists a surjective map $g : \mathcal{Z} \to \mathcal{W}$ to some set $\mathcal{W}$ and probability distributions $p_{W|X}$ and $p_{Z|WY}$ where W is over the alphabet $\mathcal{W}$, such that $p_{Z|WY}(z|w, y) = 0$ if $w \neq g(z)$, and for all $x \in \mathcal{X}, y \in \mathcal{Y}, z \in \mathcal{Z}$,*

$$p_{Z|XY}(z|x, y) = p_{W|X}(g(z)|x) \cdot p_{Z|WY}(z|g(z), y).$$

In that case, $p_{Z|XY}$ has a unique-transcript secure protocol in which Alice sends w sampled according to $p_{W|X=x}$ and Bob sends back z according to $p_{Z|W=w,Y=y}$.

Proof. It is easy to see that if $g, p_{W|X}, p_{Z|WY}$ as in the statement exist, then the protocol described is indeed a unique-transcript protocol for $p_{Z|XY}$: its output is distributed correctly, and since $p_{Z|WY}(z|w, y) = 0$ if $w \neq g(z)$, the only transcript resulting in the output z is of the form $(g(z), z)$. A unique-transcript protocol is always a secure protocol since the transcript can be simulated from the output by a (deterministic) simulator.

We prove the other direction below. Suppose we are given a two-round protocol Π_0 for $p_{Z|XY}$, with the two messages denoted by a and b. Then, we can construct a secure protocol Π in which Bob computes the second message b as before, but sends out $z = \mathrm{out}_{\Pi_0}(a, b)$, and both Alice and Bob output z: clearly Π has the same output as Π_0 and is also secure since the transcript of Π can be simulated from a (simulated) transcript of Π_0 by applying a deterministic function to it.

Π is defined by probability distributions $\Pr_\Pi[a|x]$ and $\Pr_\Pi[z|a, y]$. For convenience, we define $\alpha(a, x) := \Pr_\Pi[a|x]$ and $\beta(a, z, y) := \Pr_\Pi[z|a, y]$. Also, since $p_{Z|XY}$ is simple, let us write $p_{Z|XY}(z|x, y) = \rho(x, z) \cdot \sigma(y, z)$.

Before proceeding further, note that for a transcript $m = (a, z)$, from (2), we have

$$\Pr_\Pi[m|x, y] = \mu(m)\rho(x, z)\sigma(y, z) = \alpha(a, x)\beta(a, z, y).$$

If $\Pr_\Pi[m|x, y] > 0$ for some x, y, then by considering the above equality for (x, y) as well as (x', y), and dividing the latter by the former (which is non-zero), we get that for all $x' \in \mathcal{X}$,

$$\frac{\rho(x', z)}{\rho(x, z)} = \frac{\alpha(a, x')}{\alpha(a, x)}. \tag{12}$$

We define an equivalence relation $\equiv$ over $\mathcal{Z}$ as follows:

$$z_1 \equiv z_2 \text{ if } \exists c > 0, \forall x \in \mathcal{X}, \rho(x, z_1) = c\rho(x, z_2).$$

We let $\mathcal{W}$ be the set of equivalence classes of $\equiv$, and define $g : \mathcal{Z} \to \mathcal{W}$ which maps z to its equivalence class. Thus, $z_1 \equiv z_2$ iff $g(z_1) = g(z_2)$. We also define a function h that maps the first message a in a transcript to an element in $\mathcal{W}$, as follows:

$$h(a) = g(z) \text{ if } \exists x, y \text{ s.t. } \Pr_\Pi[a, z|x, y] > 0.$$

For h to be well-defined, we need that each a has a unique value $h(a)$ that satisfies the above condition. Suppose $z_1, z_2 \in \mathcal{Z}$ are such that $\Pr_\Pi[a, z_1|x_1, y_1] > 0$ and $\Pr_\Pi[a, z_2|x_2, y_2] > 0$ (from some $(x_1, y_1), (x_2, y_2) \in \mathcal{X} \times \mathcal{Y}$). By applying (12) to these, we get that for all $x' \in \mathcal{X}$

$$\frac{\rho(x', z_1)}{\rho(x_1, z_1)} = \frac{\alpha(a, x')}{\alpha(a, x_1)} \quad \text{and} \quad \frac{\rho(x', z_2)}{\rho(x_2, z_2)} = \frac{\alpha(a, x')}{\alpha(a, x_2)}.$$

Hence $\forall x' \in \mathcal{X}$, $\rho(x', z_2) = c\rho(x', z_1)$, where $c := \frac{\rho(x_2,z_2)\alpha(a,x_1)}{\alpha(a,x_2)\rho(x_1,z_1)}$. All the factors in c are positive (as they appear in $\Pr_\Pi[a, z_1|x_1, y_1] \cdot \Pr_\Pi[a, z_2|x_2, y_2] > 0$), and also independent of x'. Thus $z_1 \equiv z_2$ and $g(z_1) = g(z_2)$, making $h(a)$ well defined.

$p_{W|X}$ is defined as follows: given x, sample a as in Π, and output $w = h(a)$. That is, $p_{W|X}(w|x) = \sum_{a:h(a)=w} \alpha(a, x)$. Finally, we define $p_{Z|WY}$ as follows: given w, we argue that we can reverse sample a without access to x (so that $w = h(a)$), and then use the protocol Π to sample z from (a, y). That is, we define a distribution over a given w, by the probability

$$\eta(a, w) := \begin{cases} \frac{\alpha(a,x)}{\sum_{a':h(a')=w} \alpha(a',x)} & \text{if } h(a) = w \\ 0 & \text{otherwise,} \end{cases}$$

where *any* $x \in \mathcal{X}$ such that $\sum_{a':h(a')=w} \alpha(a', x) > 0$ is used. This is well defined because, by (12), switching from x to x' amounts to multiplying both the numerator and the denominator by the same factor (namely, $\frac{\rho(x',z)}{\rho(x,z)}$ for any $z \in g^{-1}(w)$). Then, $p_{Z|WY}(z|w, y) = \sum_a \eta(a, w)\beta(a, z, y)$. We verify that, when $w = g(z)$,

$$\begin{aligned} p_{W|X}(w|x) \cdot p_{Z|WY}(z|w, y) &= \left(\sum_{a:h(a)=w} \alpha(a, x) \right) \left(\sum_a \eta(a, w)\beta(a, z, y) \right) \\ &= \sum_a \alpha(a, x)\beta(a, z, y) = \Pr_\Pi[z|x, y] = p_{Z|XY}(z|x, y). \end{aligned}$$

□

5 Complexity of Randomized Functions

We point out a couple of complexity aspects in which randomized functions differ from deterministic functions. This also points to the difficulty in characterization of securely computable function in the case of randomized functions.

5.1 Smaller Simple Functions which are not Securely Computable

It is well-known that simple functions are not all securely computable, even for deterministic functions, with a first example given by Beaver [Bea89], with an output alphabet of size 5. Without overloading the output symbols (see Remark 1), this turns out to be the smallest output alphabet for a simple deterministic function that is not securely realizable. (With symbol-overloading, the smallest possible alphabet size for a simple deterministic function that is not securely computable is 4, as shown in [MPR13].)

But for randomized functions, we see a higher level of complexity arising even with an output size of 3. In Figure 1 we show an example of a simple function with ternary output alphabet (without resorting to symbol-overloading) that is not securely computable, i.e., it does not satisfy the characterization given in Theorem 4.

5.2 Limits of Unique-Transcript Protocols

It follows from Section 3 that all securely computable randomized functions with a ternary output kernel can be computed using unique-transcript protocols. Also, it follows from Section 4 that all randomized functions securely computable by two-round protocols are in fact securely computable using two-round unique-transcript protocols. Also, the characterization by Kushilevitz [Kus89] showed that all securely computable

		y_1			y_2		
		z_1	z_2	z_3	z_1	z_2	z_3
	z_1	2/9			5/18		
x_1	z_2		4/9			2/9	
	z_3			1/3			1/2
	z_1	1/3			5/12		
x_2	z_2		5/12			5/24	
	z_3			1/4			3/8

Figure 1 This function is not securely computable as it does not satisfy the condition from Theorem 4. However, it is simple, with the functions ρ and σ given by

	z_1	z_2	z_3
x_1	1/3	2/3	1
x_2	1/2	5/8	3/4

and

	z_1	z_2	z_3
y_1	2/3	2/3	1/3
y_2	5/6	1/3	1/2

.

deterministic functions have unique-transcript secure protocols. Thus one may reasonably suspect that all securely computable functions have unique-transcript secure protocols.

However, we show that in some sense, the above results give the limits of unique-transcript protocols: If we go just beyond the above conditions – namely, ternary output, 2-round computable, deterministic – then we can indeed find securely computable functions that do not have any unique-transcript secure protocol. In Appendix B, we demonstrate a simple randomized function with an output alphabet of size 4, securely computable by a 3-round protocol, such that it has no unique-transcript secure protocols.

Acknowledgments

Deepesh Data's research was supported in part by the Microsoft Research India Ph.D. Fellowship. Manoj Prabhakaran wishes to thank Hemanta Maji and Amit Sahai for an earlier collaboration on the same problem. The observation in Section 5.2 was made (using a different example) during that work as well.

References

[Bea89] Donald Beaver. Perfect privacy for two-party protocols. In Joan Feigenbaum and Michael Merritt, editors, *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 65–77. American Mathematical Society, 1989. 1, 2, 13

[Dat16] Deepesh Data. Secure computation of randomized functions. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 3053–3057, 2016. 4

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31. ACM, 1988. 1

[Kil00] Joe Kilian. More general completeness theorems for secure two-party computation. In *Proc. 32th STOC*, pages 316–324. ACM, 2000. 4

[KMPS14] Daniel Kraschewski, Hemanta K. Maji, Manoj Prabhakaran, and Amit Sahai. A full characterization of completeness for two-party randomized function evaluation. In *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, pages 659–676, 2014. 1

[KMQR09] Robin Künzler, Jörn Müller-Quade, and Dominik Raub. Secure computability of functions in the IT setting with dishonest majority and applications to long-term security. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 238–255. Springer, 2009. 4

[Kus89] Eyal Kushilevitz. Privacy and communication complexity. In *FOCS*, pages 416–421. IEEE, 1989. 1, 2, 4, 6, 13

[MPR09] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Complexity of multi-party computation problems: The case of 2-party symmetric secure function evaluation. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2009. 4

[MPR12] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A unified characterization of completeness and triviality for secure function evaluation. In *Progress in Cryptology - INDOCRYPT 2012*, pages 40–59, 2012. 1, 2, 5

[MPR13] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Complexity of multi-party computation functionalities. In Manoj Prabhakaran and Amit Sahai, editors, *Secure Multi-Party Computation*, volume 10 of *Cryptology and Information Security Series*, pages 249–283. IOS Press, 2013. 2, 4, 13

A Security for Simple Symmetric Functions

In this section we prove Lemma 1.

Proof of Lemma 1. Firstly, by perfect correctness, for any x, y, we require that with probability one, $\mathrm{out}^A_\Pi(m, x) = \mathrm{out}^B_\Pi(m, y)$, where m is produced by Π on input (x, y). Suppose if there is an $x_0 \in \mathcal{X}$, $y_0 \in \mathcal{Y}$ such that $\Pr_\Pi[m|x_0, y_0] > 0$.

Since we can write $\Pr_\Pi[m|x, y] = \alpha(m, x)\beta(m, y)$, we have $\alpha(m, x_0)\beta(m, y_0) > 0$. Also, for all x such that $\alpha(m, x) > 0$, we have a positive probability of Π producing m on input (x, y_0) and hence we must have $\mathrm{out}^A_\Pi(m, x) = \mathrm{out}^B_\Pi(m, y_0)$ with probability 1 (which requires them to be deterministic). Similarly, for all y such that $\beta(m, y) > 0$, we have $\mathrm{out}^B_\Pi(m, y) = \mathrm{out}^A_\Pi(m, x_0)$ with probability 1. Letting $\mathrm{out}_\Pi(m) := \mathrm{out}^A_\Pi(m, x_0) = \mathrm{out}^B_\Pi(m, y_0)$ (which must be deterministic), we have that $\mathrm{out}^A_\Pi(m, x) = \mathrm{out}^B_\Pi(m, y) = \mathrm{out}_\Pi(m)$ if $\Pr_\Pi[m|x, y] > 0$.

Now we prove the second part. Note that since Π computes $p_{Z|XY}$, we have that for all x, y, z, $\Pr_\Pi[z|x, y] = p_{Z|XY}(z|x, y)$. Consider any x, y, z such that $p_{Z|XY}(z|x, y) > 0$. For all m such that $\mathrm{out}_\Pi(m, x) \neq z$, we can set $S(m, z) = 0$. So, suppose $\mathrm{out}_\Pi(m) = z$. Then

$$\Pr_\Pi[m|x, y, z] = \frac{\Pr_\Pi[m, z|x, y]}{\Pr_\Pi[z|x, y]} = \frac{\Pr_\Pi[m|x, y]}{p_{Z|XY}(z|x, y)} = \frac{\alpha(m, x)\beta(m, y)}{\rho(z, x)\sigma(z, y)},$$

where we wrote $\Pr_\Pi[m, z|x, y] = \Pr_\Pi[m|x, y]$ (since $z = \mathrm{out}_\Pi(m)$), $\Pr_\Pi[m|x, y] = \alpha(m, x)\beta(m, y)$ (since Π is a protocol) and $p_{Z|XY}(z|x, y) = \rho(z, x)\sigma(z, y)$ (since $p_{Z|XY}$ is a simple function). Using the security guarantee for a symmetric function (Definition 1, with $z_A = z_B = z$), we get

$$S_1(m, x, z) = S_2(m, y, z) = \frac{\alpha(m, x)}{\rho(z, x)} \cdot \frac{\beta(m, y)}{\sigma(z, y)}.$$

		y_1				y_2			
		z_1	z_2	z_3	z_4	z_1	z_2	z_3	z_4
x_1	z_1	2/9				5/18			
	z_2		31/108				31/216		
	z_3			17/108				17/216	
	z_4				1/3				1/2
x_2	z_1	1/3				5/12			
	z_2		31/360				31/720		
	z_3			119/360				119/720	
	z_4				1/4				3/8

Figure 2 This simple randomized function $p_{Z|XY}$ is securely computable by a 3 round protocol (given in Figure 3) that is not unique-transcript, but cannot be securely computed using any unique-transcript protocol (with any number of rounds).

Now, fixing (m, z) as above, consider all (x, y) such that $p_{Z|XY}(z|x, y) > 0$. If the above expression is 0 for all choices of (x, y), then we can simply set $S(m, z) = 0$. Otherwise, there is some x such that $S_1(m, x, z) \neq 0$. Then, considering the above expression for that x, we get that $\frac{\beta(m,y)}{\sigma(z,y)}$ equals a quantity that is independent of y (and hence is a function of (m, z) alone). Similarly, by considering $S_2(m, y, z)$, we get that $\frac{\alpha(m,x)}{\rho(z,x)}$ is a function of (m, z) alone. Hence we have

$$S_1(m, x, z) = S_2(m, y, z) = S(m, z).$$

□

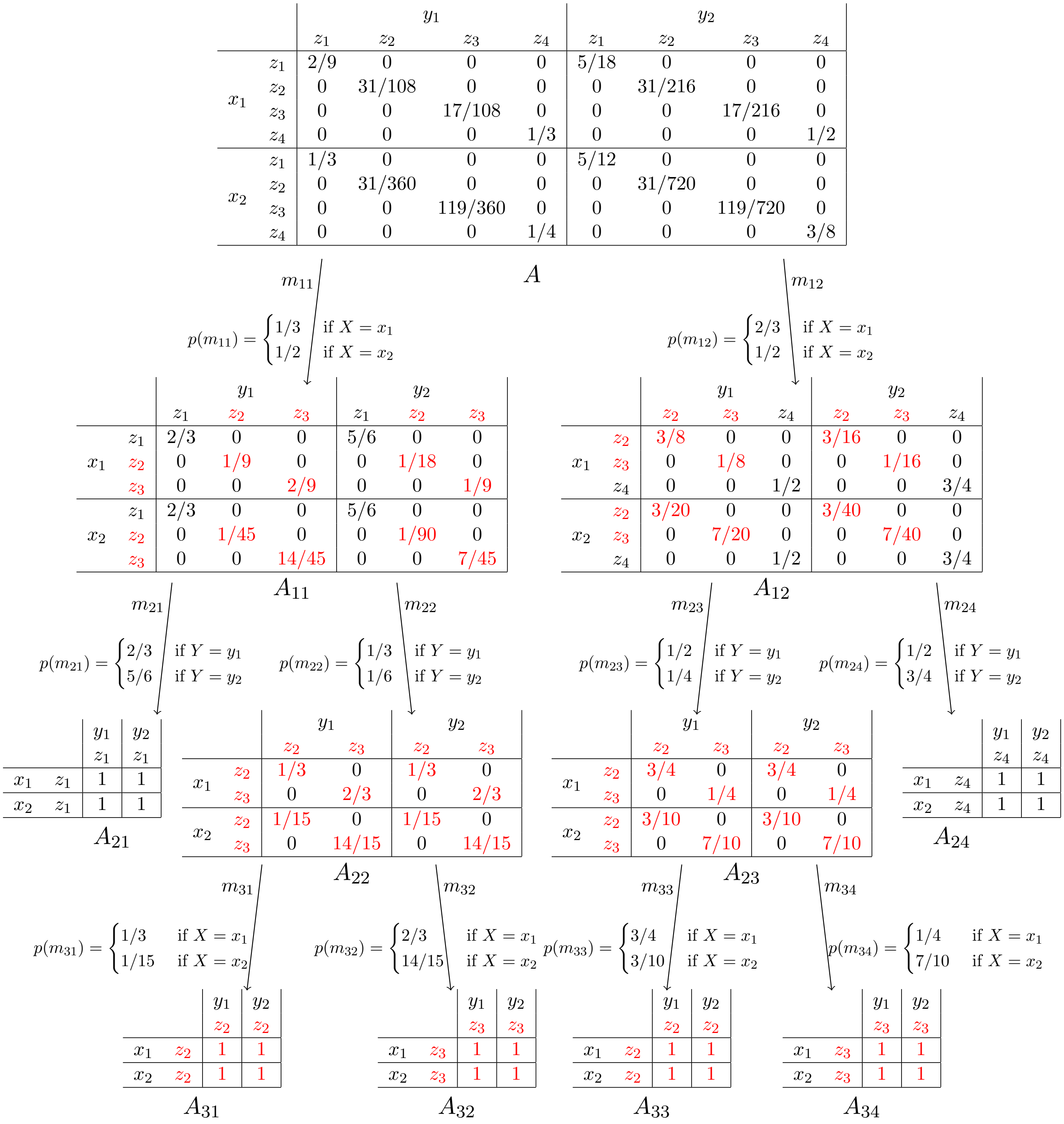
B Example for Section 5.2

Theorem 6. *There exists a randomized function that can be securely computed using a 3-round protocol, but cannot be securely computed using a unique-transcript protocol with any number of rounds.*

Proof. Consider the simple randomized function $p_{Z|XY}$ given in Figure 2. This can be securely computed, and a 3-round protocol for that is given in Figure 3. Note that this protocol is not unique-transcript. First we show that the protocol given in Figure 3 is secure, i.e., it is correct and perfectly private; and later we show that no unique-transcript protocol can securely compute this function with any number of rounds.

Correctness: It follows from the fact that for every x, y, z, $p_{Z|XY}(z|x, y)$ is equal to the sum of the probabilities on different paths leading to leaves labelled as z, where probability of a path is equal to the product of the probabilities (corresponding to the particular x and y) appearing on the edges along that path.

Privacy: Consider an arbitrary $k \in [4]$. We need to show that for any transcript m, $p(m|x_i, y_j, z_k)$ must be the same for every $i, j \in [2]$, $k \in [4]$. This trivially holds for every $z \in \mathcal{Z} \setminus \{z_2, z_3\}$, because there is a unique path from root to the leaf corresponding to z, which means that output being z itself determines the whole transcript. But for z_2 and z_3 there are two possible transcripts. Fix any $z \in \{z_2, z_3\}$, say, z_2; similar argument holds for $z = z_3$ as well. There are two transcripts (m_{11}, m_{22}, m_{31}) and (m_{12}, m_{23}, m_{33}) for z_2, implying two distinct paths. In order to show that the protocol is perfectly private, we need to show that $p(m_{11}, m_{22}, m_{31}|x_i, y_j, z_2)$ is same for all $i, j \in [2]$. It can be easily verified that this is indeed the case, which implies that the protocol described in Figure 3 is perfectly private.



A

		y_1				y_2			
		z_1	z_2	z_3	z_4	z_1	z_2	z_3	z_4
x_1	z_1	2/9	0	0	0	5/18	0	0	0
	z_2	0	31/108	0	0	0	31/216	0	0
	z_3	0	0	17/108	0	0	0	17/216	0
	z_4	0	0	0	1/3	0	0	0	1/2
x_2	z_1	1/3	0	0	0	5/12	0	0	0
	z_2	0	31/360	0	0	0	31/720	0	0
	z_3	0	0	119/360	0	0	0	119/720	0
	z_4	0	0	0	1/4	0	0	0	3/8

$$p(m_{11}) = \begin{cases} 1/3 & \text{if } X = x_1 \\ 1/2 & \text{if } X = x_2 \end{cases}$$

$$p(m_{12}) = \begin{cases} 2/3 & \text{if } X = x_1 \\ 1/2 & \text{if } X = x_2 \end{cases}$$

A_{11}

		y_1			y_2		
		z_1	z_2	z_3	z_1	z_2	z_3
x_1	z_1	2/3	0	0	5/6	0	0
	z_2	0	1/9	0	0	1/18	0
	z_3	0	0	2/9	0	0	1/9
x_2	z_1	2/3	0	0	5/6	0	0
	z_2	0	1/45	0	0	1/90	0
	z_3	0	0	14/45	0	0	7/45

A_{12}

		y_1			y_2		
		z_2	z_3	z_4	z_2	z_3	z_4
x_1	z_2	3/8	0	0	3/16	0	0
	z_3	0	1/8	0	0	1/16	0
	z_4	0	0	1/2	0	0	3/4
x_2	z_2	3/20	0	0	3/40	0	0
	z_3	0	7/20	0	0	7/40	0
	z_4	0	0	1/2	0	0	3/4

$$p(m_{21}) = \begin{cases} 2/3 & \text{if } Y = y_1 \\ 5/6 & \text{if } Y = y_2 \end{cases}$$

$$p(m_{22}) = \begin{cases} 1/3 & \text{if } Y = y_1 \\ 1/6 & \text{if } Y = y_2 \end{cases}$$

$$p(m_{23}) = \begin{cases} 1/2 & \text{if } Y = y_1 \\ 1/4 & \text{if } Y = y_2 \end{cases}$$

$$p(m_{24}) = \begin{cases} 1/2 & \text{if } Y = y_1 \\ 3/4 & \text{if } Y = y_2 \end{cases}$$

A_{21}

		y_1	y_2
		z_1	z_1
x_1	z_1	1	1
x_2	z_1	1	1

A_{22}

		y_1		y_2	
		z_2	z_3	z_2	z_3
x_1	z_2	1/3	0	1/3	0
	z_3	0	2/3	0	2/3
x_2	z_2	1/15	0	1/15	0
	z_3	0	14/15	0	14/15

A_{23}

		y_1		y_2	
		z_2	z_3	z_2	z_3
x_1	z_2	3/4	0	3/4	0
	z_3	0	1/4	0	1/4
x_2	z_2	3/10	0	3/10	0
	z_3	0	7/10	0	7/10

A_{24}

		y_1	y_2
		z_4	z_4
x_1	z_4	1	1
x_2	z_4	1	1

$$p(m_{31}) = \begin{cases} 1/3 & \text{if } X = x_1 \\ 1/15 & \text{if } X = x_2 \end{cases}$$

$$p(m_{32}) = \begin{cases} 2/3 & \text{if } X = x_1 \\ 14/15 & \text{if } X = x_2 \end{cases}$$

$$p(m_{33}) = \begin{cases} 3/4 & \text{if } X = x_1 \\ 3/10 & \text{if } X = x_2 \end{cases}$$

$$p(m_{34}) = \begin{cases} 1/4 & \text{if } X = x_1 \\ 7/10 & \text{if } X = x_2 \end{cases}$$

A_{31}

		y_1	y_2
		z_2	z_2
x_1	z_2	1	1
x_2	z_2	1	1

A_{32}

		y_1	y_2
		z_3	z_3
x_1	z_3	1	1
x_2	z_3	1	1

A_{33}

		y_1	y_2
		z_2	z_2
x_1	z_2	1	1
x_2	z_2	1	1

A_{34}

		y_1	y_2
		z_3	z_3
x_1	z_3	1	1
x_2	z_3	1	1

Figure 3 This describes a 3-round protocol for securely computing $p_{Z|XY}$, denoted by matrix A, where the first message is sent by Alice. This protocol is not unique-transcript: both A_{11} and A_{12} have z_2, z_3 in common, and that is highlighted in red color. The meaning of the probabilities on the edges is as follows: If Alice's input is x_1, then she sends m_{11} as the first message with probability 1/3 and m_{12} as the first message with probability 2/3. If Alice's input is x_2, then she sends m_{11} as the first message with probability 1/2 and m_{12} as the first message with probability 1/2. If Alice sends m_{11}, then the problem reduces to securely computing A_{11}, and if Alice sends m_{12}, then the problem reduces to securely computing A_{12}. Suppose Alice reduces the problem to A_{11}. Now it is Bob's turn to send a message. If Bob's input is y_1, he sends m_{21} with probability 2/3 and m_{22} with probability 1/3, and so on ... In the end, at the leaf nodes there is only one possible z_i to output, and they output that element with probability 1.

Now we show that no *unique-transcript* protocol (with any number of rounds) can securely compute $A := p_{Z|XY}$. Note that in a unique-transcript protocol, during any round, the party who is sending a message makes a partition of the output alphabet, and sends the other party the part (i.e., the reduced output alphabet) in which the output should lie. We show below that neither Alice nor Bob can make a partition in the first round itself. This implies that no unique-transcript protocol exists for securely computing A with any number of rounds.

- **Alice cannot partition $\mathcal{Z}$:** Suppose, to the contrary, that Alice can partition $\mathcal{Z}$ in the first round; and, assume, w.l.o.g., that Alice makes two parts $\mathcal{Z} = \mathcal{Z}_1 \uplus \mathcal{Z}_2$.

  Let $m_i,\ i = 1, 2$ denote the message that Alice sends to Bob in order to restrict the output to $\mathcal{Z}_i,\ i = 1, 2$. This implies that $p_{M_1|XYZ}(m_1|x_i, y_j, z \in \mathcal{Z}_1) = 1$ and $p_{M_1|XYZ}(m_2|x_i, y_j, z \in \mathcal{Z}_2) = 1$, for every $i, j \in \{1, 2\}$. We show below that if Alice partitions $\mathcal{Z} = \mathcal{Z}_1 \uplus \mathcal{Z}_2$, then the following must hold for every $i \in \{1, 2\}$:

$$\sum_{z\in\mathcal{Z}_1} p_{Z|XY}(z|x_i, y_1) = \sum_{z\in\mathcal{Z}_1} p_{Z|XY}(z|x_i, y_2), \tag{13}$$

$$\sum_{z\in\mathcal{Z}_2} p_{Z|XY}(z|x_i, y_1) = \sum_{z\in\mathcal{Z}_2} p_{Z|XY}(z|x_i, y_2). \tag{14}$$

  It can be easily verified that the matrix A does not satisfy the above two conditions for any non-trivial and disjoint $\mathcal{Z}_1$, $\mathcal{Z}_2$, which is a contradiction: since $\mathcal{Z} = \mathcal{Z}_1 \uplus \mathcal{Z}_2$ and $|\mathcal{Z}| = 4$, one of the following must hold: (i) either $|\mathcal{Z}_1|$=1 or $|\mathcal{Z}_2|$=1, or (ii) either $|\mathcal{Z}_1|$=2 or $|\mathcal{Z}_2|$=2. This verification can be done easily even exhaustively. We show (13) and (14) below. In the following, i belongs to $\{1, 2\}$.

$$\begin{aligned} p_{M_1|XY}(m_1|x_i, y_1) &= \sum_{z\in\mathcal{Z}_1} p_{M_1 Z|XY}(m_1, z|x_i, y_1) + \sum_{z\in\mathcal{Z}_2} p_{M_1 Z|XY}(m_1, z|x_i, y_1) \\ &= \sum_{z\in\mathcal{Z}_1} p_{Z|XY}(z|x_i, y_1) \underbrace{p_{M_1|XYZ}(m_1|x_i, y_1, z)}_{=1} \\ &\qquad\qquad + \sum_{z\in\mathcal{Z}_2} p_{Z|XY}(z|x_i, y_1) \underbrace{p_{M_1|XYZ}(m_1|x_i, y_1, z)}_{=0} \\ &= \sum_{z\in\mathcal{Z}_1} p_{Z|XY}(z|x_i, y_1) \end{aligned} \tag{15}$$

  Similarly we can show the following:

$$p_{M_1|XY}(m_1|x_i, y_2) = \sum_{z\in\mathcal{Z}_1} p_{Z|XY}(z|x_i, y_2), \tag{16}$$

$$p_{M_1|XY}(m_2|x_i, y_1) = \sum_{z\in\mathcal{Z}_2} p_{Z|XY}(z|x_i, y_1), \tag{17}$$

$$p_{M_1|XY}(m_2|x_i, y_2) = \sum_{z\in\mathcal{Z}_2} p_{Z|XY}(z|x_i, y_2). \tag{18}$$

  Since Alice sends the first message, which means that the Markov chain $M_1 - X - Y$ holds. This implies that $p_{M_1|XY}(m_j|x_i) = p_{M_1|XY}(m_j|x_i, y_1) = p_{M_1|XY}(m_j|x_i, y_2)$ for every $j \in \{1, 2\}$. Now comparing (15) & (16) gives (13), and (17) & (18) gives (14).

- **Bob cannot partition $\mathcal{Z}$:** Switching the roles of Alice and Bob with each-other in the above argument and using the fact that for every partition $\mathcal{Z} = \mathcal{Z}_1 \uplus \mathcal{Z}_2$, the matrix A does not satisfy the following two conditions, we can prove that Bob also cannot partition $\mathcal{Z}$. In the following, i belongs to $\{1, 2\}$.

$$\sum_{z \in \mathcal{Z}_1} p_{Z|XY}(z|x_1, y_i) = \sum_{z \in \mathcal{Z}_1} p_{Z|XY}(z|x_2, y_i),$$
$$\sum_{z \in \mathcal{Z}_2} p_{Z|XY}(z|x_1, y_i) = \sum_{z \in \mathcal{Z}_2} p_{Z|XY}(z|x_2, y_i).$$

This completes the proof of Theorem 6. □