# Fairness in Two-Party Computation

## Characterizing Fair Functions

Nikolaos Makriyannis

TESI DOCTORAL UPF / 2016

Directora de la tesi

Prof. Vanesa Daza,
Department of Information and Communication Technologies

Universitat Pompeu Fabra
Barcelona

The PhD committee was appointed by the Rector of Universitat Pompeu Fabra on .............................................., 2016.

President: **Iftach Haitner**

Member: **Claudio Soriente**

Secretary: **Carla Ràfols Salvador**

Reserve Members:

**Oriol Farràs Ventura**
**Javier Herranz Sotoca**

The doctoral defense was held on ........................................................, 2016, at Universitat Pompeu Fabra and scored as ...................................................

PRESIDENT MEMBER

SECRETARY

*To my mother.*

# Abstract/Resum

*Secure two-party computation* is a classic problem in cryptography. It involves two parties computing a function of their private inputs, and only revealing what the output suggests. Additional security requirements may include *fairness*, which states that either all parties receive output, or no one does. A seminal result from the 1980's demonstrates that fairness cannot be guaranteed for all functions, and only recently have certain functions been shown to be computable with fairness. The two results naturally give rise to a distinction between *fair* functions and *unfair* ones. In this work, we investigate the characterization of such functions in the two-party setting. In the end, we obtain a full characterization for Boolean functions, and we develop a number of useful techniques for characterizing arbitrary fair functions.

*Secure two-party computation* és un problema clàssic en criptografia. Dos participants acorden calcular una funció de les seves entrades privades, de manera que només es revela el que se'n derivi del resultat. Altres requisits de seguretat poden incloure *fairness*, que exigeix que o bé tots els participants obtenen el resultat, o ningú ho fa. Un resultat fonamental de la dècada dels 80 demostra que la propietat no es pot garantir per a totes les funcions, i només recentment s'ha demostrat que algunes sí que tenen aquesta propietat. Els dos resultats donen lloc a una distinció entre les funcions que són *fair*, i les que no ho són. En aquest treball, investiguem la caracterització d'aquestes funcions en l'entorn de dos participants, obtenint una caracterització completa de funcions Booleanes. A més a més, desenvolupem una sèrie de tècniques útils per caracteritzar qualsevol funció.

# Contents

# List of Figures

Chapter 1

# Introduction

Steve, a most industrious person, decides to sell a digital artefact over the internet. Steve advertises his product on a popular message board, and invites all interested parties to bid on it. Only two parties, Bob and Barbara, reply to the advertisement. Instead of revealing their bids, the potential buyers communicate the following request: if they were to make an offer, the amount they would be willing to pay should be disclosed (and transferred) to the seller alone[1], *only if* their bid was the highest, and *only after* they received the item. Without hesitation, Steve suggests that the three of them engage in secure multi-party computation.

Secure multiparty computation (MPC) is one of the gems of modern cryptography. It allows several distrusting parties to perform seemingly impossible tasks, like the one described above, solely by interacting with one another. It is useful to think of these tasks as functions/functionalities that take an input from each party and return an output to each party. In our example, the buyers inputs' correspond to their offers, the seller's input is the item on sale, and their outputs are functions of these inputs – the seller receives payment, i.e. the maximum of the buyers' inputs, the winner obtains the item, i.e. the seller's input, and the loser obtains a value indicating that he lost.

Another useful abstraction is to assume that the parties perform tasks by means of protocols, i.e. sets of instructions that the parties follow in order to complete the task. Obviously, the protocol must satisfy certain security requirements. To speak very loosely, we say that a protocol is secure if

[1]i.e. not the other buyer, nor potential eavesdroppers.

the underlying task exhibits certain properties even if some of the parties deviate from the instructions. Let us describe some of these properties.

- *Correctness.* The first property is perhaps the most obvious one. It stipulates that the protocol does what it is supposed to do. In our case, it means that at the end of the protocol, the seller receives payment corresponding to the highest bid, the loser is informed that he lost, and the winner obtains the "correct" item.

- *Privacy.* As one might expect, privacy is roughly equivalent to the notion that secret things should be kept secret. However, there is a caveat. In our example, it would appear that the parties should learn nothing about each others bids. Since the parties know their own bids, by the mere fact of winning or losing, the parties can deduce upper/lower bounds on one another's bids. This is unavoidable because of correctness. Thus, a more precise definition of privacy is to state that each party may learn only what his input/output suggests.

- *Independence of Inputs.* The third requirement stipulates that no party may choose his input as a function of other parties' inputs. In our example, it means that the buyers cannot outbid each other by deviating from the protocol's instructions.

- *Fairness.* By cleverly exploiting the protocols' flaws, suppose that Barbara, the highest bidder, manages to acquire the item without giving Steve any recompense. Alternatively, suppose that Steve gets away with the money without handing the item to Barbara. In both cases, the outcome of the protocol is unfair. By contrast, a protocol is *fair* if either everybody receives output or no one does.

- *Guaranteed Output Delivery.* The last property stipulates that no subset of parties can prevent the others from obtaining their output. If Bob, suspecting he is the lowest bidder, prevents Steve and Barbara from finalizing the transaction, then output delivery is violated.

## Framing the Problem

Next, let us discuss the parties' ability to communicate, and the potential threats that the parties may pose.

**The Communication Model.** At the very least, one must assume that the parties have access to a point-to-point network enabling the parties to communicate with one another. Alternatively, the parties may have access to more elaborate technologies, such as broadcast channels or public-key infrastructures. The *communication model* is the description of all processes/devices that allow the parties to communicate. We remark that a communication model may also include trusted parties in the form of servers that cannot be tampered with, nor can they be made to deviate from their intended purposes.

It goes without saying that the particularities of the model may be exploited by the cryptographer. However, protocols that achieve high degrees of security with little "help" from the model are clearly superior to protocols that rely on its characteristic features. Therefore, in designing protocols, the sensible approach is to make few and/or realistic assumptions that apply across different communication models. For example, given the way most people communicate e.g. over the internet, broadcast channels are regarded as realistic, whereas trusted parties are not.

While the distinction between realistic and unrealistic models is meaningful, we stress that models of the latter type are not without their merits. First off, dealing with unrealistic models is a legitimate scientific pursuit if realistic ones are simply too hard to be reckoned with. Second, the security of schemes in realistic settings is often reducible[2] to the security of other schemes in unrealistic settings, which may prove advantageous considering that unrealistic models often lend themselves to simpler and more intuitive analyses.

**The Adversary.** It is customary to attribute all undesirable behaviour to an abstract entity, known as the *adversary*, corrupting a subset of parties that she utterly controls[3], and to model the adversary by framing her capabilities and motives. In the spirit of the previous discussion, the adversarial model should be broad enough to capture as many threats as possible. Thus, adversaries with extraordinary computational power, sophisticated corrupting capabilities, and motives unknown, provide the highest degree of generality.

Still, there are advantages in considering weaker or limited adversaries. Two

[2]i.e. the latter implies the former.

[3]This assumption captures the worst-case scenario where all dishonest parties form a single coalition. There are cases where this assumption may be too strong, and adversarial models with non-cooperating coalitions of corrupted parties may be more suitable [24,64].

types that feature prominently in this work are *passive* and *fail-stop*. Passive adversaries do *not* deviate from the protocol's instructions. Rather, their goal is to extract sensitive information while maintaining an honest façade. On the other hand, fail-stop adversaries are essentially passive adversaries with the extra capability of aborting the protocol early.

## Security & The Ideal-World Paradigm

Historically, satisfactory definitions of security have been remarkably tricky to come by. Classic approaches usually focus on specific requirements (like privacy), or narrow adversarial models. By contrast, modern definitional frameworks that emanate from the *ideal-world* paradigm [40] take a more holistic approach to security. One such framework, known as *stand-alone*, gives rise to the security definitions that appear in this work. We discuss it next in greater detail.

The cryptographer is tasked with demonstrating that a protocol is secure. The first order of business is to consider an idealized scenario, known as the *ideal model*, involving the parties and the adversary, as well as a newly defined algorithmic entity, the *trusted party*, that interacts with all participants, including the adversary, in a predefined and controlled way.

**The Ideal Model.** To begin with, the honest parties send their inputs to the trusted party, while the corrupted parties send values of the adversary's choice. After receiving all the inputs, the trusted party computes the outputs, and interacts with the adversary in accordance with the cryptographer's rules that were laid out at the beginning of the exercise. For example, the trusted party may leak private information to the adversary and/or let the adversary decide which parties are to receive output – or the trusted party may do nothing of the sort. In any case, we stress that whatever the trusted party concedes to the adversary is a feature of the model, not a flaw, since the interactions of the trusted party and the adversary are fully determined by the cryptographer. At the end, in accordance with the previous step, the trusted party sends outputs to the parties.

**Security Definition (Informal) .** A protocol is secure with respect to the ideal model if executing the protocol is *no worse* than invoking a trusted party, and having all the participants interact as per the ideal model.

The ideal model is essentially a thought experiment on the part of the cryptographer. As such, all aspects of the model, including the adversary, fall under the cryptographer's discretion. With that in mind, we make a couple observations regarding adversary in the ideal model. The adversary is privy to the inputs/outputs of the corrupted parties, and the values that the latter send to the trusted party are chosen by the adversary. Other than that, the adversary is limited to whatever the trusted party might concede *as per the specifications of the model.* Thus, the cryptographer's security objectives dictate the features of the ideal model. For instance, all the requirements discussed at the beginning of the introduction (privacy, fairness, ...) are captured by a model, known as the ideal model with *full-security*, where the trusted party pays no attention to the adversary.

A protocol is secure if every (real) adversary is essentially "confined" to the restrictive framework of the ideal model. More precisely, a protocol is secure if every adversary admits an ideal-world analogue, i.e.

- Anything that can be extracted from the protocol, can be inferred from information available to the adversary in the ideal model.
- The ideal-world adversary can replicate the effects of the real-world adversary on the honest parties' outputs.

To sum up, security in the ideal-world paradigm is defined by describing a comprehensive mental picture of the cryptographer's security aims. Then, for any specific protocol, security is determined by assessing whether the protocol emulates this mental picture faithfully. The paradigm's appeal stems from this elegant process.

To conclude, we mention that the definitional framework above admits various extensions (as in [23]) with security notions that are not captured by the stand-alone framework. However, these notions are beyond the scope of our work. For our purposes, the definitions that emanate from the stand-alone framework suffice.

## State of the Art

Protocols satisfying the highest degree of security are known to exist [15,25, 41,70], as long as more than half of the parties are honest. In other words, whenever the corrupted parties form a *strict* minority, there are protocols that emulate the ideal model with full security, and thus these protocols satisfy all the requirements described at the beginning of the introduction.

However, mostly[4] because of fairness, all bets are off in the presence of a dishonest (relative) majorities. This limitation was known even before to the advent of MPC and the ideal-world paradigm. Fairness was first studied in the context of signature exchange, where two parties exchange digital signatures on a pre-agreed document. Fairness in this setting is of paramount importance, since it would be highly undesirable for one party to be bound to contents of the document, and for the other not to. Even and Yacobi [35] showed that, for this task, *the strongest notion of fairness* is out of the question. Specifically, in any signature-exchange protocol, there is a point where one party is privy to information that allows him to produce the relevant output, and the other party is not. Thus, a protocol where one party obtains output if both do is ruled out for signature exchange. We mention that protocols satisfying weaker notions of fairness are known for this task [18, 33].

Later, Cleve [26] showed that the limitation described above is somewhat inherent to secure computation in the presence dishonest majorities: even something as seemingly mundane as two-party coin-tossing, i.e. the task of two parties computing a common uniform random bit, *is not* computable with fairness. Cleve's result relies on an intuition similar to the one described above. If two parties are performing some interactive task, and the parties go back and forth exchanging messages in a *non-simultaneous* way, it would appear that there must be a point where one party has enough information to compute the output while the other party does not. By simply aborting at that point, fairness is clearly breached. As we shall see later on, this intuition does not apply to all MPC tasks. Following Cleve's impossibility, two complementary directions of investigation emerged.

The first one discards fairness altogether. *Unfairness* is formalized by letting the adversary decide who receives output in the ideal model. The resulting model, known as *security-with-abort*, is meaningful given that it guarantees correctness, privacy and independence of inputs, and there are protocols [41] that emulate the model faithfully for any number of corrupted parties, under suitable assumptions. We stress that most secure-with-abort protocols (like

[4]For our purposes, it is reasonable to attribute the lack of full-security to fairness. In arbitrary settings however, output delivery plays an important and distinct role. c.f. [29, 30]

the one from [41]) are unfair even when the dishonest parties are in the minority. A line of work, known as *best of both worlds*, attempts to combine full-security and security-with-abort by means of protocols that emulate one model or the other depending on the ratio of honest-to-dishonest parties. A number of papers [11, 52, 54] show that such protocols exist as long as the relevant ratio does not exceed certain values.

The second direction of investigation strives to preserve fairness albeit in a weaker/alternative form. Alternative notions of fairness roughly fall under three categories. In the first one, arguably the most natural one, protocols are (weakly) fair if the probability of breaching fairness is deemed tolerable. In the second category, (weak) fairness is defined by introducing entities, in the form of trusted parties or a preexisting networks, that can either restore fairness or deter the parties from breaching it. Finally, notions from the third category depart from the malicious/honest dichotomy of the cryptographic realm, by introducing game-theoretic utility functions and/or reputation systems for the parties.

**Gradual Release, Probabilistic Fairness & $1/p$-security.** The parties either learn the output slowly (Gradual Release) [18–20,31,33,34,36,38,39, 51,68,69,73] *or* their confidence in knowing the output grows as the protocol approaches its termination (Probabilistic Fairness) [10,14,27,42,61,71]. We refer to [43] for a detailed survey of the literature on these topics. For a more modern approach, we mention the $1/p$-security notion of Gordon and Katz [48] that follows the usual real-vs-ideal definitional framework. By relaxing how faithfully a protocol emulates the ideal model, the authors obtain a meaningful notion of fairness.

**Optimistic Fairness & MPC with Penalties.** In the optimistic model, one assumes the presence of an offline trusted party, dubbed the judge, that can restore fairness whenever it is breached. The optimistic model is well studied, and there is a plethora of definitions and feasibility/infeasibility results that pertain to it [6–9, 21, 32, 37, 56, 58–60, 65, 67]. On the other hand, in the penalty model, aggrieved parties do not obtain their output by resorting to a judge, or in fact by any other means. Rather, the parties are compensated in the form of some crypto-currency, like BitCoin [16, 55, 57]. Needless to say, this approach requires some sort of preexisting network.

Both optimistic fairness and MPC with penalties are application-driven and may prove useful to the non-academic world. That being said, from a theoretical perspective, both approaches somewhat defeat the purpose of secure computation – the objective of secure computation is to do away with trusted parties and preexisting setups.

**Other Notions of Fairness.** Inspired by [50], and finding itself at the intersection of cryptography and game theory, rational cryptography does not shy away from speculating about the parties' behavior. If the cryptographer knows the parties' preferences because he/she is privy to their utility functions, then protocols that build appropriate incentives by means of cryptographic tools provide meaningful notions of security. Alternatively, one might use reputations systems to achieve similar ends [5].

## Complete Fairness

The motivation behind secure-with-abort and weakly-fair computation is the notion that fairness is impossible for functions other than trivial ones, i.e. constant functions or functions where only one party is assigned an output. In [45], Gordon, Hazay, Katz and Lindell show that the notion is false. In this section, we survey the literature pertaining to the strongest form of fairness, also known as *complete fairness.*

**Impossibility of Fairness.** Coin-Tossing is the functionality that takes no inputs, and hands the same uniform random bit to a number of parties. Cleve [26] showed that in any two-party coin-tossing protocol, at least one party can *bias* the other party's output, i.e. the output is tilted toward 1 or 0. This seemingly unrelated fact has profound implications for fairness. In particular, it implies that coin-tossing is not computable with fairness, since the parties' outputs from any fair realization of the coin-tossing functionality cannot be biased. By extension, fairness is ruled out for any function that implies coin-tossing, such as XOR. The work of Asharov, Lindell, and Rabin [4] tackles this question in depth, and provides a characterization of all functions that imply coin-tossing. The characterization is known as the *balanced criterion.*

Agrawal and Prabhakaran [1] generalized Cleve's result to arbitrary *sampling functionalities.* A sampling functionality is a randomized function in which two parties, with no inputs, sample two bits. The functionality is said to be non-trivial if the resulting bits exhibit statistical correlation. In [1], it is shown that any non-trivial sampling functionality is *not* computable with fairness in the two-party setting. In addition, the authors analyze sampling functionalities systematically in the context of fairness, and they explore the relationship between sampling functionalities and the task of exchanging secret bits.

Cleve's work is also central to the topic of *optimally fair coin-tossing* [26,28], i.e. designing protocols that minimize the bias that the parties may inflict. Cleve showed that the bias is lower-bounded by a value that is inversely proportional to the number of rounds, for any coin-tossing protocol. Malkin, Naor and Segev [66] confirmed that Cleve's bound is tight asymptotically. In the multi-party setting, Beimel, Omri and Orlov [13] deduced asymptotically optimal bounds, assuming the corrupted parties do not exceed two-thirds of the total participants. Haitner and Tsfadia [49] deduced *almost*-optimal bounds for the three-party setting with corrupted majority.

**Possibility of Fairness.** The topic of complete fairness was single-handedly revitalized by the work of Gordon, Hazay, Katz and Lindell [45]. Far from being impossible, Gordon et al. showed that for certain two-party functions, there exist protocols that are fully secure, and thus satisfy the strongest notion of fairness. Such functions are inherently *fair*, as opposed to *unfair* functions like XOR. The question of characterizing functions with respect to full security was reopened. In [45], a distinction is made between XOR-embedded[5] and non XOR-embedded functions. Functions of the latter type, which includes OR and the greater-than function, are shown to be fair. Yet XOR-embedded functions are not necessarily excluded from fully-secure computation. Gordon et al. propose a specific protocol, referred to as GHKL throughout the present thesis, that computes a certain XOR-embedded function with full security.

Asharov [2] builds on the work of Gordon et al. by showing that, contrary to what might be expected, an extraordinary amount o functions are fair. To prove this result, Asharov delves into the security analysis of protocol GHKL, and deduces sufficient criteria for the protocol to be fully secure. Then, by applying classic results from affine geometry and linear algebra, the author elegantly demonstrates that almost all functions with unequal-size domains[6] satisfy the criteria. On the flip side, using similar arguments, the security analysis of Gordon et al. is shown to fail for almost all functions with equal-size domains.

In the multi-party setting, Gordon and Katz [47] showed that three-party majority function and $n$-party OR are computable with full security when

[5]A function is XOR-embedded if restricting the function to a subset of inputs yields the XOR function.

[6]i.e. one party has more inputs than the other.

all-but-one parties are corrupted. Finally, we mention [46] that discusses complete primitives for fairness, i.e. minimal augmentations of the communication model that enable completely fair computation.

## Our Contributions

In this work, we tackle the central question raised by Gordon et al. regarding the characterization of functions with respect to fairness. Our work is organized in two parts.

*Given a two-party function, is there an easy-to-check criterion that determines whether that function is computable with full security?*

**Part I.** In the first part, similarly to [45] and [2], we focus on a specific class of functions. Namely, functions that are *finite* – the parties have a finite amount of inputs, *symmetric* – both parties receive the same output, *Boolean* – the output consists of a bit, and *deterministic* – the output is fully determined by the inputs. By abusing terminology, functions in this class are referred to as Boolean functions.

Tight security analysis of GHKL.

Like Asharov, we embark on an in depth analysis of GHKL and its security proof, and we deduce sufficient criteria for the protocol to be fully-secure. The criteria of [2] follow from ours as a special case. Then, we show that our criteria are necessary. That is to say GHKL falls short of full-security, for any function that does not meet our criteria. We emphasize that our criteria were found independently of Asharov's.

Generalization of the Balanced Criterion of Asharov, Lindel and Rabin.

As mentioned earlier, certain functions are excluded from fully-secure computation because they imply coin-tossing. In Chapter 4, we present a technique to implement non-trivial sampling functionalities by means of certain Boolean functions. Since fully-secure computation of sampling functionalities is ruled out by [1], the technique results in a necessary condition for fairness. The condition, dubbed the *semi-balanced* criterion, gives rise to a family of *unfair* functions that were previously unaccounted .

> A new protocol, based on GHKL, that computes *all* fair Boolean functions with full security.

At the beginning of Chapter 5, we show that a gap remains in the characterization, i.e. there are functions that lie outside the newly encountered family of unfair functions *and* the family of functions for which GHKL is fully secure. By the end of the chapter, we show that modifying GHKL results in a protocol that is fully secure for *all* the functions that lie in the gap. While our modification may be viewed as generalization of GHKL, we believe that it departs from the spirit of Gordon et al.'s protocol. The problem of characterizing Boolean functions is settled.

**Theorem (Informal) .** A Boolean function is computable with full security if and only if the all-one vector or the all-zero vector belong to the affine span of either the rows or the columns of the matrix describing the function.

As a conclusion to Part I, we point out that our main result extends to a complete characterization of – randomized two-party Boolean functions – multi-party symmetric Boolean functionalities in the presence of relative dishonest majorities (assuming broadcast). We emphasize that most previous works in this area (with the exception of [47]) assume that the corrupted parties form a strict minority.

> The exact (non-asymptotic) trade-off between bias and round-complexity for two-party coin-tossing.

In Chapter 4, we make a brief detour to the topic of optimally fair coin-tossing. We show that by derandomizing the protocol of Malkin, Naor and Segev, the theoretical bound from Cleve's seminal paper is reached.

**Part II.** Needless to say, Part II aims at characterizing functions from a broader class of functions – arbitrary functions that are *finite*. However, we emphasize that our goals are rather more ambitious. The second half of our exposition aspires to lay the foundation for a framework that allows for the systematic analysis of a functions's fairness. To this end, we "deconstruct" certain concepts from Part I, and insights that emerge are applied toward extracting necessary and/or sufficient criteria for fairness. Unfortunately, the second part of our work contains a number of unresolved issues. Some of these issues may be of independent interest, and they are discussed in the concluding chapter.

> What does provable unfairness boil down to?

The question gives rise to a couple of concepts that we refer to as *locking strategies* and *sampling attacks*. As the choice of names may give away, locking strategies and sampling attacks are algorithmic processes that we associate with the honest party and the adversary, respectively. We emphasize that both processes appear repeatedly in Part I, even though they are only defined at the beginning of Part II. Locking strategies and sampling attacks are added to the nomenclature in order to systematize the techniques from the first part of our exposition. In Chapter 6, locking strategies provide meaningful context to the negative criteria from Chapter 4, and they allow for an intuitive generalization of the semi-balanced criterion to arbitrary functions.

> A notion of security that lies between security-with-abort and full-security.

Intermediate notions of security are worthy of our consideration for the possibility that they might provide a route toward full security. This is precisely what Chapter 7 is all about. We show that certain protocols satisfying a peculiar security requirement can be used as building blocks in the design of fully secure protocols. This new requirement, referred to as *security against sampling attacks*, relates to sampling attacks as well as locking strategies.

> Designing fully secure protocols for *asymmetric* Boolean functions.

To illustrate the usefulness of our approach, we focus on asymmetric Boolean functions as a case study. We show how to construct suitable protocols that give rise to fully secure ones. Chapter 7 mostly revolves around an algorithm that we use in this regard. In particular, for any asymmetric Boolean function, our algorithm either comes up with the description of a suitable protocol, or it returns that it failed to do so. In the former case, the function is computable with full security. In the latter case, we suspect that the function is unfair, even though a proof escapes us at the moment.

## Publications

Most of our results from Chapters 3 and 4 were published in [63]. Our efforts were recognized by the program committee of the Security and Cryptography for Networks Conference in the form of a Best Paper award. The characterization from Chapter 5 was published in [3].

At the time these lines are being written, the contents of Part II do not appear in the literature, and we are actively pursuing the publication of the relevant results. Finally, some preliminary results that are not included in this thesis can be found in [62].

Chapter 2

# Preliminaries

Some familiarity with probability theory and linear algebra is taken for granted.

## 2.1 Notation

Let $\mathbb{N}$ denote the set of natural numbers and let $n \in \mathbb{N}$ denote the security parameter. A function $\mu(\cdot) = \mathsf{negl}(\cdot)$ is *negligible* if it vanishes faster than any (positive) inverse-polynomial. A distribution ensemble $X = \{X(a,n)\}_{a \in \Delta, n \in \mathbb{N}}$ is an infinite sequence of random variables indexed by $\Delta$ and $\mathbb{N}$. Two distribution ensembles, $X$ and $Y$, are *computationally indistinguishable* if for every non-uniform polynomial-time algorithm $D$, there exists a negligible function $\mu$ such that for every $a$ and $n$

$$|\Pr[D(X(a,n)) = 1] - \Pr[D(Y(a,n)) = 1]| \leq \mu(n).$$

Furthermore, we say that $X$ and $Y$ are *statistically close* if for all $a$ and $n$, the following sum is upper-bounded by a negligible function in $n$:

$$\frac{1}{2} \cdot \sum_{s} |\Pr[X(a,n) = s] - \Pr[Y(a,n) = s]|,$$

where $s$ ranges over the support of either $X(a,n)$ or $Y(a,n)$. We write $X \overset{c}{\equiv} Y$ when the ensembles are computationally indistinguishable and $X \overset{s}{\equiv} Y$ when they are statistically close.

Let $P_1, P_2, \ldots, P_m$ denote the parties. An *$m$-party functionality*[1] or *function* $f = (f_1, \ldots, f_m)$ is a random process that maps $m$-tuples of inputs (one for each party) to $m$-tuples of random variables called outputs (one for each party). The domain of $f$ is denoted $X = X_1 \times \cdots \times X_m$. Moreover, we say that $f$ is symmetric if $f_1 = \cdots = f_m$ i.e. all parties output the same value, and asymmetric otherwise.

An *$m$-party protocol* $\Pi$ computing functionality $f$ is a polynomial-time protocol satisfying the following. On global input $1^n$ – the security parameter, and private inputs $x_1 \in X_1, \ldots, x_m \in X_m$ – handed by the parties, the joint distribution of the outputs in an honest execution of $\Pi$ is statistically close to $f(x_1, \cdots, x_m) = (f_1(x_1, \ldots, x_m), \ldots, f_m(x_1, \cdots, x_m))$. The parties in the protocol run in polynomial time in the security parameter $n$.

Following the usual convention, we introduce an adversary $\mathcal{A}$, which is given an auxiliary input $z$ and corrupts a subset of the parties. The adversary is static, that is, it chooses the subset it corrupts prior to the execution of the protocol. The adversary is assumed to be malicious and computationally bounded. For a protocol $\Pi$ computing $f$, let

$$(\mathrm{Out}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}, \mathrm{View}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi})(1^n, x_1, \ldots, x_m)$$

denote the joint distribution of the honest parties' outputs and the adversary's view during an execution of $\Pi$, where $x_1, \ldots, x_m$ are the prescribed inputs, $1^n$ is the security parameter (in unary representation), and the view of the adversary consists of the auxiliary input, the inputs of the parties it controls, their random inputs, and the messages they receive during the execution of the protocol.

**Notation and Results from Linear Algebra.** Let $\mathbb{R}$ denote the real field, and fix $\ell$, $k \in \mathbb{N}$. Column vectors are denoted by bold letters, e.g. $\mathbf{v}$ or $\mathbf{1}_k$ (the all-one vector), and matrices are denoted by capital letters, e.g. $M$, $P$. We distinguish row vectors and transposes using the superscript $^T$. The $i$-th entry of a vector $\mathbf{v}$ is denoted $\mathbf{v}(i)$, and we write $\mathbf{v} \in \mathbb{R}^k$ to indicate that $\mathbf{v}$ has $k$ entries. The entry at the intersection of the $i$-th row and $j$-th column of a matrix $M$ is denoted $M(i,j)$, and we write $M \in \mathbb{R}^{\ell \times k}$ to indicate that $M$ has size $\ell \times k$. Given two matrices $M$ and $M'$ of same size, write $M * M'$ for the entry-wise (Hadamard) product.

We say that $\mathbf{w} \in \mathbb{R}^\ell$ is in the image of $M$, denoted $\mathbf{w} \in \mathrm{im}(M)$, if there exists $\mathbf{u} \in \mathbb{R}^k$ such that $M\mathbf{u} = \mathbf{w}$. We say that $\mathbf{v} \in \mathbb{R}^k$ is in the kernel of

[1]The definition of a functionality is tailored to our purposes.

$M$, denoted $\mathbf{v} \in \ker(M)$, if $M\mathbf{v} = \mathbf{0}_\ell$. Vectors $\mathbf{u}$ and $\mathbf{v} \in \mathbb{R}^k$ are said to be orthogonal if $\langle \mathbf{u} \,|\, \mathbf{v} \rangle = \mathbf{u}^T \mathbf{v} = 0$. Similarly, sets $\mathbf{U}$ and $\mathbf{V} \subseteq \mathbb{R}^k$ are said to be orthogonal if $\mathbf{u}$ and $\mathbf{v}$ are orthogonal, for every $(\mathbf{u}, \mathbf{v}) \in \mathbf{U} \times \mathbf{V}$. Let $\mathbf{U} + \mathbf{V} = \{\mathbf{u} + \mathbf{v} \,|\, (\mathbf{u}, \mathbf{v}) \in \mathbf{U} \times \mathbf{V}\}$.

**Theorem 2.1** (Fundamental Theorem of Linear Algebra)**.** *For any* $M \in \mathbb{R}^{\ell \times k}$*, it holds that* $\mathrm{im}(M^T)$ *and* $\ker(M)$ *are orthogonal, and* $\mathrm{im}(M^T) + \ker(M) = \mathbb{R}^k$*.*

Let $\mathbf{V} \subseteq \mathbb{R}^\ell$ be a finite set of vectors and write $\langle \mathbf{V} \rangle$ for the vector space generated by $\mathbf{V}$ i.e. the set of vectors obtained as linear combinations of the elements of $\mathbf{V}$. We say that $\mathbf{w} \in \mathbb{R}^\ell$ is an *affine combination* of the elements in $\mathbf{V}$ if there exist scalars $\{a_\mathbf{v}\}_{\mathbf{v} \in \mathbf{V}}$ such that $\mathbf{w} = \sum_{\mathbf{v} \in \mathcal{V}} a_\mathbf{v} \cdot \mathbf{v}$ and $\sum_{\mathbf{v} \in \mathcal{V}} a_\mathbf{v} = 1$, and all $\mathbf{w}$'s obtained that way form the *affine span* of $\mathbf{V}$. Write $\mathcal{H}(M^T)$ for the affine span of the rows of $M \in \mathbb{R}^{\ell \times k}$. Finally, for an arbitrary matrix $P \in \mathbb{R}^{k \times k}$, we say that $P$ is an affine endomorphism of $\mathcal{H}(M^T)$ if $P \cdot \mathbf{u} \in \mathcal{H}(M^T)$, for every $\mathbf{u} \in \mathcal{H}(M^T)$.

**Lemma 2.2.** *For any* $M \in \mathbb{R}^{\ell \times k}$*, it holds that* $\mathbf{0}_k$ *is an affine combination of the rows of* $M$ *if and only if* $\mathbf{1}_\ell$ *is not a linear combination of the columns of* $M$*.*

*Proof.* Write $M'$ for the matrix obtained from $M$ by concatenating the all-1 column. It holds that $\mathbf{1}_\ell$ is not a linear combination of the columns of $M$ if and only if $\mathrm{rank}(M') = \mathrm{rank}(M) + 1$. Equivalently, $\dim(\ker(M')) = \dim(\ker(M)) - 1$ and there exists $\mathbf{u} \in \mathbb{R}^\ell$ such that $M^T \mathbf{u} = \mathbf{0}_k$ and $\mathbf{1}_\ell^T \cdot \mathbf{u} \neq 0$. □

## 2.2 Security Definition

Let $f = (f_1, \ldots, f_m)$ be an $m$-party functionality and let $\Pi$ be a protocol for computing $f$. Further assume that an adversary corrupts a fixed subset $B \subseteq \{P_1, \ldots, P_m\}$ of the parties. Security in multiparty computation is defined via an *ideal model*. We assume that parties have access to a trusted party $\mathcal{T}$ that computes the functionality for them, and we attempt to show that protocol $\Pi$ emulates this idealized scenario. In Figure 2.1, we describe the *ideal model with full security*.

**Inputs:** Each party $P_i$ holds $1^n$ and $x_i \in X_i$. The adversary is given an auxiliary input $z \in \{0,1\}^*$. The trusted party $\mathcal{T}$ has no input.

**Parties send inputs:** Each honest party sends its input to $\mathcal{T}$, each corrupted party sends a value of the adversary's choice. Write $(x'_1, \ldots, x'_m)$ for the tuple of inputs received by $\mathcal{T}$.

**The trusted party performs computation:** If any $x'_i$ is not in the appropriate domain (or was not sent at all), then $\mathcal{T}$ reassigns the aberrant input to some default value. Write $(x'_1, \ldots, x'_m)$ for the tuple of inputs after (possible) reassignment. The trusted party then chooses a random string $r$ and computes $f(x'_1, \ldots, x'_m; r)$.

**Trusted party sends outputs:** Each party $P_i$ receives $f_i(x'_1, \ldots, x'_m; r)$.

**Outputs:** Each honest party outputs whatever $\mathcal{T}$ sent him, the corrupted parties output nothing, and the adversary outputs a probabilistic polynomial-time function of its view.

Figure 2.1: The Ideal Model with Full Security.

Let $\mathcal{S}$ be an adversary in the ideal world, given auxiliary input $z$ and corrupting a subset $B$ of the parties. Let $\mathrm{Out}^{\mathsf{Ideal}}_{\mathcal{S}(z),f}(1^n, x_1, \ldots, x_m)$ denote the honest parties' outputs in the ideal model, where $x_1, \ldots, x_m$ are the prescribed inputs and $1^n$ is the security parameter. Let $\mathrm{View}^{\mathsf{Ideal}}_{\mathcal{S}(z),f}(1^n, x_1, \ldots, x_m)$ denote the adversary's *output* in the ideal model.

**Definition 2.3.** Let $\Pi$ be a protocol for computing $f$. We say that $\Pi$ computes $f$ *with full security* tolerating coalitions of size at most $t$ if for every non-uniform polynomial time adversary $\mathcal{A}$ controlling a set $B$ of at most $t$ parties in the real model, there exists a non-uniform polynomial time adversary $\mathcal{S}$ (called the simulator) controlling $B$ in the ideal model such that

$$\left\{ \left( \mathrm{Out}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}, \mathrm{View}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi} \right) (1^n, x_1, \ldots, x_m) \right\}_{\substack{(x_1,\ldots,x_m)\in X,\\ z\in\{0,1\}^*, n\in\mathbb{N}}}$$
$$\overset{c}{\equiv} \left\{ \left( \mathrm{Out}^{\mathsf{Ideal}}_{\mathcal{S}(z),f}, \mathrm{View}^{\mathsf{Ideal}}_{\mathcal{S}(z),f} \right) (1^n, x_1, \ldots, x_m) \right\}_{\substack{(x_1,\ldots,x_m)\in X,\\ z\in\{0,1\}^*, n\in\mathbb{N}}} .$$

In effect, showing that the above distribution ensembles are computationally indistinguishable implies that, in the plain model, the information acquired by the adversary together with her influence over the honest parties' outputs is no worse than what can be achieved in an idealized situation.

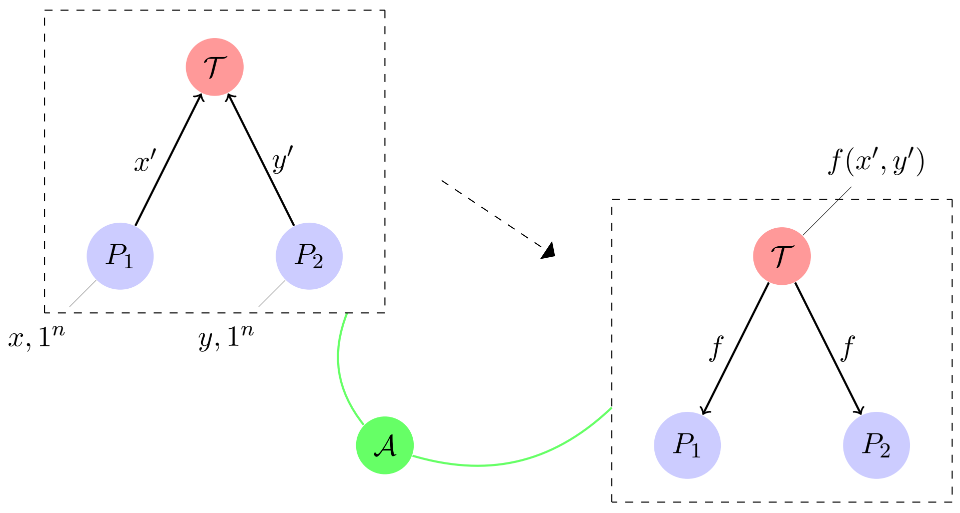

Figure 2.2: Visual Illustration of the Two-Party Fully-Secure Model.

**Cryptographic Attacks.** From the security definition, it follows that a protocol is *not* fully-secure if some (real) adversary does not admit a ideal-world analogue. Typically, the adversary follows an algorithmic process, in the real model, that induces a view and/or an effect on the honest party's outputs, clearly beyond the simulator's capabilities in the ideal model. We refer to any such process as an attack.

As discussed in the introduction, if the corrupted parties are in the minority, then essentially any task can be carried out with full security. Specifically, if the parties have access to broadcast, and at least half of them are honest, then any multiparty functionality is computable with full security [70]. The statement remains true even without broadcast, as long as two thirds of the parties are honest [15, 25]. Finally, if we allow cryptographic assumptions, then any multiparty functionality is computable with full security [41], as long as half of the parties are honest.

| % Corrupted Parties | $< 1/3$ | $< 1/2$ |
|---|---|---|
| *with Broadcast* | | Statistical Security |
| *without Broadcast* | Statistical Security | Computational Security |

If the adversary corrupts a majority of parties, then theoretical limitations relating to fairness prevent fully secure computation, for many functionalities. This happens by default in the two-party setting. By doing away with fairness, what remains is the notion of security-with-abort, which is formalized by letting the simulator decide which parties receive output in the ideal model. The definition for security-with-abort follows from the relevant model in the same way that Definition 2.3 follows from the ideal model with full security. For more details, see Appendix A p.152.

**Theorem 2.4** (GMW87). *Under suitable cryptographic assumptions, any functionality is computable with abort, for any number of corrupted parties.*

## 2.3 The Hybrid Model

Breaking down a difficult task into a number of easier subtasks is arguably the most useful problem-solving technique there is. However, this technique is only applicable when *the whole corresponds to the sum of the parts.* Luckily for us, MPC tasks abide by this rule. Let $f^{(1)}, \ldots, f^{(t)}$ and $f$ denote multiparty functionalities. The *hybrid model with*[2] *ideal access to* $f^{(1)}, \ldots, f^{(t)}$ is the communication model obtained by augmenting the real-model with a trusted party that the parties may call upon to compute any one of the $f^{(i)}$, and the participants' interaction play out according to a specific ideal model that has been fixed in advance. Thus, *a hybrid protocol* for computing $f$ is a protocol for computing $f$ in the relevant communication model. Without loss of generality, any hybrid protocol proceeds in sequence of at most polynomially-many iterations such that, at each iteration, the parties either have a real-world interaction, or they make a single call to the trusted party.

A hybrid protocol $\Pi$ computes $f$ with full security if for every adversary in the hybrid model, there is a simulator $\mathcal{S}$ in the ideal model such that the joint distribution of the adversary's view and honest parties' outputs in the ideal and hybrid model are indistinguishable. By virtue of proposition [22] below, secure protocols in the hybrid model give rise to secure protocols in the plain model.

**Proposition 2.5.** *For every $i \in \{1, \ldots, t\}$, suppose that (real) protocol $\rho_i$ computes $f^{(i)}$ with full-security. If hybrid protocol* $\Pi$ *computes $f$ with full*

[2]If no confusion arises, functions $f^{(i)}$ are omitted.

*security, then real protocol* $\Pi^{\rho_1,\ldots,\rho_m}$ *computes* $f$ *with full security, where* $\Pi^{\rho_1,\ldots,\rho_m}$ *is obtained from* $\Pi$ *by replacing ideal calls to the trusted party with the appropriate protocol.*

For positive results, it goes without saying that the hybrid-model approach makes sense only if every invocation of the trusted party is readily replaceable with a (real) protocol. Even then, the hybrid model is advantageous insofar that it conceals all things irrelevant to the analysis, and it rids itself of the minutiae of the plain-model. Alternatively, by turning the method on its head, hybrid models also provide convenient routes towards impossibility results. Whereas the positive direction asks whether the overlying task is feasible, based on the subtasks, the negative direction asks whether certain subtasks are impossible, based on the overlying task.

## 2.4 The Online Dealer Model

For aesthetic as well as practical reasons, the communication model underlying most of our work is of the hybrid variety, and any claims pertaining to the plain model follow from a hybrid-model argument. The model we use is known as *Online Dealer*, and it involves a trusted party, dubbed the *dealer*, that performs all the computations for the parties, and the parties receive messages exclusively through the dealer.

To show that a given function is fair, we define an appropriate protocol as per Figure 2.3, and we argue that the (hybrid) protocol is fully secure. A protocol in the online dealer model is defined by assigning a function $r \in \mathsf{poly}(n)$ for the number of rounds, where $n$ denotes the security parameter, and by describing the probability distribution of the parties' *backup sequences*, for all possible inputs. The backup sequences of $P_1$ and $P_2$ are denoted $(a_1, \ldots, a_r)$ and $(b_1, \ldots, b_r)$, respectively. As the name suggests, $a_i$ (resp. $b_i$) denotes the relevant party's output, if the opponent were to abort at round $i$ (resp. $i + 1$).

**Quirks of the Online Dealer Model.** Any protocol in the online dealer model admits a concise and intuitive description. In addition, the model is convenient in that the adversary is restricted to fail-stop behavior. We argue that neither of these assumptions incurs loss of generality, in relation to the plain-model.

**Inputs:** Party $P_1$ (resp. $P_2$) holds $1^n$ and $x \in X$ (resp. $y \in Y$). The dealer $\mathcal{D}$ has no input.

**Parties compute backup outputs:** $P_1$ (resp. $P_2$) is instructed to compute $a_0$ (resp. $b_0$).

**Parties send inputs:** Parties are instructed to send their inputs to the dealer. Write $(x', y')$ for the inputs received by $\mathcal{D}$.

**The dealer performs computation:** The dealer computes bits $a_1, \dots, a_r$ and $b_1, \dots, b_r$, where $r \in \mathbb{N}$ depends on the security parameter.

**The dealer sends outputs:** For $i = 1 \dots r$

1. The dealer hands $a_i$ to the first party. Then, $P_1$ instructs $\mathcal{D}$ to *abort* or *continue*. In the first case, the dealer sends $\perp$ to both parties and halts. Otherwise go to next step.
2. The dealer hands $b_i$ to the second party. Then, $P_2$ instructs $\mathcal{D}$ to *abort* or *continue*. In the first case, the dealer sends $\perp$ to both parties and halts. Otherwise go to next step.

**Outputs:** Parties are instructed to output the last bit they successfully constructed/received.

Figure 2.3: The Online Dealer Model

For one, in light of standard authentication techniques and secure-with-abort computation, it is sensible to assume that anything beyond fail-stop bears no consequence to a (real) protocol's security. Specifically, by means of an *unfair* preprocessing step, the parties may generate a number of random values, in the form of keys, shares and tags, allowing the parties to keep tabs on one another. Then, by unfairly exchanging some of these values in a predefined way, information available to the parties at any give point may be restricted to whatever the cryptographer intended for. Thus, any protocol along those lines is only vulnerable to fail stop attacks, *at worst.*

While theoretically sound, the approach just described comes at a heavy price. The (real) protocol's description must carefully keep track of all the relevant values, even though the values only serve the one purpose. Other aspects of the protocol are often obscured, and the resulting security analyses may appear convoluted. By contrast, notice that the online dealer model achieves similar ends, while avoiding all the hassle.

As discussed in the previous section, a hybrid model is useful if it serves as a user-friendly alternative to the real model, which the online dealer model certainly does. There is a standard transformation from this model to the plain one, c.f. Appendix B p.153, and thus Proposition 2.5 applies to this model.

**Remark 2.6.** The ideal call in the online dealer model is reactive, i.e. it receives inputs and gives outputs over multiple iterations, whereas the underlying functionalities in a hybrid model are non-reactive. This minor inconsistency may be resolved in one of two ways. Either by noting that Proposition 2.5 applies to reactive functionalities as well, or by replacing the online dealer model with the equivalent offline dealer model from Appendix B p. 153. The latter is hybrid according to our definition.

Throughout our work, by abusing terminology, the terms *plain*, *real* and *online dealer* are used interchangeably.

Part I

# The Characterization of Boolean Functions

## Special Notation for Part I

Throughout Part I, $f$ denotes a two-party Boolean function. The input domain of $f$ will are denoted $X \times Y$, instead of $X_1 \times X_2$, and we assume that $X = \{1, \dots, \ell\}$, $Y = \{1, \dots, k\}$. In addition, to every function $f$, we associate a matrix $M \in \mathbb{R}^{\ell \times k}$ such that $M(x, y) = f(x, y)$. Finally, the $x$-th row and $y$-th column of $M$ are denoted $\mathsf{row}_x^T$ and $\mathsf{col}_y$, respectively. We say that a vector is *monochromatic* if it is equal to the all-1 or all-0 vector.

Chapter 3

# Protocol GHKL

Parties $P_1$ and $P_2$ wish compute a boolean function $f : X \times Y \to \{0, 1\}$ on inputs $x$ and $y$, respectively, by means of protocol GHKL. We ask whether GHKL computes $f$ with full security. The purpose of the present chapter is to prove the following theorem.

**Theorem 3.1.** *GHKL computes $f$ with full security if and only if $\mathbf{v} \mapsto \big(M^T \cdot \mathbf{1}_\ell/\ell\big) * \mathbf{v}$ is an endomorphism of $\mathcal{H}(M^T)$.*

Protocol GHKL proceeds in a sequence of rounds such that the parties receive *backup outputs* one after the other. Prior to round $i^*$, referred to as the *special* or *threshold* round, the parties' backup outputs are computed by choosing a fresh random input for the other party. For every round after $i^*$, the parties are handed the correct output. At any given round, if either party aborts, the remaining party is instructed to output the last backup output he received. It goes without saying that the parties are (mostly) ignorant of the value of $i^*$, which is chosen by the dealer according to the geometric distribution with parameter $\alpha$.

The present chapter is organized as follows. We begin with the security analysis that appears in [45], and which gives rise to a set of equations that the simulator is required to solve. Then, we show that solutions can be found efficiently if a simple criterion is met. Finally, in the last section, we show that GHKL is susceptible to an attack, for any function that does not satisfy the criterion. Interestingly, this means that the simulation of GHKL is essentially unique, and our analysis is tight.

**Protocol GHKL**

1. The parties $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[a]
2. The dealer chooses $i^* \geq 1$ according to the geometric distribution with probability $\alpha$.
3. The dealer computes $\mathsf{out} = f(x, y)$, and for $0 \leq i \leq r$

$$a_i = \begin{cases} f(x, \widetilde{y}^{(i)}) \text{ where } \widetilde{y}^{(i)} \in_U Y & \text{if } i < i^* \\ \mathsf{out} & \text{otherwise} \end{cases}$$

and

$$b_i = \begin{cases} f(\widetilde{x}^{(i)}, y) \text{ where } \widetilde{x}^{(i)} \in_U X & \text{if } i < i^* \\ \mathsf{out} & \text{otherwise.} \end{cases}$$

4. The dealer gives $b_0$ to $P_2$.
5. For $i = 1, \ldots, r$, where $r(n) = \alpha^{-1} \cdot \omega(\log(n))$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

[a]If $x$ is not in the appropriate domain or $P_1$ does not hand an input, then the dealer sends $f(\hat{x}, y)$ (where $\hat{x}$ is a default value) to $P_2$, which outputs this value and the protocol is terminated. The case of an inappropriate $y$ is dealt analogously.

Figure 3.1: Protocol GHKL for Computing $f$.

## 3.1 The Analysis of Gordon et al.

The probability that $i^* > r$ is $(1-\alpha)^r$. So, if $r = \alpha^{-1} \cdot \omega(\log n)$, then the protocol is correct with overwhelming probability. Next, note that a corrupted $P_2$ poses no threat to the protocol. Any malicious behaviour on the part of $P_2$ is utterly harmless, given that $P_1$ receives the correct output first, no matter what. In fact, a corrupted $P_2$ can be simulated regardless of the parameters, for *any* function. On the other hand, precisely because $P_1$ receives the correct output first, an adversary $\mathcal{A}$ controlling $P_1$ may potentially breach fairness by guessing the threshold round and quitting.

Let us begin with a high-level description of the simulator. The adversary hands $x$ to the simulator for the computation. The simulator chooses $i^*$

**The Simulator $\mathcal{S}$ for Protocol GHKL**

- The adversary $\mathcal{A}$ gives its input $x$ to the simulator.[a]
- The simulator chooses $i^* \geq 1$ according to the geometric distribution with probability $\alpha$.
- For $i = 1, \ldots, i^* - 1$:
  - The simulator gives $a_i = f(x, \widetilde{y}^{(i)})$ to the adversary $\mathcal{A}$, where $\widetilde{y}^{(i)}$ is chosen according to the uniform distribution.
  - If $\mathcal{A}$ aborts, then the simulator chooses an input $x_0$ according to a distribution $\mathbf{x}_x^{(a_i)}$ (which depends on the input $x$ and backup output $a_i$), gives $x_0$ to the trusted party, outputs the bits $a_1, \ldots, a_i$, and halts.
- At round $i = i^*$, the simulator gives $x$ to the trusted party and gets the output $a = f(x, y)$.
- For $i = i^*, \ldots, r$: The simulator gives $a_i = a$ to the adversary $\mathcal{A}$, if $\mathcal{A}$ aborts, then the simulator outputs the bits $a_1, \ldots, a_i$ and halts.
- The simulator outputs the bits $a_1, \ldots, a_r$ and halts.

[a]If the adversary gives an inappropriate $x$ (or no $x$), then the simulator sends some default $\hat{x} \in X$ to the trusted party, outputs the empty string, and halts.

Figure 3.2: The Simulator $\mathcal{S}$ for Protocol GHKL.

according to the geometric distribution with parameter $\alpha$, and, for $i = 1 \ldots i^* - 1$, the simulator generates backup outputs for $P_1$ exactly as dealer does in the protocol, i.e. by choosing an input for the second party uniformly at random. The backup outputs are handed one-by-one to the adversary, thus simulating the rounds of the protocol. At any point, if the adversary aborts, the simulator choose $x_0$ according to a probability vector $\mathbf{x}_x^{(a)}$, where $a$ denotes the last value that was handed to the adversary. The simulator sends $x_0$ to the trusted party, outputs the sequence of bits he handed to the adversary, and halts. If the simulator runs out of values, and the adversary is still active, then the simulator sends $x$ to the trusted party and receives the correct output. Let $\mathsf{out}$ denote the output. For $i = i^* \ldots r$, the simulator hands $\mathsf{out}$ to the adversary, thus simulating the remaining rounds of the protocol. At any point, if the adversary aborts, the simulator outputs the sequence of bits he handed to the adversary and halts. See Figure 3.2 for the complete description of the simulator.

Recall that the protocol is secure if the adversary's view and the honest party's output in the real and ideal worlds are indistinguishable. Observe that the corrupted party's backup sequences are identically distributed, in both worlds. The question becomes how does the simulator replicate the effect of an early abort on the honest party's output, given that he can only do so by choosing an appropriate input for $P_1$. Thus, the strategy boils down to the existence of $\mathbf{x}_x^{(a)}$ that will do the trick. Define $\mathbf{c}_x^{(0)}, \mathbf{c}_x^{(1)}$ such that

$$\mathbf{c}_x^{(0)}(y) \overset{\text{def}}{=} \begin{cases} \mathbf{q}(y) & \text{if } f(x,y)=1 \\ \dfrac{\alpha \cdot \mathbf{q}(y)}{(1-\alpha)\cdot(1-\mathbf{p}(x))} + \mathbf{q}(y) & \text{otherwise} \end{cases},$$

$$\mathbf{c}_x^{(1)}(y) \overset{\text{def}}{=} \begin{cases} \mathbf{q}(y) & \text{if } f(x,y)=0 \\ \dfrac{\alpha \cdot (\mathbf{q}(y)-1)}{(1-\alpha)\cdot \mathbf{p}(x)} + \mathbf{q}(y) & \text{otherwise} \end{cases},$$

and

$$\mathbf{q} \overset{\text{def}}{=} M^T \cdot \mathbf{1}_\ell, \qquad \mathbf{p} \overset{\text{def}}{=} M \cdot \mathbf{1}_k.$$

**Theorem 3.2** (Gordon et al. [44]). *Using the notation above, if for some $\alpha \in (0,1)$, and all $x \in \{1,\ldots,\ell\}$, there exist probability vectors $\mathbf{x}_x^{(0)}, \mathbf{x}_x^{(1)} \in \mathbb{R}^\ell$ such that*

$$M^T \cdot \mathbf{x}_x^{(a)} = \mathbf{c}_x^{(a)},$$

*then GHKL computes $f$ with full security.*

*Proof.* First, observe that the real and ideal backup-sequences of $P_1$ are identically distributed. It follows that the adversary quits in the real world if and only if she quits in the ideal world. Next, if the adversary aborts after $i^*$ has been surpassed, then the honest party's output and the adversary's view are identically distributed in the real and ideal worlds. Finally, assuming the adversary aborts at round $i \leq i^*$, $P_1$'s view up to round $i-1$ is independent of the tuple consisting of the honest party's output and the corrupted party's backup at round $i$. Thus, to show that the relevant ensembles are indistinguishable, it suffices to show that for an adversary quitting at round $i \leq i^*$, the $i$-th backup output of $P_1$ together and the honest party's output admit the same joint distribution in the real and ideal model.

Let $(a_i, b)^{\mathsf{Real}}$ and $(a_i, b)^{\mathsf{Ideal}}$ denote the $i$-th backup output of $P_1$ and the honest party's output in the relevant model. By analyzing all possible outcomes, we show that $(a_i, b)^{\mathsf{Real}}$ and $(a_i, b)^{\mathsf{Ideal}}$ are identically distributed.

Write $\mathbf{c}_x^{(a)}(y)$ for the probability that $P_2$'s outputs is equal to 1 in the ideal world, given that the simulator sent an input according to $\mathbf{x}_x^{(a)}$ and $P_2$ holds $y$ as an input.

**First Case:** $(0,0)$**.** In the real world,

$$\Pr\left[(a_i,b)^{\mathsf{Real}}=(0,0)\right]=\alpha(1-f(x,y))(1-\mathbf{q}(y))+(1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{q}(y)),$$

In the ideal world,

$$\Pr\left[(a_i,b)^{\mathsf{Ideal}}=(0,0)\right]=\alpha(1-f(x,y))+(1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{c}_x^{(0)}(y)),$$

Thus, unless $\mathbf{p}(x)=1$ (in which case the two probabilities are obviously equal), equality holds if

$$\mathbf{c}_x^{(0)}(y)=\mathbf{q}(x)+\frac{\alpha}{(1-\alpha)(1-\mathbf{p}(x))}\cdot(1-f(x,y))\mathbf{q}(x)$$

**Second Case:** $(1,0)$**.** In the real world,

$$\Pr\left[(a_i,b)^{\mathsf{Real}}=(1,0)\right]=\alpha f(x,y)(1-\mathbf{q}(y))+(1-\alpha)\mathbf{p}(x)(1-\mathbf{q}(y)),$$

In the ideal world,

$$\Pr\left[(a_i,b)^{\mathsf{Ideal}}=(1,0)\right]=(1-\alpha)\mathbf{p}(x)(1-\mathbf{c}_x^{(1)}(y)),$$

Thus, unless $\mathbf{p}(x)=0$ (in which case the two probabilities are obviously equal), equality holds if

$$\mathbf{c}_x^{(1)}(y)=\mathbf{q}(x)+\frac{\alpha}{(1-\alpha)\mathbf{p}(x)}\cdot f(x,y)(1-\mathbf{q}(x))$$

**Third Case:** $(0,1)$**.** In the real world,

$$\Pr\left[(a_i,b)^{\mathsf{Real}}=(0,1)\right]=\alpha(1-f(x,y))\mathbf{q}(y)+(1-\alpha)(1-\mathbf{p}(x))\mathbf{q}(y),$$

In the ideal world,

$$\Pr\left[(a_i,b)^{\mathsf{Ideal}}=(0,1)\right]=(1-\alpha)(1-\mathbf{p}(x))\mathbf{c}_x^{(0)}(y),$$

Thus, unless $\mathbf{p}(x)=1$ (in which case the two probabilities are obviously equal), equality holds if

$$\mathbf{c}_x^{(0)}(y)=\mathbf{q}(x)+\frac{\alpha}{(1-\alpha)(1-\mathbf{p}(x))}\cdot(1-f(x,y))\mathbf{q}(x)$$

**Fourth Case:** $(1,1)$**.** In the real world,

$$\Pr\left[(a_i,b)^{\mathsf{Real}}=(1,1)\right]=\alpha f(x,y)\mathbf{q}(y)+(1-\alpha)\mathbf{p}(x)\mathbf{q}(y),$$

In the ideal world,

$$\Pr\left[(a_i,b)^{\mathsf{Ideal}}=(1,1)\right]=\alpha f(x,y)+(1-\alpha)\mathbf{p}(x)\mathbf{c}_x^{(1)}(y),$$

Thus, unless $\mathbf{p}(x)=0$ (in which case the two probabilities are obviously equal), equality holds if

$$\mathbf{c}_x^{(1)}(y)=\mathbf{q}(y)+\frac{\alpha}{(1-\alpha)\mathbf{p}(x)}\cdot f(x,y)(1-\mathbf{q}(y))$$

□

Equivalently, in matrix form, Theorem 3.2 can be stated as follows: if for some $\alpha\in(0,1)$, and all $x\in\{1,\ldots,\ell\}$, there exist *probability vectors* $\mathbf{x}_x^{(0)}$ and $\mathbf{x}_x^{(1)}$ such that

$$\begin{aligned}M^T\cdot\mathbf{x}_x^{(0)}&=\mathbf{q}+\lambda_x^{(0)}\cdot((\mathbf{1}-\mathsf{row}_x)*\mathbf{q})\\ M^T\cdot\mathbf{x}_x^{(1)}&=\mathbf{q}+\lambda_x^{(1)}\cdot(\mathsf{row}_x*(\mathbf{1}-\mathbf{q}))\end{aligned}\tag{3.1}$$

then GHKL computes $f$ with full security, where

$$\lambda_x^{(a)}=\begin{cases}\dfrac{\alpha}{(1-\alpha)(1-\mathbf{p}(x))} & \text{if } a=0\wedge\mathbf{p}(x)\neq 1\\ \dfrac{\alpha}{(1-\alpha)\mathbf{p}(x)} & \text{if } a=1\wedge\mathbf{p}(x)\neq 0\\ 0 & \text{if } \mathbf{p}(x)=1-a\end{cases}$$

and $\cdot*\cdot$ denotes the entry-wise (Hadamard) product.

## 3.2 The GHKL Criterion

By the distributive property of the Hadamard product, (3.1) is equivalent to

$$\begin{aligned}M^T\cdot\mathbf{x}_x^{(0)}&=\mathbf{q}+\lambda_x^{(0)}\cdot(\mathbf{q}-\mathbf{q}*\mathsf{row}_x)\\ M^T\cdot\mathbf{x}_x^{(1)}&=\mathbf{q}+\lambda_x^{(1)}\cdot(\mathsf{row}_x-\mathbf{q}*\mathsf{row}_x)\end{aligned}$$

Since $\mathbf{q}^T = (1/\ell, \dots, 1/\ell) \cdot M$, deduce that

$$M^T \cdot \mathbf{x}_x^{(0)} = M^T (1 + \lambda_x^{(0)}) \cdot \mathbf{1}_\ell/\ell - \lambda_x^{(0)} \cdot (\mathbf{q} * \mathsf{row}_x)$$
$$M^T \cdot \mathbf{x}_x^{(1)} = M^T (1 + \lambda_x^{(1)}) \cdot \left( \frac{\mathbf{1}_\ell/\ell + \lambda_x^{(1)} \mathbf{e}_x}{(1 + \lambda_x^{(1)})} \right) - \lambda_x^{(1)} (\mathbf{q} * \mathsf{row}_x),$$

and that

$$\mathbf{q} * \mathsf{row}_x = M^T \left( \frac{1 + \lambda_x^{(0)}}{\lambda_x^{(0)}} \cdot \mathbf{1}_\ell/\ell - \frac{1}{\lambda_x^0} \cdot \mathbf{x}_x^{(0)} \right).$$

Knowing that $\mathbf{1}_\ell/\ell$ and $\mathbf{x}_x^{(0)}$ are probability vectors, it follows that $\mathbf{q} * \mathsf{row}_x$ is an affine combination of the rows of $M$. Conversely, suppose there exists $\mathbf{v} \in \mathbb{R}^\ell$ such that $M^T \cdot \mathbf{v} = \mathbf{q} * \mathsf{row}_x$ and $\sum_i \mathbf{v}(i) = 1$. Let us show that we can find probability vectors satisfying (3.1). Fix $\alpha \in (0, 1)$ and define

$$\widetilde{\mathbf{x}}_x^{(0)} = (1 + \lambda_x^{(0)}) \cdot \mathbf{1}_\ell/\ell - \lambda_x^{(0)} \mathbf{v}$$
$$\widetilde{\mathbf{x}}_x^{(1)} = (1 + \lambda_x^{(1)}) \cdot \left( \frac{\mathbf{1}_\ell/\ell + \lambda_x^{(1)} \mathbf{e}_x}{(1 + \lambda_x^{(1)})} \right) - \lambda_x^{(1)} \mathbf{v}$$

If the vectors above are positive, then we are done. Otherwise, note that

$$\widetilde{\mathbf{x}}_x^{(a)}(y) \geq 1/\ell - \lambda_x^{(a)} \mathbf{v}(y),$$

for every $j \in \{1, \dots, \ell\}$. Since $\lambda_x^{(a)}$ approaches 0 as $\alpha$ tends to 0, there exists $\alpha_0 \in (0, 1)$ such that

$$\lambda_x^{(a)} \leq \frac{1}{\ell} \cdot \frac{1}{\max_j \{\mathbf{v}(j)\}} \ ,$$

for every $\alpha \leq \alpha_0$ and $x \in X$. The theorem below follows from our discussion.

**Theorem 3.3** (GHKL Criterion). *If $\mathbf{v} \mapsto \mathbf{q} * \mathbf{v}$ is an endomorphism of $\mathcal{H}(M^T)$, then there exists $\alpha \in (0, 1)$ such that GHKL computes $f$ with full security.*

### 3.2.1 Alternative Formulation of the Criterion

**Lemma 3.4.** *Assuming it exists, let $\mathbf{b}$ denote the unique pre-image of $\mathbf{1}_\ell$ by $M$ that belongs to $\mathrm{im}(M^T)$, and let $Q$ denote an arbitrary diagonal matrix. It holds that $Q$ is an endomorphism of $\mathcal{H}(M^T)$ if and only if*

1. *$Q$ is a (linear) endomorphism of $\mathrm{im}(M^T)$.*

2. $\mathbf{b}$ *is a fixed point of* $Q$, *i.e.* $Q \cdot \mathbf{b} = \mathbf{b}$.

*If* $\mathbf{b}$ *does not exist then the equivalence holds with the first item alone.*

*Proof.* ($\Rightarrow$) Suppose that $Q$ is an endomorphism of $\mathcal{H}(M^T)$. By definition, $Q \cdot \mathsf{row}_x \in \operatorname{im}(M^T)$ and since $Q$ is a linear map, the first item is trivially true. For the second item, we know that for some affine vector $\mathbf{v}$, it holds that $Q \cdot \mathsf{row}_x = M^T \cdot \mathbf{v}$. Since $\sum_i \mathbf{v}(i) = 1$, it follows that for every $x \in \{1, \ldots, \ell\}$,

$$\mathbf{b}^T \cdot QM^T \cdot \mathbf{e}_x = 1 \ .$$

Equivalently, $MQ \cdot \mathbf{b} = \mathbf{1}_\ell$, and thus $Q \cdot \mathbf{b} = \mathbf{b} + \mathbf{v}_0$, for some $\mathbf{v}_0 \in \ker(M)$. To conclude, note that $Q$ is an endomorphism of $\operatorname{im}(M)^T$, and thus $\mathbf{b} + \mathbf{v}_0 \in \operatorname{im}(M)^T$ implies that $\mathbf{v}_0 = \mathbf{0}$. ($\Leftarrow$) From the first item, it follows that for every $x$, there exists $\mathbf{v}$ such that $M^T \mathbf{v} = Q \cdot \mathsf{row}_x$. Using the fact that $Q \cdot \mathbf{b} = \mathbf{b}$, deduce that

$$\begin{aligned} \mathbf{b}^T \cdot M^T \cdot \mathbf{v} = \mathbf{b}^T \cdot Q \cdot \mathsf{row}_x \quad \Rightarrow \\ (1, \ldots, 1) \cdot \mathbf{v} = \mathbf{b}^T \cdot \mathsf{row}_x = 1 \ \ , \end{aligned}$$

, and thus $\mathbf{v}$ is affine. To conclude, we show that if $\mathbf{b}$ does not exist then $\mathcal{H}(M^T) = \operatorname{im}(M^T)$, and thus the equivalence holds with the first item alone. If $\mathbf{1}_\ell \notin \operatorname{im}(M)$, then there exists $\mathbf{v}_0 \in \ker(M)$ such that $\langle \mathbf{1}_\ell \,|\, \mathbf{v}_0 \rangle \neq 0$. Hence, $\mathbf{v} = M^T \mathbf{u}$ implies that

$$\mathbf{v} = M^T \left( \mathbf{u} + \frac{1 - \langle \mathbf{1}_\ell \,|\, \mathbf{u} \rangle}{\langle \mathbf{1}_\ell \,|\, \mathbf{v}_0 \rangle} \cdot \mathbf{v}_0 \right),$$

and thus $\mathbf{v}$ is an affine combination of the rows of $M$. □

**Proposition 3.5.** *Let* $Q$ *denote the diagonal matrix such that* $Q(i,i) = \mathbf{q}(i)$. *It holds that* $Q$ *is an endomorphism of* $\mathcal{H}(M^T)$ *if and only if*

1. $Q$ *is an (linear) endomorphism of* $\operatorname{im}(M^T)$,

2. *no linear combination of the non-monochromatic columns of* $M$ *yields* $\mathbf{1}_\ell$.

*Proof.* Assume the contrary and let $\mathbf{b}$ denote the unique pre-image[1] of $\mathbf{1}_\ell$ that belongs to $\operatorname{im}(M^T)$. Assume that $Q$ is an endomorphism of $\mathcal{H}(M^T)$

[1] If $\mathbf{1}_\ell \notin \operatorname{im}(M)$, then the proposition follows trivially from the lemma.

and apply the previous lemma. Since $Q \cdot \mathbf{b} = \mathbf{b}$, and $Q$ is diagonal, it follows that $\mathbf{b}(y) \neq 0$ implies $\mathbf{q}(y) = 1$. Furthermore, since $\mathbf{q}(y) = \langle \mathbf{1}_\ell/\ell \,|\, \mathsf{col}_y \rangle$, we deduce that $\mathbf{b}(y) \neq 0$ implies $\mathsf{col}_y = \mathbf{1}_\ell$. Without loss of generality, suppose that the first $k'$ columns contain $\mathbf{1}_\ell$ in their linear span, and not one of them is monochromatic. To find a contradiction, we show that $\mathbf{b}(y) \neq 0$, for some $y \in \{1, \ldots, k'\}$. Let $\mathbf{v} = (v_1, \ldots, v_{k'}, 0, \ldots, 0)$, such that $M \cdot \mathbf{v} = \mathbf{1}_\ell$. It follows that $\mathbf{v} = \mathbf{b} + \mathbf{v}_0$, for some $\mathbf{v}_0 \in \ker(M)$. Consequently, $\langle \mathbf{v} \,|\, \mathbf{b} \rangle = \langle \mathbf{b} \,|\, \mathbf{b} \rangle > 0$, and we conclude that at least one of the first $k'$ entries of $\mathbf{b}$ is non-zero.

For the converse, since the non-monochromatic columns of $M$ do not span $\mathbf{1}_\ell$, it holds that $\mathbf{b}(y) \neq 0$ implies $\mathsf{col}_y = \mathbf{1}$. Let

$$\mu = \Big\{ y \in \{1, \ldots, k\} \,\Big|\, \mathbf{b}(y) \neq 0 \wedge \mathsf{col}_y \neq \mathbf{1}_\ell \Big\},$$

and compute

$$\begin{aligned} \sum_{y \in \mu} \mathbf{b}(y) \cdot \mathsf{col}_y &= \sum_{y} \mathbf{b}(y) \cdot \mathsf{col}_y - \sum_{y \notin \mu} \mathbf{b}(y) \cdot \mathbf{1}_\ell \\ &= M \cdot \mathbf{b} - \left( \sum_{y \notin \mu} \mathbf{b}(y) \right) \cdot \mathbf{1}_\ell = \left( 1 - \sum_{y \notin \mu} \mathbf{b}(y) \right) \cdot \mathbf{1}_\ell \ . \end{aligned} \tag{3.2}$$

Define $\mathbf{b}'$ such that $\mathbf{b}'(y) = \mathbf{b}(y)$ if $y \in \mu$, and 0 otherwise. By assumption, since the non-monochromatic columns of $M$ do not span $\mathbf{1}_\ell$, equation (3.2) implies that $1 - \sum_{y \notin \mu} \mathbf{b}(y) = 0$, and thus $\mathbf{b}' \in \ker(M)$. However, since $\langle \mathbf{b} \,|\, \mathbf{b}' \rangle = \sum_{y \in \mu} \mathbf{b}(y)^2$, we deduce that $\mathbf{b}' = \mathbf{0}$, and that $\mu$ is the empty set. Finally, knowing that $\mathbf{q}(y) = \langle \mathbf{1}_\ell/\ell \,|\, \mathsf{col}_y \rangle$, we conclude that $Q \cdot \mathbf{b} = \mathbf{b}$. □

**Corollary 3.6** (The Simplified Criterion)**.** *Using the notation above, matrix $Q$ is an endomorphism of $\mathcal{H}(M^T)$ if and only if $M\mathbf{v} = MQ\mathbf{v}$, for every $\mathbf{v}$ such that $M \cdot \mathbf{v} \in \langle \mathbf{1}_\ell \rangle$.*

*Proof.* Since $Q$ is diagonal, $Q$ is an endomorphism of $\mathrm{im}(M^T)$ if and only if $Q$ is an endomorphism of $\ker(M)$ i.e. $\forall \mathbf{v} \in \ker(M)$, $MQ\mathbf{v} = \mathbf{0}_\ell$. On the other hand, assuming it exists, let $\mathbf{b}$ be as in Lemma 3.4. If $Q$ is an endomorphism of $\mathrm{im}(M^T)$, then $\mathbf{1}_\ell = M\mathbf{b} = MQ\mathbf{b}$ implies that $\mathbf{b}$ is a fixed point of $Q$, since $\mathbf{b}$ is unique. □

Notice how easy it is to verify whether a given function satisfies the GHKL criterion. Let $\mathbf{B}$ denote an arbitrary basis of $\{\mathbf{v} \in \mathbb{R}^k \,|\, M\mathbf{v} \in \langle \mathbf{1}_\ell \rangle\}$. It suffices to check whether $MQ\mathbf{v} = M\mathbf{v}$, for every $\mathbf{v} \in \mathbf{B}$.

## 3.3 Limits of GHKL

In this section, we show that the converse of Theorem 3.3 is true, and we dedicate most of this section to the proof of the following proposition.

**Proposition 3.7.** *Let $Q$ be as in Proposition 3.5. If $Q$ is not an endomorphism of $\mathcal{H}(M^T)$, then for any $\alpha$ such that $\alpha^{-1} \in \mathsf{poly}(n)$, it holds that GHKL is not fully-secure.*

As an example, consider the function described by

$$M = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} .$$

Recall that $\mathbf{q} = M^T \cdot \mathbf{1}/\ell$ and $Q \cdot \mathbf{v} = \mathbf{q} * \mathbf{v}$. By the fundamental theorem of Linear Algebra, it holds that $\mathbf{v} \in \operatorname{im}(M^T)$ if and only if $\sum(-1)^i \cdot \mathbf{v}(i) = 0$. Observe that $\mathbf{q} * \mathsf{row}_1 \notin \operatorname{im}(M^T)$, and thus $Q$ is *not* an endomorphism of $\mathcal{H}(M^T)$. We show that GHKL is susceptible to an attack for this particular function, regardless of the choice of $\alpha$.

Suppose that $P_2$ chooses an input uniformly at random, and flips his output if $y \in \{y_1, y_3\}$. Notice that, by following these instructions in the ideal model, $P_2$ is guaranteed to output a uniform random bit, no matter what. Back to the real model, assume that $P_1$ plays $x_1$ and quits immediately if $a_1 = 0$, or upon seeing $a_2$, otherwise. Let's compute $\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1\right]$ under these assumptions, where $\mathsf{out}_2^{\mathsf{R}}$ denotes $P_2$'s output in the real model.

$$\begin{aligned}
\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1\right] &= \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 1\right] + \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* \neq 1\right] \\
&= \alpha \cdot \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \,\middle|\, i^* = 1\right] + (1-\alpha) \cdot 1/2 \\
&= \alpha \cdot \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge y \in \{1,2\} \,\middle|\, i^* = 1\right] \\
&\qquad + \alpha \cdot \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge y \in \{3,4\} \,\middle|\, i^* = 1\right] + (1-\alpha) \cdot 1/2 \\
&= \alpha \cdot (1/2 \cdot 1/2 + 1/2 \cdot 3/8) + (1-\alpha) \cdot 1/2 \\
&= 1/2 - \alpha \cdot 1/16 \ .
\end{aligned}$$

For the general case, we prove that GHKL is susceptible to an attack for any function that does not satisfy the hypothesis of the Corollary 3.6. Fix

$\mathbf{v}$ such that $M \cdot \mathbf{v} \in \langle \mathbf{1}_\ell \rangle$ and $M\mathbf{v} \neq MQ\mathbf{v}$. Let $\delta_2 \cdot \mathbf{1}_\ell = M\mathbf{v}$, and, without loss of generality, suppose that $\mathbf{e}_{x_1}^T M\mathbf{v} \neq \mathbf{e}_{x_1}^T MQ\mathbf{v}$. Consider protocol $\Pi$ in the $f$-hybrid model.

**Protocol** $\Pi$

1. **Inputs**: $P_1$ holds $x \in \{1, \ldots, \ell\}$, $P_2$'s input domain is empty.
2. $P_2$ chooses $y \in \{1, \ldots, k\}$ with probability $|\mathbf{v}(y)|$.
3. The parties invoke the trusted party computing $f$ on inputs $x$ and $y$. The trusted party hands $\phi = f(x, y)$ to both parties.
4. **Outputs**: The first party outputs $\phi$, the second party outputs $\phi$ if $\mathbf{v}(y) > 0$ and $1 - \phi$ otherwise.

Figure 3.3: Protocol $\Pi$ in the $f$-hybrid Model.

**Claim 3.8.** *In the hybrid model with ideal access to $f$, it holds that*

$$\Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1\right] = \delta_2 + \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)| \ ,$$

*where $\mathsf{out}_2^{\mathsf{H}}$ denote $P_2$'s.*

*Proof.* Write $(x', y')$ for the pair of inputs that $P_1$ and $P_2$ hand to the trusted party.

$$\begin{aligned}
\Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1 \,\middle|\, x' = x\right] &= \sum_{y \in Y} |\mathbf{v}(y)| \cdot \Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1 \,|\, x' = x \wedge y' = y\right] \\
&= \sum_{\mathbf{v}(y)\geq 0} \mathbf{v}(y) \cdot \Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1 \,|\, x' = x \wedge y' = y\right] \\
&\qquad + \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)| \cdot \Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1 \,|\, x' = x \wedge y' = y\right] \\
&= \sum_{\mathbf{v}(y)\geq 0} \mathbf{v}(y) \cdot f(x, y) + \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)| \cdot (1 - f(x, y)) \\
&= \mathbf{e}_x^T \cdot M\mathbf{v} + \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)|
\end{aligned}$$

□

In the real model, consider a corrupt $P_1$ that deviates from the protocol as follows: Use $x_1$ as an input and quit at round 1 or 2 if $a_1 = 1$ or $a_1 = 0$, respectively. Let's compute the real-world probability distribution of $P_2$'s output. Write $\mathsf{out}_2^{\mathsf{R}}$ for $P_2$'s output in the plain-model.

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1\right] = \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 1\right] + \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* > 1\right] .$$

If $i^* > 1$, then $a_1$ is independent of both $b_0$ and $b_1$. Thus,

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* > 1\right] = (1-\alpha) \cdot \left(\delta_2 + \sum_{\mathbf{v}(i)<0} |\mathbf{v}(i)|\right) . \tag{3.3}$$

On the other hand,

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 1\right] = \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge a_1 = 0 \wedge i^* = 1\right] + \\ \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge a_1 = 1 \wedge i^* = 1\right] .$$

So, let us compute each of the terms above.

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge a_1 = 1 \,\middle|\, i^* = 1\right] = \\ \sum_{\mathbf{v}(y)\geq 0} \mathbf{v}(y) \cdot f(x_1, y) \cdot \left(\mathbf{1}_\ell^T/\ell \cdot \mathsf{col}_y\right) + \\ \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)| \cdot f(x_1, y) \cdot \left(1 - \mathbf{1}_\ell^T/\ell \cdot \mathsf{col}_y\right) ,$$

and

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge a_1 = 0 \,\middle|\, i^* = 1\right] = \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)| \cdot (1 - f(x_1, y)) .$$

It follows that

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 1\right] = \alpha \cdot \left(\mathbf{e}_{x_1}^T M Q \mathbf{v} + \sum_{\mathbf{v}(y)<0} |\mathbf{v}(y)|\right) . \tag{3.4}$$

Combine (3.3) and (3.4), and conclude that if $\alpha$ is noticeable, then $P_2$'s output is noticeably far from the prescribed distribution. □

Chapter 4

# Inputless Functionalities

An inputless functionality is a random process that takes no inputs and returns outputs to the parties according to some joint probability distribution. Every known limitation of fairness can be attributed to inputless functionalities. That being said, fair computation of inputless functionalities is a strange topic in our view. If we take step back from the ideal-world paradigm, it would appear that inputless functionalities are trivial as far as fairness is concerned. Recall that an informal way to define fairness is to say that “one party learns the correct output, if everybody does”. When the parties are not holding any inputs, it is unclear what the “correct” output is - if there is one at all. As we shall see later on, this intuition is false.

Chapter 4 begins with the generalization of Cleve’s theorem by Agrawal and Prabhakaran [1]. Then, we discuss optimally fair coin-tossing, and we show that Cleve’s lower bound for the bias is tight non-asymtotically. Finally, we return to Boolean functions, and we present the *balanced* criterion of Asharov, Lindell and Rabin that we generalize into the *semi-balanced* criterion.

The present chapter is peculiar in that it deviates from the characterization of Boolean functions. We do so in order to settle a long-standing question regarding the exact trade-off between bias and round-complexity in two-party coin-tossing. Another peculiarity of the chapter is that our own contributions appear in the middle of the chapter and then at the end. The choice is deliberate and allows for a smoother exposition.

**Coin-Tossing.** Processes that emulate the tossing of a physical coin by informational means alone are known as coin-tossing schemes. The design of such schemes is a classic cryptographic problem dating back to the early days of modern cryptography [17] circa 1981. In the jargon of MPC, Coin-tossing is the inputless functionality that returns the same uniform bit to a number of parties. On the other hand, *sampling* is a generic term applied to any inputless functionality, other than coin-tossing.

Prior to the advent of simulation based security, it was customary to consider each security issue separately. For inputless functionalities, perhaps the most important security issue relates to the probability distribution of the honest party's output. Namely, in any sampling protocol, the probability distribution of the honest party's output should be as close as possible to the functionality's prescribed distribution - for coin-tossing that would be the uniform distribution. This security requirement is captured by a metric called the *bias*, which is a positive function of the protocols' round-complexity. The further away the bias is from 0, the easier it is for the adversary to effectively choose the honest party's output. In particular, a bias of 0 indicates that the output is distributed according to the functionality's specifications, no matter what. Alas, no such protocol exists in the plain model. This remarkable result, that we owe to Cleve [26], has profound consequences for fairness. To illustrate, consider Blum's protocol in the online dealer model from Figure 4.1.

**Blum's Coin-Tossing**

1. The dealer hands a uniform random bit to $P_1$, say $b$.
2. Party $P_1$ decides to either continue or abort the protocol.
   - $P_1$ **continues**: $P_2$ receives $b$ from the dealer.
   - $P_1$ **aborts**: $P_2$ receives $b'$ from the dealer, where $b'$ is a uniform random bit and is independent of $b$.
3. **Outputs**: The parties output whatever they received from the dealer.

Figure 4.1: Blum's Coin-Tossing Protocol in the Online Dealer Model.

Suppose that $P_1$ is corrupted, and that he aborts the computation whenever $b = 0$. In that case, $P_2$ outputs $b'$. Write $\mathsf{out}_2$ for $P_2$'s output and compute $\Pr[\mathsf{out}_2 = 1] = \Pr[\mathsf{out}_2 = 1 \wedge b = 0] + \Pr[\mathsf{out}_2 = 1 \wedge b = 1] =$

$$= \Pr\left[b' = 1 \wedge b = 0\right] + \Pr[b = 1]$$
$$= \frac{1}{4} + \frac{1}{2} = \frac{3}{4} \ .$$

Observe that even though $b$ and $b'$ are uniform random bits, the probability distribution of $P_2$'s output is 1/4 for 0 and 3/4 for 1. In the jargon of coin-tossing, $P_2$'s output suffers a bias of $1/4 = |\Pr[\mathsf{out}_2 = 1] - 1/2|$.

As we shall see in the following section, the attack above is not specific to Blum's protocol. Cleve [26] showed that any coin-tossing protocol is susceptible to a variant of this attack, and any coin-tossing protocol has a bias of *at least* $O(1/r)$, where $r$ denotes the number of rounds. This result was generalised by Agrawal and Prabhakaran [1] for any (non-trivial) sampling protocol. On the flip side, variants of Blum's protocol guarantee a bias of *at most* $O(1/\sqrt{r})$, and, by means of a special-round protocol, Malkin, Naor and Segev [66] matched Cleve's bound asymptotically. However, there is a discrepancy between the bounds of [26] and [66] by a factor of 1/2, and the question of identifying the *exact* trade-off (i.e. non-asymptotically) remained open. In Section 4.2, we settle the question by showing that Cleve's bound is the correct one.

## 4.1 The Characterization of Inputless Functionalities

We visit Cleve's theorem and its generalization by Agrawal and Prabhakaran. We begin by defining (Boolean) sampling functionalities (Figure 4.2).

**The Sampling Functionality $\mathfrak{S}(\mathcal{D})$**

- **Parameter:** Distribution $\mathcal{D}$ over $\{0,1\}^2$.
- **Inputs:** Empty for both parties.
- **Outputs:** $P_1$ and $P_2$ receive bits $a$ and $b$ respectively, such that $(a, b) \leftarrow \mathcal{D}$.

Figure 4.2: The Sampling Functionality.

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ denote the marginal distributions of the parties' outputs. In other words, $\mathcal{D}_i$ denotes the distribution of $P_i$'s output that is implied by $\mathcal{D}$. Furthermore, let $\mathcal{D}_1 \times \mathcal{D}_2$ denote the distribution over $\{0,1\}^2$ that samples $a$ and $b$ independently according to $\mathcal{D}_1$ and $\mathcal{D}_2$ respectively. Finally, let $\mathsf{SD}(\mathcal{D}, \times)$ denote the statistical distance between $\mathcal{D}$ and $\mathcal{D}_1 \times \mathcal{D}_2$.

**Claim 4.1.** *Suppose that $a$ and $b$ are sampled according to $\mathcal{D}$, and that $a'$ and $b'$ are sampled according to $\mathcal{D}_1 \times \mathcal{D}_2$. It holds that $|\Pr[a = b] - \Pr[a' = b']| = \mathsf{SD}(\mathcal{D}, \times)$.*

*Proof.* First, note that $\Pr[a = b] - \Pr[a' = b'] =$

$$\frac{\Pr[a = b] - \Pr[a' = b']}{2} + \frac{\Pr[a' \neq b'] - \Pr[a \neq b]}{2} .$$

Next, observe that $\Pr[(a, b) = (i, i)] - \Pr[(a', b') = (i, i)]$ shares the same sign with $-\Pr[(a, b) = (j, 1 - j)] + \Pr[(a', b') = (j, 1 - j)]$ since the two expressions are in fact equal. Putting everything together, it holds that

$$\Bigg| \sum_{i \in \{0,1\}} \Pr[(a, b) = (i, i)] - \Pr[(a, b) = (i, i)] + \sum_{j \in \{0,1\}} \Pr\big[(a', b') = (j, 1 - j)\big] - \Pr[(a, b) = (j, 1 - j)] \Bigg| =$$

$$\sum_{i \in \{0,1\}} |\Pr[(a, b) = (i, i)] - \Pr[(a, b) = (i, i)]| + \sum_{j \in \{0,1\}} |\Pr\big[(a', b') = (j, 1 - j)\big] - \Pr[(a, b) = (j, 1 - j)]| .$$

□

**Definition 4.2** (Bias)**.** Let $\Pi$ be a protocol for computing $\mathfrak{S}(\mathcal{D})$. Write $\mathsf{out}_i$ for the output of $P_i$, and suppose that $P_{3-i}$ is corrupted. The bias of the honest party's output is defined to be

$$\big|\Pr[\mathsf{out}_i = 1] - \Pr[z = 1 \mid z \leftarrow \mathcal{D}_i]\big| .$$

By extension, the bias of the protocol refers to the maximum bias over all possible corruptions.

The next theorem relates the bias of any given sampling protocol to its round complexity. The proof that we provide is similar to the ones that appear in the literature [1, 26]. The idea is to consider a specific family of attacks, and to argue that *on average* these attacks inflict a noticeable bias on the honest party's output.

Our approach differs in that we analyse the attacks independently of one another. Specifically, we pinpoint the conditions under which any attack is successful, and we apply the averaging argument to these conditions (rather than the biases).

**Theorem 4.3** (Cleve's Theorem [1, 26]). *Let $\Pi$ be a correct protocol for computing $\mathfrak{S}(\mathcal{D})$ in the plain model. It holds that* $\mathsf{bias} \geq \mathsf{SD}(\mathcal{D}, \times)/4r$, *where $r$ denotes the number of rounds.*

*Proof.* We model $\Pi$ as follows: Each round consists of a single interaction between the parties. Party $P_2$ sends a message followed by $P_1$ who waits for the message to arrive and replies. At the beginning of round $i$, parties $P_1$ and $P_2$ hold bits $a_{i-1}$ and $b_{i-1}$, respectively. If either party aborts at round $i$, the remaining party is instructed to output the bit he has at hand. Upon receiving their $i$-th round message, the parties update their back-up outputs to $a_i$ and $b_i$ respectively.

We assume that the protocol is a correct, and, as such, $a_r$ and $b_r$ are sampled according to distribution $\mathcal{D}$. We also assume that $a_0$ and $b_0$ are sampled according to $\mathcal{D}_1 \times \mathcal{D}_2$. We argue that this last assumption is sensible even though it incurs some loss of generality. On one hand, $a_0$ and $b_0$ are independent random variables given that they are constructed before any communication occurs. On the other hand, if $a_0$ (resp. $b_0$) is not sampled according to $\mathcal{D}_1$ (resp. $\mathcal{D}_2$) then, by aborting immediately, an adversary corrupting $P_2$ (resp. $P_1$) will successfully bias the honest party's output.

Suppose there is a noticeable difference between $\Pr[a_i = b_{i-1}]$ and $\Pr[a_i = b_i]$. We claim that one of the following adversaries corrupting $P_1$ can bias $P_2$'s output:

- $\mathcal{A}_i^{(0)}$ quits at round $i$ if $a_i \neq 0$, or at round $i+1$ if not.
- $\mathcal{A}_i^{(1)}$ quits at round $i$ if $a_i \neq 1$, or at round $i+1$ if not.

Let $\varepsilon = |\Pr[a_i = b_i] - \Pr[a_i = b_{i-1}]|$. Let's compute the bias of each of these adversaries. First, note that

$$\Pr\left[\mathsf{out} = 0 \,\middle|\, \mathcal{A}_i^{(0)}\right] = \Pr[a_i = 1 \wedge b_{i-1} = 0] + \Pr[a_i = 0 \wedge b_i = 0]$$
$$\Pr\left[\mathsf{out} = 1 \,\middle|\, \mathcal{A}_i^{(1)}\right] = \Pr[a_i = 0 \wedge b_{i-1} = 1] + \Pr[a_i = 1 \wedge b_i = 1]$$

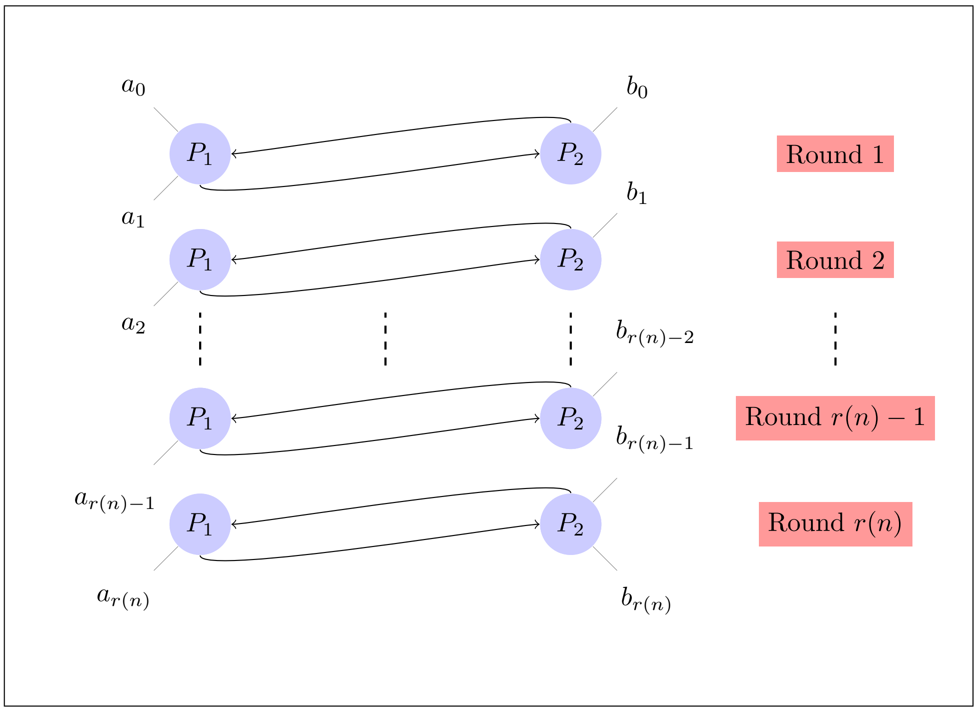

Figure 4.3: Generic Protocol with Well Defined Outputs.

Thus, on average

$$\begin{aligned}
\mathsf{bias} &\geq \frac{1}{2}\left|\Pr\left[\mathsf{out}=0 \,\middle|\, \mathcal{A}_i^{(0)}\right] - (1-q) + \Pr\left[\mathsf{out}=1 \,\middle|\, \mathcal{A}_i^{(1)}\right] - q\right| \\
&\geq \frac{1}{2}\cdot\left|\Pr\left[a_i \neq b_{i-1}\right] + \Pr\left[a_i = b_i\right] - 1\right| \\
&\geq \frac{1}{2}\cdot\left|\Pr\left[a_i = b_i\right] - \Pr\left[a_i = b_{i-1}\right]\right| = \frac{\varepsilon}{2}.
\end{aligned}$$

where $q = \Pr\left[b = 1 \,|\, b \leftarrow \mathcal{D}_2\right]$. Similarly, if there is a noticeable $\varepsilon$-gap between $\Pr\left[b_i = a_{i+1}\right]$ and $\Pr\left[b_i = a_i\right]$, then an adversary corrupting $P_2$ can bias the first party's output by at least $\varepsilon/2$. For $i \in \{1, \ldots, r\}$ define

$$\begin{cases}
\varepsilon_i^{(1)} = \left|\Pr\left[a_i = b_i\right] - \Pr\left[a_i = b_{i-1}\right]\right| \\
\varepsilon_i^{(2)} = \left|\Pr\left[b_{i-1} = a_i\right] - \Pr\left[b_{i-1} = a_{i-1}\right]\right|
\end{cases},$$

and let $\varepsilon$ denote the arithmetic mean of $\{\varepsilon_i^{(1)}, \varepsilon_i^{(2)}\}_{i=1\dots r}$. It follows that

$$\begin{aligned}
\varepsilon &= \frac{1}{2r} \cdot \left( \sum_{i=1}^{r} \varepsilon_i^{(1)} + \varepsilon_i^{(2)} \right) \\
&\geq \frac{1}{2r} \cdot \left| \sum_{i=1}^{r} \Pr[a_i = b_i] - \Pr[a_i = b_{i-1}] + \Pr[b_{i-1} = a_i] - \Pr[b_{i-1} = a_{i-1}] \right| \\
&\geq \frac{1}{2r} \cdot \left| \sum_{i=1}^{r} -\Pr[a_{i-1} = b_{i-1}] + \Pr[b_i = a_i] \right| \\
&= \frac{1}{2r} \cdot \Big| \Pr[b_r = a_r] - \Pr[a_0 = b_0] \Big| = \frac{\mathsf{SD}(\mathcal{D}, \times)}{2r}
\end{aligned}$$

The last equality follows from Claim 4.1. We conclude that at least one of the $\varepsilon_i^{(j)}$ is greater than $\mathsf{SD}(\mathcal{D}, \times)/2r$, meaning that there exists an adversary that can bias the honest party's output by at least $\mathsf{SD}(\mathcal{D}, \times)/4r$. □

**Corollary 4.4.** *Unless $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$, functionality $\mathfrak{S}(\mathcal{D})$ is not computable with complete fairness.*

## 4.2 The Exact Trade-off between Bias and Round-Complexity

Cleve's theorem stipulates that any protocol realizing the coin-tossing functionality suffers a bias of at least $1/8r$. The MNS protocol of Malkin, Naor and Segev is known to reach this bound *asymptotically.* Specifically, in [66], the authors present a protocol with a bias of at most $1/4r$. We show that Cleve's bound is the correct one, assuming the existence of oblivious transfer.

The protocol of Malkin, Naor and Segev can be viewed as a variant of GHKL computing $(\emptyset, \emptyset) \mapsto \lambda$, where $\lambda$ is a uniform bit, and with $i^*$ chosen uniformly at random instead of according to a geometric distribution. In more detail, prior to the special round, each party observes a sequence of identically and independently sampled bits. From the special round and onwards, both sequences stabilize to the same fixed value chosen uniformly at random and independently of the backups that preceded. We stress that, prior to $i^*$, the dealer taps into fresh randomness to compute the parties' backup outputs. This is precisely what enables the adversary to inflict a bias that is strictly greater than what Cleve's bound suggests.

**MNSdrnd**

1. The dealer fixes the number of rounds $r$, where $r(n) \in \mathsf{poly}(n)$.
2. The dealer chooses $i^* \in \{1, \ldots, r(n)\}$ according to the *uniform* distribution.
3. The dealer samples independently $\mathsf{out} \in_U \{0,1\}$ and $\widetilde{b} \in_U \{0,1\} \in$, and constructs sequences $(a_1, \ldots, a_r)$, $(b_1, \ldots, b_r)$ such that

$$a_i = \begin{cases} \mathsf{out} + i^* - i \mod 2 & \text{if } i < i^* \\ \mathsf{out} & \text{if } i \geq i^* \end{cases},$$

and

$$b_i = \begin{cases} \widetilde{b} & \text{if } i < i^* \\ \mathsf{out} & \text{if } i \geq i^* \end{cases}.$$

4. The dealer hands $a_0 \in_U \{0,1\}$ to $P_1$, and $b_0 \in_U \{0,1\}$ to $P_2$.
5. For $i = 1, \ldots, r$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

Figure 4.4: A Derandomized Variant of the MNS Protocol.

We define a variant of protocol MNS obtained by derandomization (Figure 4.4 p. 48). Using the notation from the figure, notice that the players observe the following sequences:

$$\begin{array}{llcccccccc} P_1 & : \cdots & \overline{\mathsf{out}} & \mathsf{out} & \overline{\mathsf{out}} & \mathsf{out} & \overline{\mathsf{out}} & \mathsf{out} & \mathsf{out} & \cdots \\ P_2 & : \cdots & \widetilde{b} & \widetilde{b} & \widetilde{b} & \widetilde{b} & \widetilde{b} & \mathsf{out} & \mathsf{out} & \cdots \\ & & & & & & & \uparrow & & \\ & & & & & & & i^* & & \end{array}$$

where $\overline{\mathsf{out}}$ denotes $\mathsf{out} \oplus 1$. Let's compute the bias for a class of adversaries with fail-stop strategies that depend only on the value of $a_1$. Write $\mathbf{A}_1$ to denote the class and let $\mathcal{A} \in \mathbf{A}_1$. Furthermore, let $\Delta_1$ and $\Delta_0$ be two probability distributions over $\{1, \ldots, r, r+1\}$ and write $p_i^{(a)}$ for the probability that $\mathcal{A}$ quits at round $i$, given that $a_i = a$. In other words, if $a_1 = 0$ then the adversary quits at round $i_0 \leftarrow \Delta_0$. If $a_1 = 1$ then the adversary quits at round $i_1 \leftarrow \Delta_1$. Write $\mathsf{out}_2$ for $P_2$'s output. Let's compute the bias caused $\mathcal{A}$. Observe that

$$\begin{aligned}\Pr[\mathsf{out}_2 = 1] &= \Pr[a_1 = 0 \wedge \mathsf{out}_2 = 1] + \Pr[a_1 = 1 \wedge \mathsf{out}_2 = 1] \\ &= \frac{1}{2} \cdot \Pr[\mathsf{out}_2 = 1 \mid a_1 = 0] + \frac{1}{2} \cdot \Pr[\mathsf{out}_2 = 1 \mid a_1 = 1],\end{aligned}$$

and

$$\begin{aligned}&\Pr[\mathsf{out}_2 = 1 \mid a_1 = a] = \\ &\qquad \sum_{i \text{ odd}} p_i^{(a)} \cdot \Pr[b_{i-1} = 1 \mid a_1 = 1] + \sum_{i \text{ even}} p_i^{(a)} \cdot \Pr[b_{i-1} = 1 \mid a_1 = a] \quad .\end{aligned}$$

**Claim 4.5.** *For every $i \in \{1, \ldots, r+1\}$ and $a \in \{0,1\}$, it holds that*

$$\Pr[b_{i-1} = 1 \mid a_1 = a] = \begin{cases} \frac{1}{2} & i \text{ odd} \\ \frac{1}{2} + (2a-1) \cdot \frac{1}{2r} & i \text{ even} \end{cases} .$$

*Proof.*

$$\begin{aligned}\Pr[b_{i-1} = 1 \mid a_1 = 0] &= \Pr[i^* \text{ even} \wedge i^* \le i-1] + \frac{1}{2} \cdot \Pr[i^* \ge i] \\ &= \begin{cases} \frac{i-1}{2r} + \frac{r-i+1}{2r} & i \text{ odd} \\ \frac{i-2}{2r} + \frac{r-i+1}{2r} & i \text{ even} \end{cases} = \begin{cases} \frac{1}{2} & i \text{ odd} \\ \frac{1}{2} - \frac{1}{2r} & i \text{ even} \end{cases} .\end{aligned}$$

Similarly,

$$\begin{aligned}\Pr[b_{i-1} = 1 \mid a_1 = 1] &= \Pr[i^* \text{ odd} \wedge i^* \le i-1] + \frac{1}{2} \cdot \Pr[i^* \ge i] \\ &= \begin{cases} \frac{i-1}{2r} + \frac{r-i+1}{2r} & i \text{ odd} \\ \frac{i}{2r} + \frac{r-i+1}{2r} & i \text{ even} \end{cases} = \begin{cases} \frac{1}{2} & i \text{ odd} \\ \frac{1}{2} + \frac{1}{2r} & i \text{ even} \end{cases} .\end{aligned}$$

□

Let $p^{(a)} = \sum_{i \text{ even}} p_i^{(a)}$ and deduce that

$$\begin{aligned}\Pr[\mathsf{out}_2 = 1 \mid a_1 = a] &= (1 - p^{(a)}) \cdot \frac{1}{2} + p^{(a)} \cdot \left(\frac{1}{2} - \frac{1}{2r}\right) \\ &= \frac{1}{2} - p^{(a)} \cdot \frac{1}{2r} \quad .\end{aligned}$$

And thus

$$\begin{aligned}\Pr\left[\mathsf{out}_2 = 1\right] &= \frac{1}{4} - p^{(0)} \cdot \frac{1}{4r} + \frac{1}{4} + p^{(1)} \cdot \frac{1}{4r} \\ &= \frac{1}{2} + \frac{1}{4r}\left(p^{(1)} - p^{(0)}\right) \ .\end{aligned}$$

It follows that $\mathsf{bias} = \frac{1}{4r} \cdot \left|p^{(1)} - p^{(0)}\right|$ and we conclude that the maximum over all the members of $\mathbf{A}_1$ is $1/(4r)$.

To conclude, we argue that no adversary can do better than the one described above. Let $\mathcal{A}$ denote an arbitrary adversary corrupting $P_1$. Since the protocol is defined in the dealer model, the adversary is restricted to fail-stop behavior. In turn, a fail-stop strategy depends on the adversary's view which in our case is a (deterministic) function of $a_1$ and $i^*$. Notice that the view leaks the value of $i^*$ (if it contains two successive backup outputs that are equal), and so the adversary knows with certainty whether $i^* < i$, at any given round $i$. In addition, if $i \leq i^*$, then the adversary's view at round $i$ only depends on $a_1$. Without loss of generality, we can model $\mathcal{A}$ by describing the probabilities of aborting at round $i$. Namely, assuming that the adversary is still active at round $i$, the table below describes the probability that the adversary quits at that round, depending on $a$ and $i^*$.

| Round | $i \leq i^*$ | $i = i^* + 1$ | $i \leq i^* + 2$ | $i \leq i^* + 3$ | $\dots$ | $i = i^* + r - 1$ |
|---|---|---|---|---|---|---|
| 1 | $q_1^{(a)-}$ | $\times$ | $\times$ | $\times$ | $\dots$ | $\times$ |
| 2 | $q_2^{(a)-}$ | $q_2^{(a)+_1}$ | $\times$ | $\times$ | $\dots$ | $\times$ |
| 3 | $q_3^{(a)-}$ | $q_3^{(a)+_1}$ | $q_3^{(a)+_2}$ | $\times$ | $\dots$ | $\times$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ |
| $r-1$ | $q_{r-1}^{(a)-}$ | $q_{r-1}^{(a)+_1}$ | $q_{r-1}^{(a)+_2}$ | $q_{r-2}^{(a)+_3}$ | $\dots$ | $\times$ |
| $r$ | $q_r^{(a)-}$ | $q_r^{(a)+_1}$ | $q_r^{(a)+_2}$ | $q_r^{(a)+_3}$ | $\dots$ | $q_r^{(a)+_{r-1}}$ |

where $q_i^{(a)+_i}$, $q_i^{(a)-} \in [0,1]$. Since an abort on the part of $\mathcal{A}$ does not influence $P_2$'s output whenever $i^*$ is surpassed, it follows that the bias caused by the adversary described by the table above is identical to the bias caused by an adversary $\mathcal{A}'$ described by the table below.

| Round | $a_1 = a \wedge i \leq i^*$ | $a_1 = a \wedge i > i^*$ |
|---|---|---|
| 1 | $q_1^{(a)-}$ | $\times$ |
| 1 | $q_2^{(a)-}$ | $q_2^{(a)-}$ |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $r$ | $q_r^{(a)-}$ | $q_r^{(a)-}$ |

.

Deduce that $\mathcal{A}' \in \mathbf{A}_1$. Finally, by randomizing which of the parties gets to be $P_1$ and $P_2$ – this can be achieved by having the dealer "inform" the parties who goes first – we arrive at the following conclusion.

**Theorem 4.6.** *Under suitable cryptographic assumptions, there exists a coin-tossing protocol with bias at most* $1/8r$.

## 4.3 Semi-Balanced Functions

We return to the question of characterizing functions that take input. As a warm-up, let us consider the XOR function with associated matrix

$$M = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} .$$

We argue that XOR is not computable with fairness. Assume the contrary and consider the following process in the hybrid model with ideal access to XOR: Parties $P_1$ and $P_2$ choose their inputs uniformly at random, they invoke the trusted party, and they output whatever they receive from the trusted party. Notice that the process just described defines a *fully-secure* coin-toss in the hybrid model with ideal access to XOR. Consequently, a fully-secure realization of XOR implies the existence of a fully-secure coin-tossing scheme, in contradiction with Corollary 4.4. Fair computation is thus ruled out for this function.

**Strictly Balanced Functions.** While coin-tossing was known to be reducible to many functions (such as XOR), a systematic analysis of these functions first appears in the work of Asharov, Lindell and Rabin [4]. The authors identify a family of functions (that we define bellow), and they show that not only do they all imply coin-tossing, but any function outside the family does not (information-theoretically) imply coin-tossing. As usual, let $f$ be an arbitrary Boolean function with associated matrix $M \in \{0,1\}^{\ell \times k}$.

**Definition 4.7** (Strictly-Balanced)**.** We say that $f$ is right strictly-balanced if there exists probability vector $\mathbf{q} \in \mathbb{R}^k$ and real number $\delta \in (0,1)$ such that $M \cdot \mathbf{q} = \delta \cdot \mathbf{1}_\ell$. Similarly, $f$ is left strictly-balanced if $f^T$ is right strictly-balanced. Finally, we say that $f$ is strictly balanced if $f$ is left and right strictly-balanced.

Suppose that $f$ is strictly balanced. As per the definition above, there exists probability vectors $\mathbf{p}$, $\mathbf{q}$ and real numbers $\delta_1$, $\delta_2 \in (0,1)$ such that $\mathbf{p}^T \cdot M = \delta_1 \cdot \mathbf{1}_k^T$ and $M \cdot \mathbf{q} = \delta_2 \cdot \mathbf{1}_\ell$. As one might expect, vectors $\mathbf{p}$ and $\mathbf{q}$ describe probability distributions over the parties' inputs. Next, we show that $f$ implies non-trivial sampling.

Consider the following in the hybrid model with ideal access to $f$: The parties invoke the trusted party after choosing their inputs according to $\mathbf{p}$ and $\mathbf{q}$, respectively, and they output whatever they receive from the trusted party. Write $\mathsf{out}_1$ and $\mathsf{out}_2$ for the parties' outputs. Since $\mathsf{out}_1 = \mathsf{out}_2$ and $\delta_i \neq 0, 1$ by assumption, it follows that the parties' outputs are statistically dependent, and thus coin-tossing is reducible[1] to $f$.

To motivate our generalization of the balanced criterion, consider the following example. Let $f$ be the function with associated matrix

$$M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} .$$

We remark that neither $f$ nor $f^T$ satisfy the GHKL criterion given that both the rows and the columns of $M$ span the all-1 vector (c.f. Theorem 3.5 p. 36). In addition, function $f$ is not strictly balanced since $M \cdot \mathbf{q} = \delta \cdot \mathbf{1}_\ell$, for some $\delta > 0$, implies that $\mathbf{q}(2) < 0$.

Assuming there exists a completely fair realization of function $f$, suppose that $P_1$ chooses among $\{x_1, x_3, x_4\}$ with equal probability, $P_2$ chooses $y_4$ with probability $2/5$, or one of his other inputs with probability $1/5$. Players compute $f$ on the chosen inputs and receive $\lambda$. Define

$$\mathsf{out}_1 = \lambda, \qquad \mathsf{out}_2 = \begin{cases} 1 - \lambda & \text{if } P_2 \text{ chose } y_2 \\ \lambda & \text{otherwise} \end{cases} .$$

[1]We remark that the reduction described above is not from coin-tossing to $f$, but rather from functionality $(\emptyset, \emptyset) \mapsto (z, z)$ to $f$, where $\Pr[z = 1] = \delta \in (0,1)$. Nevertheless, any functionality of the latter type implies coin-tossing via the celebrated Von Neumann technique [72].

A quick analysis reveals that $\Pr[\mathsf{out}_1 = 1] = 1/3$, $\Pr[\mathsf{out}_2 = 1] = 2/5$, and that the two are equal with probability $4/5$. If $\mathsf{out}_1$ and $\mathsf{out}_2$ were independent, they would be equal with probability $8/15$, which is not the case. Finally, it is not hard to see that any malicious behaviour by either party cannot affect the probability distribution of the other party's output. It follows that $f$ is not computable with fairness, since the process described above results in correlated outputs.

Let $M$ be an arbitrary Boolean matrix of size $\ell \times k$, and suppose that the parties compute $M$ by means of a trusted party. Further assume that $P_2$ chooses an input according to some fixed distribution, and that $P_2$ outputs a bit that depends on his input and the bit he received from the trusted party. We ask under what conditions is the resulting bit independent of the $P_1$'s input. Let us introduce some notation. Write $\mathsf{out}_2$ for $P_2$'s output. Define $\mathbf{q}^{(0)}(y)$ and $\mathbf{q}^{(1)}(y) \in \mathbb{R}$ such that

$$\mathbf{q}^{(b)}(y) = \Pr[\mathsf{out}_2 = 1 \wedge \mathsf{input} = y \mid f(x,y) = b] \quad ,$$

and write $\mathbf{q}^{(b)}$ for the resulting vector in $\mathbb{R}^k$.

**Claim 4.8.** *Using the notation above, it holds that* $\mathsf{out}_2$ *is independent of* $P_1$*'s input if and only if*

$$M \cdot (\mathbf{q}^{(1)} - \mathbf{q}^{(0)}) = \delta \cdot \mathbf{1}_\ell \quad ,$$

*for some* $\delta \in \mathbb{R}$. *Furthermore,* $\Pr[\mathsf{out}_2 = 1] = \delta + \sum_y \mathbf{q}^{(0)}(y)$.

*Proof.* Write $x'$ for the input handed by $P_1$ to the trusted party. Since $\mathsf{out}_2$ is a bit, it suffices to show that there exists $\delta' \in [0,1]$ such that $\Pr[\mathsf{out}_2 = 1 \mid x' = x] = \delta'$, for every every $x \in X$. Observe that

$$\begin{aligned}\Pr\left[\mathsf{out}_2 = 1 \,\middle|\, x' = x\right] &= \sum_y \Pr\left[\mathsf{out}_2 = 1 \wedge \mathsf{input} = y \,\middle|\, x' = x\right] \\ &= \sum_y \mathbf{q}^{(0)}(y) \cdot (1 - f(x,y)) + \mathbf{q}^{(1)}(y) \cdot f(x,y)\end{aligned}$$

□

We extend all of the above to the first party, and we write $\mathsf{out}_1$, $\mathbf{p}^{(0)}$, $\mathbf{p}^{(1)}$ for $P_1$'s analogues of $\mathsf{out}_2$, $\mathbf{q}^{(0)}$, $\mathbf{q}^{(1)}$. In addition, assume there exist $\delta_1$, $\delta_2 \in \mathbb{R}$ such that

$$\begin{aligned}(\mathbf{p}^{(1)} - \mathbf{p}^{(0)})^T \cdot M &= \delta_1 \cdot \mathbf{1}_k^T \\ M \cdot (\mathbf{q}^{(1)} - \mathbf{q}^{(0)}) &= \delta_2 \cdot \mathbf{1}_\ell \quad .\end{aligned}$$

**Proposition 4.9.** *Using the notation above, the following statements are equivalent.*

1. $\mathsf{out}_1$ *and* $\mathsf{out}_2$ *are statistically dependent.*
2. $\delta_1 \neq 0$ *and* $\sum_y \mathbf{q}^{(1)}(y) - \mathbf{q}^{(0)}(y) \neq \delta_2$.
3. $\delta_2 \neq 0$ *and* $\sum_x \mathbf{p}^{(1)}(x) - \mathbf{p}^{(0)}(x) \neq \delta_1$.

*Proof.* First consider $(\mathbf{p}^{(1)} - \mathbf{p}^{(0)})^T \cdot M \cdot (\mathbf{q}^{(1)} - \mathbf{p}^{(0)})$. By the associative property of matrix multiplication, it follows that

$$\delta_2 \cdot \left( \sum_x \mathbf{p}^{(1)}(x) - \mathbf{p}^{(0)}(x) \right) = \delta_1 \cdot \left( \sum_y \mathbf{q}^{(1)}(y) - \mathbf{q}^{(0)}(y) \right) \quad . \tag{4.1}$$

Returning to the question at hand, we know that $\mathsf{out}_1$ and $\mathsf{out}_2$ are independent if and only if

$$\Pr\left[\mathsf{out}_1 = 1 \,\wedge\, \mathsf{out}_2 = 1\right] = \Pr\left[\mathsf{out}_1 = 1\right] \cdot \Pr\left[\mathsf{out}_2 = 1\right] \tag{4.2}$$

Note that the left hand side is equal to $\mathbf{p}^{(0)\,T}(1-M)\mathbf{q}^{(0)} + \mathbf{p}^{(1)\,T}M\mathbf{q}^{(1)}$, and the right hand side is equal to $\left(\delta_1 + \sum_x \mathbf{p}^{(0)}(x))\right)\left(\delta_2 + \sum_x \mathbf{q}^{(0)}(x))\right)$. By substituting $\mathbf{p}^{(1)T}M$ for $\delta_1 \cdot \mathbf{1}_k^T + \mathbf{p}^{(0)T}M$, we deduce that Equation (4.2) is equivalent to

$$\delta_1\delta_2 = \delta_1 \cdot \left( \sum_y \mathbf{q}^{(1)}(y) - \mathbf{q}^{(0)}(y) \right) \quad .$$

In light of Equation (4.1), this concludes the proof. □

**Definition 4.10.** Let $f$ be a finite boolean function with matrix representation $M$. We say that $f$ is right semi-balanced if

$$\exists \mathbf{q} \in \mathbb{R}^k \text{ such that } \begin{cases} M\mathbf{q} = \mathbf{1}_\ell \\ \sum_y \mathbf{q}(y) \neq 1 \end{cases} \quad .$$

Similarly, $f$ is left semi-balanced if $f^T$ is right semi-balanced. Finally, we say that $f$ is semi-balanced if $f$ is left and right semi-balanced.

**Theorem 4.11.** *If $f$ is semi-balanced, then $f$ is not computable with complete fairness.*

*Proof.* We show that any secure protocol computing $f$ with respect to the ideal model with complete fairness, implies the existence of a secure (non-private) protocol computing $\mathfrak{S}(\mathcal{D})$, for some $\mathcal{D} \neq \mathcal{D}_1 \times \mathcal{D}_2$. Assuming $f$ is left and right semi-balanced, fix $\mathbf{q} \in \mathbb{R}^k$, $\mathbf{p} \in \mathbb{R}^\ell$ such that

$$\begin{cases} \sum_x |\mathbf{p}(x)| = 1 \\ \mathbf{p}^T M = (\delta_1, \dots, \delta_1) \\ \sum_x \mathbf{p}(x) \neq \delta_1 \end{cases} , \qquad \begin{cases} \sum_y |\mathbf{q}(y)| = 1 \\ M\mathbf{q} = (\delta_2, \dots, \delta_2)^T \\ \sum_y \mathbf{q}(y) \neq \delta_2 \end{cases} .$$

Consider protocol $\Pi$ in the $f$-hybrid model from figure 4.5.

**Protocol $\Pi$ in the $f$-hybrid model**

- **Inputs:** On empty inputs, $P_1$ chooses $x$ with probability $|\mathbf{p}(x)|$, $P_2$ chooses column $y$ with probability $|\mathbf{q}(y)|$.
- **Invoke Trusted Party:** $P_1$, $P_2$ invoke the trusted party on inputs $x$ and $y$, respectively. As per the ideal model with complete fairness, both parties receive a bit $b$.
- **Outputs:** If $\mathbf{p}(x) < 0$, then $P_1$ outputs $1 - b$. Similarly, if $\mathbf{q}(y) < 0$, then $P_2$ outputs $1 - b$. Otherwise, the parties are instructed to output $b$.

Figure 4.5: Reduction from Sampling to $f$.

Observe that $\Pi$ is a secure (non-private) realization of $\mathfrak{S}(\mathcal{D})$, for some $\mathcal{D} \neq \mathcal{D}_1 \times \mathcal{D}_2$, by defining

$$\mathbf{q}^{(0)}(y) = \begin{cases} 0 & \text{if } \mathbf{q}(y) \geq 0 \\ |\mathbf{q}(y)| & \text{otherwise} \end{cases}, \quad \mathbf{q}^{(1)}(y) = \begin{cases} \mathbf{q}(y) & \text{if } \mathbf{q}(y) \geq 0 \\ 0 & \text{otherwise} \end{cases},$$

and

$$\mathbf{p}^{(0)}(x) = \begin{cases} 0 & \text{if } \mathbf{p}(x) \geq 0 \\ |\mathbf{p}(x)| & \text{otherwise} \end{cases}, \quad \mathbf{p}^{(1)}(x) = \begin{cases} \mathbf{p}(x) & \text{if } \mathbf{p}(x) \geq 0 \\ 0 & \text{otherwise} \end{cases},$$

and applying Proposition 4.9. □

Chapter **5**

# The Characterization

In this chapter, we complete the characterization of Boolean symmetric functions. We begin by showing that the criteria of Chapters 3 & 4 account for *almost all* functions. Then, we discuss a particular function that does not meet any of the criteria encountered so far. Finally, we propose a new protocol, and we argue that it computes all unaccounted functions with full security. Interestingly, by carefully choosing the parameters of the new protocol, we show that it computes *all* fair functions with full security.

## 5.1 The Fairness Landscape so Far

Let $f$ be a finite Boolean function with associated matrix $M \in \{0,1\}^{\ell \times k}$. Define $\mathbf{q}^T = \mathbf{1}_\ell^T/\ell \cdot M$ and $\mathbf{p} = M \cdot \mathbf{1}_k/k$. We ask whether $f$ is fair or not. To this end, we have two criteria at our disposal.

- **GHKL Criterion:** If either $\mathbf{v} \mapsto \mathbf{q} * \mathbf{v}$ is an endomorphism of $\mathcal{H}(M^T)$, or $\mathbf{u} \mapsto \mathbf{p} * \mathbf{u}$ is an endomorphism of $\mathcal{H}(M)$, then the function is computable with fairness.
- **Sampling Criterion:** If the function is semi-balanced, then it is not computable with fairness.

We show that the criteria account for *almost all* Boolean functions.

**Theorem 5.1.** *Suppose that $|X| \geq |Y|$ and let $f$ be a random element of $2^{X \times Y}$. If $|X| \neq |Y|$, then $f$ is fair with probability greater than $1 - \mathsf{negl}(|X|)$. If $|X| = |Y|$, then $f$ is unfair with probability greater than $1 - \mathsf{negl}(|X|)$.*

The theorem follows as a corollary of the discussion below. A slightly weaker variant was first observed by Asharov in [2]. The proof we propose goes along the same lines as Asharov's, albeit with greater attention to certain details. Drawing inspiration from [74], we prove the theorem by applying Komlós' Theorem [53], in combination with a couple of lemmas.

**Theorem 5.2** (Komlós' Theorem)**.** *Let $M$ be a $0/1$-matrix of size $d \times d$. With probability greater than $1 - \mathsf{negl}(d)$, matrix $M$ is non-singular.*

**Proposition 5.3.** *If $\mathbf{1}_\ell \notin \mathrm{im}(M)$ and $\ker(M) = \{\mathbf{0}_k\}$, then $f$ satisfies the GHKL criterion. On the other hand, if $M$ is a square matrix i.e. $\ell = k$, and $M$ and $\overline{M}$ are both invertible, then $f$ is semi-balanced.*

*Proof.* The first part of the claim is just a trivial instance of Lemma 3.4 (p. 36). For the second part of the claim, since $M$ is full-rank, there exists $\mathbf{p}$ such that $M \cdot \mathbf{p} = \mathbf{1}$. Observe that $\sum_i \mathbf{p}(i) \neq 1$, otherwise $\overline{M} \cdot \mathbf{p} = \mathbf{0}_\ell$ and thus $\ker(\overline{M}) \neq \{\mathbf{0}_k\}$. It follows that $f$ is right semi-balanced. We apply the same argument to $M^T$ and we conclude that $f$ is semi-balanced. □

Already, we see that almost all functions $f : X \times Y \to \{0,1\}$ with $|X| = |Y|$ are *not* computable with fairness. By Komlós' Theorem, the probability that either $M$ or $\overline{M}$ is singular is at most $2 \cdot \mathsf{negl}(|X|)$. Thus, by Proposition 5.3, the probability that a random function is semi-balanced is greater than $1 - 2 \cdot \mathsf{negl}(|X|) = 1 - \mathsf{negl}(|X|)$.

**Lemma 5.4** (Ziegler [74])**.** *Let $\mathbf{v}_1, \dots \mathbf{v}_t$ be arbitrary vectors in $\{0,1\}^k$. Construct $\mathbf{v}'_1, \dots \mathbf{v}'_t$ by flipping all the bits indexed by $\mu \subseteq \{1, \dots, \ell\}$. In other words, for every $i \in \{1, \dots, k\}$,$\mathbf{v}'_i(j) = 1 - \mathbf{v}_i(j)$ if $j \in \mu$, and $\mathbf{v}'_i(j) = \mathbf{v}_i(j)$ otherwise. It holds that $\mathbf{v}_2 - \mathbf{v}_1, \dots \mathbf{v}_t - \mathbf{v}_1$ are linearly independent if and only if $\mathbf{v}'_2 - \mathbf{v}'_1, \dots, \mathbf{v}'_t - \mathbf{v}'_1$ are as well.*

*Proof.* Without loss of generality, suppose that $\mu = \{1, \dots, j\}$ and write $\mathbf{v}_i^T = (\mathbf{v}_i^T[\mu], \mathbf{v}_i^T[\overline{\mu}])$ in order to distinguish between coordinates in $\mu$ and $\overline{\mu}$. Observe that $(\mathbf{v}'_i)^T = (\mathbf{1}_j^T - \mathbf{v}_i^T[\mu], \mathbf{v}_i^T[\overline{\mu}])$ and thus

$$\mathbf{v}_i - \mathbf{v}_1 = \begin{pmatrix} \mathbf{v}_i[\mu] - \mathbf{v}_1[\mu] \\ \mathbf{v}_i[\overline{\mu}] - \mathbf{v}_1[\overline{\mu}] \end{pmatrix}, \qquad \mathbf{v}'_i - \mathbf{v}'_1 = \begin{pmatrix} -\mathbf{v}_i[\mu] + \mathbf{v}_1[\mu] \\ \mathbf{v}_i[\overline{\mu}] - \mathbf{v}_1[\overline{\mu}] \end{pmatrix} .$$

Clearly, any linear combination that annihilates one set of vectors also annihilates the other. □

**Lemma 5.5** (Ziegler [74]). *Let $M$ be a random $0/1$-matrix of size $\ell \times k$, with $\ell > k$. With probability greater than $1 - \mathsf{negl}(\ell - 1)$, it holds that $\mathbf{1}_\ell \notin \mathrm{im}(M)$ and $\ker(M) = \{\mathbf{0}_k\}$.*

*Proof.* First, note that it suffices to prove the claim for $k = \ell - 1$. Let $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$ denote the rows of $M$. Construct $\mathbf{v}'_2, \ldots \mathbf{v}'_\ell$ such that $\mathbf{v}'_i(j) = 1 - \mathbf{v}_i(j)$ if $\mathbf{v}_1(j) = 1$, and $\mathbf{v}'_i(j) = \mathbf{v}_i(j)$ otherwise. By applying the previous claim together with Komlós' Theorem, deduce that $\mathbf{v}_2 - \mathbf{v}_1, \ldots \mathbf{v}_\ell - \mathbf{v}_1$ are linearly independent with probability at least $1 - \mathsf{negl}(\ell - 1)$. Next, we show that $\mathbf{0}_k$ is an affine combination of $\{\mathbf{v}_1, \ldots, \mathbf{v}_\ell\}$, assuming $\mathbf{v}_2 - \mathbf{v}_1, \ldots \mathbf{v}_\ell - \mathbf{v}_1$ are linearly independent. Since $\ell > k$, there exist $\gamma_1, \ldots, \gamma_\ell$ (not all zero) such that $\sum_i \gamma_i \mathbf{v}_i = \mathbf{0}_k$. If $\sum_i \gamma_i = 0$ then

$$\begin{aligned}\left(-\sum_{i\geq 2} \gamma_i\right) \cdot \mathbf{v}_1 + \sum_{i \geq 2} \gamma_i \mathbf{v}_i &= \sum_{i\geq 2} \gamma_i(\mathbf{v}_i - \mathbf{v}_1) \\ &= \mathbf{0}_k\end{aligned}$$

which is a contradiction. Thus, there exist $\gamma_1, \ldots, \gamma_\ell$ with $\sum_i \gamma_i \neq 0$ such that $\sum_i \gamma_i \mathbf{v}_i = \mathbf{0}_k$.

In summary, with probability greater than $1 - \mathsf{negl}(\ell - 1)$, matrix $M$ has rank $k$, and $\mathbf{0}_k$ is an affine combination of the rows of $M$. If so, then $\ker(M) = \{\mathbf{0}_k\}$ and $\mathbf{1}_\ell \notin \mathrm{im}(M)$. On one hand, $M$ has maximal rank and since $\ell \geq k$, it follows that $\ker(M) = \{\mathbf{0}_k\}$. On the other hand, if there exists $\mathbf{b} \in \mathbb{R}^k$ such that $M \cdot \mathbf{b} = \mathbf{1}_\ell$, then $0 = \mathbf{0}_k^T \cdot \mathbf{b} = (\gamma_1, \ldots, \gamma_\ell) \cdot M \cdot \mathbf{b} = (\gamma_1, \ldots, \gamma_\ell) \cdot \mathbf{1}_\ell \neq 0$, which is a contradiction. □

### 5.1.1 Unaccounted Functions

Consider function $f$ described by the following matrix

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

Function $f$ clearly lies in the gap. It is easy to check that the function is not semi-balanced, and since the all-1 vector belongs to the linear span of the rows and columns of $M$, and none of them are monochromatic, neither is $f$

computable with GHKL. In the remainder of this section, we show that the arguments from Chapter 4 do not apply to this function. Define inputless functionality $\mathcal{F}$ such that $P_1$ and $P_2$ receive $(x, z_1)$ and $(y, z_2)$, respectively, according to the probabilities described by the following table.

| $(x, z_1) \setminus (y, z_2)$ | $(0, 0)$ | $(0, 1)$ | $(1, 0)$ | $(1, 1)$ |
|---|---|---|---|---|
| $(0, 0)$ | 1/9 | 0 | 0 | 1/9 |
| $(0, 1)$ | 0 | 1/3 | 1/9 | 0 |
| $(1, 0)$ | 0 | 1/9 | 0 | 0 |
| $(1, 1)$ | 1/9 | 0 | 0 | 1/9 |

Drawing inspiration from the impossibility result of the previous chapter, consider protocol $\Pi'$ computing $\mathcal{F}$ in the $f$-hybrid model (Figure 5.1).

**Protocol** $\Pi'$

- **Inputs:** Empty for both parties.
- $P_1$ (resp. $P_2$) chooses $x \in \{1, 3, 4\}$ (resp. $y \in \{1, 3, 4\}$) uniformly at random.
- Parties invoke the trusted party for computing $f$ and receive $b \in \{0, 1\}$ from the trusted party.
- **Outputs:** $P_1$ (resp. $P_2$) outputs $(1, 1 - b)$ if $x = 1$ (resp. $y = 1$) and $(0, b)$ otherwise.

Figure 5.1: Protocol $\Pi'$ for Computing $\mathcal{F}$ in the $f$-hybrid Model

**Claim 5.6.** *An honest execution of protocol $\Pi'$ is a correct realization of functionality $\mathcal{F}$. Furthermore, $\mathcal{F}$ satisfies the following.*

- *Any two of $\{x, y, z_1, z_2\}$ are independent random variables.*
- $\begin{cases} \Pr[x = 1] = \Pr[y = 1] = 1/3 \\ \Pr[z_1 = 1] = \Pr[z_2 = 1] = 2/3 \end{cases}$ .
- $x \oplus z_1 = y \oplus z_2$.

*Proof.* Immediate. □

The first thing to note is that, contrary to semi-balanced functions, $z_1$ and $z_2$ are not statistically dependent, and thus the semi-balanced criterion does not apply here. Assuming a fully secure realization of $f$, observe that no adversary corrupting $P_1$ (resp. $P_2$) can attack the *marginal* distributions of $y$ and $z_2$ (resp. $x$ and $z_1$).

Turning to $\mathcal{F}$, we stress that it cannot be implemented securely. Indeed, $\mathcal{F}$ is a sampling functionality, and there always exists an adversary corrupting either $P_1$ or $P_2$ that can bias the honest party's output, for any realization of $\mathcal{F}$. In other words, the adversary can always alter the joint distribution of $(x, z_1)$ - or $(y, z_2)$.

That being said, we would like to show something slightly stronger. It seems that, in any realization of $\mathcal{F}$, the adversary should be able to attack the marginal distributions of the first or second output of the honest party. The reason for this is that each party's (global) output contains information about both the first and second output of the opponent. Alas, this intuition is false. In the remainder of this section we argue why it is wrong. Consider protocol $\Pi$ in the online dealer model (Figure 5.2).

**Protocol $\Pi$**

1. $P_1$ (resp. $P_2$) computes $x$ (resp. $y$) such that $\Pr[x = 1] = 1/3$ (resp. $\Pr[y = 1] = 1/3$).
2. Parties invoke the dealer on inputs $x$ and $y$. If $P_1$ fails to engage with the dealer (or invokes the dealer on an aberrant value), the dealer sends an abort symbol to $P_2$ and halts. $P_2$ outputs $(y, y \oplus 1)$. If $P_2$ fails to engage with the dealer (or invokes the dealer on an aberrant value), $P_1$ outputs $(x, z_1')$ where $z_1'$ is sampled independently such that $\Pr[z_1' = 1] = 2/3$.
3. The dealer chooses $z$ according to the prescribed distribution, and computes $z_1 = z \oplus x$ and $z_2 = z \oplus y$.
    a) The dealer sends $z_1$ to $P_1$. If $P_1$ aborts, the dealer sends an abort symbol to $P_2$ and halts. $P_2$ outputs $(y, y \oplus 1)$.
    b) The dealer sends $z_2$ to $P_2$ and halts.
4. Parties output $(x, z_1)$ and $(y, z_2)$ respectively.

Figure 5.2: Protocol $\Pi$ for Computing $\mathcal{F}$.

**Claim 5.7.** *No adversary can attack the marginal distributions of the honest party's output bits.*

*Proof.* First, notice that a corrupt $P_2$ cannot affect $P_1$'s (global) output distribution. In fact, an adversary corrupting $P_2$ can be simulated in the ideal model with full security. Thus, we need only consider the case where the adversary corrupts $P_1$. Since $y$ is chosen before any interaction occurs, we deduce that the marginal distribution of $P_2$'s first output cannot be attacked. For the second bit, it is not hard to see that the adversary's attacking strategy boils down to the following:

1. The adversary chooses $x$ whichever way she wants.
2. Upon seeing $z_1$, the adversary decides to abort depending on her view thus far.

Recall that $z_2$ does not depend on $x$, so as long as $y$ is chosen according to the prescribed distribution, $z_2$ will also satisfy its prescribed distribution. Now consider the following probabilities.

$$\Pr\left[y = 1 \mid (x, z_1)\right] = \begin{cases} 1/2 & \text{if } x = z_1 \\ 0 & \text{if } (x, z_1) = (1, 0) \\ 1/4 & \text{if } (x, z_1) = (0, 1) \end{cases} .$$

Thus,

$$\Pr\left[z_2 = 1 \mid (x, z_1)\right] = \begin{cases} 1/2 & \text{if } x = z_1 \\ 1 & \text{if } (x, z_1) = (1, 0) \\ 3/4 & \text{if } (x, z_1) = (0, 1) \end{cases} .$$

Finally, recall that $P_2$'s backup for $z_2$ (in case of an early abort) is $1 \oplus y$. Notice that the probability distribution of the latter, given $x$ and $z_1$, corresponds exactly to the probability distribution of $z_2$ conditioned on the same $x$ and $z_1$. □

## 5.2 Protocol FairTwoParty

In this section, we describe a new protocol for computing functions fairly. As usual, let $f : X \times Y \to \{0, 1\}$ be a finite Boolean function with the associated matrix $M$ of size $\ell \times k$. Without loss of generality, $X = \{1, \dots, \ell\}$ and $Y = \{1, \dots, k\}$. Our protocol is based on the protocol of Gordon, Hazay, Katz and Lindell. As such, we take a moment to "deconstruct" GHKL. Recall that security is almost a trivial matter when $P_2$ is corrupted. Therefore, all the comments below are made with a corrupted $P_1$ in mind.

Throughout this thesis, protocols are defined by specifying the parties' backup outputs at any given round. In light of the hybrid-model technique, this approach does not incur any loss of generality. Designing protocols that are secure is thus an exercise in choosing the *appropriate* backup outputs. In GHKL, the backup outputs depend on the function, the parties inputs, *and* the special round $i^*$. Namely, once the special round has been assigned, $P_2$'s backup outputs $(b_0, b_1, \dots, b_r)$ are constructed as follows. If $i \geq i^*$ then $b_i = f(x, y)$, and if $i < i^*$ then $b_i = f(\widetilde{x}^{(i)}, y)$, where $\widetilde{x}^{(i)}$ denotes[1] an element of $X$ chosen uniformly at random.

One interpretation is to say that, prior to the special round, $P_2$ computes his backup output *by choosing an input for* $P_1$ *that is independent of his own input.* This approach is directly inspired by the ideal model, where the simulator hands an input for $P_1$ to the trusted that is independent of $P_2$'s input[2]. So, a natural generalization of GHKL involves modifying the distribution of $\widetilde{x}^{(i)}$. Instead of the uniform distribution, $\widetilde{x}^{(i)}$ may be chosen according to a more elaborate probability distribution. This generalization was noticed independently by Asharov [2] and Makriyannis [63]. It results in variant of GHKL that is meaningful, as was shown in [63]: there is a net benefit in considering different distributions, and these distributions can be found efficiently. For further details, the reader is referred to Appendix C p. 157.

However, we argue that this is not the right way to go. One one hand, we note that this generalization is somewhat superfluous. Before round $i^*$, $P_1$'s view is independent of $P_2$'s backup output. Specifically, because $(a_0, \dots, a_{i^*-1})$ is independent of $(b_0, \dots, b_{i^*-2})$, changing the probability distribution of $(b_0, \dots, b_{i^*-2})$ is inconsequential to the security analysis, as long as the distribution remains consistent with $P_2$'s input. On the other hand, we claim that this generalization is rather restrictive. When we define $b_{i^*-1} = f(\widetilde{x}, y)$, where $\widetilde{x} \leftarrow \Delta(X)$ and $\Delta(X)$ denotes an arbitrary distribution over $X$, then the probability distribution of $b_{i^*-1}$ is limited to those distributions that can be constructed that way.

[1]As the superscript suggests, a fresh input is chosen at every round.

[2]We are merely highlighting the "independence of inputs" security requirement.

Our protocol addresses both of these issues by leaving $(b_0, \ldots, b_{i^*-2})$ as is, and letting $b_{i^*-1}$ take an arbitrary distribution. We believe that this modification goes against the spirit of GHKL. Indeed, when computing his backup output at round $i^* - 1$, it can no longer be said that $P_2$ chooses an input for $P_1$ because (in general) no such input exists, i.e. $\forall \Delta(X)$, $\exists y \in Y$ such that $\Pr[b_{i^*-1} = 1 \mid y] \neq \Pr[f(\widetilde{x}, y) = 1 \mid \widetilde{x} \leftarrow \Delta(X)]$.

**Remark 5.8.** It would seem that we are opening ourselves to one of the following traps: Either $b_{i*-1}$ is inconsistent with $P_2$'s output, e.g. if $b_{i^*-1} = 0$, and there exists $y \in Y$ such that for all $x \in X$, $f(x, y) = 1$, and thus the protocol's correctness comes into question, *or* the implicit input $\widetilde{x}$ that $P_2$ uses for $P_1$ (i.e. $f(\widetilde{x}, y) = b_{i^*-1}$) *depends* on $y$, in contradiction with our discussion regarding *independence of inputs*. These issues will be addressed at the end of this section. For now, we draw the attention of the reader to the formal description of Protocol FairTwoParty$_\sigma$ (Figure 5.3).

Our protocol differs from the GHKL protocol in that $b_{i^*-1} = \sigma$ and the fact that $i^* \geq 2$ to accommodate for this change (cf. Steps 3 and 4 in Figure 5.3). For some functions we choose $\sigma = 0$ and for others we choose $\sigma = 1$; the choice of $\sigma$ depends only on the function and is independent of the inputs. This seemingly small change will enable to compute all Boolean functions that are fair.

**The Round-Complexity & The Choice of $\sigma$.** There are two parameters in Protocol FairTwoParty$_\sigma$ that are unspecified – the parameter $\alpha$ of the geometric distribution and the number of rounds $r$. We show that there exists a constant $\alpha_0$ (which depends on $f$) such that taking any $\alpha \leq \alpha_0$ guarantees full security (provided that $f$ satisfies some conditions). As for the number of rounds $r$, even if both parties are honest the protocol fails if $i^* > r$ (where $i^*$ is chosen with geometric distribution). The probability that $i^* > r$ is $(1 - \alpha)^r$. So, if $r = \alpha^{-1} \cdot \omega(\log n)$ (where $n$ is the security parameter), the probability of not reaching $i^*$ is negligible.

Before we proceed with the security analysis of the protocol, we remark that computing $f$ with $\sigma = 1$ is equivalent to computing $1 - f$ with $\sigma = 0$ and flipping the output from the computation. Thus, FairTwoParty$_1$ computes $f$ with full security if and only if FairTwoParty$_0$ computes $1 - f$ with full security. Therefore, without loss of generality, we restrict ourselves to $\sigma = 0$.

**Protocol** $\textsc{FairTwoParty}_\sigma$

1. The parties $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[a]
2. The dealer chooses $i^* \geq 2$ according to the geometric distribution with probability $\alpha$.
3. The dealer computes $\mathsf{out} = f(x, y)$, and for $0 \leq i \leq r$

$$a_i = \begin{cases} f(x, \widetilde{y}^{(i)}) \text{ where } \widetilde{y}^{(i)} \in_U Y & \text{if } i < i^* \\ \mathsf{out} & \text{otherwise} \end{cases}$$

and

$$b_i = \begin{cases} f(\widetilde{x}^{(i)}, y) \text{ where } \widetilde{x}^{(i)} \in_U X & \text{if } i < i^* - 1 \\ \sigma & \text{if } i = i^* - 1 \\ \mathsf{out} & \text{otherwise.} \end{cases}$$

4. The dealer gives $b_0$ to $P_2$.
5. For $i = 1, \ldots, r$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

[a]If $x$ is not in the appropriate domain or $P_1$ does not hand an input, then the dealer sends $f(\hat{x}, y)$ (where $\hat{x}$ is a default value) to $P_2$, which outputs this value and the protocol is terminated. The case of an inappropriate $y$ is dealt analogously.

Figure 5.3: Protocol $\textsc{FairTwoParty}_\sigma$ for Computing $f$.

### 5.2.1 Security Analysis

**Theorem 5.9.** *Let $M$ be the associated matrix of a Boolean function $f$. If $\mathbf{0}_k$ is an affine combination of the rows of $M$, then there is a constant $\alpha_0 > 0$ such that Protocol $\textsc{FairTwoParty}_0$ with $\alpha \leq \alpha_0$ is a fully-secure protocol for $f$.*

It is easy to see that the protocol is secure against a corrupted $P_2$, using a simulator similar to [45]. Intuitively, this follows directly from the fact that $P_2$ always gets the output after $P_1$. Next, we demonstrate that the protocol is secure against a corrupted $P_1$ by constructing a simulator for every adversary $\mathcal{A}$ in the real world controlling $P_1$. The simulator we construct is given black-box access to $\mathcal{A}$ and we denote it by $\mathcal{S}^{\mathcal{A}}$. The simulation is described in Figure 7.2. It operates along the same lines as in the proof of [45]. The main difference is in the analysis of the simulator, where we prove that there exist distributions $(\mathbf{x}_x^{(a)})_{x \in X, a \in \{0,1\}}$, which are used by the simulator to choose the input that $P_1$ gives to the trusted party.

**The Simulator $\mathcal{S}$ for Protocol** FairTwoParty$_0$

- The adversary $\mathcal{A}$ gives its input $x$ to the simulator.[a]
- The simulator chooses $i^* \geq 2$ according to the geometric distribution with probability $\alpha$.
- For $i = 1, \ldots, i^* - 1$:
  - The simulator gives $a_i = f(x, \widetilde{y}^{(i)})$ to the adversary $\mathcal{A}$, where $\widetilde{y}^{(i)}$ is chosen according to the uniform distribution.
  - If $\mathcal{A}$ aborts, then the simulator chooses an input $x_0$ according to a distribution $\mathbf{x}_x^{(a_i)}$ (which depends on the input $x$ and the last bit that was chosen), gives $x_0$ to the trusted party, outputs the bits $a_1, \ldots, a_i$, and halts.
- At round $i = i^*$, the simulator gives $x$ to the trusted party and gets the output $a = f(x, y)$.
- For $i = i^*, \ldots, r$: The simulator gives $a_i = a$ to the adversary $\mathcal{A}$, if $\mathcal{A}$ aborts, then the simulator outputs the bits $a_1, \ldots, a_i$ and halts.
- The simulator outputs the bits $a_1, \ldots, a_r$ and halts.

[a] If the adversary gives an inappropriate $x$ (or no $x$), then the simulator sends some default $\hat{x} \in X$ to the trusted party, outputs the empty string, and halts.

Figure 5.4: The Simulator $\mathcal{S}$ for Protocol FairTwoParty$_0$.

We next prove the correctness of the simulator, that is, if $(\mathbf{0}_k)^T$ is an affine combination of the rows of $M$, then the output of the simulator and the output of $P_2$ in the ideal world are distributed as the adversary's view and the output of the honest $P_2$ in the real world. The simulator generates an output identical to the view of $P_1$ in the execution of the protocol in the real world. First, it chooses $i^*$ as in Protocol FairTwoParty$_0$, i.e., according to a geometric distribution. Up to round $i^* - 1$, the backup outputs are uncorrelated to the input of the honest party. That is, for all $i < i^*$ the output $a_i = f(x, \widetilde{y})$ is chosen with a uniformly random $\widetilde{y}$. Starting with round $i = i^*$, the backup outputs are correlated to the true input of the honest party, and are set as $a_i = f(x, y)$, exactly as in the real execution.

As a result the adversary seeing the same view, $\mathcal{A}$ aborts in the simulation if and only if it aborts in the protocol. Furthermore, if the adversary aborts after round $i^*$, the output of $P_2$ in the protocol and in the simulator is identical – $f(x, y)$. The only difference between the simulation and the protocol is the way that the output of $P_2$ is generated when the adversary aborts in a round $i \leq i^*$. If the adversary aborts in round $i^*$, the output of $P_2$ in Protocol FairTwoParty$_0$ is $b_{i^*-1} = 0$, while the the output of $P_2$ in the simulation is $f(x, y)$. To compensate for this difference, in the simulation the output of $P_2$ if the adversary aborts in a round $1 \leq i \leq i^* - 1$ is $f(x_0, y)$, where $x_0$ is chosen according to a distribution $\mathbf{x}_x^{(a_i)}$ to be carefully defined later in the proof.

To conclude, we only have to compare the distributions $(\text{View}_{\mathcal{A}}, \mathsf{out}_2)$ in the real and ideal worlds given that the adversary aborts in a round $1 \leq i \leq i^*$. In the remainder of the proof, we let $i$ be the round in which the adversary aborts, and we assume that $1 \leq i \leq i^*$. The view of the adversary in both worlds is $a_1, \ldots, a_i$. Notice that in both worlds $a_1, \ldots, a_{i-1}$ are identically distributed and are independent of $(a_i, \mathsf{out}_2)$. Consequently, we only need to compare the distribution of $(a_i, \mathsf{out}_2)$ in each world.

Let us introduce the following notation

$$\mathbf{q}^T = \mathbf{1}_\ell^T/\ell \cdot M, \qquad \mathbf{p} = M \cdot \mathbf{1}_k/k. \tag{5.1}$$

For example, $\mathbf{q}(y)$ is the probability that $f(\widetilde{x}, y) = 1$, when $\widetilde{x}$ is uniformly distributed. Furthermore, for every $x \in X$, $y \in Y$, and $a \in \{0, 1\}$, define $\mathbf{c}_x^{(a)}(y) \triangleq \Pr[f(x', y) = 1]$, where $x'$ is chosen according to distribution $\mathbf{x}_x^{(a)}$. That is, $\mathbf{c}_x^{(a)}(y)$ is the probability that the output of $P_2$ in the simulation is 1 when the the adversary aborts in round $i < i^*$, the input of $P_1$ is $x$, the input of $P_2$ is $y$, and $a_i = a$. Finally, we define vector $\mathbf{c}_x^{(a)} \triangleq (\mathbf{c}_x^{(a)}(y))_{y \in Y}$. Using this notation,

$$\mathbf{x}_x^{(a)T} M = \mathbf{c}_x^{(a)T}, \tag{5.2}$$

where we represent distribution $\mathbf{x}_x^{(a)} = (\mathbf{x}_x^{(a)}(x'))_{x' \in X}$ by a column vector. Next, we analyze the four options for the values of $(a_i, \mathsf{out}_2)$.

**First case:** $(a_i, \mathsf{out}_2) = (0, 0)$**.** In the real world $(a_i, \mathsf{out}_2) = (0, 0)$ if one of the following two events occurs:

- $i < i^*$, $a_i = f(x, \widetilde{y}) = 0$, and $\mathsf{out}_2 = b_{i-1} = f(\widetilde{x}, y) = 0$. The probability of this event is $(1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{q}(y))$.
- $i = i^*$, $a_{i^*} = f(x,y) = 0$, and $\mathsf{out}_2 = b_{i^*-1} = 0$. Recall that in Protocol $\textsc{FairTwoParty}_0$ $b_{i^*-1} = 0$ with probability 1. The probability of this event is $\alpha \cdot (1 - f(x,y)) \cdot 1$ (that is, it is 0 if $f(x,y) = 1$ and it is $\alpha$ otherwise).

Therefore, in the real world $\Pr[(a_i, \mathsf{out}_2) = (0,0)] = (1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{p}(y)) + \alpha(1-f(x,y))$. On the other hand, in the ideal world $(a_i, \mathsf{out}_2) = (0,0)$ if one of the following two events occurs:

- $i < i^*$, $a_i = f(x, \widetilde{y}) = 0$, and $\mathsf{out}_2 = f(x_0, y) = 0$. The probability of this event is $(1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{c}_x^{(0)}(y))$.
- $i = i^*$, $a_{i^*} = f(x,y) = 0$, and $\mathsf{out}_2 = f(x,y) = 0$. The probability of this event is $\alpha \cdot (1 - f(x,y))$.

Therefore, in the ideal world $\Pr[(a_i, \mathsf{out}_2) = (0,0)] = (1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{c}_x^{(0)}(y)) + \alpha(1-f(x,y))$. To get full security we need that these two probabilities in the two worlds are the same, that is,

$$(1-\alpha)(1-\mathbf{p}(x))(1-\mathbf{q}(y)) + \alpha(1-f(x,y)) = \\ (1-\alpha)(1-p_x)(1-\mathbf{c}_x^{(0)}(y)) + \alpha(1-f(x,y)) \ ,$$

i.e.,

$$\mathbf{c}_x^{(0)}(y) = \mathbf{q}(y) \ . \tag{5.3}$$

As this is true for every $y$, we deduce, using Equation (5.1) and Equation (5.2), that

$$\mathbf{x}_x^{(0)T} M = \mathbf{c}_x^{(0)T} = \mathbf{1}_\ell/\ell \cdot M \ . \tag{5.4}$$

Thus, the uniform distribution, i.e. $\mathbf{x}_x^{(0)} = \mathbf{1}_\ell/\ell$, satisfies these constraints.

**Second case:** $(a_i, \mathsf{out}_2) = (0,1)$. In the real world $\Pr[(a_i, \mathsf{out}_2) = (0,1)] = (1-\alpha)(1-p_x)s_y$ (in the real world $\mathsf{out}_2 = 0$ when $i = i^*$). On the other hand, in the ideal world $\Pr[(a_i, \mathsf{out}_2) = (0,1)] = (1-\alpha)(1-p_x)q_x^{(0)}(y)$ (in the ideal world $a_{i^*} = \mathsf{out}_2 = f(x,y)$). The probabilities in the two worlds are equal if Equation (5.3) holds and thus it suffices that $\mathbf{x}_x^{(0)} = \mathbf{1}_\ell/\ell$.

**Third case:** $(a_i, \mathsf{out}_2) = (1,0)$**.** In the real world $\Pr[(a_i, \mathsf{out}_2) = (1,0)] = (1-\alpha)\mathbf{p}(x)(1-\mathbf{q}(y)) + \alpha \cdot f(x,y) \cdot 1$. On the other hand, in the ideal world $\Pr[(a_i, \mathsf{out}_2) = (1,0)] = (1-\alpha)\mathbf{p}(x)(1-\mathbf{c}_x^{(1)}(y))$. The probabilities in the two worlds are equal when

$$(1-\alpha)\mathbf{p}(x)(1-\mathbf{q}(y)) + \alpha f(x,y) = (1-\alpha)\mathbf{p}(x)(1-\mathbf{c}_x^{(1)}(y)).$$

If $p_x = 0$, then $f(x,y) = 0$ and we are done. If not, equality holds if and only if

$$\mathbf{c}_x^{(1)}(y) = \mathbf{q}(y) - \frac{\alpha f(x,y)}{(1-\alpha)\mathbf{p}(x)}. \tag{5.5}$$

As this is true for every $y$, we deduce, using Equation (5.1) and Equation (5.2), that

$$\begin{aligned} \mathbf{x}_x^{(1)T} M = \mathbf{c}_x^{(1)} &= \mathbf{1}_\ell/\ell \cdot M - \frac{\alpha}{(1-\alpha)\mathbf{p}(x)} \cdot \mathsf{row}_x^T \\ &= \left(\mathbf{1}_\ell/\ell - \frac{\alpha}{(1-\alpha)\mathbf{p}(x)} \cdot \mathbf{e}_x\right) M, \end{aligned} \tag{5.6}$$

where $\mathsf{row}_x$ is the row of $M$ labelled by the input $x$, and $\mathbf{e}_x$ is the $x$-th unit vector. Before analyzing when there exists a probability vector $\mathbf{x}_x^{(1)}$ solving Equation (5.6), we remark that if the equalities hold for the first 3 cases of the values for $(a_i, \mathsf{out}_2)$, the equality of the probabilities must also hold for the case $(a_i, \mathsf{out}_2) = (1,1)$. In the rest of the proof we show that there exist probability vectors $\mathbf{x}_x^{(1)}$ for every $x \in X$ solving Equation (5.6).

**Claim 5.10.** *Fix $x \in X$ and let $\alpha$ be a sufficiently small constant. If $\mathbf{0}_k$ is an affine combination of the rows of $M$, then there exists a probability vector $\mathbf{x}_x^{(1)}$ solving Equation (5.6).*

*Proof.* By the conditions of the claim, there exists a vector $\mathbf{u} \in \mathbb{R}^\ell$ such that $\mathbf{u}^T \cdot M = \mathbf{0}_k^T$ and $\sum_{i \in X} \mathbf{u}(i) = 1$. Define vector $\mathbf{x}_x^{(1)} = \mathbf{1}_\ell/\ell + \frac{\alpha}{(1-\alpha)\mathbf{p}(x)}(\mathbf{u} - \mathbf{e}_x)$. Observe that $\mathbf{x}_x^{(1)}$ is a solution to Equation (5.6) since $\mathbf{u}^T \cdot M = \mathbf{0}_k^T$. We need to show that it is a probability vector. First,

$$\begin{aligned} \sum_{i \in X} \mathbf{x}_x^{(1)}(i) = \mathbf{1}_\ell^T \cdot \mathbf{x}_x^{(1)} &= \mathbf{1}_\ell^T \cdot \mathbf{1}_\ell/\ell + \mathbf{1}_\ell^T \cdot \frac{\alpha}{(1-\alpha)\mathbf{p}(x)}(\mathbf{u} - \mathbf{e}_x) \\ &= 1 + \frac{\alpha}{(1-\alpha)\mathbf{p}(x)}(1-1) = 1. \end{aligned}$$

Second, $\mathbf{x}_x^{(1)}(i) \geq 1/\ell - \frac{\alpha}{(1-\alpha)\mathbf{p}(x)}(1 + |\mathbf{u}(i)|)$. Thus, as long as

$$\alpha \leq \min_{x \in X \,|\, \mathbf{p}(x) \neq 0} \left\{ \frac{\mathbf{p}(x)/\ell}{1 + \max_{i \in X}\{|\mathbf{u}(i)|\} + \mathbf{p}(x)/\ell} \right\} ,$$

then[3] $\mathbf{x}_x^{(1)}$ is a probability vector, for any $x \in X$.

□

To sum up, we showed that if $\mathbf{0}_k$ is an affine combination of the rows of $M$, then there exists probability vectors solving Equation (5.6). Furthermore, these probability vectors can be found efficiently. Using these probability distributions in the simulator we constructed proves that protocol FairTwoParty$_0$ is fully secure for $f$. □

### 5.2.2 Limits of Protocol FairTwoParty

**Theorem 5.11.** *If $\mathbf{0}_\ell$ is not an affine combination of the rows of $M$, then for any $\alpha$ such that $\alpha^{-1}(n) \in \mathsf{poly}(n)$, protocol* FairTwoParty$_0$ *is susceptible to an attack.*

*Proof.* First, from Lemma 2.2 p. 17, it follows that that $\mathbf{1}_\ell \in \text{im}(M)$. So, fix $\mathbf{b}$ such that $\sum_i |\mathbf{b}(i)| = 1$ and $M \cdot \mathbf{b} = \delta \cdot \mathbf{1}_\ell$, for some $\delta \neq 0$. Now, in the hybrid model with ideal access to $M$, suppose that $P_2$ chooses an input and flips the output according to $\mathbf{b}$. It holds that, regardless of the input chosen by $P_1$,

$$\Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1\right] = \delta + \sum_{\mathbf{b}(i)<0} |\mathbf{b}(i)| \ .$$

Replace the call to the trusted party with an execution of FairTwoParty$_0$. Consider a corrupted $P_2$ that quits immediately upon receiving $a_2$. Observe that $P_2$ outputs $b_1$ if $\mathbf{b}(y) \geq 0$ and $1 - b_1$ otherwise. Let us compute

$$\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1\right] = \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 2\right] + \Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* \neq 2\right]$$

[3] We remark that since the size of the function's input-domain is constant, vector $\mathbf{u}$ is fixed and $\alpha$ is a constant.

Note that $\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* \neq 2\right] = (1-\alpha)\left(\delta + \sum_{\mathbf{b}(i)<0} |\mathbf{b}(i)|\right)$, and that $\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1 \wedge i^* = 2\right] = \Pr\left[i^* = 2\right] \cdot \Pr\left[\mathbf{b}(y) < 0\right]$. Thus,

$$\begin{aligned}\Pr\left[\mathsf{out}_2^{\mathsf{R}} = 1\right] &= \alpha \cdot \sum_{\mathbf{b}(y)<0} |\mathbf{b}(y)| + (1-\alpha)\left(\delta + \sum_{\mathbf{b}(i)<0} |\mathbf{b}(i)|\right) \\ &= \Pr\left[\mathsf{out}_2^{\mathsf{H}} = 1\right] - \alpha\delta \ .\end{aligned}$$

□

Recall that GHKL computes $f$ with full security if and only if $\mathbf{v} \mapsto \left(M^T \cdot \mathbf{1}_k\right) * \mathbf{v}$ is a (linear) endomorphism of $\mathrm{im}(M^T)$ *and* the non-monochromatic columns of $M$ do not span the all-1 vector.

**Claim 5.12.** *If GHKL computes $f$ with full security then* FairTwoParty$_0$ *computes at least one of $\{f, f^T, 1 - f^T\}$ with full security.*

*Proof.* Suppose that $M$ contains monochromatic columns. It holds that,

- if $\mathbf{0}_\ell \in \{\mathsf{col}_y\}_{y \in Y}$ then FairTwoParty$_0$ computes $f^T$ with full security,
- if $\mathbf{1}_\ell \in \{\mathsf{col}_y\}_{y \in Y}$ then FairTwoParty$_0$ computes $1 - f^T$ with full security.

If $M$ does not contain monochromatic columns, the second condition of the GHKL criterion stipulates that $\mathbf{1}_\ell \notin \mathrm{im}(M)$ and thus $\mathbf{0}_k$ is an affine combination of the rows of $M$. It follows that FairTwoParty$_0$ computes $f$ with full security. □

**Revisiting Remark 5.8.** We now return to our discussion regarding the choice of $b_{i^*-1}$. It seems odd that one can fix $b_{i^*-1} = 0$ and still be able to compute at least as many functions as the GHKL protocol. In Remark 5.8, two issues were raised, one relating the protocol's correctness (the honest party's output), the other to the *independence of inputs* security requirement.

To address these issues, we must ask what implicit input is chosen for $P_1$ at round $i^* - 1$, i.e. to construct $b_{i^*-1}$. As an example, consider the function

$f$ given by the matrix below.

$$M = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Since $(-1, 1, 0, 1) \cdot M = (0, 0, 0, 0)$, it follows that FairTwoParty$_0$ computes $f$ with full security. The implicit input for $P_1$ must be encoded by a probability vector $\mathbf{u}$ such that $\mathbf{u}^T \cdot M = (0, 0, 0, 0)$. However, a quick analysis reveals that no such vector exists (!?). The apparent paradox arises because our question is somewhat misplaced. Since the adversary is oblivious to the value of $i^*$, by quitting at round $i$, the honest party's output is a probabilistic combination of $b_{i<i^*}$ and the correct output. The paradox is resolved by observing that the implicit input used in $b_{i<i^*}$ is encoded by the vector

$$\mathbf{u} = \begin{pmatrix} 1/4 - 5/4\alpha \\ 1/4 + 3/4\alpha \\ 1/4 - 1/4\alpha \\ 1/4 + 3/4\alpha \end{pmatrix} = (1-\alpha) \cdot \begin{pmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{pmatrix} + \alpha \begin{pmatrix} -1 \\ 1 \\ 0 \\ 1 \end{pmatrix},$$

which is a probability vector since $b_{i<i^*}$ and $b_1$ are identically distributed. The reason this phenomenon[4] does not occur in GHKL is that $b_{i^*-1}$ and $b_{i<i^*}$ are identically distributed.

## 5.3 Characterizing Boolean Functions

**Claim 5.13.** *Protocol* FairTwoParty$_1$ *computes $f$ with full security if and only if $\mathbf{1}_k$ is an affine combination of the rows of $M$.*

*Proof.* Note that $\mathbf{1}_k$ is an affine combination of the rows of $M$ if and only if $\mathbf{0}_k$ is an affine combination of the rows of $\mathbf{1}_{\ell \times k} - M$. To conclude, recall that FairTwoParty$_1$ computes $f$ with full security if and only if FairTwoParty$_0$ computes $1 - f$ with full security. □

Next, we present a equivalent statement of the semi-balanced criterion.

[4] It is worth mentioning that one way to interpret this phenomenon is to say that the implicit input in $b_{i^*-1}$ is sampled with "negative probabilities".

**Claim 5.14.** *Function $f$ is semi-balanced if and and only if neither of the following statements apply to $f$:*

- *$\mathbf{0}_k$ or $\mathbf{1}_k$ is an affine combination of the rows of $M$.*
- *$\mathbf{0}_\ell$ or $\mathbf{1}_\ell$ is an affine combination of the columns of $M$.*

*Proof.* We show that if $\mathbf{0}_k$ or $\mathbf{1}_k$ is an affine combination of the rows of $M$, then $f$ is not right semi-balanced. Recall that a function is right semi-balanced if there exists $\mathbf{q} \in \mathbb{R}^k$ such that $\sum_y \mathbf{q}(y) \neq 1$ and $M\mathbf{q} = \mathbf{1}_\ell$. We know that $\mathbf{1}_\ell \notin \mathrm{im}(M)$ if and only if $\mathbf{0}_k$ is an affine combination of the rows of $M$. It remains to show that if $\mathbf{1}_k$ is an affine combination of the rows, then any linear combination of columns that yields $\mathbf{1}_\ell$ is affine. Indeed, let $\mathbf{a} \in \mathbb{R}^\ell$ such that $\sum_x \mathbf{a}(x) = 1$ and $\mathbf{a}^T M = \mathbf{1}_k^T$. Suppose there exists $\mathbf{b} \in \mathbb{R}^\ell$ such that $M \cdot \mathbf{b} = \mathbf{1}_\ell$. Then,

$$1 = \mathbf{a}^T \cdot \mathbf{1}_\ell = \mathbf{a}^T M \cdot \mathbf{b} = \mathbf{1}_k^T \cdot \mathbf{b} = \sum_y \mathbf{b}(y) \ .$$

We apply the same arguments to $M^T$ and we conclude that if $\mathbf{0}_\ell$ or $\mathbf{1}_\ell$ is an affine combination of the columns of $M$, then $f$ is not left semi-balanced. $\square$

The theorem below follows as a corollary.

**Theorem 5.15** (Characterization of Boolean Functions)**.** *Function $f$ is computable with with full-security if and only if $f$ is not semi-balanced.*

### 5.3.1 Randomized Functions

The characterization of randomized Boolean functions follows effortlessly from our discussion so far. Let $f : X \times Y \to \Delta(\{0,1\})$ denote a randomized Boolean function. We claim that the arguments from Chapters 4 & 5 apply almost verbatim to randomized functions. Define associated matrix $M \in \mathbb{R}^{\ell \times k}$ such that $M(x,y) = \Pr[f(x,y) = 1]$, and extend the semi-balanced definition. Functionality $f$ is semi-balanced if there exist non-affine vectors $\mathbf{p}$ and $\mathbf{q}$ such that $M^T \cdot \mathbf{p} = \mathbf{1}_k$ and $M \cdot \mathbf{q} = \mathbf{1}_\ell$.

**Theorem 5.16.** *Functionality $f$ is computable with with full-security if and only if $f$ is not semi-balanced.*

The details are left to reader.

### 5.3.2 The Multi-Party Setting

We can also extend the characterization to the multiparty setting when at most half of the parties are corrupted. Let $f : X_1 \times \ldots \times X_{2t} \to \Delta(\{0, 1\})$ denote a randomized symmetric functionality. Since a multi-party protocol is a two-party protocol in particular, it follows that every function obtained by partitioning the input domains into two equal sets must be fair in the two-party setting. Conversely, using secret sharing techniques [11, 12], we can devise a multi-party protocol which from the adversary's perspective, or the honest parties taken as a whole, essentially amounts to protocol $\textsc{FairTwoParty}_\sigma$. Thus, one can show that the necessary condition is sufficient. For $\mu \in \binom{[2t]}{t}$, write $f_\mu : X_\mu \times Y_\mu \to \Delta(\{0, 1\})$ for the two-party functionality where $X_\mu = \underset{i \in \mu}{\times} X_i$, $Y_\mu = \underset{j \in \mu \setminus [2t]}{\times} X_i$ and $f_\mu(\vec{x}, \vec{y}) = f(\vec{x}, \vec{y})$.

**Theorem 5.17.** *Multiparty functionality $f$ is computable with with full-security in the presence of relative corrupted majorities if and only if two-party functionality $f_\mu$ is not semi-balanced, for every $\mu \in \binom{[2t]}{t}$.*

We refer to [3] for an in-depth exposition.

# Part II

# An Analytical Framework for Fairness

## Special notation for Part II

For practical reasons, we reboot some of the notation. Let $[m]$ denote the set $\{0, \ldots, m-1\}$. We focus on arbitrary (possibly randomized) finite functions of the form $f : X \times Y \to \{0, \ldots, m-1\}^2$. Letters $\alpha$, $\beta$ denote values in $[m]$. In Chapter 7, the geometric distribution is chosen with parameter $\gamma$, instead of $\alpha$ as in Part I.

**Matrix Representation.** For any $f$, define $m^2$ matrices $\{M^{(a,b)}\}_{0 \leq a,b \leq m-1}$ where $M^{(a,b)}(x, y) = \Pr\left[f(x, y) = (a, b)\right]$. In addition, for every $\mu$ and $\nu \subseteq [m]$, let

$$M^{(\mu,\nu)} = \sum_{\substack{i \in \mu \\ j \in \nu}} M^{(i,j)} \enspace .$$

In other words, the entries of $M^{(\mu,\nu)}$ are the probabilities that the parties outputs' belong to $\mu \times \nu$ on the corresponding inputs. If $\mu$ or $\nu$ are equal to $[m]$, then we write $M^{(*,\nu)}$ and $M^{(\mu,*)}$ respectively, instead of $M^{(\mu,\nu)}$. Given an arbitrary matrix $C$, the $i$-th row and $j$-th column of $C$ will be denoted $[\,C\,]_{i,*}$ and $[\,C\,]_{*,j}$, respectively.

**Notation $*$.** We use this notation in a number of somewhat inconsistent but convenient ways that we believe are very easy to distinguish. Like Part I, $A * B$ denotes the entrywise (Hadamard) product of two matrices. As mentioned above, the notation may denote rows/columns of matrices, and it may also be used to denote a summation. Also, the notation may be used as a place-holder for inputs, either to avoid a universal quantifier, or to indicate that a value has not been assigned by the relevant party. Finally, $i^*$ denotes the special round of a threshold protocol – it is symbol in its own right and should not bring about any confusion. In any case, which of these interpretations applies should be obvious from the context.

# Transition from Part I to Part II

In Part I, we characterized symmetric Boolean functions with respect to full-security thanks to two complementary results/tools: The semi-balanced criterion that relies on Cleve's Theorem, and protocol FairTwoParty, a variant of GHKL. While these suffice for the characterization of Boolean functions, we cannot help but find that the insights they provide are lacking, as far as fairness is concerned. For one thing, the semi-balanced criterion relies on a technicality[5], and it is far from clear why it plays such an important role in fully secure computation.

In our view, the same can be said of GHKL. The security of the protocol is often attributed to its core characteristic – the special round $i^*$ – by arguing that the adversary never knows whether the special round has been reached, and that the all-or-nothing (memoryless) property of the geometric distribution guarantees that the probability of her guessing $i^*$ correctly doesn't get "too big". While this may be true whenever GHKL is fully secure, it says nothing about the shortcomings of the protocol. Specifically, it is unclear why the protocol requires tweaks in order to compute all the remaining functions that we now know to be fair.

In the end, the biggest flaw in the analysis so far is the lack of generality. Straightforward generalizations of the semi-balanced criterion and protocol FairTwoParty shed very little light on the characterization of arbitrary

[5]i.e. skilful or efficient way of doing or achieving something. Our use of the term is anything but disparaging.

functionalities. It must be said that our intention is *not* to downplay the importance of Cleve's theorem or GHKL. Rather, we are suggesting that each of these tools represents the core of some concept/notion we have yet to grasp.

**Example.** Define $f(x,y) = (f_1(x,y), f_2(x,y))$, where $f_1$, $f_2$ are given by the following matrices

$$M^{(1,*)} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \quad M^{(*,1)} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}.$$

The example serves to illustrate the limitations of our exposition thus far. It appears that only trivial sampling functionalities reduce[6] to $f$. What's more, protocols computing $f$ that follow naturally from GHKL and FairTwoParty fall short of full-security, yet a protocol with a most bizarre feature is fully-secure for $f$.

**Limits of** FairTwoParty **and GHKL.** Suppose the parties compute $f$ by executing an asymmetric variant of GHKL. Write $(a_0, \ldots, a_r)$ and $(b_0, \ldots, b_r)$ for the backup outputs of $P_1$ and $P_2$, respectively. Suppose that $P_2$ chooses his input uniformly at random from $\{y_1, y_2\}$. For any protocol computing $f$ with full security, this particular choice of inputs results in $P_2$ obtaining a uniform bit as an output, regardless of the actions of $P_1$.

We claim that a corrupt $P_1$ can bias $P_2$'s output towards 0, and thus the protocol does not compute $f$ with full security. Consider an adversary $\mathcal{A}$ using $x_3$ as an input, and aborting at the first or second round, if $a_1 = 1$ or $a_1 = 0$, respectively. Let's compute the probability that $P_2$'s output is equal to 1 in the presence of $\mathcal{A}$.

$$\Pr[\mathrm{out}_2 = 1] = \Pr[\mathrm{out}_2 = 1 \wedge i^* \neq 1] + \Pr[\mathrm{out}_2 = 1 \wedge i^* = 1] \quad ,$$

and thus

$$\Pr[\mathrm{out}_2 = 1] = (1-\alpha)\frac{1}{2} + \alpha \cdot \Big(\Pr[a_1 = 1 \wedge b_0 = 1 \,|\, i^* = 1] + \Pr[a_1 = 0 \wedge b_1 = 1 \,|\, i^* = 1]\Big)$$

$$(1-\alpha)\frac{1}{2} + \alpha \cdot \left(\frac{1}{4} + 0\right) = \frac{1}{2} - \alpha\frac{1}{4}.$$

[6]More precisely, augmenting the balanced criterion to asymmetric functions does not account for $f$.

Notice that $\mathcal{A}$ can guess $i^*$ with non-negligible probability ($\mathcal{A}$ is betting that $i^* = 1$), and since $P_1$ obtains $a_{i^*}$ before $P_2$ obtains $b_{i^*}$, it follows that $P_2$'s output suffers a bias. Now, in the spirit of protocol $\textsc{FairTwoParty}_\sigma$, suppose that $i^* \geq 2$ and $\Pr\left[b_{i^*-1} = 1 \,|\, y' = y\right] = p_y$, where $y'$ denotes the input handed to the dealer by $P_2$. Define $\mathcal{A}^{(0)}$ and $\mathcal{A}^{(1)}$ such that $\mathcal{A}^{(a)}$ uses $x_{3+a}$ as an input, and aborts at the second or third round if $a_2 = 1 - a$ or $a_2 = a$, respectively. For each case, let's compute the probability that $P_2$'s output is equal to 1 in the presence of each of these adversaries.

$$\begin{aligned}
\Pr\left[\mathrm{out}_2 = 1 \,|\, \mathcal{A}^{(a)}\right] &= \Pr\left[\mathrm{out}_2 = 1 \wedge i^* > 2\right] + \Pr\left[\mathrm{out}_2 = 1 \wedge i^* = 2\right] \\
&= (1-\alpha)\cdot\frac{1}{2} + \alpha\cdot\Big(\Pr\left[a_2 = 1 \wedge b_{2-a} = 1 \,|\, i^* = 2\right] \\
&\qquad\qquad + \Pr\left[a_2 = 0 \wedge b_{1+a} = 1 \,|\, i^* = 2\right]\Big) \\
&= \begin{cases} (1-\alpha)\cdot\dfrac{1}{2} + \alpha\cdot\dfrac{p_{y_1}}{2} & \text{if } a = 1 \\ (1-\alpha)\cdot\dfrac{1}{2} + \alpha\cdot\left(\dfrac{1}{2} + \dfrac{p_{y_1}}{2}\right) & \text{if } a = 0 \end{cases}
\end{aligned}$$

Conclude that for any value of $p_{y_1} \in [0, 1]$, at least one of the attacks results in a bias. We remark that a corrupted $P_1$ can bias $P_2$'s output regardless of how the function is described, i.e. the choice of associated matrices. Specifically, flipping some of the rows of $M^{(1,*)}$ and/or some of the columns of $M^{(*,1)}$ is not helpful, since the attacks can be easily modified to successfully bias $P_2$'s output, and, switching the players' roles is not helpful either, since one matrix is the transpose of the other.

**A Special Round Protocol with a Twist.** Consider Protocol FairTwoPartySpecial from Figure 5.5. We note that it is not susceptible[7] to any of the attacks mentioned above since, for $x \in \{x_3, x_4\}$, party $P_1$ receives his output only after $P_2$ receives his. Protocol FairTwoPartySpecial shares the familiar trait of having a special round, but unlike GHKL and FairTwoParty where only $P_1$ may pose a threat, our newest protocol gives the advantage to one party or the other depending on $P_1$'s input. In Appendix D p. 159, we show that FairTwoPartySpecial computes $f$ with full-security. Alternatively, a significantly shorter proof is provided at the end of Chapter 7 that relies on the main result from that chapter.

[7]We encourage to reader to consider what happens if the adversary chooses $x_1$ or $x_2$, and carries out a similar attack, or if a corrupted $P_2$ attempts to do so.

**Protocol** FAIRTWOPARTYSPECIAL

1. $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[8]
2. The dealer chooses $i^* \geq 1$ according to a geometric distribution with probability $\alpha$.
3. The dealer computes $\mathrm{out}_1 = f_1(x,y)$, $\mathrm{out}_2 = f_2(x,y)$ and for $0 \leq i \leq r$

$$a_i = \begin{cases} f_1(x, \widetilde{y}^{(i)}) \text{ where } \widetilde{y}^{(i)} \in_U Y & \text{if } i < i^* \\ f_1(x, \widetilde{y}^{(i)}) \text{ where } \widetilde{y}^{(i)} \in_U Y & \text{if } i = i^* \text{ and } x \in \{x_3, x_4\} \\ \mathrm{out}_1 & \text{otherwise} \end{cases}$$

$$b_i = \begin{cases} f_2(\widetilde{x}^{(i)}, y) \text{ where } \widetilde{x}^{(i)} \in_U X & \text{if } i < i^* \\ \mathrm{out}_2 & \text{otherwise.} \end{cases}$$

4. The dealer gives $b_0$ to $P_2$.
5. For $i = 1, \ldots, r$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

Figure 5.5: Protocol FAIRTWOPARTYSPECIAL for Computing $f$.

**Motivation for Part II.** The purpose of Part II is to generalize the characterization to arbitrary functionalities. We intend for the second half our exposition to be much more insightful. The topic of *Attacks on Fairness* serves as a starting point for Part II. We show that the different attacks from Part I are actually of the same "type", i.e. all of them follow a similar pattern, and they rely on a specific hybrid-model technique described by Figure 5.6. Next, we revisit each of these attacks.

In Section 5.2.2, we showed that $\text{FAIRTWOPARTY}_0$ is not fully-secure for any function whose associated matrix contains the all-1 vector in its image. In the proof, $P_2$ constructs an *ideally unbiased* bit as a deterministic function of his input/output. Then, by simply quitting at round 2, the adversary causes a bias. In Section 3.3, we showed that the GHKL-criterion is necessary by proving that, whenever it is not met for some function, a process similar to the one above excludes GHKL from computing the function with full security. The process involves an honest $P_2$ constructing an *ideally unbiased* bit, which the adversary can bias with a quitting strategy that is only slightly more elaborate than the one for protocol $\text{FAIRTWOPARTY}_0$.

Interestingly, the process from Figure 5.6 is also the crux of Cleve's Theorem

**Impossibility via Hybrid-Model**

1. The honest party chooses an input according to some pre-determined distribution.
2. The corrupted party is handed an input of the adversary's choice.
3. The parties execute the protocol, while the adversary caries out a fail-stop attack.
4. **Output**: the honest party applies a (possibly randomized) function to the input/output from the protocol and outputs a bit.

The purpose of Items 1 and 4 is to define *in the hybrid model* an output for the honest party – not necessarily equal to the output of the protocol/functionality – that is independent of the corrupted party's input. In other words, in the presence of a trusted party, Items 1 and 4 result in an (hybrid) output with a fixed probability distribution, regardless of the adversary's course of action. Using the terminology from coin-tossing, Items 1 and 4 result in an *ideally unbiased* bit. Since the distribution of the output is invariant in the hybrid model, to show that a protocol is not fully-secure, it suffices to specify an attack for Item 3 with a noticeable effect on the (real) distribution of the output, i.e. a *bias*.

Figure 5.6: Sketch of Impossibility Proof.

and its generalization (Section 4.1). Recall that in the proof of Cleve's Theorem, each round of a non-trivial sampling protocol is assigned to an adversary that quits at that round or the next depending on the backup output of the assigned round. Using an averaging argument, it is shown that at least one of the attacks results in a bias.

**Locking Strategies & Sampling Attacks.** For convenience, we introduce a couple of terms to the nomenclature. Items 1 and 4 will be referred to as a *locking strategy*, Item 3 will be referred to as a *sampling attack*. A sampling attack differs from an arbitrary fail-stop attack in that the former specifically aims at biasing the output resulting from some locking strategy. We say that a sampling attack (or the adversary) disrupts a locking strategy, if the attack is successful. Most of Part II discusses the implications of the proposition below.

**Proposition (Informal) .** If a protocol is fully secure, then no adversary may disrupt a locking strategy with a sampling attack.

Chapter 6

# Locking Strategies

Let $f : X \times Y \to [m]^2$ denote an arbitrary two-party function. Let $\Delta_i$ denote a distribution over the inputs of $P_i$, and let $\phi_i(z, c)$ denote a (possibly randomized) function, where $c \in [m]$ and $z$ denotes an element from $P_i$'s input domain. Suppose that in the hybrid model with ideal access to $f$, party $P_i$ chooses an input $z$ according to $\Delta_i$, and outputs $\phi_i(z, \mathsf{out}_i)$ – where $\mathsf{out}_i$ denotes the value sent to $P_i$ by the trusted party. We say that $(\Delta_i, \phi_i)$ is a *locking strategy* if $P_i$'s output from the procedure is independent of $P_{3-i}$'s input. To speak loosely, a locking strategy is a procedure allowing either party to "lock" the probability distribution of his output.

For the semi-balanced functions from Part I, the functions $\phi_i$ amounted to deterministic flips depending on the input, i.e. for a certain partitioning $Z_0 \sqcup Z_1$ of the input space, $z \in Z_j \Leftrightarrow \phi_i(z, c) = j \oplus c$. The strategies were deduced by solving a simple linear system. Naturally, we ask how to find relevant functions and distributions for arbitrary functions. In order to maintain the highest degree of generality, suppose that $\Phi_1$ and $\Phi_2$ denote the set of randomized functions that take inputs in $X \times [m]$ and $Y \times [m]$, respectively, and output elements in $\mathbb{N}$. Assume there exist $(\phi_1, \phi_2) \in \Phi_1 \times \Phi_2$ and distributions $\Delta_1$, $\Delta_2$ such that, in the hybrid model with ideal access to $f$,

$$\underset{x \leftarrow \Delta_1}{\phi_1} (x, f_1(x, *)), \qquad \underset{y \leftarrow \Delta_2}{\phi_2} (y, f_2(*, y))$$

are independent of $P_2$'s and $P_1$'s choice of input, respectively.

**Claim 6.1.** *Using the notation above, it follows that for every $\alpha \in \mathbb{N}$, and* *every $y,\ y' \in Y$*

$$\Pr_{x \leftarrow \Delta_1}[\phi_1(x, f_1(x,y)) = \alpha] = \Pr_{x \leftarrow \Delta_1}\left[\phi_1(x, f_1(x,y')) = \alpha\right] \ .$$

*Similarly, for every $\beta \in \mathbb{N}$, and every $x,\ x' \in X$*

$$\Pr_{y \leftarrow \Delta_2}[\phi_2(y, f_2(x,y)) = \beta] = \Pr_{y \leftarrow \Delta_2}\left[\phi_2(y, f_2(x',y)) = \beta\right] \ .$$

By definition, $(\Delta_1, \phi_1)$ and $(\Delta_2, \phi_2)$ yield statistically dependent outputs if there exists $(\alpha, \beta) \in \mathbb{N}^2$ such that

$$\Pr_{\substack{x \leftarrow \Delta_1 \\ y \leftarrow \Delta_2}}[(\phi_1(x, f_1(x,y)), \phi_2(y, f_2(x,y))) = (\alpha, \beta)] \neq \Pr_{x \leftarrow \Delta_1}[\phi_1(x, f_1(x,*)) = \alpha] \cdot \Pr_{y \leftarrow \Delta_2}[\phi_2(y, f_2(*,y)) = \beta] \ . \tag{6.1}$$

In line with the discussion from Chapter 4, we investigate how to find pairs $(\Delta_1, \phi_1)$ and $(\Delta_2, \phi_2)$ based on $f$. We begin by effectively restricting the function-space to those functions that have Boolean output. If there exists $(\alpha, \beta)$ as in Equation (6.1), then it is easy to see that pairs $(\Delta_1, \widetilde{\phi_1})$, $(\Delta_2, \widetilde{\phi_2})$ yield statistically dependent outputs, where

$$\widetilde{\phi_1}(x,a) = \begin{cases} 1 & \text{if } \phi_1(x,a) = \alpha \\ 0 & \text{otherwise} \end{cases} , \qquad \widetilde{\phi_2}(y,b) = \begin{cases} 1 & \text{if } \phi_2(y,b) = \beta \\ 0 & \text{otherwise} \end{cases} .$$

From now on, assume that $\phi_i : Z \times [m] \to \{0,1\}$.

**Definition 6.2.** We say that $\sigma_1 = (\Delta_1, \phi_1)$ is a *locking strategy* for $P_1$ with respect to $f$ if for every $y,\ y' \in Y$ it holds that

$$\Pr_{x \leftarrow \Delta_1}[\phi_1(x, f_1(x,y)) = 1] = \Pr_{x \leftarrow \Delta_1}\left[\phi_1(x, f_1(x,y')) = 1\right] \ .$$

Similarly, we say that $\sigma_2 = (\Delta_2, \phi_2)$ is a *locking strategy* for $P_2$ with respect to $f$ if for every $x,\ x' \in X$ it holds that

$$\Pr_{y \leftarrow \Delta_2}[\phi_2(y, f_2(x,y)) = 1] = \Pr_{y \leftarrow \Delta_2}\left[\phi_2(y, f_1(x',y)) = 1\right] \ .$$

Next, we generalize the semi-balanced criterion for arbitrary functions.

**Definition 6.3.** We say that $f$ is semi-balanced if there exist locking strategies $\sigma_1 = (\Delta_1, \phi_1)$ and $\sigma_2 = (\Delta_2, \phi_2)$ for $P_1$ and $P_2$ respectively such that the resulting outputs are statistically dependent, i.e.

$$\Pr_{\substack{x \leftarrow \Delta_1 \\ y \leftarrow \Delta_2}} [\phi_1(x, f_1(x,y)) = 1 \wedge \phi_2(y, f_2(x,y)) = 1] \neq \Pr_{x \leftarrow \Delta_1} [\phi_1(x, f_1(x,*)) = 1] \cdot \Pr_{y \leftarrow \Delta_2} [\phi_2(y, f_2(*,y)) = 1] \ .$$

In particular, we say that strategies $\sigma_1$ and $\sigma_2$ are dependent.

**Remark 6.4.** To alleviate notation, an economical approach is taken with regards to denoting summations. Specifically, the limits of sums are almost never denoted in full detail. In particular, "$\mathbf{z}_\xi(z) > 0$" in the lower limit of a sum indicates a summation over $\xi$ and $z$ such that $\mathbf{z}_\xi(z) > 0$, and other inequalities follow the same logic. Furthermore, "$\xi \,|\, c \in \xi$" in the lower limit of a sum indicates a summation over all $\xi$ such that $c \in \xi$.

## 6.1 Locking Strategies are (Quasi) Linear

We show that locking strategies are endowed with a quasi-linear structure, i.e. we can embed them into a vector space. This is highly desirable since using vectors and matrices, instead of distributions and functions, greatly simplifies the analysis. To show that such an embedding exists, we interpret $\phi_1$ and $\phi_2$ as probabilistic combinations of deterministic indicator functions. First, we introduce further notation. For every $y$, let $d_y = \Pr[y_0 = y \,|\, y_0 \leftarrow \Delta_2]$. Furthermore, let $\{b_0^{(y)}, b_1^{(y)}, \ldots, b_{m-1}^{(y)}\}$ be an ordering of $[m]$ such that

$$\Pr\left[\phi_2(y, b_0^{(y)}) = 1\right] \leq \Pr\left[\phi_2(y, b_1^{(y)}) = 1\right] \leq \cdots \leq \Pr\left[\phi_2(y, b_{m-1}^{(y)}) = 1\right] \ .$$

For $i \in \{-1, \ldots, m\}$ define $p_i^{(y)}$ such that

$$p_i^{(y)} = \begin{cases} 0 & \text{if } i = -1 \\ \Pr\left[\phi_2(y, b_i^{(y)}) = 1\right] & \text{if } 0 \leq i \leq m-1 \\ 1 & \text{if } i = m \end{cases} \ .$$

Observe that, for $i \in \{0, \ldots, m\}$, it holds that

$$\phi_2(y, b) = \tau_i^{(y)}(b) \quad \text{with probability } p_i^{(y)} - p_{i-1}^{(y)} \ ,$$

where $\tau_i^{(y)}(b) = 1$ if $b \in \{b_i^{(y)}, \dots, b_{m-1}^{(y)}\}$ and 0 otherwise. Let $\mathcal{M}$ consist of all the non-empty subsets of $\{0, 1 \dots, m-1\}$ of size strictly less than $m/2$, and the subsets of size $m/2$ that do not contain 0. Define vectors $\{\mathbf{y}_\mu\}_{\mu \in \mathcal{M}} \subseteq \mathbb{R}^k$, such that

$$\mathbf{y}_\mu(y) = \begin{cases} d_y \cdot (p_i^{(y)} - p_{i-1}^{(y)}) & \text{if } \tau_i^{(y)}(b) = 1 \Leftrightarrow b \in \mu \\ d_y \cdot (-p_i^{(y)} + p_{i-1}^{(y)}) & \text{if } \tau_i^{(y)}(b) = 1 \Leftrightarrow b \notin \mu \\ 0 & \text{otherwise} \end{cases} .$$

Finally, define vectors $\mathbf{y}_\emptyset$ and $\mathbf{y}_{[m]}$ such that

$$\mathbf{y}_\emptyset(y) = d_y \cdot (1 - p_{m-1}^{(y)}) \ ,$$
$$\mathbf{y}_{[m]}(y) = d_y \cdot p_0^{(y)} \ .$$

**Claim 6.5.** *Vectors $\{\mathbf{y}_\mu\}_{\mu \in \mathcal{M}}$ are well defined.*

*Proof.* $\{\mathbf{y}_\mu\}_{\mu \in \mathcal{M}}$ are well defined if every entry is accounted for exactly once. It suffices to show that the following sets have cardinality at most 1 and at least one of them is empty

$$\left\{ i \in \{1, \dots, m-1\} \;\middle|\; \tau_i^{(y)}(b) = 1 \Leftrightarrow b \in \mu \right\} \ ,$$
$$\left\{ i \in \{1, \dots, m-1\} \;\middle|\; \tau_i^{(y)}(b) = 1 \Leftrightarrow b \notin \mu \right\} \ . \quad (6.2)$$

Observe that for every $y$, for every $i \neq j$, it holds that $\tau_i^{(y)} \neq \tau_j^{(y)}$. Thus, the sets above have cardinality at most 1. It remains to show that at least one of the two is empty, i.e. for every $i, j \in [m]$, it holds that $\tau_i^{(y)} \neq 1 - \tau_j^{(y)}$. Note that

$$\tau_*^{(y)}(b) = 1 \Leftrightarrow b \in \{b_*^{(y)}, \dots, b_{m-1}^{(y)}\}$$

and thus $\tau_i^{(y)}(b_0^{(y)}) = \tau_j^{(y)}(b_0^{(y)})$, since $i, j \geq 1$. □

**Proposition 6.6.** *It holds that the procedure from Figure 6.1 is functionally equivalent to $\sigma_2$. In addition,*

$$\mathbf{1}_{\ell \times k} \cdot \mathbf{y}_{[m]} + \sum_{\mu \in \mathcal{M}} M^{(*,\mu)} \cdot \mathbf{y}_\mu = \left( \delta_2 - \sum_{\mathbf{y}_\mu(y) < 0} |\mathbf{y}_\mu(y)| \right) \cdot \mathbf{1}_\ell \ ,$$

**Locking Strategies from Vectors**

- **Offline.**
  1. With probability $\mathbf{y}_\emptyset(y)$ choose $y$ and $\tau$ such that $\tau \equiv 0$.
  2. With probability $\mathbf{y}_{[m]}(y)$ choose $y$ and $\tau$ such that $\tau \equiv 1$.
  3. With probability $|\mathbf{y}_\mu(y)|$ choose $y$ and $\tau$ such that
     a) If $\mathbf{y}_\mu(y) \geq 0$,
     $$\tau(b) = \begin{cases} 1 & \text{if } b \in \mu \\ 0 & \text{otherwise} \end{cases} .$$
     b) If $\mathbf{y}_\mu(y) < 0$,
     $$\tau(b) = \begin{cases} 1 & \text{if } b \notin \mu \\ 0 & \text{otherwise} \end{cases} .$$
- **Online.** $P_2$ sends $y$ to the trusted party. The latter replies with $b = f_2(x, y)$, where $x$ denotes the input chosen by $P_1$.
- **Output.** $P_2$ outputs $\tau(b)$.

Figure 6.1: How to Obtain Locking Strategies from Vectors.

*where $\delta_2$ denotes the probability that strategy $\sigma_2$ returns* 1*. Conversely, up to a multiplicative factor, vectors $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}} \cup \{\mathbf{y}_\emptyset, \mathbf{y}_{[m]}\}$ such that $\mathbf{y}_\emptyset, \mathbf{y}_{[m]} \geq 0$ define a locking strategy if*

$$\sum_{\mu\in\mathcal{M}} M^{(*,\mu)} \cdot \mathbf{y}_\mu = \delta_2 \cdot \mathbf{1}_\ell \ ,$$

*for some $\delta_2 \in \mathbb{R}$.*

*Proof.* Fix an arbitrary $x$ and observe that

$$\sum_{\mathbf{y}_\mu(y)<0} \Pr\left[f_2(x,y) \in \mu\right] \cdot \mathbf{y}_\mu(y) = \sum_{\mathbf{y}_\mu(y)<0} \left(1 - \Pr\left[f_2(x,y) \notin \mu\right]\right) \cdot \mathbf{y}_\mu(y)$$

By construction, $\delta_2$ is equal to

$$\sum_{\mathbf{y}_\mu(y)\geq 0} \Pr\left[f_2(x,y) \in \mu\right] \cdot \mathbf{y}_\mu(y) + \sum_{\mathbf{y}_\mu(y)<0} \Pr\left[f_2(x,y) \notin \mu\right] \cdot |\mathbf{y}_\mu(y)| + \sum_{y\in Y} \mathbf{y}_{[m]}(y) \ .$$

We omit the proof of the converse. □

### 6.1.1 Dependent Strategies

So far, we showed that strategies for $P_2$ are encoded by vectors $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}} \cup \{\mathbf{y}_\emptyset, \mathbf{y}_{[m]}\}$ that satisfy $\sum_{\mu\in\mathcal{M}} M^{(*,\mu)} \cdot \mathbf{y}_\mu = \delta_2 \cdot \mathbf{1}_\ell$, for some $\delta_2 \in \mathbb{R}$. Extend all of the above to the first party, and deduce that strategies for $P_1$ are encoded by vectors $\{\mathbf{x}_\mu\}_{\mu\in\mathcal{M}} \cup \{\mathbf{x}_\emptyset, \mathbf{x}_{[m]}\}$ that satisfy $\sum_{\mu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,*)} = \delta_1 \cdot \mathbf{1}_k^T$,for some $\delta_1 \in \mathbb{R}$.

**Proposition 6.7.** *Using the notation above, it holds that the associated locking strategies yield statistically dependent outputs if and only if*

$$\sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}_\nu \neq \delta_1\delta_2 \ .$$

*Proof.* To prove the claim, we show that

$$\Pr\left[(\mathsf{out}_1, \mathsf{out}_2) = (1,1)\right] = \sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}_\nu$$

$$+ \delta_1 \cdot \left( \langle \mathbf{y}_{[m]} \,|\, \mathbf{1}_k \rangle + \sum_{\mathbf{y}_\nu(y)<0} |\mathbf{y}_\nu(y)| \right) + \delta_2 \cdot \left( \langle \mathbf{x}_{[m]} \,|\, \mathbf{1}_\ell \rangle + \sum_{\mathbf{x}_\mu(x)<0} |\mathbf{x}_\mu(x)| \right)$$

$$+ \left( \langle \mathbf{x}_{[m]} \,|\, \mathbf{1}_\ell \rangle + \sum_{\mathbf{x}_\mu(x)<0} |\mathbf{x}_\mu(x)| \right) \cdot \left( \langle \mathbf{y}_{[m]} \,|\, \mathbf{1}_k \rangle + \sum_{\mathbf{y}_\nu(y)<0} |\mathbf{y}_\nu(y)| \right)$$

So, for every $(\mu,\nu) \in \mathcal{M}^2$, define matrix $\widehat{M}^{(\mu,\nu)}$ such that

$$\widehat{M}^{(\mu,\nu)}(x,y) = \begin{cases} \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu \times \nu\right] & \text{if } \mathbf{x}_\mu(x) \geq 0, \mathbf{y}_\nu(y) \geq 0 \\ \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu \times \overline{\nu}\right] & \text{if } \mathbf{x}_\mu(x) \geq 0, \mathbf{y}_\nu(y) < 0 \\ \Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu} \times \nu\right] & \text{if } \mathbf{x}_\mu(x) < 0, \mathbf{y}_\nu(y) \geq 0 \\ \Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu} \times \overline{\nu}\right] & \text{if } \mathbf{x}_\mu(x) < 0, \mathbf{y}_\nu(y) < 0 \end{cases} .$$

Let $|\mathbf{x}_\mu|$ and $|\mathbf{y}_\nu|$ denote the vectors obtained from $\mathbf{x}_\mu$ and $\mathbf{y}_\nu$ by taking the absolute value of all the entries. Clearly, $\Pr\left[(\mathsf{out}_1, \mathsf{out}_2) = (1,1)\right] =$

$$\sum_{\mu,\nu\in\mathcal{M}} |\mathbf{x}_\mu|^T \cdot \widehat{M}^{(\mu,\nu)} \cdot |\mathbf{y}_\nu| + \left( \delta_1 + \sum_{\mathbf{x}_\mu(x)<0} |\mathbf{x}_\mu(x)| \right) \cdot \langle \mathbf{y}_{[m]} \,|\, \mathbf{1}_k \rangle$$

$$+ \left( \delta_2 + \langle \mathbf{y}_{[m]} \,|\, \mathbf{1}_k \rangle + \sum_{\mathbf{y}_\mu(y)<0} |\mathbf{y}_\mu(y)| \right) \cdot \langle \mathbf{x}_{[m]} \,|\, \mathbf{1}_\ell \rangle \ .$$

Next, $|\mathbf{x}_\mu|^T \cdot \widehat{M}^{(\mu,\nu)} \cdot |\mathbf{y}_\nu| =$

$$\sum_{\substack{\mathbf{x}_\mu(x)\geq 0\\ \mathbf{y}_\nu(y)\geq 0}} \mathbf{x}_\mu(x)\cdot \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\nu\right]\cdot \mathbf{y}_\nu(y)$$
$$+ \sum_{\substack{\mathbf{x}_\mu(x)\geq 0\\ \mathbf{y}_\nu(y)< 0}} \mathbf{x}_\mu(x)\cdot \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\overline{\nu}\right]\cdot |\mathbf{y}_\nu(y)|$$
$$+ \sum_{\substack{\mathbf{x}_\mu(x)< 0\\ \mathbf{y}_\nu(y)\geq 0}} |\mathbf{x}_\mu(x)|\cdot \Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu}\times\nu\right]\cdot \mathbf{y}_\nu(y)$$
$$+ \sum_{\substack{\mathbf{x}_\mu(x)< 0\\ \mathbf{y}_\nu(y)< 0}} |\mathbf{x}_\mu(x)|\cdot \Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu}\times\overline{\nu}\right]\cdot |\mathbf{y}_\nu(y)| \ .$$

Now, note that

$$\begin{aligned}
&\Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\overline{\nu}\right] = \\
&\qquad \Pr\left[f_1(x,y)\in\mu\right] - \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\nu\right]\\
&\Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu}\times\nu\right] = \\
&\qquad \Pr\left[f_2(x,y)\in\nu\right] - \Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\nu\right] \ ,
\end{aligned}$$

and that

$$\begin{aligned}
&\Pr\left[(f_1(x,y), f_2(x,y)) \in \overline{\mu}\times\overline{\nu}\right] = \\
&\quad 1+\Pr\left[(f_1(x,y), f_2(x,y)) \in \mu\times\nu\right] - \Pr\left[f_1(x,y)\in\mu\right] - \Pr\left[f_2(x,y)\in\nu\right] \ .
\end{aligned}$$

From all of the above, it follows that $\sum_{\mu,\nu\in\mathcal{M}} |\mathbf{x}_\mu|^T \cdot \widehat{M}^{(\mu,\nu)} \cdot |\mathbf{y}_\nu| =$

$$\begin{aligned}
&\sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)}\cdot \mathbf{y}_\nu + \sum_{\mathbf{y}_\nu(y)<0} |\mathbf{y}_\nu(y)|\cdot \sum_{\mu} \mathbf{x}_\mu^T \left[M^{(\mu,*)}\right]_{*,y}\\
&\quad + \sum_{\mathbf{x}_\mu(x)<0} |\mathbf{x}_\mu(x)|\cdot \sum_{\nu} \left[M^{(*,\nu)}\right]_{x,*} \mathbf{y}_\nu + \sum_{\substack{\mathbf{x}_\mu(x)<0\\ \mathbf{y}_\nu(y)<0}} |\mathbf{x}_\mu(x)|\cdot|\mathbf{y}_\nu(y)|\\
&= \sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)}\cdot \mathbf{y}_\nu + \delta_1 \cdot \sum_{\mathbf{y}_\nu(y)<0} |\mathbf{y}_\nu(y)|\\
&\quad + \delta_2 \cdot \sum_{\mathbf{x}_\mu(x)<0} |\mathbf{x}_\mu(x)| + \sum_{\substack{\mathbf{x}_\mu(x)<0\\ \mathbf{y}_\nu(y)<0}} |\mathbf{x}_\mu(x)|\cdot|\mathbf{y}_\nu(y)|
\end{aligned}$$

□

## 6.2 Locking Strategies made Simple

To conclude, we restrict locking strategies to ones that are deemed "simple". Observe that the party applying a locking strategies chooses from among $2^m/2+1$ maps. In light of Proposition 6.7, let us assume that $\mathbf{y}_\emptyset = \mathbf{y}_{[m]} = \mathbf{0}_k$, in effect restricting the number of maps to $2^m/2-1$. In what follows, we show that we can restrict the number of maps to $m-1$.

Using the notation from the previous section, consider a locking strategy represented by vectors $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$. Let us move a few terms around in the expression below.

$$\begin{aligned}\sum_{\mu\in\mathcal{M}} M^{(*,\mu)}\cdot\mathbf{y}_\mu &= \sum_{\mu\in\mathcal{M}}\left(\sum_{b\in\mu} M^{(*,b)}\right)\cdot\mathbf{y}_\mu \\ &= M^{(*,0)}\cdot\left(\sum_{\mu\,|\,0\in\mu}\mathbf{y}_\mu\right) + \sum_{b=1}^{m-1} M^{(*,b)}\cdot\left(\sum_{\mu\,|\,b\in\mu}\mathbf{y}_\mu\right) \quad .\end{aligned}$$

Now, since $M^{(*,0)} = \mathbf{1}_{\ell\times k} - \sum_{b=1}^{m-1} M^{(*,b)}$, it follows that

$$\sum_{\mu\in\mathcal{M}} M^{(*,\mu)}\cdot\mathbf{y}_\mu = \left(\sum_{\mu\,|\,0\in\mu}\langle\mathbf{y}_\mu\,|\,\mathbf{1}_k\rangle\right)\cdot\mathbf{1}_\ell + \sum_{b=1}^{m-1} M^{(*,b)}\cdot\left(\sum_{\mu\,|\,b\in\mu}\mathbf{y}_\mu - \sum_{\mu\,|\,0\in\mu}\mathbf{y}_\mu\right) \tag{6.3}$$

Define[1] vectors $\{\mathbf{y}_b\}_{b\in[m]}$ such that

$$\mathbf{y}_b = \sum_{\mu\,|\,b\in\mu}\mathbf{y}_\mu - \sum_{\mu\,|\,0\in\mu}\mathbf{y}_\mu \quad .$$

We note that these vectors encode a new locking strategy. In accordance with the notation so far, the locking strategy associated with $\{\mathbf{y}_b\}_{b\in[m]}$ is functionally equivalent to the locking strategy associated with $\{\mathbf{y}'_\mu\}_{\mu\in\mathcal{M}}$, where

$$\mathbf{y}'_\mu = \begin{cases}\mathbf{y}_b & \text{if } \mu=\{b\} \\ \mathbf{0}_k & \text{otherwise}\end{cases} \quad .$$

[1]No confusion should arise between $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$ and $\{\mathbf{y}_b\}_{b\in[m]}$. One family consists of vectors indexed over *sets*, the other family is indexed over *integers*.

The fact that the resulting vectors encode a locking strategy follows from Equation (6.3), the fact that $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$ encode a locking strategy, and Proposition 6.6. Immediately, we see that $\{\mathbf{y}_b\}_{b\in[m]}$ is much "simpler" than $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$, since we have exponentially fewer maps to choose from. Next, we show that any loss of generality that this simplification incurs does not affect our analysis of the semi-balanced criterion.

**Proposition 6.8.** *Let $\{\mathbf{x}_\mu\}_{\mu\in\mathcal{M}}$, $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$ and $\{\mathbf{y}_b\}_{b\in[m]}$ be as above. The outputs from the locking strategies associated with $\{\mathbf{x}_\mu\}_{\mu\in\mathcal{M}}$ and $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$ are statistically dependent if and only the outputs from the strategies associated with $\{\mathbf{x}_\mu\}_{\mu\in\mathcal{M}}$ and $\{\mathbf{y}_b\}_{b\in[m]}$ are as well.*

*Proof.* Since $\{\mathbf{x}_\mu\}_{\mu\in\mathcal{M}}$ and $\{\mathbf{y}_\mu\}_{\mu\in\mathcal{M}}$ are locking strategies, let $\delta_1$, $\delta_2 \in \mathbb{R}$ such that $\delta_1 \cdot \mathbf{1}_k^T = \sum_{\mu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,*)}$ and $\delta_2 \cdot \mathbf{1}_\ell = \sum_{\nu\in\mathcal{M}} M^{(*,\nu)} \cdot \mathbf{y}_\nu$. The proof relies on applying Proposition 6.7. Let us compute

$$\begin{aligned}
\sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}'_\nu &= \sum_\mu \sum_{b=1}^{m-1} \mathbf{x}_\mu^T \cdot M^{(\mu,b)} \cdot \mathbf{y}_b \\
&= \sum_\mu \sum_{b=1}^{m-1} \mathbf{x}_\mu^T \cdot M^{(\mu,b)} \cdot \left( \sum_{\nu\,|\,b\in\nu} \mathbf{y}_\nu - \sum_{\nu\,|\,0\in\nu} \mathbf{y}_\nu \right) \\
&= \sum_\mu \mathbf{x}_\mu^T \cdot \left( \sum_{b=1}^{m-1} \sum_{\nu\,|\,b\in\nu} M^{(\mu,b)} \cdot \mathbf{y}_\nu - \sum_{b=1}^{m-1} \sum_{\nu\,|\,0\in\nu} M^{(\mu,b)} \cdot \mathbf{y}_\nu \right) \\
&= \sum_\mu \mathbf{x}_\mu^T \cdot \left( \sum_\nu \sum_{b\in\nu\setminus\{0\}} M^{(\mu,b)} \cdot \mathbf{y}_\nu - \sum_{\nu\,|\,0\in\nu} \sum_{b=1}^{m-1} M^{(\mu,b)} \cdot \mathbf{y}_\nu \right) .
\end{aligned}$$

Now, observe that $\sum_{b=1}^{m-1} M^{(\mu,b)} = \sum_{b=0}^{m-1} M^{(\mu,b)} - M^{(\mu,0)} = M^{(\mu,*)} - M^{(\mu,0)}$.

$$\sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}'_\nu =$$

$$\sum_\mu \mathbf{x}_\mu^T \cdot \left( \sum_\nu \sum_{b\in\nu\setminus\{0\}} M^{(\mu,b)} \cdot \mathbf{y}_\nu + \sum_{\nu\,|\,0\in\nu} \left( M^{(\mu,0)} - M^{(\mu,*)} \right) \cdot \mathbf{y}_\nu \right) .$$

Thus,

$$\begin{aligned}
\sum_{\mu,\nu\in\mathcal{M}} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}'_\nu &= \sum_{\mu} \mathbf{x}_\mu^T \cdot \left( \sum_{\nu} \sum_{b\in\nu} M^{(\mu,b)} \cdot \mathbf{y}_\nu - M^{(\mu,*)} \cdot \sum_{\nu \,|\, 0\in\nu} \mathbf{y}_\nu \right) \\
&= \sum_{\mu,\nu} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}_\nu - \left( \sum_{\mu} \mathbf{x}_\mu^T \cdot M^{(\mu,*)} \right) \sum_{\nu \,|\, 0\in\nu} \mathbf{y}_\nu \\
&= \sum_{\mu,\nu} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}_\nu - \delta_1 \cdot \mathbf{1}_k^T \cdot \sum_{\nu \,|\, 0\in\nu} \mathbf{y}_\nu \\
&= \sum_{\mu,\nu} \mathbf{x}_\mu^T \cdot M^{(\mu,\nu)} \cdot \mathbf{y}_\nu - \delta_1 \cdot \sum_{\nu \,|\, 0\in\nu} \langle \mathbf{y}_\nu \,|\, \mathbf{1}_k \rangle \quad .
\end{aligned}$$

To conclude, observe that by Equation (6.3),

$$\sum_{\nu\in\mathcal{M}} M^{(*,\nu)} \cdot \mathbf{y}'_\nu = \left( \delta_2 - \sum_{\nu \,|\, 0\in\nu} \langle \mathbf{y}_\nu \,|\, \mathbf{1}_k \rangle \right) \cdot \mathbf{1}_\ell \quad .$$

□

### 6.2.1 Main Theorem

Let $\mathbf{L}_2$ denote an arbitrary basis of the vector space consisting of all vectors $\mathbf{y}$ of the form $\mathbf{y}^T = (\mathbf{y}_1^T, \ldots, \mathbf{y}_{m-1}^T)$ such that

$$\left( M^{(*,1)} \,\middle\|\, M^{(*,2)} \,\middle\|\, \cdots \,\middle\|\, M^{(*,m-1)} \right) \cdot \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_{m-1} \end{pmatrix} = \delta_2 \cdot \mathbf{1}_\ell \quad ,$$

for some $\delta_2 \in \mathbb{R}$. Similarly let $\mathbf{L}_1$ denote an arbitrary basis of the vector space consisting of all vectors $\mathbf{x}$ of the form $\mathbf{x}^T = (\mathbf{x}_1^T, \ldots, \mathbf{x}_{m-1}^T)$ such that

$$\left( \mathbf{x}_1^T, \ldots, \mathbf{x}_{m-1}^T \right) \cdot \begin{pmatrix} M^{(1,*)} \\ \vdots \\ M^{(m-1,*)} \end{pmatrix} = \delta_1 \cdot \mathbf{1}_k^T \quad ,$$

for some $\delta_1 \in \mathbb{R}$.

**Theorem 6.9.** *Function $f$ is semi-balanced if and only if there exist $\mathbf{x} \in \mathbf{L}_1$ and $\mathbf{y} \in \mathbf{L}_2$ such that*

$$\sum_{a,b=1}^{m-1} \mathbf{x}_a^T \cdot M^{(a,b)} \cdot \mathbf{y}_b \neq \delta_1 \delta_2 \ ,$$

*where $\delta_1 \cdot \mathbf{1}_k^T = \sum_{a=1}^{m-1} \mathbf{x}_a^T \cdot M^{(a,*)}$ and $\delta_2 \cdot \mathbf{1}_\ell = \sum_{b=1}^{m-1} M^{(*,b)} \cdot \mathbf{y}_b$ .*

In this chapter, we defined locking strategies and we generalized the semi-balanced criterion. Locking strategies are algorithmic processes associated with functions, that enable the parties to output bits that are independent of each others' inputs. Strategies are defined by means of the communication model with ideal access to a given function $f$ i.e. the *hybrid model.* We showed that they are encoded by vectors and that, under certain conditions, they yield statistically dependent outputs, rendering the associated function unfair by Cleve.

**Remark 6.10.** The GHKL-criterion stipulates that the protocol computes $f$ with full security if and only if $\mathbf{v} \mapsto \big(M^T \cdot \mathbf{1}_k/k\big) * \mathbf{v}$ is an endomorphism of $\mathcal{H}(M^T)$. Using the new terminology, GHKL computes $f$ with full security if for every $\mathbf{v} \in \langle \mathbf{L}_2 \rangle$, the strategies associated with $\mathbf{v}$ and $\big(M^T \cdot \mathbf{1}_k/k\big) * \mathbf{v}$ are functionally equivalent.

CHAPTER 7

# Sampling Attacks

Recall the attacks from the proof of Cleve's Theorem. Depending on the corrupted party's backup output at some predetermined round, the adversary either aborts the computation immediately, *or* carries on honestly for exactly one more round. In this chapter, we discuss a distinct but somewhat related class of attacks that we refer to as sampling attacks. Informally, in a sampling attack, the adversary observes a subset of the corrupted party's backup outputs (of constant-size in the security parameter), and aborts either upon receiving the last back-up in the subset *or* not at all, i.e. carries on honestly. It is worth mentioning that sampling attacks do not include the attacks attributed to Cleve, although they are qualitatively "stronger", as we shall see later in the chapter.

The main result of this chapter is a generic transformation from a subclass of constant-round passively secure protocols to the class of fully-secure protocols. A protocol belongs to the relevant subclass if it satisfies an additional security requirement pertaining to locking strategies and sampling attacks.

## 7.1 Security against Sampling Attacks

Knowing that locking strategies are encoded by vectors, let $\mathbf{y}$ denote some locking strategy for $P_2$, and $\mathbf{L}_2$ denote an arbitrary basis of the relevant vector space. Suppose that $\mathcal{A}$ corrupts $P_1$ and define $\{I_n\}_{n\in\mathbb{N}}$ such that $I_n \subset \mathbb{N}$ and $|I_n| < \infty$, and $\exists n_0 \in \mathbb{N}$ such that $|I_n| = |I_{n_0}|$, for every $n \geq n_0$. Without loss of generality, assume that $I_n = \{j_0, j_1, \dots, j_{i_n}\}$, and that $j_0 < j_1 \dots < j_{i_n}$.

**Definition 7.1.** A *sampling attack* is fail-stop attack parametrized by $\{I_n\}_{n\in\mathbb{N}}$. The adversary executes the protocol with an input of her own choosing, observes the subsequence $(a_{j_0}, a_{j_1}, \ldots, a_{j_{i_n}})$ of the backup sequence, and, depending on the adversary's specifications, quits either at round $j_{i_n}$ or not at all, i.e. carries on honestly.

*Under what conditions does a sampling attack disrupt $P_2$'s locking strategy?*

Write $\widehat{\mathsf{out}}_2$ for the output resulting from the locking strategy in an execution of $\Pi$. Protocol $\Pi$ is *secure against sampling attacks* if for every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, for every $\mathcal{A}$ carrying out a sampling attack, it holds that $\left|\Pr\left[\widehat{\mathsf{out}}_2 = 1\right] - \delta_2\right| = \mathsf{negl}(n)$, where $\delta_2$ denotes the probability that $P_2$ outputs 1 in the hybrid model with ideal access to $f$. Next, for every $\{I_n\}_{n\in\mathbb{N}}$, for every $x \in X$, for every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, define $\mathbf{a}^-_{I_n} = (a_{j_0}, \ldots, a_{j_{i_n}}, \widehat{b}_{j_{i_n}-1})$ and $\mathbf{a}^+_{I_n} = (a_{j_0}, \ldots, a_{j_{i_n}}, \widehat{b}_r)$, where $\widehat{b}_*$ denotes the bit obtained from $b_*$ by applying $\mathbf{y}$.

**Proposition 7.2.** *The protocol is secure against sampling attacks if and only if for every $\{I_n\}_{n\in\mathbb{N}}$ as above, for every $x \in X$, for every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, it holds that random variables $\mathbf{a}^-_{I_n}$ and $\mathbf{a}^+_{I_n}$ are statistically close.*

*Proof.* In pursuit of a contradiction, say there exists $\{I_n\}_{n\in\mathbb{N}}$, and $(\vec{\alpha}_{I_n}, \beta_n) \in [m]^{|I_n|} \times \{0,1\}$ and $t \in \mathsf{poly}$ such that for an infinite number of $n$'s

$$\left|\Pr\left[\mathbf{a}^+_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right] - \Pr\left[\mathbf{a}^-_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right]\right| \geq \frac{1}{t(n)}$$

Now consider the following attack: If $(a_{j_0}, \ldots, a_{j_{i_n}}) = \vec{\alpha}_{I_n}$, quit at round $j_{i_n}$. Otherwise proceed honestly. We show that the attack results in a bias. Let's compute the probability that the honest party's output is equal to $\beta_n$.

$$\begin{aligned}\Pr\left[\widehat{\mathsf{out}}_2 = \beta_n\right] &= \Pr\left[\vec{a}_{I_n} = \vec{\alpha}_{I_n} \wedge \widehat{b}_{j_{i_n}-1} = \beta_n\right] + \Pr\left[\vec{a}_{I_n} \neq \vec{\alpha}_{I_n} \wedge \widehat{b}_r = \beta_n\right] \\ &= \Pr\left[\vec{a}_{I_n} = \vec{\alpha}_{I_n} \wedge \widehat{b}_{i-1} = \beta_n\right] - \Pr\left[\vec{a}_{I_n} = \vec{\alpha}_{I_n} \wedge \widehat{b}_r = \beta_n\right] \\ &\qquad + \Pr\left[\widehat{b}_r = \beta_n\right] \qquad (7.1)\end{aligned}$$

By correctness

$$\left|\Pr\left[\widehat{b}_r = \beta_n\right] - \Pr\left[\widehat{f}_2 = \beta_n\right]\right| \leq \mathsf{negl}(n) \ . \qquad (7.2)$$

Combining (7.1) and (7.2), and using the triangle inequality, it follows that

$$\begin{aligned}\left|\Pr\left[\widehat{\mathsf{out}}_2 = \beta_n\right] - \Pr\left[\widehat{f}_2 = \beta_n\right]\right| \geq \\ \left|\Pr\left[\widehat{\mathsf{out}}_2 = \beta_n\right] - \Pr\left[\widehat{b}_r = \beta_n\right]\right| - \left|\Pr\left[\widehat{b}_r = \beta_n\right] - \Pr\left[\widehat{f}_2 = \beta_n\right]\right| \\ \geq \\ \left|\Pr\left[\mathbf{a}^+_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right] - \Pr\left[\mathbf{a}^-_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right]\right| - \mathsf{negl}(n) \\ \geq \frac{1}{t(n)} - \mathsf{negl}(n) \geq \frac{1}{c \cdot t(n)} \ ,\end{aligned}$$

for any fixed $c > 1$ and $n$ large enough. Turning to the converse, suppose that for every $\{I_n\}_{n\in\mathbb{N}}$, for every $x$ and $\mathbf{y}$, and every $(\vec{\alpha}_{I_n}, \beta_n) \in [m]^{|I_n|} \times \{0,1\}$ it holds that

$$\left|\Pr\left[\mathbf{a}^+_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right] - \Pr\left[\mathbf{a}^-_{I_n} = (\vec{\alpha}_{I_n}, \beta_n)\right]\right| \leq \mathsf{negl}(n) \ .$$

Let $p_{\vec{\alpha}_{I_n}}$ denote the probability that the adversary quits at round $j_{i_n}$ having observed $\vec{\alpha}_{I_n}$. Let's compute the bias. By the triangle inequality,

$$\begin{aligned}\left|\Pr\left[\widehat{\mathsf{out}}_2 = 1\right] - \Pr\left[\widehat{f}_2 = 1\right]\right| \leq \\ \left|\Pr\left[\widehat{\mathsf{out}}_2 = 1\right] - \Pr\left[\widehat{b}_r = 1\right]\right| + \left|\Pr\left[\widehat{b}_r = 1\right] - \Pr\left[\widehat{f}_2 = 1\right]\right| \ .\end{aligned} \tag{7.3}$$

And thus

$$\begin{aligned}(7.3) &\leq \left|\sum_{\vec{\alpha}_{I_n}} \Pr\left[\widehat{\mathsf{out}}_2 = 1 \wedge \vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] - \Pr\left[\widehat{b}_r = 1 \wedge \vec{a}_{I_n} = \vec{\alpha}_{I_n}\right]\right| + \mathsf{negl}(n) \\ &\leq \sum_{\vec{\alpha}_{I_n}} p_{\vec{\alpha}_{I_n}} \cdot \left|\Pr\left[\mathbf{a}^-_{i} = (\vec{\alpha}_{I_n}, 1)\right] - \Pr\left[\mathbf{a}^+_{I_n} = (\vec{\alpha}_{I_n}, 1)\right]\right| + \mathsf{negl}(n) \\ &\leq \mathsf{negl}(n) \ .\end{aligned}$$

The first inequality follows by a simple expansion. The second inequality follows from the triangle inequality and the fact that

$$\begin{aligned}\Pr\left[\widehat{\mathsf{out}}_2 = 1 \wedge \vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] = \\ p_{\vec{\alpha}_{I_n}} \cdot \Pr\left[\widehat{b}_{j_{i_n}-1} = 1 \wedge \vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] + (1 - p_{\vec{\alpha}_{I_n}}) \cdot \Pr\left[\widehat{b}_r = 1 \wedge \vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] \ .\end{aligned}$$

The last inequality follows from the hypothesis of the claim and the fact that $|I_n|$ is constant in the security parameter. □

**Sampling Attacks vis-à-vis Fairness.** An informal definition of fairness states that *one party obtains output if and only if both do.* We ask if security against sampling attacks admits a statement along those lines. Fix $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$ and write $\widehat{f_2}$ for $P_2$'s output resulting from $\mathbf{y}$ in the hybrid model with ideal access to $f$. Observe that Proposition 7.2 admits the following interpretation: the protocol is susceptible to sampling attacks if, for some input $x \in X$, the adversary perceives a noticeable difference between the probability distributions of $\widehat{f_2}$ and $\widehat{b}_{j_{i_n}-1}$. Alternatively, one could say that a protocol is secure against sampling attacks if, at any given round, whatever the adversary can infer about about the honest party's output[1], the information is already contained in the honest party's backup output. To illustrate, consider the symmetric Boolean function given by

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} .$$

The second party has a unique locking strategy encoded by vector $\mathbf{q} = (-1, 1, 0, 1)^T$. Consider the single-round protocol defined by means of the backup outputs $(a_0, a_1)$ and $(b_0, b_1)$ such that $a_0 = f(x, \widetilde{y})$, where $\widetilde{y} \in_U Y$, $b_0 = 1$ and $a_1 = b_1 = f(x, y)$. Write $\widehat{b}_*$ for the bit obtained from $b_*$ by applying strategy $\mathbf{q}$. In the hybrid model, $P_2$'s output is equal to 1 with probability 2/3, regardless of $P_1$'s choice of input. On the other hand, in the real model, observe that

| $x \setminus \Pr[\widehat{b}_i = 1 \mid a_1 = \alpha]$ | $\alpha = 0$ | $\alpha = 1$ |
|---|---|---|
| $x_1$ | 1/2 | 1 |
| $x_2$ | 1/2 | 1 |
| $x_3$ | 1/2 | 1 |
| $x_4$ | $\rightarrow\leftarrow$ | 2/3 |

for $i \in \{0, 1\}$, and thus from $P_1$'s perspective, $\widehat{b}_1$ and $\widehat{b}_0$ are identically distributed.

**Sampling vs Cleve-type attacks.** We show that if a protocol is secure against sampling attacks, then it is also secure against Cleve-type attacks. Using the notation above and going back to the proof of Cleve's Theorem in Chapter 4, a protocol is secure against Cleve-type attacks if, for every $x \in$

[1]Resulting from the locking strategy.

$X$, for every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, and every $i \in \mathbb{N}$, the following pairs are statistically close

$$(a_i, \widehat{b}_{i-1}), \qquad (a_i, \widehat{b}_i) \ .$$

If the protocol is secure against sampling attacks, it follows that

$$(a_i, a_{i+1}, \widehat{b}_i), \quad (a_i, a_{i+1}, \widehat{b}_r)$$

are close, and obviously $(a_i, \widehat{b}_i)$ and $(a_i, \widehat{b}_r)$ are as well. Now, once again because the protocol is secure against sampling attacks, we have that $(a_i, \widehat{b}_{i-1})$ and $(a_i, \widehat{b}_r)$ are close, and since statistical closeness is transitive, we conclude that $(a_i, \widehat{b}_i)$ and $(a_i, \widehat{b}_{i+1})$ are close as well. Hence, sampling attacks are *at least as good as* Cleve-type attacks.

### 7.1.1 Sampling Attacks in Linear-Algebraic Terms

In this section, we show how security against sampling attacks can be expressed in linear-algebraic terms. First, we define closeness for vectors. Let $\{\mathbf{v}_n\}_{n\in\mathbb{N}}$ and $\{\mathbf{u}_n\}_{n\in\mathbb{N}}$ denote two families of vectors indexed by $\mathbb{N}$. We say that $\mathbf{v}_n$ is *close* to $\mathbf{u}_n$ if $\|\mathbf{u}_n - \mathbf{v}_n\| \leq \mathsf{negl}(n)$.

**Definition 7.3.** For every $\{I_n\}_{n\in\mathbb{N}}$, for every $\vec{\alpha}_{I_n} = (\alpha_{j_1}, \dots, \alpha_{I_{j_{i_n}}}) \in [m]^{j_{i_n}+1}$ with $j_{i_n} \geq 1$, and every $\beta \in [m]$, define matrices $B_-^{(\vec{\alpha}_{I_n},\beta)}$, $B_+^{(\vec{\alpha}_{I_n},\beta)} \in \mathbb{R}^{\ell\times k}$ such that

$$B_-^{(\vec{\alpha}_{I_n},\beta)}(x,y) = \Pr\left[(\vec{a}_{I_n}(x,y), b_{j_{i_n}-1}(x,y))) = (\vec{\alpha}_{I_n},\beta)\right]$$
$$B_+^{(\vec{\alpha}_{I_n},\beta)}(x,y) = \Pr\left[(\vec{a}_{I_n}(x,y), b_r(x,y))) = (\vec{\alpha}_{I_n},\beta)\right] \ .$$

Similarly, for every $\vec{\beta}_i = (\beta_{j_1}, \dots, \beta_{j_{i_n}}) \in [m]^i$ and every $\alpha \in [m]$ define matrices $A_-^{(\alpha,\vec{\beta}_{I_n})}$, $A_+^{(\alpha,\vec{\beta}_{I_n})} \in \mathbb{R}^{\ell\times k}$ such that

$$A_-^{(\alpha,\vec{\beta}_{I_n})}(x,y) = \Pr\left[\left(a_{j_{i_n}}(x,y), \vec{b}_{I_n}(x,y)\right) = (\alpha, \vec{\beta}_{I_n})\right]$$
$$A_+^{(\alpha,\vec{\beta}_{I_n})}(x,y) = \Pr\left[\left(a_r(x,y), \vec{b}_{I_n}(x,y)\right) = (\alpha, \vec{\beta}_{I_n})\right] \ .$$

**Theorem 7.4.** *Protocol $\Pi$ is secure against sampling attacks if and only if*

- *for every* $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, *for every* $\{I_n\}_{n\in\mathbb{N}}$, *for every* $\vec{\alpha}_{I_n} \in [m]^{|I_n|}$, *the vector below is close to* $\mathbf{0}_\ell$.

$$\left(B_+^{(\vec{\alpha}_{I_n},1)} - B_-^{(\vec{\alpha}_{I_n},1)} \,\Big\|\, \cdots \,\Big\|\, B_+^{(\vec{\alpha}_{I_n},m-1)} - B_-^{(\vec{\alpha}_{I_n},m-1)}\right) \cdot \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_{m-1} \end{pmatrix} \tag{7.4}$$

- *for every* $\mathbf{x} \in \langle \mathbf{L}_1 \rangle$, *for every* $\{I_n\}_{n\in\mathbb{N}}$, *for every* $\vec{\beta}_{I_n} \in [m]^{|I_n|}$, *the vector below is close to* $\mathbf{0}_k^T$.

$$\left(\mathbf{x}_1^T, \dots, \mathbf{x}_{m-1}^T\right) \cdot \begin{pmatrix} A_+^{(1,\vec{\beta}_{I_n})} - A_-^{(1,\vec{\beta}_{I_n})} \\ \vdots \\ A_+^{(m-1,\vec{\beta}_{I_n})} - A_-^{(m-1,\vec{\beta}_{I_n})} \end{pmatrix} \tag{7.5}$$

*Proof.* Write $\widehat{f}_2$ for $P_2$'s output resulting from some strategy $\mathbf{y}^T = (\mathbf{y}_1^T, \dots, \mathbf{y}_{m-1}^T)$ and let $|\mathbf{y}|$ denote the vector obtained from $\mathbf{y}$ by taking the absolute value of all the entries. In addition, define $\widehat{B}_+^{(\vec{\alpha}_{I_n},\beta)}$ and $\widehat{B}_-^{(\vec{\alpha}_{I_n},\beta)}$ such that

$$\widehat{B}_+^{(\vec{\alpha}_{I_n},\beta)}(x,y) = \begin{cases} \Pr\left[(\vec{a}_{I_n}(x,y), b_r(x,y)) = (\vec{\alpha}_{I_n}, \beta)\right] & \text{if } \mathbf{y}_\beta(y) \geq 0 \\ \Pr\left[\vec{a}_{I_n}(x,y) = \vec{\alpha}_{I_n} \ \wedge \ b_r(x,y) \neq \beta\right] & \text{if } \mathbf{y}_\beta(y) < 0 \end{cases},$$

$$\widehat{B}_-^{(\vec{\alpha}_{I_n},\beta)}(x,y) = \begin{cases} \Pr\left[(\vec{a}_{I_n}(x,y), b_{j_{i_n}-1}(x,y)) = (\vec{\alpha}_{I_n}, \beta)\right] & \text{if } \mathbf{y}_\beta(y) \geq 0 \\ \Pr\left[\vec{a}_{I_n}(x,y) = \vec{\alpha}_{I_n} \ \wedge \ b_{j_{i_n}-1}(x,y) \neq \beta\right] & \text{if } \mathbf{y}_\beta(y) < 0 \end{cases}.$$

Let us compute the probability that $(\vec{a}_{I_n}, \widehat{f}_2)$ is equal to $(\vec{\alpha}_{I_n}, 1)$.

$$\begin{aligned} \Pr\left[(\vec{a}_{I_n}, \widehat{f}_2) = (\vec{\alpha}_{I_n}, 1)\right] &= \sum_{b=1}^{m-1} \widehat{B}_+^{(\vec{\alpha}_{I_n},\beta)} \cdot |\mathbf{y}_b| \\ &= \sum_{b=1}^{m-1} B_+^{(\vec{\alpha}_{I_n},\beta)} \cdot \mathbf{y}_b + \sum_{\mathbf{y}_b(j)<0} |\mathbf{y}_b(j)| \cdot \left[B_+^{(\vec{\alpha}_{I_n},*)}\right]_{*,j} \end{aligned}$$

$$\begin{aligned} \Pr\left[(\vec{a}_{I_n}, \widehat{b}_{j_{i_n}-1}) = (\vec{\alpha}_{I_n}, 1)\right] &= \sum_{b=1}^{m-1} \widehat{B}_-^{(\vec{\alpha}_{I_n},\beta)} \cdot |\mathbf{y}_b| \\ &= \sum_{b=1}^{m-1} B_-^{(\vec{\alpha}_{I_n},\beta)} \cdot \mathbf{y}_b + \sum_{\mathbf{y}_b(j)<0} |\mathbf{y}_b(j)| \cdot \left[B_-^{(\vec{\alpha}_{I_n},*)}\right]_{*,j} \end{aligned}$$

Notice that

$$\Pr\left[(\vec{a}_{I_n}, \widehat{f}_2) = (\vec{\alpha}_{I_n}, 1)\right] = \Pr\left[\vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] - \Pr\left[(\vec{a}_{I_n}, \widehat{f}_2) = (\vec{\alpha}_{I_n}, 0)\right]$$
$$\Pr\left[(\vec{a}_{I_n}, \widehat{b}_{j_{i_n}-1}) = (\vec{\alpha}_{I_n}, 1)\right] = \Pr\left[\vec{a}_{I_n} = \vec{\alpha}_{I_n}\right] - \Pr\left[(\vec{a}_{I_n}, \widehat{b}_{i-1}) = (\vec{\alpha}_{I_n}, 0)\right],$$

and thus $(\vec{a}_{I_n}, \widehat{f}_2)$ is close to $(\vec{a}_{I_n}, \widehat{b}_{j_{i_n}-1})$ if the vector in (7.4) is close to $\mathbf{0}_\ell$. We omit the proof for an honest $P_1$. □

## 7.2 Towards Full Security

Needless to say, our goal is to design protocols that are fully secure. We show that constant-round protocols that satisfy passive security *and* security against sampling attacks are easily transformed into fully secure protocols. Let $\Pi$ be a protocol for computing $f$ satisfying all the relevant properties. We model the protocol in the usual way. The parties's backup outputs for $\Pi$ will be denoted $(c_0, \ldots, c_{r'})$ and $(d_0, \ldots, d_{r'})$, respectively, where $r'$ denotes the number of rounds.

We assume that $r'$ is *constant* in the security parameter. This assumption is desirable for reasons that will become clear in the next section, and it is good enough for our purposes. Nevertheless, the question of determining the optimal round complexity for protocols that are passively secure and secure against sampling attacks may be of independent interest. We conjecture that *for the class of functions we have in mind*, i.e. finite & constant-size domain, if such protocols exist, then constant-round protocols exhibiting the same security features should exist as well.

**Conjecture 7.5.** *For any finite function $f$, passive security and security against sampling attacks can either be achieved in a constant number of rounds or not at all.*

Since the protocol is assumed to be passively secure, there exist simulators, that we denote $\{\mathcal{S}_i^{\mathsf{p}}\}_{i\in\{1,2\}}$, that can recreate the backup sequences in the ideal model. In addition, since the protocol is constant-round, it follows that the ideal sequences are statistically close to the real ones. Formally,

$$(c_0, \ldots, c_{r'}, d_{r'})^{\mathsf{Real}} \overset{s}{\equiv} (c_0, \ldots, c_{r'}, f_2)^{\mathsf{Ideal}}$$
$$(d_0, \ldots, d_{r'}, c_{r'})^{\mathsf{Real}} \overset{s}{\equiv} (d_0, \ldots, d_{r'}, f_1)^{\mathsf{Ideal}} \quad .$$

Since the protocol is constant-round, by assumption, Theorem 7.4 applies to $\Pi$ in a very straightforward way. Using the notation from the previous section,

- For every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, for every $i = 1, \dots, r'$, for every $\vec{\alpha}_i \in [m]^{i+1}$, the vector below is close $\mathbf{0}_\ell$.
$$\left( B_+^{(\vec{\alpha}_i,1)} - B_-^{(\vec{\alpha}_i,1)} \middle\| \cdots \middle\| B_+^{(\vec{\alpha}_i,m-1)} - B_-^{(\vec{\alpha}_i,m-1)} \right) \cdot \mathbf{y} \ .$$
- For every $\mathbf{x} \in \langle \mathbf{L}_1 \rangle$, for every $i = 0, \dots, r' - 1$, for every $\vec{\beta}_i \in [m]^{i+1}$, the vector below is close to $\mathbf{0}_k^T$.
$$\mathbf{x}^T \cdot \begin{pmatrix} A_+^{(1,\vec{\beta}_i)} - A_-^{(1,\vec{\beta}_i)} \\ \vdots \\ A_+^{(m-1,\vec{\beta}_i)} - A_-^{(m-1,\vec{\beta}_i)} \end{pmatrix} \ .$$

We are going to *combine* the main ingredient of the GHKL protocol – the threshold round $i^*$ – with the protocol above. Specifically, we are going to instruct the parties to run a protocol such that, at some point in the execution, unbeknownst to them, the parties begin running $\Pi$.

This is achieved by choosing a random threshold round according to a geometric distribution. Prior to that round, the parties exchange backup outputs that are independent of each other, and, once the threshold round has been reached, the parties exchange backups according to the specifications of $\Pi$. Formally, consider protocol SecSamp2Fair($\Pi$) from Figure 7.1. For the new protocol, $i^* \geq r' + 1$ is chosen according to a geometric distribution with parameter $\gamma$. If $i < i^* - r'$, then $a_i$ and $b_i$ are independent of one another. If $i^* - r' \leq i < i^*$, then $a_i$ and $b_i$ are equal to $c_{i-i^*+r'}$ and $d_{i-i^*+r'}$, respectively. Finally, if $i \geq i^*$, then $a_i$ and $b_i$ are equal to $f_1(x, y)$ and $f_2(x, y)$, respectively.

## 7.3 Security Analysis

We only deal with the case where $P_1$ is corrupted. The other case is virtually analogous. Write $\mathcal{A}$ for the adversary corrupting $P_1$. We begin with a high-level description of the simulator. The simulator $\mathcal{S}$ chooses $i^*$ according to the specifications of the protocol, and simulates the rounds of the protocol

**Protocol** $\textsc{SecSamp2Fair}(\Pi)$

1. The parties $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[a]
2. The dealer executes $\Pi$ locally on inputs $x$ and $y$ and security parameter $n$. Write $(c_0, \ldots, c_{r'})$ and $(d_0, \ldots, d_{r'})$ for the sequences of backup outputs computed by the dealer.
3. The dealer chooses $i^* \geq r' + 1$ according to the geometric distribution with parameter $\gamma$.
4. The dealer computes $(\mathsf{out}_1, \mathsf{out}_2) = (f_1(x,y), f_2(x,y))$, and for $0 \leq i \leq r$

$$a_i = \begin{cases} f_1(x, \widetilde{y}^{(i)}) \text{ where } \widetilde{y}^{(i)} \in_U Y & \text{if } i < i^* - r \\ c_{i-(i^*-r')} & \text{if } i^* - r' \leq i < i^* \\ \mathsf{out}_1 & \text{otherwise} \end{cases}$$

and

$$b_i = \begin{cases} f_2(\widetilde{x}^{(i)}, y) \text{ where } \widetilde{x}^{(i)} \in_U X & \text{if } i < i^* - r \\ d_{i-(i^*-r')} & \text{if } i^* - r' \leq i < i^* \\ \mathsf{out}_2 & \text{otherwise.} \end{cases}$$

5. The dealer gives $b_0$ to $P_2$.
6. For $i = 1, \ldots, r$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

[a]If $x$ is not in the appropriate domain or $P_1$ does not hand an input, then the dealer sends $f(\hat{x}, y)$ (where $\hat{x}$ is a default value) to $P_2$, which outputs this value and the protocol is terminated. The case of an inappropriate $y$ is dealt analogously.

Figure 7.1: Protocol $\textsc{SecSamp2Fair}(\Pi)$ for Computing $f$.

as follows. Prior to iteration/round $i^* - r'$, the simulator generates backup outputs in exactly the same way as the dealer does in the real model. If the adversary decides to abort, $\mathcal{S}$ sends $x_0 \in X$ to the trusted party, where $x_0$ is sampled according to probability vector $\mathbf{z}_x^{(\vec{\alpha}_{r'})} \in \mathbb{R}^\ell$. As the notation suggests, $\mathbf{z}_x^{(\vec{\alpha}_{r'})}$ depends on $x$ (the input handed by the adversary for the computation) and the last $r' + 1$ backup outputs computed by the simulator. At iteration $i^* - r'$, assuming the adversary is still active, the simulator hands $x$ to the trusted party, and receives output $a = f_1(x, y)$. In order to reconstruct the next values of the backup sequence, the simulator invokes $\mathcal{S}_2^{\mathsf{p}}$, and hands one-by-one to $\mathcal{A}$ the values computed by $\mathcal{S}_2^{\mathsf{p}}$. At every iteration following $i^*$, the simulator hands $a$ to $\mathcal{A}$. At any given point, if the adversary aborts, the simulator outputs the sequence of values he handed to $\mathcal{A}$, and halts.

**Why does this work?** By definition, the simulator's output together with the honest party's output in the ideal model is required to be indistinguishable from the adversary's view and the honest party's output in the real model. In our case, the adversary's view corresponds to the sequence of backup outputs she observes. Notice that the backup up sequences of each world are statistically close, which follows from the way $i^*$ is chosen in both worlds, the passive security of $\Pi$, and the fact that prior to $i^* - r'$ the backup outputs in the real and ideal world are identically distributed. The hard part is to argue that there exists $\mathbf{z}_x^{(\vec{\alpha}_{r'})}$ from which the simulator can sample from. As we shall see, the existence of $\mathbf{z}_x^{(\vec{\alpha}_{r'})}$ follows from a corollary of the fundamental theorem of Linear Algebra, which comes into play because of the security against sampling attacks assumption.

As a warm-up to the proof, we verify that introducing the threshold round does not affect security against sampling attacks. In Protocol $\Pi$, an immediate abort on the part of $P_1$ results in $P_2$ outputting $d_0$. However, by introducing the threshold round, aborting immediately upon receiving $c_0$ (i.e. $a_{i^*-r'}$) results in $P_2$ outputting $b_{i^*-r'-1}$. Write $\widehat{b}_*$ for the bit obtained from $b$ by applying some locking strategy $\mathbf{y}$. It suffices to show that $(c_0, \widehat{b}_{i^*-r'-1})$ and $(c_0, \widehat{b}_r)$ are close, for every $x \in X$ and every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$. Define[2] matrices $B_-^{(\alpha,\beta)}$ and $B_+^{(\alpha,\beta)} \in \mathbb{R}^{\ell \times k}$ such that

$$B_-^{(\alpha,\beta)}(x, y) = \Pr\left[(c_0(x, y), b_{i^*-r'-1}(x, y)) = (\alpha, \beta)\right] \ ,$$

[2]We remark that these matrices were not previously defined.

**The simulator $\mathcal{S}$ for Protocol** SecSamp2Fair(Π)

- The adversary $\mathcal{A}$ gives its input $x$ to the simulator.[a]
- The simulator chooses $i^* \geq r' + 1$ according to the geometric distribution with parameter $\gamma$.
- For $i = 1, \dots, i^* - r' - 1$:
  - The simulator gives $a_i = f(x, \widetilde{y}^{(i)})$ to the adversary $\mathcal{A}$, where $\widetilde{y}^{(i)}$ is chosen according to the uniform distribution.
  - If $\mathcal{A}$ aborts, then the simulator chooses an input $x_0$ according to a distribution $\mathbf{z}_x^{(a_{i-\rho},\dots,a_i)}$ (which depends on the input $x$ and the last sequence $\rho + 1$ values that the simulator generated, where $\rho = \min(r', i)$), gives $x_0$ to the trusted party, outputs the bits $a_1, \dots, a_i$, and halts.
- At round $i = i^* - r'$, the simulator gives $x$ to the trusted party and gets the output $a = f_1(x, y)$.
  - The simulator constructs $(a_{i^*-r'}, \dots, a_{i^*-1})$ such that $a_{i^*-r'+j} = \widehat{a}_j$ by invoking $\mathcal{S}_2^{\mathsf{p}}$ on input $x$, output $a = f_1(x, y)$ and security parameter $n$.
- For $i = i^* - r', \dots, i^* - 1$: The simulator gives $a_i$ to the adversary $\mathcal{A}$, if $\mathcal{A}$ aborts, then the simulator outputs the bits $a_1, \dots, a_i$ and halts.
- For $i = i^*, \dots, r$: The simulator gives $a_i = a$ to the adversary $\mathcal{A}$, if $\mathcal{A}$ aborts, then the simulator outputs the bits $a_1, \dots, a_i$ and halts.
- The simulator outputs the bits $a_1, \dots, a_r$ and halts.

[a]If the adversary gives an inappropriate $x$ (or no $x$), then the simulator sends some default $\hat{x} \in X$ to the trusted party, outputs the empty string, and halts.

Figure 7.2: The Simulator $\mathcal{S}$ for Protocol SecSamp2Fair(Π).

and

$$B_+^{(\alpha,\beta)}(x,y) = \Pr\left[(c_0(x,y), b_{i^*}(x,y)) = (\alpha,\beta)\ .\right]$$

We argue that, for any $\mathbf{y} \in \mathbf{L}_2$,

$$\left[B_+^{(\alpha,0)} - B_-^{(\alpha,0)} \middle\| \cdots \middle\| B_+^{(\alpha,m-1)} - B_-^{(\alpha_{r'},m-1)}\right]_{x,*} \cdot \mathbf{y} = \mathbf{0}_\ell\ . \tag{7.6}$$

Since $\Pi$ is defined in the plain model, and $c_0$ is constructed before any communication occurs, it follows that $c_0$ is a function of $P_1$'s input and $P_1$'s local coins. Similarly, $b_{i^*-r'-1}(x,y)$ can be viewed as a function of $P_2$'s input and $P_2$'s local coins. Hence, $c_0(x,y)$ is independent of $b_{i^*-r'-1}(x,y)$. Next, notice that $c_0$ is independent of $\widehat{f_2}$, where the latter denotes $P_2$'s output resulting from $\mathbf{y}$ in the hybrid model. Indeed, $P_2$'s output from $\mathbf{y}$ is independent of $x$, and $f_2(x,y)$ is independent of $P_1$'s local coins. Thus,

$$\Pr\left[(c_0(x,\mathbf{y}), \widehat{f_2}(\widetilde{x},\mathbf{y})) = (\alpha,\beta) \,\middle|\, \widetilde{x} \in_U X\right] = \\ \Pr\left[c_0(x) = \alpha\right] \cdot \Pr\left[\widehat{f_2}(\widetilde{x},\mathbf{y}) = \beta \,\middle|\, \widetilde{x} \in_U X\right] \tag{7.7}$$

and

$$\Pr\left[(c_0(x,\mathbf{y}), \widehat{f_2}(x,\mathbf{y})) = (\alpha,\beta)\right] = \Pr\left[c_0(x) = \alpha\right] \cdot \Pr\left[\widehat{f_2}(x,\mathbf{y}) = \beta\right]. \tag{7.8}$$

Equations (7.7) and (7.8) imply Equation (7.6). Moving on, recall that for $i = 1 \ldots r'$ matrices $B_-^{(\alpha_0,\ldots,\alpha_i,\beta)}$ and $B_+^{(\alpha_0,\ldots,\alpha_i,\beta)}$ denote

$$B_-^{(\alpha_0\ldots\alpha_i,\beta)}(x,y) = \Pr\left[(c_0,\ldots,c_i,d_{i-1})(x,y) = (\alpha_0,\ldots,\alpha_i,\beta)\right]$$
$$B_+^{(\alpha_0\ldots\alpha_i,\beta)}(x,y) = \Pr\left[(c_0,\ldots,c_i,d_{r'})(x,y) = (\alpha_0,\ldots,\alpha_i,\beta)\right]$$

Now, define $p_x^{(\alpha)} = \Pr\left[f_1(x,\widetilde{y}) = \alpha \,|\, \widetilde{y} \in_U Y\right]$. To alleviate notation, we will omit the security parameter.

As mentioned earlier, the corrupted party's backup sequences in the real and ideal world are statistically close. Therefore, if the adversary quits in the real world, then the adversary quits in the ideal world as well with all but negligible probability – and vice versa. The whole point of the simulation is to show that early aborts do not breach security. In particular, if the adversary quits *after* round $i^*$, then the relevant distributions[3] in the real and ideal world are statistically close. Our analysis only deals with aborts that take place prior to round $i^*$.

Regarding the adversary's view, we only focus on the last $r' + 1$ elements of the corrupted party's backup sequence. Having assumed that $i^*$ has not been surpassed, anything prior the last $r' + 1$ elements is essentially noise, and it has no bearing on the security analysis. For every sequence of elements $\vec{\alpha}_{r'} \in [m]^{r'+1}$ and every $\beta \in [m]$, we compute the probability that the adversary's view and honest party's output in the real world is equal to $(\vec{\alpha}_{r'}, \beta)$, and we express the result in terms of the $B_-^{(\ ,\ )}$-matrices. Similarly, for the ideal world, we compute the probability that the simulator's output and honest party's output is equal to $(\vec{\alpha}_{r'}, \beta)$, and we express the result in terms of the $B_+^{(\ ,\ )}$-matrices and vector $\mathbf{z}_x^{(\vec{a}_{r'})}$.

The point of the exercise is to obtain (linear) constraints for vector $\mathbf{z}_x^{(\vec{a}_{r'})}$. Then, we ask if the constraints are satisfiable, and, if so, whether solutions can be found efficiently. The second question can be readily answered. If an appropriate solution exists, the simulator can compute it efficiently. Indeed, the simulator can approximate the probability distribution of all possible sequences of size $r' + 1$, and, assuming it exists, the simulator computes $\mathbf{z}_x^{(\vec{a}_{r'})}$ by solving a linear system of size $|X| \times (m-1) \cdot |Y|$. Thus, it suffices to show that $\mathbf{z}_x^{(\vec{a}_{r'})}$ exists. The security features of $\Pi$ come into play in this regard.

An early abort on the part of the adversary alters the conditional[4] probability distribution of the honest party's output. Security against sampling attacks guarantees that the output remains consistent with the function at hand. Thus, by introducing a threshold round and fine-tuning its parameter, we restrict[5] the distribution of the output until it falls within the range of the function, and the simulator can match it with an appropriate input. In that case, an early abort is essentially equivalent to lying about one's input from the beginning. Our analysis relies on a statistical variant of the Fundamental Theorem of Linear Algebra and a couple of arguments from topology.

We remark that, while the case where the adversary aborts before round $r'$ needs special consideration, in reality, the only difference is that $\mathbf{z}_x^{(\vec{a}_i)}$

[3]The adversary's view together with the honest party's output.

[4]conditioned on the adversary's view.

[5]Obviously, this argument does no apply to unfair functions, like semi-balanced function.

depends on fewer elements. The analysis is largely the same and we do not address this case any further. Finally, we assume that $p_x^{(\alpha)} \neq 0$, for every $\alpha \in [m]$ and $x \in X$. This assumption allows for a smoother exposition by disregarding degenerate cases.

### 7.3.1 Real vs Ideal.

For every sequence $\vec{\alpha}_{r'} = (\alpha_0, \ldots, \alpha_{r'}) \in [m]^{r'+1}$ and every $\beta \in [m]$, we compute the probability that the adversary observes $\vec{\alpha}_{r'}$ and the honest party outputs $\beta$.

**Claim 7.6.** *In the real model, it holds that*

$$\begin{aligned}\Pr\left[(a_{i-r'}, \ldots, a_i, b_{i-1})^{\mathsf{Real}} = (\vec{\alpha}_{r'}, \beta) \,\middle|\, i \leq i^*\right] = \\ (1-\gamma)^{r'+1} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'})} \cdot \mathbf{q}^{(\beta)T} + \gamma(1-\gamma)^{r'} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'-1})} \cdot \left[B_-^{(\alpha_{r'},\beta)}\right]_{x,*} + \\ \cdots + \gamma(1-\gamma) \cdot p_x^{(\alpha_0)} \cdot \left[B_-^{(\alpha_1 \ldots \alpha_{r'},\beta)}\right]_{x,*} + \gamma \cdot \left[B_-^{(\vec{\alpha}_{r'},\beta)}\right]_{x,*} ,\end{aligned}$$

*where* $\mathbf{q}^{(\beta)} = M^{(*,\beta)T} \cdot \mathbf{1}_\ell$.

*Proof.* Simple expansion over possible values of $i^*$. □

Define $\mathbf{c}_x^{(\vec{\alpha}_{r'},\beta)} \in \mathbb{R}^k$ such that $\mathbf{c}_x^{(\vec{\alpha}_{r'},\beta)}(y) = \Pr\left[f_2(x_0, y_i) = \beta \,\middle|\, x_0 \leftarrow \mathbf{z}_x^{(\vec{\alpha}_{r'})}\right]$ .

**Claim 7.7.** *In the ideal model, it holds that*

$$\begin{aligned}\Pr\left[(a_{i-r'}, \ldots, a_i, f_2)^{\mathsf{Ideal}} = (\vec{\alpha}_{r'}, \beta) \,\middle|\, i \leq i^*\right] = \\ (1-\gamma)^{r'+1} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'})} \cdot \mathbf{c}_x^{(\vec{\alpha}_{r'},\beta)T} + \gamma(1-\gamma)^{r'} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'-1})} \cdot \left[B_+^{(\alpha_{r'},\beta)}\right]_{x,*} + \\ \cdots + \gamma(1-\gamma) \cdot p_x^{(\alpha_0)} \cdot \left[B_+^{(\alpha_1 \ldots \alpha_{r'},\beta)}\right]_{x,*} + \gamma \cdot \left[B_+^{(\vec{\alpha}_{r'},\beta)}\right]_{x,*} .\end{aligned}$$

Thus, we require that $\mathbf{c}_x^{(\vec{\alpha}_{r'},\beta)T}$ is close to

$$\mathbf{q}^{(\beta)T} + \sum_{i=0}^{r'} \lambda_i(\gamma, \vec{\alpha}_{r'}) \cdot \left[B_-^{(\alpha_{r'-i} \ldots \alpha_{r'},\beta)} - B_+^{(\alpha_{r'-i} \ldots \alpha_{r'},b)}\right]_{x,*} ,$$

where

$$\lambda_i(\gamma, \vec{\alpha}_{r'}) = \frac{\gamma(1-\gamma)^{r'-i} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'-i-1})}}{(1-\gamma)^{r'+1} \cdot p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'})}} = \frac{\gamma}{(1-\gamma)^{i+1}} \cdot \frac{1}{p_x^{(\alpha_{r'-i})} \cdots p_x^{(\alpha_{r'})}} \ .$$

**Proposition 7.8.** *If there exists probability vector* $\mathbf{z}_x^{(\vec{\alpha}_{r'})} \in \mathbb{R}^k$ *such that* $\mathbf{z}_x^{(\vec{\alpha}_{r'})T} \cdot \left(M^{(*,1)} \,\middle\|\, \cdots \,\middle\|\, M^{(*,m-1)}\right)$ *is close to*

$$\left(\mathbf{q}^{(1)T} \,\middle\|\, \cdots \,\middle\|\, \mathbf{q}^{(m-1)T}\right) + \lambda_0 \cdot \left[B_+^{(\alpha_{r'},1)} - B_-^{(\alpha_{r'},1)} \,\middle\|\, \cdots \,\middle\|\, B_+^{(\alpha_{r'},m-1)} - B_-^{(\alpha_{r'},m-1)}\right]_{x,*} + \\ \ldots + \lambda_{r'} \cdot \left[B_+^{(\vec{\alpha}_{r'},1)} - B_-^{(\vec{\alpha}_{r'},1)} \,\middle\|\, \cdots \,\middle\|\, B_+^{(\vec{\alpha}_{r'},m-1)} - B_-^{(\vec{\alpha}_{r'},m-1)}\right]_{x,*} \ ,$$

*then Protocol* SecSamp2Fair(Π) *is fully secure.*

*Proof.* If $\mathbf{z}_x^{(\vec{\alpha}_{r'})T} \cdot M^{(*,\beta)}$ is close to $\mathbf{q}^{(\beta)T} + \left[B_+^{(\vec{\alpha}_{r'},m-1)} - B_-^{(\vec{\alpha}_{r'},m-1)}\right]_{x,*}$, for every $\beta \in [m]$, it follows that $\mathbf{z}_x^{(\vec{\alpha}_{r'})T} \cdot \left(M^{(*,0)} \,\middle\|\, M^{(*,1)} \,\middle\|\, \cdots \,\middle\|\, M^{(*,m-1)}\right)$ is close to

$$\left(\mathbf{q}^{(0)T} \,\middle\|\, \cdots \,\middle\|\, \mathbf{q}^{(m-1)T}\right) + \lambda_0 \cdot \left[B_+^{(\alpha_{r'},0)} - B_-^{(\alpha_{r'},0)} \,\middle\|\, \cdots \,\middle\|\, B_+^{(\alpha_{r'},m-1)} - B_-^{(\alpha_{r'},m-1)}\right]_{x,*} + \\ \ldots + \lambda_{r'} \cdot \left[B_+^{(\vec{\alpha}_{r'},0)} - B_-^{(\vec{\alpha}_{r'},0)} \,\middle\|\, \cdots \,\middle\|\, B_+^{(\vec{\alpha}_{r'},m-1)} - B_-^{(\vec{\alpha}_{r'},m-1)}\right]_{x,*} \ .$$

In light of the equations

$$M^{(*,0)} = \mathbf{1}_{\ell \times k} - \sum_{\beta=1}^{m-1} M^{(*,\beta)} \ , \quad \mathbf{q}^{(0)T} = \mathbf{1}_k - \sum_{\beta=1}^{m-1} \mathbf{q}^{(\beta)T}$$

$$B_+^{(\alpha_i \ldots \alpha_r,*)} - B_-^{(\alpha_i \ldots \alpha_r,*)} = \sum_{\beta=0}^{m-1} B_+^{(\alpha_i \ldots \alpha_r,\beta)} - B_-^{(\alpha_i \ldots \alpha_r,\beta)} = \mathbf{0}_{\ell \times k}$$

and the fact that $\mathbf{z}_x^{(\vec{\alpha}_{r'})}$ is a probability vector, conclude that the first item in the concatenation can be discarded. □

**Corollary 7.9.** *If there exists* $\mathbf{u}_x^{(\vec{\alpha}_{r'})}$ *of the form*

$$\begin{cases} \sum_{x_0} \mathbf{u}_x^{(\vec{\alpha}_{r'})}(x_0) = 0 \\ \forall y, \ \mathbf{u}_x^{(\vec{\alpha}_{r'})}(y) \in [1/\ell, 1 - 1/\ell] \end{cases}$$

*such that* $\mathbf{u}_x^{(\vec{\alpha}_{r'})T} \cdot \left(M^{(*,1)} \| \cdots \| M^{(*,m-1)}\right)$ *is close to*

$$\lambda_0 \cdot \left[B_+^{(\alpha_{r'},1)} - B_-^{(\alpha_{r'},1)} \middle\| \cdots \middle\| B_+^{(\alpha_{r'},m-1)} - B_-^{(\alpha_{r'},m-1)}\right]_{x,*} + \cdots + \lambda_{r'} \cdot \left[B_+^{(\vec{\alpha}_{r'},1)} - B_-^{(\vec{\alpha}_{r'},1)} \middle\| \cdots \middle\| B_+^{(\vec{\alpha}_{r'},m-1)} - B_-^{(\vec{\alpha}_{r'},m-1)}\right]_{x,*} ,$$

*then Protocol* SecSamp2Fair(Π) *is fully secure.*

*Proof.* Define $\mathbf{u}_x^{(\vec{\alpha}_{r'})} = \mathbf{z}_x^{(\vec{\alpha}_{r'})} - \mathbf{1}_\ell/\ell$ and note that

$$\mathbf{1}_\ell^T/\ell \cdot \left(M^{(*,1)} \middle\| \cdots \middle\| M^{(*,m-1)}\right) = \left(\mathbf{q}^{(1)T} \middle\| \cdots \middle\| \mathbf{q}^{(m-1)T}\right) .$$

□

### 7.3.2 Four Useful Lemmas

**Lemma 7.10.** *Let $M$ be an arbitrary matrix and let $\{\mathbf{v}_n\}_{n\in\mathbb{N}}$ be a family of vectors such that $|\langle \mathbf{c} \,|\, \mathbf{v}_n\rangle| = \mathsf{negl}(n)$, for every $\mathbf{c} \in \ker(M)$. Then, there exist $\{\mathbf{z}_n\}_{n\in\mathbb{N}}$ such that $M^T\mathbf{z}_n$ is close to $\mathbf{v}_n$.*

*Proof.* From standard linear algebra, we know that there exist $\{\mathbf{u}_n\}_{n\in\mathbb{N}}$ and $\{\mathbf{c}_n\}_{n\in\mathbb{N}}$ such that $\mathbf{u}_n \in \mathrm{im}(M^T)$, $\mathbf{c}_n \in \ker(M)$ and $\mathbf{v}_n = \mathbf{u}_n + \mathbf{c}_n$. Now, let $\mathbf{K}$ denote an arbitrary orthonormal basis of $\ker(M)$ and observe that, for any $\mathbf{c} \in \mathbf{K}$,

$$\begin{aligned} |\langle \mathbf{c} \,|\, \mathbf{v}_n\rangle| &= |\langle \mathbf{c} \,|\, \mathbf{u}_n\rangle + \langle \mathbf{c} \,|\, \mathbf{c}_n\rangle| \\ &= |\langle \mathbf{c} \,|\, \mathbf{c}_n\rangle| = \mathsf{negl}(n) \ , \end{aligned}$$

and thus $\|\mathbf{c}_n\|^2 = \sum_{c\in\mathcal{K}} \langle \mathbf{c} \,|\, \mathbf{c}_n\rangle^2 = |\mathbf{K}| \cdot (\mathsf{negl}(n))^2 = \mathsf{negl}(n)$. □

**Lemma 7.11.** *Let $M \in \mathbb{R}^{d\times d'}$ be an arbitrary matrix and let $\{\mathbf{c}_n\}_{n\in\mathbb{N}}$ denote some family of vectors in $\mathbb{R}^{d'}$. There exists $\{\mathbf{u}_n\}_{n\in\mathbb{N}}$ such that $\sum_j \mathbf{u}_n(j) = 0$ and $M^T\mathbf{u}_n$ is close to $\mathbf{c}_n$ if and only if $\langle \mathbf{c}_n \,|\, \mathbf{v}\rangle = \mathsf{negl}(n)$, for every $\mathbf{v}$ satisfying $M \cdot \mathbf{v} \in \langle \mathbf{1}_d\rangle$.*

*Proof.* ($\Leftarrow$) Let $\mathbf{w}_n = \mathbf{c}_n^T - \mathbf{u}_n^T \cdot M$. By assumption, $\|\mathbf{w}_n\| = \mathsf{negl}(n)$. Deduce that for any $\mathbf{v}$ such that $M \cdot \mathbf{v} = \delta \cdot \mathbf{1}_d$, it holds that

$$\begin{aligned} \langle \mathbf{c}_n \,|\, \mathbf{v}\rangle &= (\mathbf{u}_n^T \cdot M + \mathbf{w}_n^T) \cdot \mathbf{v} \\ &= \delta \cdot \sum_j \mathbf{u}_n(j) + \mathbf{w}_n^T \cdot \mathbf{v} = 0 + \mathsf{negl}(n) \ , \end{aligned}$$

since $\sum_j \mathbf{u}_n(j) = 0$ by assumption, and $|\langle \mathbf{w}_n \,|\, \mathbf{v}\rangle| \leq \|\mathbf{w}_n\| \cdot \|\mathbf{v}\|$ by the Cauchy-Schwartz inequality. ($\Rightarrow$) For the converse, define

$$M' = \begin{pmatrix} -1 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & 0 & \dots & 1 \end{pmatrix} \cdot M \ .$$

We argue that there exists $\{\mathbf{u}_n\}_{n\in\mathbb{N}}$ of the form $\sum_j \mathbf{u}_n(j) = 0$ such that $\mathbf{u}_n^T \cdot M$ is close to $\mathbf{c}_n^T$ if and only if there exists arbitrary $\{\mathbf{u}'_n\}_{n\in\mathbb{N}}$ such that $\mathbf{u}_n'^T \cdot M$ is close to $\mathbf{c}_n^T$. Indeed, it suffices to note that the row-space of $M'$ is equal to the the image of the hyperplane $\{\mathbf{z} \in \mathbb{R}^d \mid \sum_j \mathbf{z}(j) = 0\}$ by $M^T$. Finally, note that $\ker(M') = \left\{\mathbf{v} \in \mathbb{R}^{d'} \,\middle|\, M \cdot \mathbf{v} \in \langle \mathbf{1}_d \rangle \right\}$ and conclude using the previous lemma. □

**Lemma 7.12.** *Let $M \in \mathbb{R}^{d\times d'}$ be an arbitrary matrix and define*

$$\mathbf{V} = \left\{\mathbf{v} \in \mathbb{R}^{d'} \,\middle|\, M\mathbf{v} \in [-1,1]^d \wedge \mathbf{v} \in \mathrm{im}(M^T) \right\} \ .$$

*There exists $\rho \in \mathbb{R}^+$ such that $\|\mathbf{v}\| \leq \rho$, for every $\mathbf{v} \in \mathbf{V}$.*

*Proof.* Let $T : \mathrm{im}(M^T) \to \mathbb{R}^d$ such that $T(\mathbf{v}) = M\mathbf{v}$ and note that $T$ is injective and continuous. In addition, note that $[-1,1]^d \cap \mathrm{im}(M)$ is topologically compact. It follows that the preimage of $[-1,1]^d \cap \mathrm{im}(M)$ by $T$ is compact, i.e. closed and bounded. □

**Lemma 7.13.** *Let $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{u}' \in \mathbb{R}^{d-1}$ such that $\mathbf{u}^T = \left[\mathbf{u}(1) \,\|\, \mathbf{u}'^T\right]$ and $\mathbf{u}(1) = -\sum_{i=1}^{d-1} \mathbf{u}'(i)$. It holds that $\|\mathbf{u}\| \leq \|\mathbf{u}'\| \cdot \sqrt{d}$.*

*Proof.* First of all, $\|\mathbf{u}\|^2 = \langle \mathbf{1}_{d-1} \,|\, \mathbf{u}' \rangle^2 + \|\mathbf{u}'\|^2$. Then, $\langle \mathbf{u}' \,|\, \mathbf{1}_{d-1} \rangle^2 \leq \|\mathbf{u}'\| \cdot \|\mathbf{1}_{d-1}\|$ by the Cauchy-Schwartz inequality. Conclude. □

### 7.3.3 Connecting the Dots

**Theorem 7.14.** *Let $\Pi$ be a protocol for computing $f$. If $\Pi$ is constant-round, passively secure, and secure against sampling attacks, then $f$ is computable with full security.*

*Proof.* We apply Corollary 7.9 in combination with the Lemmas of the previous section. Since $\Pi$ is secure against sampling attacks, deduce that for any $i \in \{0, \ldots, r'\}$, for any $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, and any $(\alpha_{r'-i} \ldots \alpha_{r'}) \in [m]^{i+1}$, it holds that the value of

$$\left[ B_+^{(\alpha_{r'-i} \ldots \alpha_{r'},1)} - B_-^{(\alpha_{r'-i} \ldots \alpha_{r'},1)} \middle\| \cdots \middle\| B_+^{(\alpha_{r'-i} \ldots \alpha_{r'},m-1)} - B_-^{(\alpha_{r'-i} \ldots \alpha_{r'},m-1)} \right]_{x,*} \cdot \mathbf{y}$$

is close to 0. As a consequence, there exist $\{\mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}\}_{i=0}^{r'}$ such that

- $\left( \sum_{i=0}^{r'} \lambda_i \mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})T} \right) \cdot \left( M^{(*,1)} \| \cdots \| M^{(*,m-1)} \right)$ is close to

$$\lambda_0 \cdot \left[ B_+^{(\alpha_{r'},1)} - B_-^{(\alpha_{r'},1)} \middle\| \cdots \middle\| B_+^{(\alpha_{r'},m-1)} - B_-^{(\alpha_{r'},m-1)} \right]_{x,*} + \\ \cdots + \lambda_{r'} \cdot \left[ B_+^{(\vec{\alpha}_{r'},1)} - B_-^{(\vec{\alpha}_{r'},1)} \middle\| \cdots \middle\| B_+^{(\vec{\alpha}_{r'},m-1)} - B_-^{(\vec{\alpha}_{r'},m-1)} \right]_{x,*} .$$

- $\sum_{x_0 \in X} \sum_{i=0}^{r'} \lambda_i \mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}(x_0) = 0$ .

- $\exists \rho_i \in \mathbb{R}^+$ such that $\rho_i \geq \|\mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}\|$, regardless of the security parameter.

The first two items follow from Lemma 7.11. For the last item, observe that

$$\left[ B_+^{(\alpha_{r'-i} \ldots \alpha_{r'},\beta)} - B_-^{(\alpha_{r'-i} \ldots \alpha_{r'},\beta)} \right]_{x,*} \in [-1,1]^k$$

and apply Lemmas 7.12 and 7.13. It remains to show that

$$\left| \sum_{i=0}^{r'} \lambda_i \mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}(x_0) \right| \leq 1/\ell \ ,$$

for every $x_0 \in X$. Recall that

$$\lambda_i(\gamma, \vec{\alpha}_{r'}) = \frac{\gamma}{(1-\gamma)^{i+1}} \cdot \frac{1}{p_x^{(\alpha_{r'-i})} \cdots p_x^{(\alpha_{r'})}}$$

and thus

$$\left| \sum_{i=0}^{r'} \lambda_i \mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}(x_0) \right| \leq \frac{\gamma}{(1-\gamma)^{r'+1}} \cdot \frac{(r'+1) \cdot \max(\rho_j)}{p_x^{(\alpha_0)} \cdots p_x^{(\alpha_{r'})}} \ . \tag{7.9}$$

To conclude, argue that we can fix $\gamma \in [0,1]$ such that the right-hand side of Equation (7.9) is upper-bounded by $1/\ell$. Alternatively, set $\gamma(n) = 1/\log(n)$, and deduce that

$$\left|\sum_{i=0}^{r'} \lambda_i \mathbf{u}_{x,i}^{(\vec{\alpha}_{r'})}(x_0)\right| \leq 1/\ell \ ,$$

for $n$ large enough. □

## 7.4 Applications

In this chapter, we showed how to build fully-secure protocols starting with a constant-round passively secure protocol satisfying an additional requirement pertaining to sampling attacks. The most logical step to take next, is to investigate how to design such protocols (c.f. Chapter 8).

We remark that fully secure protocols from Part I can be viewed as special cases of our approach. Below, we specify the underlying constant-round, passively secure, and secure against sampling attack protocol for every protocol we encountered so far. We conclude with a short proof of full security for protocol FairTwoPartySpecial.

- **Protocol** GHKL.

| Round | $P_1$ | $P_2$ |
|---|---|---|
| 0 | $a_0$ | $b_0$ |
| 1 | $a_1$ | $b_1$ |

0. $(a_0, b_0) = (f(x, \widetilde{y}), f(\widetilde{x}, y))$, where $\begin{cases} \widetilde{x} \in_U X \\ \widetilde{y} \in_U Y \end{cases}$ .
1. $(a_1, b_1) = (f(x, y), f(x, y))$.

- **Protocol** FairTwoParty$_\sigma$.

| Round | $P_1$ | $P_2$ |
|---|---|---|
| 0 | $a_0$ | $b_0$ |
| 1 | $a_1$ | $b_1$ |

0. $(a_0, b_0) = (f(x, \widetilde{y}), \sigma)$, where $\widetilde{y} \in_U Y$.

1. $(a_1, b_1) = (f(x, y), f(x, y))$.

- **Protocol** FairTwoPartySpecial**.**

| Round | $P_1$ | $P_2$ |
|---|---|---|
| 0 | $a_0$ | $b_0$ |
| 1 | $a_1$ | $b_1$ |
| 2 | $a_2$ | $b_2$ |

0. $(a_0, b_0) = (f_1(x, \widetilde{y}), f_2(\widetilde{x}, y))$, where $\begin{cases} \widetilde{x} \in_U X \\ \widetilde{y} \in_U Y \end{cases}$.

1. $(a_1, b_1) = \begin{cases} (f_1(x, \widetilde{y}'), f_2(x, y)) & \text{if } x \in \{x_1, x_2\}, \text{ where } \widetilde{y}' \in_U Y \\ (f_1(x, y), f_2(x, y)) & \text{if } x \in \{x_3, x_4\} \end{cases}$.

2. $(a_2, b_2) = (f_1(x, y), f_2(x, y))$.

### 7.4.1 Short Proof of Security for FairTwoPartySpecial

In light of Theorem 7.14, it suffices to show that the 3-round protocol underlying protocol FairTwoPartySpecial is passively secure and secure against sampling attacks. The fact that the protocol is passively secure is immediate. Recall function $f$ defined by means of the following matrices.

$$M^{(1,*)} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad M^{(*,1)} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}.$$

Observe that each party effectively has a unique locking strategy, namely

$$\mathbf{p} = \mathbf{q} = \begin{pmatrix} 1/2 \\ 1/2 \\ 0 \\ 0 \end{pmatrix}.$$

Now, the distributions of $(a_1, b_0)$ and $(a_1, b_2)$ are described by the matrices below

$$B_-^{(0,1)} = \begin{pmatrix} 0 & 0 & 0 & 1/2 \\ 3/4 & 1/4 & 1/2 & 0 \\ 3/8 & 1/8 & 1/4 & 1/4 \\ 3/8 & 1/8 & 1/4 & 1/4 \end{pmatrix}, \quad B_+^{(0,1)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 1/2 & 1/2 \end{pmatrix}$$

$$B_-^{(1,1)} = \begin{pmatrix} 3/4 & 1/4 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 \\ 3/8 & 1/8 & 1/4 & 1/4 \\ 3/8 & 1/8 & 1/4 & 1/4 \end{pmatrix}, \quad B_+^{(1,1)} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 1/2 & 1/2 \end{pmatrix}$$

From an honest $P_2$'s perspective, the protocol is secure against sampling attacks since $b_2 = b_1$ and $(B_+^{(\alpha,1)} - B_-^{(\alpha,1)}) \cdot \mathbf{q} = \mathbf{0}_4$, for every $\alpha \in \{0,1\}$. Similarly, the distributions of $(a_1, b_1)$ and $(a_2, b_1)$ are described by the matrices below

$$A_-^{(1,0)} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1/2 & 1/2 & 1/2 \\ 1/2 & 0 & 0 & 0 \end{pmatrix}, \quad A_+^{(1,0)} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$A_-^{(1,1)} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 1/2 & 1/2 \end{pmatrix}, \quad A_+^{(1,1)} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

Hence, from an honest $P_1$'s perspective, the protocol is secure against sampling attacks since $\mathbf{p}^T \cdot (A_+^{(1,\beta)} - A_-^{(1,\beta)}) = \mathbf{0}_4^T$, for every $\beta \in \{0,1\}$. □

Chapter **8**

# The Asymmetric Case

Our analysis of locking strategies and sampling attacks culminates in Theorem 7.14 from the previous chapter. The theorem states that, in order to demonstrate that a given function is computable with full security, it suffices to design a constant-round, passively-secure protocol that is secure against sampling attacks. In this chapter, we look for relevant protocols for asymmetric Boolean functions. We propose an algorithm that takes the function as input, and, depending on the termination step, either returns the description of an appropriate protocol, or, it returns that it failed to do so.

We begin by visiting some mathematical tools and a few useful lemmas. Next, we define a game involving the parties computing $f$ and the dealer. The game simulates the last interaction in a correct protocol computing $f$, and whose purpose is for the dealer to hand a backup[1] output to the disadvantaged party *without* compromising any of the security requirements. Finally, largely as an extension of the game, we obtain an algorithm for designing constant-round protocols that are passively secure and secure against sampling attacks. Using the tools and the lemmas from Section 8.1, we demonstrate that our algorithm satisfies correctness.

**Speculative Remark.** For what it is worth, numerical results on small cases indicate that our algorithm accounts for the overwhelmingly majority of non semi-balanced functions. We also encountered a handful of non semi-balanced functions for which our algorithm fails to come up with a suitable protocol. These functions are noteworthy because we suspect that

[1]Other than the actual output.

their unknown status cannot be attributed to potential shortcomings of our algorithm. We believe that our algorithm is as good at finding suitable protocols as can be expected. In support of this point, towards the end of the chapter, we show that one of these functions seems to be computable with fairness, but only at the expense of privacy.

## 8.1 Irreducible Locking Strategies

Let $f : X \times Y \to \{0,1\}^2$ denote some Boolean asymmetric (possibly randomized) finite function. Since $f$ is asymmetric, it has four associated matrices $M^{(0,0)}$, $M^{(0,1)}$, $M^{(1,0)}$, $M^{(1,1)} \in [0,1]^{\ell \times k}$. Recall that locking strategies for $P_1$ and $P_2$ correspond to elements of the following vector spaces.

$$\begin{aligned}\langle \mathbf{L}_1 \rangle &= \left\{ \mathbf{x} \in \mathbb{R}^\ell \,\middle|\, \exists \delta_1 \in \mathbb{R} : \mathbf{x}^T M^{(1,*)} = \delta_1 \cdot \mathbf{1}_k^T \right\} \ , \\ \langle \mathbf{L}_2 \rangle &= \left\{ \mathbf{y} \in \mathbb{R}^k \,\middle|\, \exists \delta_2 \in \mathbb{R} : M^{(*,1)} \mathbf{y} = \delta_2 \cdot \mathbf{1}_\ell \right\} \ ,\end{aligned}$$

where $\mathbf{L}_1$ and $\mathbf{L}_2$ denote arbitrary bases of each space. Without loss of generality, assume $|\mathbf{L}_1| = s_1$ and $|\mathbf{L}_2| = s_2$. Locking strategies endow a matrix with a matroid structure, in the same way that linear dependence does. We define the matroid by means of its *minimally dependent sets*, i.e. circuits.

**Definition 8.1.** We say that the columns of $M^{(*,1)}$ indexed by $Y' \subseteq Y$ are *minimally dependent* if

- $\left\{ M^{(*,1)} \mathbf{e}_y \right\}_{y \in Y'} \cup \{ \mathbf{1}_\ell \}$ are linearly dependent,
- for every $y_0 \in Y'$, it holds that $\left\{ M^{(*,1)} \mathbf{e}_y \right\}_{y \in Y' \setminus \{y_0\}} \cup \{ \mathbf{1}_\ell \}$ are linearly independent.

Similarly, we say that the rows of $M^{(1,*)}$ indexed by $X' \subseteq X$ are *minimally dependent* if

- $\left\{ \mathbf{e}_x^T M^{(1,*)} \right\}_{x \in X'} \cup \left\{ \mathbf{1}_k^T \right\}$ are linearly dependent,
- for every $x_0 \in X'$, it holds that $\left\{ \mathbf{e}_x^T M^{(1,*)} \right\}_{x \in X' \setminus \{x_0\}} \cup \left\{ \mathbf{1}_k^T \right\}$ are linearly independent.

**Proposition 8.2.** *Suppose that the columns of $M^{(*,1)}$ indexed by $Y' \subseteq Y$ are minimally dependent. Up to a multiplicative factor, there exists a unique $\mathbf{q} \in \mathbb{R}^k \setminus \{\mathbf{0}_k\}$ such that $M^{(*,1)}\mathbf{q} \in \langle \mathbf{1}_\ell \rangle$ and* $\operatorname{supp}(\mathbf{q}) = Y'$.

*Proof.* By definition, there exists $\mathbf{q} \in \mathbb{R}^k$ such that $M^{(*,1)}\mathbf{q} \in \langle \mathbf{1}_\ell \rangle$ and $\operatorname{supp}(\mathbf{q}) = Y'$. The non-trivial task is to show that this vector is unique, *up* to a multiplicative factor. Suppose there exists $\mathbf{q}'$ such that $\operatorname{supp}(\mathbf{q}') \subseteq Y'$ and $M^{(*,1)}\mathbf{q}' \in \langle \mathbf{1}_\ell \rangle$. In pursuit of a contradiction, assume that $\mathbf{q}' \neq \lambda \mathbf{q}$, for every $\lambda \in \mathbb{R}$. Equivalently, there exists $i$, $j \in Y'$ such that $\mathbf{q}(i) = \lambda_i \mathbf{q}'(i)$ and $\mathbf{q}(j) = \lambda_j \mathbf{q}'(j)$, with $\lambda_i \neq \lambda_j$. Without loss of generality, say that $\lambda_i \neq 0$ and define $\mathbf{q}'' = \lambda_i \cdot \mathbf{q}' - \mathbf{q}$. Deduce that $M^{(*,1)}\mathbf{q}'' \in \langle \mathbf{1}_\ell \rangle$ and $\operatorname{supp}(\mathbf{q}'') \subsetneq Y'$, in contradiction with the fact that the columns indexed by $Y'$ are minimally dependent. □

**Definition 8.3.** If $\mathbf{q} \in \mathbb{R}^k$ is as in Proposition 8.2, we say that $\mathbf{q}$ is *irreducible.*

**Proposition 8.4.** *There exists a basis of $\langle \mathbf{L}_2 \rangle$ consisting of irreducible strategies.*

*Proof.* It is a well known result from Linear Algebra that any generating set contains a basis. Thus, we prove the Claim by showing that irreducible locking strategies form a generating set. Let $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$ and consider $\operatorname{supp}(\mathbf{y})$. Let $\mu_1, \dots, \mu_{t_\mathbf{y}}$ denote all the subsets of $\operatorname{supp}(\mathbf{y})$ that index minimally dependent columns, and write $\mathbf{q}_1, \dots, \mathbf{q}_{t_\mathbf{y}}$ for the associated unique *irreducible* locking strategies. We show that $\mathbf{y} \in \langle \mathbf{q}_1, \dots, \mathbf{q}_{t_\mathbf{y}} \rangle$ by constructing a sequence of locking strategies $\mathbf{y}_0, \dots, \mathbf{y}_{s_\mathbf{y}}$ such that

$$\begin{cases} \mathbf{y}_0 = \mathbf{y} \\ \mathbf{y}_{j+1} = \mathbf{y}_j - \alpha_j \cdot \mathbf{q}^{(j)} \\ \mathbf{y}_{s_\mathbf{y}} = \mathbf{0}_\ell \end{cases},$$

where $\alpha_j \in \mathbb{R}$ and $\mathbf{q}^{(j)} \in \{\mathbf{q}_1, \dots, \mathbf{q}_{t_\mathbf{y}}\}$. Let $\mathbf{q}^{(0)}$ be an arbitrary element of $\{\mathbf{q}_1, \dots, \mathbf{q}_{t_\mathbf{y}}\}$ and fix $j_0$ such that $\mathbf{q}^{(0)}(j_0) \neq 0$. Define $\mathbf{y}_1 = \mathbf{y} - \frac{\mathbf{y}(j_0)}{\mathbf{q}^{(0)}(j_0)} \cdot \mathbf{q}^{(0)}$. Notice that $\mathbf{y}_1$ is a locking strategy and that $\operatorname{supp}(\mathbf{y}_1) \subsetneq \operatorname{supp}(\mathbf{y})$. Since $\mathbf{y}_1$ is a locking strategy, it follows that $\mu^{(1)} \subset \operatorname{supp}(y_1)$, for some $\mu^{(1)} \in \{\mu_1, \dots, \mu_{t_\mathbf{y}}\}$. Write $\mathbf{q}^{(1)}$ for the associated locking strategy. Similarly to what we just did, fix $j_1$ such that $\mathbf{q}^{(1)}(j_1) \neq 0$, define $\mathbf{y}_2 = \mathbf{y}_1 - \frac{\mathbf{y}_1(j_1)}{\mathbf{q}^{(1)}(j_1)} \cdot \mathbf{q}^{(1)}$, and notice that $\mathbf{y}_2$ is a locking strategy and that $\operatorname{supp}(\mathbf{y}_2) \subsetneq \operatorname{supp}(\mathbf{y}_1)$.

Repeat the procedure and conclude that it terminates in at most $|\mathrm{supp}(\mathbf{y})|$ steps. $\square$

Define $Y_0, \ldots, Y_{k'}$ to be a partitioning of the input domain $Y$ that we construct as follows. First, $y \in Y_0$ if $\mathbf{e}_y \perp \langle \mathbf{L}_2 \rangle$. Next, for $i \geq 1$, let $\mathbf{q}^{(i)}$ be an irreducible locking strategy such that $\mathrm{supp}(\mathbf{q}^{(i)}) \cap (Y_{i-1} \cup \ldots, \cup Y_0) = \emptyset$.

- $y \in Y_i$ if $\exists$ irreducibles $\mathbf{q}_1^{(i)}, \ldots, \mathbf{q}_{t_y}^{(i)}$ such that

$$\begin{cases} \mathbf{q}^{(i)} = \mathbf{q}_1^{(i)} \\ \mathrm{supp}(\mathbf{q}_j^{(i)}) \cap \mathrm{supp}(\mathbf{q}_{j+1}^{(i)}) \neq \emptyset \\ y \in \mathrm{supp}(\mathbf{q}_{t_y}^{(i)}) \end{cases} .$$

## 8.2 The Dealer Game

In this section, we present a game involving the parties computing $f$ and the dealer. The purpose of the game is to define a simplified variant of the security against sampling attacks requirement.

Assume that the honest party applies some locking strategy $\mathbf{y}$ while executing a protocol for computing $f$. If the protocol is secure against sampling attacks, then the adversary cannot distinguish between the correct output and the backup output of the honest party. In the worst case, the adversary is handed the output of the corrupted party before the honest party's receives his. In such an event, we ask what the honest party's backup output ought to be, other than the correct output.

Write $a_i$ (resp. $b_i$) for $P_1$'s (resp. $P_2$'s) backup output at round $i$. Let $\widehat{b}_*$ denote the bit obtained from $b_*$ by applying[2] $\mathbf{y}$, and let $r$ denotes the number of rounds. From an honest $P_2$'s perspective, we require that the pairs $(a_i, \widehat{b}_{i-1})$ and $(a_i, \widehat{b}_r)$ are statistically close, for every $x \in X$, $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$ and $i \in \{1 \ldots r\}$.

Consider the following process involving a dealer. The dealer receives inputs $x$ and $y$ from $P_1$ and $P_2$, respectively, and computes $f(x,y) = (f_1(x,y), f_2(x,y))$. Then, the dealer hands $f_1(x,y)$ to $P_1$ and a bit $b$ to $P_2$, where $b$ is a probabilistic function of $P_2$'s input and $f_2(x,y)$. We investigate how to construct $b$ with the following goals in mind.

[2]Recall that $\mathbf{y}$ encodes an input distribution but also a certain transformation.

1. *minimize* the information $b$ contains about $f_2(x, y)$
2. $(f_1, \widehat{f_2})$ is statistically close to $(f_1, \widehat{b})$, for every $x \in X$ and $\mathbf{q} \in \langle \mathbf{L}_2 \rangle$.

Let us introduce vectors $\mathbf{b}^{(0)}$, $\mathbf{b}^{(1)} \in \mathbb{R}^k$ such that

$$\mathbf{b}^{(\beta)}(y_0) = \Pr\left[b = 1 \,\middle|\, f_2(x, y) = \beta \wedge y = y_0\right] \ .$$

Fix $y \in Y$, and notice that $b \equiv f_2$ on input $y$ if $\mathbf{b}^{(0)}(y) = 0$ and $\mathbf{b}^{(1)}(y) = 1$. On the other hand, $b$ contains no information about $f_2(x, y)$ if and only if $\mathbf{b}^{(0)}(y) = \mathbf{b}^{(1)}(y)$. Consequently, our aim is for $\mathbf{b}^{(0)}$ and $\mathbf{b}^{(1)}$ to be equal on as many indices as possible. See Figure 8.1 for a concise description of the process.

**Toward Security against Sampling Attacks**

- **Offline**: The dealer chooses vectors $\mathbf{b}^{(0)}, \mathbf{b}^{(1)} \in [0, 1]^k$.
- $P_1$ chooses $x \in X$, $P_2$ samples $y$ according to $\mathbf{y}$, where $\mathbf{y}$ is a locking strategy of $P_2$'s own choosing. Parties hand their inputs to the dealer.
- The dealer computes $f(x, y) = (\alpha, \beta) \in \{0, 1\}^2$ and hands the following bits to the parties
  1. $P_1$ receives $\alpha$.
  2. $P_2$ receives $b$ such that $\Pr[b = 1] = \mathbf{b}^{(\beta)}(y)$.
- **Outputs**: $P_1$ outputs $\alpha$, $P_2$ outputs $\widehat{b}$ which denotes the bit obtained from $b$ by applying the transformation that is implicit in $\mathbf{y}$.
- **Aim of the Game**:
  - Maximize the number of $y$'s such that $\mathbf{b}^{(0)}(y) = \mathbf{b}^{(1)}(y)$,
  - $(f_1, \widehat{f_2}) \overset{s}{\equiv} (f_1, \widehat{b})$, for every $x \in X$ and $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$.

Figure 8.1: The Dealer Game for Security against Sampling Attacks.

**Claim 8.5.** *The purpose of the game from Figure 8.1 is to construct* $\mathbf{b}^{(0)}, \mathbf{b}^{(1)} \in [0, 1]^k$ *such that* $\mathbf{b}^{(0)}$ *is equal to* $\mathbf{b}^{(1)}$ *on as many indices as possible such that*

$$\begin{cases} M^{(0,0)}\left(\mathbf{b}^{(0)} * \mathbf{y}\right) + M^{(0,1)}\left(\mathbf{b}^{(1)} * \mathbf{y}\right) = M^{(0,1)}\mathbf{y} \\ M^{(1,0)}\left(\mathbf{b}^{(0)} * \mathbf{y}\right) + M^{(1,1)}\left(\mathbf{b}^{(1)} * \mathbf{y}\right) = M^{(1,1)}\mathbf{y} \end{cases} , \tag{8.1}$$

*for every* $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$.

*Proof.* The first part of the claim follows from the discussion above. For the second part of the claim, fix $x \in X$, $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$, $\alpha \in \{0,1\}$, and note that

$$\begin{aligned}\Pr\left[(f_1, \widehat{f_2}) = (\alpha, 1)\right] &= \mathbf{e}_x^T \left( \sum_{\mathbf{y}(y) \geq 0} \left[M^{(\alpha,1)}\right]_{*,y} \mathbf{y}(y) + \sum_{\mathbf{y}(y)<0} \left[M^{(\alpha,0)}\right]_{*,y} |\mathbf{y}(y)| \right) \\ &= \mathbf{e}_x^T \left( M^{(\alpha,1)}\mathbf{y} + \sum_{\mathbf{y}(y)<0} \left[M^{(\alpha,*)}\right]_{*,y} |\mathbf{y}(y)| \right)\end{aligned}$$

On the other hand, $\Pr\left[(f_1, \widehat{b}) = (\alpha, 1)\right] =$

$$\sum_{\mathbf{y}(y)\geq 0} \mathbf{e}_x^T \left( \left[M^{(\alpha,1)}\right]_{*,y} \cdot \mathbf{b}^{(1)}(y) + \left[M^{(\alpha,0)}\right]_{*,y} \cdot \mathbf{b}^{(0)}(y) \right) \mathbf{y}(y) +$$
$$\sum_{\mathbf{y}(y)<0} \left( \left[M^{(\alpha,1)}\right]_{*,y} \cdot (1 - \mathbf{b}^{(1)}(y)) + \left[M^{(\alpha,0)}\right]_{*,y} \cdot (1 - \mathbf{b}^{(0)}(y)) \right) |\mathbf{y}(y)| \ ,$$

and thus $\Pr\left[(f_1, \widehat{b}) = (\alpha, 1)\right] =$

$$\mathbf{e}_x^T \left( M^{(\alpha,0)} \left(\mathbf{b}^{(0)} * \mathbf{y}\right) + M^{(\alpha,1)} \left(\mathbf{b}^{(1)} * \mathbf{y}\right) + \sum_{\mathbf{y}(y)<0} \left[M^{(\alpha,*)}\right]_{*,y} |\mathbf{y}(y)| \right) \ .$$

To conclude, note that since $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}$ are fixed vectors, it holds that $(f_1, \widehat{f_2})$ and $(f_1, \widehat{b})$ are statistically close if and only if they are *identically distributed.* □

**From the Statistical to the Perfect Realm.** We could have defined vectors $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}$ with respect to a security parameter. However, this seemingly more general approach is not helpful. Vectors $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}$ can be viewed as solutions to linear systems satisfying inequality constraints. As such, the lemma below allows us to disregard the security parameter.

**Lemma 8.6.** *Let $M \in \mathbb{R}^{d \times d'}$ be some arbitrary matrix and let $\mathbf{c}$ be a fixed vector in $\mathbb{R}^d$. Further assume that there exist $\{\mathbf{z}_n\}_{n \in N}$ such that $\mathbf{z}_n \in [0,1]^{d'}$ and $M\mathbf{z}_n$ is close to $\mathbf{c}$. It holds that there exists $\mathbf{z} \in [0,1]^{d'}$ such that $M\mathbf{z} = \mathbf{c}$.*

*Proof.* The existence of $\mathbf{z}$ follows from the fact that the image of $[0,1]^d$ by $M$ is a closed set (in the topological sense). □

Another seemingly more general approach involves instructing the dealer to hand $\{a_n\}_{n\in\mathbb{N}}$ to $P_1$ such that $a_n$ is close to $f_1$. Again, there is no gain in pursuing such an approach. Consider families of matrices $\{A_n^{(0)}\}_{n\in\mathbb{N}}$ and $\{A_n^{(0)}\}_{n\in\mathbb{N}}$ such that $A_n^{(\alpha)}$ is close to $\mathbf{0}_{\ell\times k}$, for every $\alpha$. By the Cauchy-Schwartz inequality, it holds that $A_n^{(0)}\cdot\mathbf{b}_n^{(0)}+A_n^{(1)}\cdot\mathbf{b}_n^{(1)}$ is close to $\mathbf{0}_\ell$, for every pair of families $\{\mathbf{b}_n^{(0)}\}_{n\in\mathbb{N}}$ and $\{\mathbf{b}_n^{(1)}\}_{n\in\mathbb{N}}$ such that $\mathbf{b}_n^{(\beta)}\in[0,1]^k$. Thus, it must hold that

$$\begin{cases}\left\|M^{(0,0)}\left(\mathbf{b}_n^{(0)}*\mathbf{y}\right)+M^{(0,1)}\left(\mathbf{b}_n^{(1)}*\mathbf{y}\right)-M^{(0,1)}\mathbf{y}\right\|\leq\mathsf{negl}(n)\\ \left\|M^{(1,0)}\left(\mathbf{b}_n^{(0)}*\mathbf{y}\right)+M^{(1,1)}\left(\mathbf{b}_n^{(1)}*\mathbf{y}\right)-M^{(1,1)}\mathbf{y}\right\|\leq\mathsf{negl}(n)\end{cases},$$

and we fall back on the previous case.

Moving on, fix $Y_i\in\{Y_0,\ldots,Y_{k'}\}$ and suppose there exist $\mathbf{b}^{(0)},\mathbf{b}^{(1)}$ satisfying Equation (8.1) such that $\mathbf{b}^{(0)}(y_0)\neq 0$ or $\mathbf{b}^{(1)}(y_0)\neq 1$, for some $y_0\in Y_i$. We show that there exist $\mathbf{b}'^{(0)},\mathbf{b}'^{(1)}$ satisfying Equation (8.1) such that $\mathbf{b}'^{(0)}(y)=\mathbf{b}'^{(1)}(y)$, for every $y\in Y_i$. This is where the underlying matroid structure will come in handy.

**Proposition 8.7.** *It holds that* $\mathbf{b}^{(1)}(y)-\mathbf{b}^{(0)}(y)=\mathbf{b}^{(1)}(y_0)-\mathbf{b}^{(0)}(y_0)$, *for every* $y\in Y_i$. *In addition, vectors* $\mathbf{b}'^{(1)}$, $\mathbf{b}'^{(0)}$ *satisfy Equation* (8.1), *where*

$$\mathbf{b}'^{(b)}(y)=\begin{cases}\mathbf{b}^{(b)}(y) & \textit{if } y\notin Y_j\\ \dfrac{\mathbf{b}^{(0)}(y)}{\mathbf{b}^{(0)}(y_0)-\mathbf{b}^{(1)}(y_0)+1} & \textit{if } y\in Y_j\end{cases}.$$

*Proof.* For the first part of the claim, we make use of Proposition 8.2. The case $i=0$ is left to the reader. Let $i\geq 1$ and fix irreducible $\mathbf{q}$ such that $y_0\in\operatorname{supp}(\mathbf{q})$. We know that, for any $\mathbf{y}\in\langle\mathbf{L}_2\rangle$,

$$M^{(0,0)}\left(\mathbf{b}^{(0)}*\mathbf{y}\right)+M^{(0,1)}\left(\mathbf{b}^{(1)}*\mathbf{y}\right)=M^{(0,1)}\mathbf{y}\ , \tag{8.2}$$

$$M^{(1,0)}\left(\mathbf{b}^{(0)}*\mathbf{y}\right)+M^{(1,1)}\left(\mathbf{b}^{(1)}*\mathbf{y}\right)=M^{(1,1)}\mathbf{y}\ . \tag{8.3}$$

Let $\mathbf{y}=\mathbf{q}$ and add the two expressions.

$$\left(\mathbf{1}_{\ell\times k}-M^{(*,1)}\right)\left(\mathbf{b}^{(0)}*\mathbf{q}\right)+M^{(*,1)}\left(\mathbf{b}^{(1)}*\mathbf{q}\right)=M^{(*,1)}\mathbf{q}\ .$$

By moving a few terms around, deduce that $M^{(*,1)}\left((\mathbf{b}^{(1)}-\mathbf{b}^{(0)})*\mathbf{q}\right)\in\langle\mathbf{1}_\ell\rangle$. Consequently, by Proposition 8.2, $\mathbf{b}^{(1)}(y)-\mathbf{b}^{(0)}(y)=\mathbf{b}^{(1)}(y_0)-$

$\mathbf{b}^{(0)}(y_0)$, for every $y \in \operatorname{supp}(\mathbf{q})$. Moving on, fix an arbitrary $y \in Y_i$. We know there exists a sequence of irreducibles $\mathbf{q}_1^{(i)} \dots \mathbf{q}_{t'_y}^{(i)}$ such that

$$\begin{cases} \mathbf{q} = \mathbf{q}_1^{(i)} \\ \operatorname{supp}(\mathbf{q}_j^{(i)}) \cap \operatorname{supp}(\mathbf{q}_{j+1}^{(i)}) \neq \emptyset \\ y \in \operatorname{supp}(\mathbf{q}_{t'_y}^{(i)}) \end{cases} ,$$

Apply the same argument as above and, by induction, deduce that $\mathbf{b}^{(1)}(y) - \mathbf{b}^{(0)}(y) = \mathbf{b}^{(1)}(y_0) - \mathbf{b}^{(0)}(y_0)$. For the second part of the claim, we rely on the following observations.

- Vectors $\mathbf{b}_0^{(0)}$ and $\mathbf{b}_0^{(1)}$ satisfy equations (8.2) and (8.3), where

$$\mathbf{b}_0^{(0)} = \begin{cases} \mathbf{b}^{(0)}(y) & \text{if } y \notin Y_i \\ 0 & \text{if } y \in Y_i \end{cases}, \qquad \mathbf{b}_0^{(1)} = \begin{cases} \mathbf{b}^{(1)}(y) & \text{if } y \notin Y_i \\ 1 & \text{if } y \in Y_i \end{cases} .$$

- Solutions to Equations (8.2) and (8.3) can be combined linearly.

The second item is trivial. For the first item, we show that vectors $\mathbf{b}_0^{(0)}$ and $\mathbf{b}_0^{(1)}$ are solutions to the equations for a particular basis of $\langle \mathbf{L}_2 \rangle$. By Proposition 8.4, consider a basis of $\langle \mathbf{L}_2 \rangle$ that consists of irreducible strategies. Conclude by observing that $Y_i \cap \operatorname{supp}(\mathbf{q}') = \emptyset$, for every irreducible $\mathbf{q}'$ such that $\operatorname{supp}(\mathbf{q}) \nsubseteq Y_i$. Next, define

$$\mathbf{b}'^{(b)} = \frac{1}{\mathbf{b}^{(0)}(y_0) - \mathbf{b}^{(1)}(y_0) + 1} \cdot \mathbf{b}^{(b)} + \left(1 - \frac{1}{\mathbf{b}^{(0)}(y_0) - \mathbf{b}^{(1)}(y_0) + 1}\right) \cdot \mathbf{b}_0^{(b)} .$$

We note that $\mathbf{b}'^{(0)}$, $\mathbf{b}'^{(1)}$ admit the right expression. It remains to show that $\mathbf{b}'^{(b)}(y) \in [0,1]$, for every $y$. Since $\mathbf{b}'^{(b)}(y) = \mathbf{b}^{(b)}(y)$ if $y \notin Y_j$, it suffices to show that

$$\frac{\mathbf{b}^{(0)}(y)}{\mathbf{b}^{(0)}(y) - \mathbf{b}^{(1)}(y) + 1} \in [0,1] , \tag{8.4}$$

for $y \in Y_i$. We conclude by observing that (8.4) is equivalent to $0 \leq \mathbf{b}^{(0)}(y)$ and $\mathbf{b}^{(1)}(y) \leq 1$. □

## 8.3 Designing Fully Secure Protocols

In this section, we show how to construct passively-secure protocols that are also secure against sampling attacks. The idea is to build the backup outputs from the bottom-up, i.e. start with $a_r \equiv f_1$ and $b_r \equiv f_2$, and construct $a_{r-1}$ and $b_{r-1}$ such that $a_{r-1}$ (resp. $b_{r-1}$) only depends on $x$ and $f_1(x,y)$ (resp. $y$ and $f_2(x,y)$) without compromising security against sampling attacks.

To this end, we employ a simple minimization algorithm in combination with Proposition 8.7. Without loss of generality, we begin by assuming that $P_1$ is corrupted, and that he observes $a_r \equiv f_1(x,y)$. To define $b_{r-1}$, we run an optimization algorithm that constructs vectors $\{\mathbf{b}^{(\beta)}\}_{\beta\in\{0,1\}}$, and we delete any input $y \in Y$ for which $\mathbf{b}^{(1)}(y) - \mathbf{b}^{(0)}(y) \neq 1$. Then, in order to define $a_{r-1}$, we run an optimization algorithm that constructs vectors $\{\mathbf{a}^{(\alpha)}\}_{\alpha\in\{0,1\}}$, assuming $P_2$ is corrupted, and the party is privy to the output *only if the input he used was not deleted in the previous step.* We proceed by deleting any input $x \in X$ for which $\mathbf{a}^{(1)}(y) - \mathbf{a}^{(0)}(y) \neq 1$. We carry on in this fashion until one party runs out of inputs, *or* the process does not allow for any further deletions.

**Remark 8.8.** Getting ahead of ourselves, we note that deleted inputs cannot be used by the adversary to mount a successful sampling attack. In light of Proposition 8.7, if an input was deleted at iteration $i$, then every backup output until round $r - i$ contains no information about the output.

### 8.3.1 Additional Notation

Before we describe the algorithm, let us introduce some notation. For every $\mathbf{q} \in \mathbf{L}_2$ and $X' \subseteq X$, define

$$A_{\mathbf{q}}(X') = \begin{pmatrix} \left[M^{(0,0)} * Q\right]_{X'} & \left[M^{(0,1)}_{X'} * Q\right]_{X'} \\ \left[M^{(1,0)} * Q\right]_{X'} & \left[M^{(1,1)} * Q\right]_{X'} \\ M^{(*,0)} * Q & M^{(*,1)} * Q \end{pmatrix}, \qquad \vec{b}_{\mathbf{q}} = \begin{pmatrix} \left[M^{(0,1)}\right]_{X'} \mathbf{q} \\ \left[M^{(1,1)}\right]_{X'} \mathbf{q} \\ M^{(*,1)}\mathbf{q} \end{pmatrix}$$

where $Q = \mathbf{1}_\ell \cdot \mathbf{q}^T$ and the notation $[\ \cdot\ ]_{X'}$ indicates that only the rows indexed by $X' \subseteq X$ appear. Write $\mathbf{L}_2 = \{\mathbf{q}_1, \ldots, \mathbf{q}_{s_2}\}$ and consider the following linear system for unknown $(\mathbf{b}^{(0)T}, \mathbf{b}^{(1)T})$.

$$\begin{pmatrix} A_{\mathbf{q}_1}(X') \\ A_{\mathbf{q}_2}(X') \\ \vdots \\ A_{\mathbf{q}_{s_2}}(X') \end{pmatrix} \cdot \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \end{pmatrix} = \begin{pmatrix} \vec{b}_{\mathbf{q}_1}(X') \\ \vec{b}_{\mathbf{q}_2}(X') \\ \vdots \\ \vec{b}_{\mathbf{q}_{s_2}}(X') \end{pmatrix} \tag{8.5}$$
$$\begin{pmatrix} \mathbf{0}_k \\ \mathbf{0}_k \end{pmatrix} \leq \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \end{pmatrix} \leq \begin{pmatrix} \mathbf{1}_k \\ \mathbf{1}_k \end{pmatrix}$$

**Remark 8.9.** We remark that any solution to Equation (8.5) amounts to the following: *Assuming $P_1$ learns his output if $x \in X'$ and nothing*[3] *if $x \notin X'$, then by handing $P_2$ a backup according to $(\mathbf{b}^{(0)T}, \mathbf{b}^{(1)T})$, security against sampling attacks is preserved.*

Similarly, for every $\mathbf{p} \in \mathbf{L}_1$ and $Y \subseteq Y'$, define

$$B_{\mathbf{p}}(Y') = \begin{pmatrix} \left[M^{(0,0)T} * P\right]_{Y'} & \left[M^{(1,0)T} * P\right]_{Y'} \\ \left[M^{(0,1)T} * P\right]_{Y'} & \left[M^{(1,1)T} * P\right]_{Y'} \\ M^{(0,*)T} * P & M^{(1,*)T} * P \end{pmatrix}, \qquad \vec{a}_{\mathbf{p}} = \begin{pmatrix} \left[M^{(1,0)T}\right]_{Y'} \mathbf{p} \\ \left[M^{(1,1)T}\right]_{Y'} \mathbf{p} \\ M^{(1,*)T} \mathbf{p} \end{pmatrix}$$

where $P = \mathbf{1}_k \cdot \mathbf{p}^T$ and the notation $[\ \cdot\ ]_{Y'}$ indicates that only the rows indexed by $Y' \subseteq Y$ appear. Write $\mathbf{L}_1 = \{\mathbf{p}_1, \ldots, \mathbf{p}_{s_1}\}$ and consider the following linear system for unknown $(\mathbf{a}^{(0)T}, \mathbf{a}^{(1)T})$.

$$\begin{pmatrix} B_{\mathbf{p}_1}(Y') \\ B_{\mathbf{p}_2}(Y') \\ \vdots \\ B_{\mathbf{p}_{s_1}}(Y') \end{pmatrix} \cdot \begin{pmatrix} \mathbf{a}^{(0)} \\ \mathbf{a}^{(1)} \end{pmatrix} = \begin{pmatrix} \vec{a}_{\mathbf{p}_1}(Y') \\ \vec{a}_{\mathbf{p}_2}(Y') \\ \vdots \\ \vec{a}_{\mathbf{p}_{s_1}}(Y') \end{pmatrix} \tag{8.6}$$
$$\begin{pmatrix} \mathbf{0}_\ell \\ \mathbf{0}_\ell \end{pmatrix} \leq \begin{pmatrix} \mathbf{a}^{(0)} \\ \mathbf{a}^{(1)} \end{pmatrix} \leq \begin{pmatrix} \mathbf{1}_\ell \\ \mathbf{1}_\ell \end{pmatrix}$$

**Remark 8.10.** We remark that any solution to Equation (8.6) amounts to the following: *Assuming $P_2$ learns his output if $y \in Y'$ and nothing if $y \notin Y'$, then by handing $P_1$ a backup according to $(\mathbf{a}^{(0)T}, \mathbf{a}^{(1)T})$, security against sampling attack is preserved.*

**Building the Backup Outputs**

1. **Inputs**: $f$, $\mathbf{L}_1$, $\mathbf{L}_2$.
2. Define $X^- = X$ and $Y^- = Y \cup \{k+1\}$.
3. Minimize $-\mathbf{1}_k^T \mathbf{b}^{(0)} + \mathbf{1}_k^T \mathbf{b}^{(1)}$ subject to
$$\begin{pmatrix} A_{\mathbf{q}_1}(X^-) \\ A_{\mathbf{q}_2}(X^-) \\ \vdots \\ A_{\mathbf{q}_{s_2}}(X^-) \end{pmatrix} \cdot \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \end{pmatrix} = \begin{pmatrix} \vec{b}_{\mathbf{q}_1}(X^-) \\ \vec{b}_{\mathbf{q}_2}(X^-) \\ \vdots \\ \vec{b}_{\mathbf{q}_{s_2}}(X^-) \end{pmatrix}$$
$$\begin{pmatrix} \mathbf{0}_k \\ \mathbf{0}_k \end{pmatrix} \le \begin{pmatrix} \mathbf{b}^{(0)} \\ \mathbf{b}^{(1)} \end{pmatrix} \le \begin{pmatrix} \mathbf{1}_k \\ \mathbf{1}_k \end{pmatrix}$$
Define $Y^+$ consisting of all the $y$'s such that $\mathbf{b}^{(1)}(y) - \mathbf{b}^{(0)}(y) = 1$.
4. **If** $Y^+ = Y^-$ *or* $Y^+ = \emptyset$ stop, otherwise set $Y^- \stackrel{\mathsf{def}}{=} Y^+$ and go to Step 5.
5. Minimize $-\mathbf{1}_\ell^T \mathbf{a}^{(0)} + \mathbf{1}_\ell^T \mathbf{a}^{(1)}$ subject to
$$\begin{pmatrix} B_{\mathbf{p}_1}(Y^-) \\ B_{\mathbf{p}_2}(Y^-) \\ \vdots \\ B_{\mathbf{p}_{s_1}}(Y^-) \end{pmatrix} \cdot \begin{pmatrix} \mathbf{a}^{(0)} \\ \mathbf{a}^{(1)} \end{pmatrix} = \begin{pmatrix} \vec{a}_{\mathbf{p}_1}(Y^-) \\ \vec{a}_{\mathbf{p}_2}(Y^-) \\ \vdots \\ \vec{a}_{\mathbf{p}_{s_1}}(Y^-) \end{pmatrix}$$
$$\begin{pmatrix} \mathbf{0}_\ell \\ \mathbf{0}_\ell \end{pmatrix} \le \begin{pmatrix} \mathbf{a}^{(0)} \\ \mathbf{a}^{(1)} \end{pmatrix} \le \begin{pmatrix} \mathbf{1}_\ell \\ \mathbf{1}_\ell \end{pmatrix}$$
Define $X^+$ consisting of all the $x$'s such that $\mathbf{a}^{(1)}(x) - \mathbf{a}^{(0)}(x) = 1$.
6. **If** $X^+ = X^-$ *or* $X^+ = \emptyset$ stop, otherwise set $X^- \stackrel{\mathsf{def}}{=} X^+$ and go to Step 3.
7. **Output:** The transcript of the execution.

Figure 8.2: An Algorithm for Designing Fully-Secure Protocols.

### 8.3.2 The Algorithm

As noted earlier, the idea is to delete inputs from the parties in a sequence of iterations. Namely, we begin by running a linear program that minimizes

[3] Other than his input.

$-\mathbf{1}_k^T\mathbf{b}^{(0)} + \mathbf{1}_k^T\mathbf{b}^{(1)}$ under the constraints of Equation (8.5), with $X' = X$. At this point, we delete any input $y \in Y$ for which $\mathbf{b}^{(1)}(y) - \mathbf{b}^{(0)}(y) < 1$. Write $Y^- \subseteq Y$ for the remaining inputs. We proceed by running a linear program that minimizes $-\mathbf{1}_\ell^T\mathbf{a}^{(0)} + \mathbf{1}_\ell^T\mathbf{a}^{(1)}$ under the constraints of Equation (8.6), with $Y' = Y^-$. Again, we delete any input $x \in X$ for which $\mathbf{a}^{(1)}(x) - \mathbf{a}^{(0)}(x) < 1$. We repeat the procedure until either one of the parties runs out of inputs *or* no further deletions can be made, for either party. See Figure 8.2 for a full description of the algorithm. Next, we discuss the ramifications of the terminating step.

**Running out if Inputs.** Assume that the algorithm terminates because one of the parties ran out of inputs. Without loss of generality, say that $Y^+ = \emptyset$ and write

$$\begin{pmatrix}\mathbf{b}_0^{(0)}\\ \mathbf{b}_0^{(1)}\end{pmatrix}\cdots\begin{pmatrix}\mathbf{b}_t^{(0)}\\ \mathbf{b}_t^{(1)}\end{pmatrix}, \qquad \begin{pmatrix}\mathbf{a}_1^{(0)}\\ \mathbf{a}_1^{(1)}\end{pmatrix}\cdots\begin{pmatrix}\mathbf{a}_t^{(0)}\\ \mathbf{a}_t^{(1)}\end{pmatrix}$$

for the vectors computed in the execution of the algorithm – starting from the bottom-up, i.e. $\mathbf{b}_0^{(0)}, \mathbf{b}_0^{(1)}$ denote the *last* vectors computed for $P_2$ and $\mathbf{b}_t^{(0)}, \mathbf{b}_t^{(1)}$ denote the *first* vectors computed for $P_2$. Similarly, $\mathbf{a}_1^{(0)}, \mathbf{a}_1^{(1)}$ denote the *last* vectors computed for $P_1$ and $\mathbf{a}_t^{(0)}, \mathbf{a}_t^{(1)}$ denote the *first* vectors computed for $P_1$. Now, assume[4] that for every $i \in \{1, \ldots, t\}$, and every $j \in \{1, \ldots, \ell\}$, either $\mathbf{a}_i^{(1)}(j) - \mathbf{a}_i^{(0)}(j) = 1$ or $\mathbf{a}_i^{(1)}(j) = \mathbf{a}_i^{(0)}(j)$. Similarly, for every $i \in \{0, \ldots, t\}$, and every $j \in \{1, \ldots, k\}$, either $\mathbf{b}_i^{(1)}(j) - \mathbf{b}_i^{(0)}(j) = 1$ or $\mathbf{b}_i^{(1)}(j) = \mathbf{b}_i^{(0)}(j)$. Consider the protocol from Figure 8.3.

**Theorem 8.11.** *Protocol* PassSecSamp *is passively secure and secure against sampling attacks.*

*Proof.* Protocol PassSecSamp is passively secure since the parties' backup outputs are randomized functions of each party's input/output pair, respectively. For security against sampling attacks, it suffices to note that

$$\begin{aligned}\mathbf{b}_i^{(1)}(y) \neq \mathbf{b}_i^{(0)}(y) \quad &\Rightarrow \quad \forall j \geq i,\ \mathbf{b}_j^{(1)}(y) - \mathbf{b}_j^{(0)}(y) = 1\\ \mathbf{b}_i^{(1)}(y) - \mathbf{b}_i^{(0)}(y) \neq 1 \quad &\Rightarrow \quad \forall j \leq i,\ \mathbf{b}_j^{(1)}(y) = \mathbf{b}_j^{(0)}(y),\end{aligned}$$

[4]In light of Proposition 8.7, we can construct vectors admitting the required expression.

**Protocol** PassSecSamp

1. The parties $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[a]
2. The dealer computes $f(x, y) = (\alpha, \beta)$ and constructs $(a_0, \ldots, a_{t+1})$ and $(b_0, \ldots, b_{t+1})$ such that
   - $a_0 = f_1(x, \widetilde{y})$, where $\widetilde{y} \in_U Y$.
   - $b_0$ is sampled independently such that $\Pr[b_0 = 1] = \mathbf{b}_0^{(0)}(y) = \mathbf{b}_0^{(1)}(y)$.
   - For every $i \in \{1, \ldots, t\}$, $a_i$ and $b_i$ are bits satisfying
   $$\begin{cases} \Pr[a_i = 1] = \mathbf{a}_i^{(\alpha)}(x) \\ \Pr[b_i = 1] = \mathbf{b}_i^{(\beta)}(y) \end{cases}$$
   - $a_{t+1} = f_1(x, y)$ and $b_{t+1} = f_2(x, y)$.
3. The dealer hands $a_0$ to $P_1$ and $b_0$ to $P_2$.
4. For $i = 1, \ldots, t + 1$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

[a] If $x$ is not in the appropriate domain or $P_1$ does not hand an input, then the dealer sends $f(\hat{x}, y)$ (where $\hat{x}$ is a default value) to $P_2$, which outputs this value and the protocol is terminated. The case of an inappropriate $y$ is dealt analogously.

Figure 8.3: Protocol PassSecSamp for Computing $f$.

and

$$\begin{aligned}\mathbf{a}_i^{(1)}(x) \neq \mathbf{a}_i^{(0)}(x) \quad &\Rightarrow \quad \forall j \geq i,\ \mathbf{a}_j^{(1)}(x) - \mathbf{a}_j^{(0)}(x) = 1 \\ \mathbf{a}_i^{(1)}(x) - \mathbf{a}_i^{(0)}(x) \neq 1 \quad &\Rightarrow \quad \forall j \leq i,\ \mathbf{a}_j^{(1)}(x) = \mathbf{a}_j^{(0)}(x).\end{aligned}$$

Explicitly, the adversary either knows the output or knows nothing about it. In both cases, she will not be able to disrupt the honest party's locking strategy with a sampling attack. In the first case because of how the distributions are constructed, in the second because the only information at her disposal (i.e. the input) says nothing about the honest party's output resulting from the locking strategy. □

As an example, consider the function given by the following matrices.

$$M^{(1,*)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}, \qquad M^{(*,1)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Define

$$\begin{aligned}\mathbf{L}_1 &= \left\{\mathbf{x}_1 = (0,-1,1,0,0)^T, \mathbf{x}_2 = (1,-1,0,-1,1)^T\right\} \quad , \\ \mathbf{L}_2 &= \left\{\mathbf{y}_1 = (-1,0,0,1,0)^T, \mathbf{y}_2 = (0,-1,1,0,1)^T\right\} \quad .\end{aligned}$$

Let us walk through each step of the algorithm.

**1st Step.** The first optimization returns

$$\begin{aligned}\mathbf{b}^{(0)} &= (1,0,0,1,0)^T \\ \mathbf{b}^{(1)} &= (1,1,1,1,1)^T.\end{aligned}$$

We set $Y^+ = \{2,3,5\}$, and we go to the next step.

**2nd Step.** The second optimization returns

$$\begin{aligned}\mathbf{a}^{(0)} &= (1,1,1,1/2,1/2)^T \\ \mathbf{a}^{(1)} &= (1,1,1,1/2,1/2)^T.\end{aligned}$$

Notice that $X^+ = \emptyset$ and the algorithm terminates. Deduce that vectors $\{\mathbf{b}^{(0)}, \mathbf{b}^{(1)}\}$ and $\{\mathbf{a}^{(0)}, \mathbf{a}^{(1)}\}$ can be used to design a 2-round passively secure protocol that is secure against sampling attacks.

## 8.4 Fairness vs Privacy

We turn our attention to functions for which the algorithm returns $Y^+ \neq \emptyset$ and $X^+ \neq \emptyset$. Semi-balanced functions fall under this category. By Cleve [26], protocols that satisfy both correctness *and* security against sampling attacks do not exist in the plain model. We ask if the algorithm terminates in similar fashion for other functions. It turns out that it does. Define deterministic asymmetric function $f = (f_1, f_2)$ given by the matrices

$$M^{(1,*)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \qquad M^{(*,1)} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} .$$

Let $\mathbf{L}_1$, $\mathbf{L}_2$ denote the following bases of the parties' locking strategies.

$$\begin{aligned} \mathbf{L}_1 &= \{\mathbf{x} = (2,1,0,1,1)^T, \mathbf{x}' = (0,0,1,0,0)^T\} \ , \\ \mathbf{L}_2 &= \{\mathbf{y} = (1,1,0,1,1)^T\} \ . \end{aligned}$$

Notice that the function is not semi-balanced since,

$$M^{(*,1)} \cdot \mathbf{y} = 2 \cdot \mathbf{1}_5, \qquad \mathbf{x}_0^T \cdot M^{(1,*)} = \begin{cases} 3 \cdot \mathbf{1}_5^T & \text{if } \mathbf{x}_0 = \mathbf{x} \\ \mathbf{1}_5^T & \text{if } \mathbf{x}_0 = \mathbf{x}' \end{cases},$$

and

$$\mathbf{x}_0^T \cdot M^{(1,1)} \cdot \mathbf{y} = \begin{cases} 6 & \text{if } \mathbf{x}_0 = \mathbf{x} \\ 2 & \text{if } \mathbf{x}_0 = \mathbf{x}' \end{cases}.$$

For this function, the algorithm returns $X^+ = X^- = \{x_1, x_2, x_4, x_5\}$ and $Y^+ = Y^- = \{y_1, y_2, y_4, y_5\}$. Consequently, for any protocol $\Pi$ in the underlying communication model, the following must be true.

$$\begin{cases} \Pr\left[a_i = f_1(x_j, y) \,\middle|\, j \in \{1,2,4,5\}\right] = 1 - \mathsf{negl}(n) \\ \Pr\left[b_i = f_2(x, y_j) \,\middle|\, j \in \{1,2,4,5\}\right] = 1 - \mathsf{negl}(n) \end{cases} \implies$$
$$\begin{cases} \Pr\left[a_{i-1} = f_1(x_j, y) \,\middle|\, j \in \{1,2,4,5\}\right] = 1 - \mathsf{negl}(n) \\ \Pr\left[b_{i-1} = f_2(x, y_j) \,\middle|\, j \in \{1,2,4,5\}\right] = 1 - \mathsf{negl}(n) \end{cases},$$

where $(a_i, b_i)$ and $(a_{i-1}, b_{i-1})$ denote the parties' backup outputs at rounds $i$ and $i-1$ respectively. Since the number of rounds is polynomial in the security parameter, and the protocol is assumed to be correct, we conclude that no such protocol exists.

**Remark 8.12.** The key phrase in the discussion above is “the underlying communication model”. We assumed that the dealer hands backups based solely on the input-output pair of each party. While this assumption seems extremely sensible (after all, this is how the simulator generates the adversary’s view in the ideal model) we do not know whether it incurs a prohibitive loss of generality. Nevertheless, we suspect the conjecture below to hold true.

**Conjecture 8.13.** *If $f$ is computable with full security, then the algorithm from Figure 8.2 terminates with either one of the parties running out of inputs.*

Throughout our work, we have used “fairness” and “full-security” interchangeably. We take a moment to argue that this equivalence is perhaps misplaced. Specifically, it does not follow that fairness should imply privacy. For one, consider any of the fully secure protocols we encountered, and instruct the parties to broadcast their inputs at the end of the execution. The resulting protocols are still fair but obviously not private.

Furthermore, it is not unreasonable to expect that, for some functions, fairness can only be achieved *at the expense* of privacy. We believe this to be the case for the function above, even though a proof escapes us at this time. As evidence for our belief, we present a protocol for this function that satisfies all the requirements of full security *except* privacy. We refer the reader to Appendix E p. 164 for the definition of full security without privacy. The definition follows from a new ideal model, dubbed the *leaky model*, which is identical to the fully-secure model in every respect except that the trusted party leaks the honest party’s output to the adversary.

In the spirit of Chapter 7, we begin with a constant-round protocol that is secure against sampling attacks. In contrast with Chapter 7, we do not require the protocol to be passively secure. Then, we introduce a threshold round, and we argue that the resulting protocol is fair but not private. Consider protocol $\Pi^{\heartsuit}$ from Figure 8.4.

**Claim 8.14.** *Protocol $\Pi^{\heartsuit}$ is secure against sampling attacks.*

*Proof.* We show that $(a_1, b_0)$ and $(a_1, b_1)$ are identically distributed for every $x \in X$ and $\mathbf{q} \in \mathbf{L}_2$ and that $(a_1, b_1)$ and $(a_2, b_1)$ are identically distributed for every $\mathbf{p} \in \mathbf{L}_1$ and $y \in Y$. So, assuming $P_2$ plays according to $\mathbf{y}$, the relevant probabilities are given by the table below.

**Protocol $\Pi^{\heartsuit}$**

1. The parties $P_1$ and $P_2$ hand their inputs, denoted $x$ and $y$ respectively, to the dealer.[a]
2. The dealer computes $f(x,y) = (\alpha, \beta)$ and constructs $(a_0, a_1, a_2)$ and $(b_0, b_1, b_2)$ such that
   - $a_0 = f_1(x, \widetilde{y})$, where $\widetilde{y} \in_U Y$.
   - $b_0 \in_U \{0,1\}$ .
   - $a_1$ is bit satisfying

$$\Pr[a_1 = 1] = \begin{cases} 0 & \text{if } x = x_5 \lor \big(x = x_4 \land y \in \{y_2, y_4\}\big) \\ 1/2 & \text{if } x = x_1 \land y = y_1 \\ 1 & \text{otherwise} \end{cases} .$$

   - $b_1 = \beta$ .
   - $a_2 = \alpha$ and $b_2 = \beta$.
3. The dealer hands $a_0$ to $P_1$ and $b_0$ to $P_2$.
4. For $i = 1, 2$,
   a) The dealer gives $a_i$ to $P_1$. If $P_1$ aborts, then $P_2$ outputs $b_{i-1}$ and halts.
   b) The dealer gives $b_i$ to $P_2$. If $P_2$ aborts, then $P_1$ outputs $a_i$ and halts.

[a] If $x$ is not in the appropriate domain or $P_1$ does not hand an input, then the dealer sends $f(\hat{x}, y)$ (where $\hat{x}$ is a default value) to $P_2$, which outputs this value and the protocol is terminated. The case of an inappropriate $y$ is dealt analogously.

Figure 8.4: Protocol $\Pi^{\heartsuit}$ for Computing $f$.

| $x \setminus (a_1, b_*)$ | $(0,0)$ | $(0,1)$ | $(1,0)$ | $(1,1)$ |
|---|---|---|---|---|
| $x_1$ | $1/8$ | $1/8$ | $3/8$ | $3/8$ |
| $x_2$ | $0$ | $0$ | $1/2$ | $1/2$ |
| $x_3$ | $0$ | $0$ | $1/2$ | $1/2$ |
| $x_4$ | $1/4$ | $1/4$ | $1/4$ | $1/4$ |
| $x_5$ | $1/2$ | $1/2$ | $0$ | $0$ |

Next, we turn to an honest $P_1$ playing according to $\mathbf{x}$.

| $y \setminus (a_*, b_1)$ | $(0,0)$ | $(0,1)$ | $(1,0)$ | $(1,1)$ |
|---|---|---|---|---|
| $y_1$ | $1/5$ | $1/5$ | $1/5$ | $2/5$ |
| $y_2$ | $1/5$ | $1/5$ | $1/5$ | $2/5$ |
| $y_3$ | $2/5$ | $0$ | $2/5$ | $1/5$ |
| $y_4$ | $0$ | $2/5$ | $3/5$ | $0$ |
| $y_5$ | $2/5$ | $0$ | $1/5$ | $2/5$ |

□

Using the transformation from the previous chapter, consider SecSamp2Fair($\Pi^{\heartsuit}$). Note that $i^* \geq 3$ is chosen with parameter $\gamma$, and that backup sequences $(a_0, \dots, a_r)$ and $(b_0, \dots, b_r)$ are computed as follows.

- **Party** $P_1$.
  - $\Pr[a_i = 1] = \Pr[f_1(x, \widetilde{y}) = 1]$ where $\widetilde{y} \in_U Y$ if $i < i^* - 1$,
  - $\Pr[a_{i^*-1} = 1] = \begin{cases} 0 & \text{if } x = x_5 \vee \big(x = x_4 \wedge y \in \{y_2, y_4\}\big) \\ 1/2 & \text{if } x = x_1 \wedge y = y_1 \\ 1 & \text{otherwise} \end{cases}$ .
  - $\Pr[a_i = 1] = f_1(x, y)$ if $i \geq i^*$.
- **Party** $P_2$.
  - $\Pr[b_i = 1] = \Pr[f_2(\widetilde{x}, y) = 1]$ where $\widetilde{x} \in_U X$ if $i < i^* - 2$,
  - $\Pr[b_i = 1] = 1/2$ if $i = i^* - 2$,
  - $\Pr[b_i = 1] = f_2(x, y)$ if $i \geq i^* - 1$.

From a corrupt $P_2$'s perspective, we remark protocol $\Pi^{\heartsuit}$ is passively secure and secure against sampling attacks. It follows that SecSamp2Fair($\Pi^{\heartsuit}$) is fully secure for a corrupt $P_2$.

**Proposition 8.15.** *Discounting privacy,* SecSamp2Fair($\Pi^{\heartsuit}$) *is fully-secure for a corrupt $P_1$.*

*Proof.* The proof from the previous chapter applies here as well. The only difference is that, in order to recreate the backup sequence, the simulator must use information about the input that was leaked by the trusted party. Other than that, the remaining analysis is the same. Alternatively, see Appendix F p. 166 for a direct proof . □

From Proposition 8.15, it does *not* follow that the protocol is insecure. After all, fully secure protocols are also secure with respect to the leaky model. We show that the protocol is insecure by means of an attack. Specifically, we define a game between the honest party and the adversary, and we conclude by showing that there is a discrepancy between the probability of the adversary winning in the real world, compared to the probability of her winning in the ideal world.

**Proposition 8.16.** SecSamp2Fair($\Pi^{\heartsuit}$) *is not fully secure for a corrupt $P_1$.*

*Proof.* Consider the game from Figure 8.5 between the corrupted party $P_1$ and the honest party $P_2$. A quick analysis reveals that the optimal strategy for $\mathcal{A}$ in the ideal model is to play $x_4$, and to pick an input from one of the sets $\{y_1, y_2, y_4\}$ and $\{y_3, y_5\}$, depending on the output she received from the trusted party. It follows that $\Pr[\mathcal{A} \text{ wins } (\text{Ideal})] \leq 2/5$.

Turning to the real model, consider the following attack. $\mathcal{A}$ hands $x_4$ for the computation. If $\text{out}_1 = 1$, set $\text{guess} \in_U \{y_3, y_5\}$. If $\text{out}_1 = 0$, set $\text{guess} = y_1$ or $\text{guess} \in_U \{y_2, y_4\}$ if $a_2 = 1$ or $a_2 = 0$, respectively. Depending on the value of $i^*$, $\Pr[\mathcal{A} \text{ wins } (\text{Real}) \mid i^* \neq 2] = 2/5$ and

$$\Pr[\mathcal{A} \text{ wins } (\text{Real}) \mid i^* = 2] = \frac{2}{5} \cdot \frac{1}{2} + \frac{2}{5} \cdot \frac{1}{2} + \frac{1}{5} = \frac{3}{5} \ .$$

It follows that

$$\begin{aligned} \Pr[\mathcal{A} \text{ wins } (\text{Real})] &= (1-\gamma) \cdot \frac{2}{5} + \gamma \cdot \frac{3}{5} \\ &= \frac{2}{5} + \gamma \cdot \frac{1}{5} > 2/5 \ . \end{aligned}$$

□

**Yet Another Game**

1. The honest party chooses an input $y \in Y$ uniformly at random.
2. The corrupted party chooses $x \in X$.
3. On the inputs chosen by the parties,
   - **Real** Parties execute SecSamp2Fair($\Pi^{\heartsuit}$) for computing $f$.
   - **Ideal** Parties invoke the trusted party computing $f$.
4. The honest party accepts if $\mathsf{out}_2 = f_2(x_4, y)$.
5. The adversary outputs $\mathsf{guess} \in Y$.

The adversary wins if $P_2$ accepts *and* $\mathsf{guess} = y$.

Figure 8.5: A Passive Attack on Protocol SecSamp2Fair($\Pi^{\heartsuit}$)

# Concluding Remarks

The semi-balanced criterion in combination with protocol FairTwoParty account for whether a given finite symmetric Boolean function is fair. The characterization states that a function is fair if and only if it is not semi-balanced. The characterization was extended to randomized and multi-party functions. For the latter, we assumed that the parties have access to broadcast, and that the honest parties form a relative majority, at worst.

We defined locking strategies and sampling attacks, and we showed how the two concepts can be put to use toward the characterization of arbitrary finite functions. In particular, the semi-balanced criterion was extended to arbitrary functions, and a transformation was established from protocols satisfying "simple" security requirements to protocols that are fully secure. Specifically, the transformation involves introducing a threshold round to protocols that are passively secure and secure against sampling attacks.

Finally, we investigated the characterization of asymmetric Boolean functions, and we proposed an algorithm that builds relevant protocols for non semi-balanced functions. Unfortunately, the semi-balanced criterion and the algorithm do not account for whether every asymmetric Boolean function is fair. For one such problematic function, we showed that fairness can be achieved *at the expense of privacy.* We conclude with a short list of open problems.

Beyond locking strategies.

Locking strategies are defined in a communication model where the parties make a single call the an ideal functionality, and no other interactions is allowed to occur. We ask whether analogous-but-distinct strategies exist, if we allow plain model interactions or multiple calls to the function and its transpose.

The optimal round complexity for passive-security and security against sampling attacks.

As stated in Conjecture 7.5, for a *finite* function, we suspect that passive security and security against sampling attacks can either be achieved in a constant number of rounds or not at all.

Assumptions regarding the backup sequences in the online-dealer model.

In the game serving as a prelude to the algorithm from Chapter 8, we assumed that the dealer computes the backup outputs of each party as a randomized function of its input/output pair. It would be interesting to know whether this assumption incurs any loss of generality, and how the assumption is reconciled with passive security, if it does. If the assumption does not incur any loss of generality, then the question of characterizing asymmetric Boolean functions is settled. At the same time, it indicates that fairness and full-security are not equivalent.

Fairness vs Privacy.

As stated in Conjecture 8.13, if $f$ is computable with full security, then we suspect that the algorithm from Figure 8.2 terminates with either one of the parties running out of inputs. If so, then the function from Chapter 8 is the first example of a non semi-balanced *fair* function that is *not* fully secure. In any case, functions of unknown status require further investigation.

Other Classes of Functions.

Little is known about the characterization for functions other than Boolean ones. Designing algorithms similar to the one from Chapter 8 may be helpful in this regard. Similarly, little is known for functions with input domains of infinite size, or functionalities defined as a families of functions indexed by the security parameter. It is unclear how the framework we propose applies to such functions, if it does at all. In particular, definitions for locking strategies and sampling attacks are not immediate for such functions.

The Multiparty Case.

For reasons mentioned in Part I, most of Part II carries on to finite MPC functions in the presence of *relative* dishonest majorities. However, once the 1/2-threshold has been crossed, several difficulties arise in applying our framework. Specifically, while locking strategies and sampling attacks remain meaningful notions, the transformation from Chapter 7 may not be applicable to the multiparty case, or it may be rather more involved.

# Bibliography

[1] S. Agrawal and M. Prabhakaran. On fair exchange, fair coins and fair sampling. In *CRYPTO*, pages 259–276, 2013.

[2] G. Asharov. Towards characterizing complete fairness in secure two-party computation. In *TCC*, pages 291–316, 2014.

[3] G. Asharov, A. Beimel, N. Makriyannis, and E. Omri. Complete fairness in multi-party computation without an honest majority. In *TCC*, pages 199–228, 2015.

[4] G. Asharov, Y. Lindell, and T. Rabin. A full characterization of functions that imply fair coin tossing and ramifications to fairness. In *TCC*, pages 243–262, 2013.

[5] G. Asharov, Y. Lindell, and H. Zarosim. Fair and efficient secure multiparty computation with reputation systems. In *ASIACRYPT, Part II*, pages 201–220, 2013.

[6] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *CCS*, pages 7–17, 1997.

[7] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In *EUROCRYPT*, pages 591–606, 1998.

[8] G. Ateniese. Efficient verifiable encryption (and fair exchange) of digital signatures. In *CCS*, pages 138–146, 1999.

[9] F. Bao, R. H. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In *SP*, pages 77–85, 1998.

[10] D. Beaver and S. Goldwasser. Multiparty computation with faulty majority. In *FOCS*, pages 468–473, 1989.

[11] A. Beimel, Y. Lindell, E. Omri, and I. Orlov. $1/p$-secure multiparty computation without honest majority and the best of both worlds. pages 277–296, 2011.

[12] A. Beimel, E. Omri, and I. Orlov. Protocols for multiparty coin toss with dishonest majority. In *CRYPTO*, pages 538–557, 2010.

[13] A. Beimel, E. Omri, and I. Orlov. Protocols for multiparty coin toss with a dishonest majority. *J. of Cryptology*, 28(3):551–600, 2015.

[14] M. Ben-Or, O. Goldreich, S. Micali, and R. Rivest. A fair protocol for signing contracts. In *ICALP*, pages 43–52, 1985.

[15] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for noncryptographic fault-tolerant distributed computations. In *STOC*, pages 1–10, 1988.

[16] I. Bentov and R. Kumaresan. How to use bitcoin to design fair protocols. In *CRYPTO, Part II*, pages 421–439, 2014.

[17] M. Blum. Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, 15(1):23–27, 1983.

[18] M. Blum. How to exchange (secret) keys. *ACM Trans. Comput. Syst.*, 1(2):175–193, 1983.

[19] D. Boneh and M. Naor. Timed commitments. pages 236–254, 2000.

[20] E. F. Brickell, D. Chaum, I. B. Damgård, and J. van de Graaf. Gradual and verifiable release of a secret (extended abstract). In *CRYPTO*, pages 156–166, 1987.

[21] C. Cachin and J. Camenisch. Optimistic fair secure computation. In *CRYPTO*, pages 93–111, 2000.

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

[23] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–147, 2001.

[24] R. Canetti and M. Vald. Universally composable security with local adversaries. In *SCN*, pages 281–301, 2012.

[25] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *STOC*, pages 11–19, 1988.

[26] R. Cleve. Limits on the security of coin flips when half the processors are faulty. In *STOC*, pages 364–369, 1986.

[27] R. Cleve. Controlled gradual disclosure schemes for random bits and their applications. In *CRYPTO*, pages 573–588, 1990.

[28] R. Cleve and R. Impagliazzo. Martingales, collective coin flipping and discrete control processes. Manuscript, 1993.

[29] R. Cohen, I. Haitner, and E. Omri. Characterization of secure multiparty computation without broadcast. In *TCCa*, pages 596–616, 2016.

[30] R. Cohen and Y. Lindell. Fairness versus guaranteed output delivery in secure multiparty computation. In *ASIACRYPT*, pages 466–485, 2014.

[31] I. Damgård. Practical and provably secure release of a secret and exchange of signatures. *J. of Cryptology*, 8(4):201–222, 1995.

[32] Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic fair exchange in a multi-user setting. In *PKC*, pages 118–133, 2007.

[33] S. Even. A protocol for signing contracts. *SIGACT News*, 15(1):34–39, 1983.

[34] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.

[35] S. Even and Y. Yacobi. Relations among public key signature schemes. Report #175, Technion Israel Institute of Technology, Computer Science Department, 1980.

[36] A. Gál and P. Pudlák. Monotone complexity and the rank of matrices. *Inform. Process. Lett.*, 87:321–326, 2003.

[37] J. A. Garay, M. Jakobsson, and P. MacKenzie. Abuse-free optimistic contract signing. In *CRYPTO*, pages 449–466, 1999.

[38] J. A. Garay, P. D. MacKenzie, M. Prabhakaran, and K. Yang. Resource fairness and composability of cryptographic protocols. In *TCC*, pages 404–428, 2006.

[39] O. Goldreich. A simple protocol for signing contracts. In *CRYPTO*, pages 133–136, 1984.

[40] O. Goldreich. *Foundations of Cryptography, Voume II Basic Applications*. Cambridge University Press, 2004.

[41] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC*, pages 218–229, 1987.

[42] S. Goldwasser and L. Levin. Fair computation of general functions in presence of immoral majority. In *CRYPTO*, pages 77–93, 1991.

[43] S. D. Gordon. *On fairness in secure computation.* PhD thesis, University of Maryland, 2010.

[44] S. D. Gordon, C. Hazay, J. Katz, and Y. Lindell. Complete fairness in secure two-party computation. In *STOC*, pages 413–422, 2008.

[45] S. D. Gordon, C. Hazay, J. Katz, and Y. Lindell. Complete fairness in secure two-party computation. *J. of the ACM*, 58(6):24:1–24:37, 2011.

[46] S. D. Gordon, Y. Ishai, T. Moran, R. Ostrovsky, and A. Sahai. On complete primitives for fairness. In *TCC*, pages 91–108, 2010.

[47] S. D. Gordon and J. Katz. Complete fairness in multi-party computation without an honest majority. In *TCC*, pages 19–35, 2009.

[48] S. D. Gordon and J. Katz. Partial fairness in secure two-party computation. In *EUROCRYPT*, pages 157–176, 2010.

[49] I. Haitner and E. Tsfadia. An almost-optimally fair three-party coin-flipping protocol. In *STOC*, pages 408–416, 2014.

[50] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *STOC*, pages 623–632, 2004.

[51] R. Impagliazzo and M. Yung. Direct minimum-knowledge computations (extended abstract). In *CRYPTO*, pages 40–51, 1987.

[52] Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank. On combining privacy with guaranteed output delivery in secure multiparty computation. In *CRYPTO*, pages 483–500, 2006.

[53] J. Kahn, J. Komlós, and E. Szemerédi. On the probability that a random $\pm 1$-matrix is singular. *J. Amer Math. Soc.*, 8(1):223–240, 1995.

[54] J. Katz. On achieving the "best of both worlds" in secure multiparty computation. In *STOC*, pages 11–20, 2007.

[55] A. Kiayias, H.S. Zhou, and V. Zikas. Fair and robust multi-party computation using a global transaction ledger. In *EUROCRYPT, Part II*, pages 705–734, 2016.

[56] H. Kılınç and A. Küpçü. Optimally efficient multi-party fair exchange and fair secure multi-party computation. In *CT-RSA*, pages 330–349, 2015.

[57] R. Kumaresan and I. Bentov. How to use bitcoin to incentivize correct computations. In *CCS*, pages 30–41, 2014.

[58] A. Küpçü and A. Lysyanskaya. Optimistic fair exchange with multiple arbiters. In *ESORICS*, pages 488–507, 2010.

[59] A. Küpçü and A. Lysyanskaya. Usable optimistic fair exchange. In *CT-RSA*, pages 252–267, 2010.

[60] Y. Lindell. Legally-enforceable fairness in secure two-party computation. In *CT-RSA*, pages 121–137, 2008.

[61] M. Luby, S. Micali, and C. Rackoff. How to simultaneously exchange a secret bit by flipping a symmetrically-biased coin. In *FOCS*, pages 11–21, 1983.

[62] N. Makriyannis. On the fairness of finite boolean functions. In *RECSI*, 2012.

[63] N. Makriyannis. On the classification of finite boolean functions up to fairness. In *SCN*, pages 135–154, 2014.

[64] U. Maurer and R. Renner. Abstract cryptography. In *ICS*, pages 1–21, 2011.

[65] S. Micali. Simple and fast optimistic protocols for fair electronic exchange. In *PODC*, pages 12–19, 2003.

[66] T. Moran, M. Naor, and G. Segev. An optimally fair coin toss. In *TCC*, pages 1–18, 2009.

[67] B. Pfitzmann, M. Schunter, and M. Waidner. Optimal efficiency of optimistic contract signing. In *PODC*, pages 113–122, 1998.

[68] B. Pinkas. Fair secure two-party computation. In *EUROCRYPT*, pages 87–105, 2003.

[69] M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981. Available online in the Cryptology ePrint Archive, Report 2005/187.

[70] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *STOC*, pages 73–85, 1989.

[71] U. V. Vazirani and V. V. Vazirani. Trapdoor pseudo-random number generators, with applications to protocol design. In *FOCS*, pages 23–30, 1983.

[72] J. von Neumann. Various Techniques Used in Connection with Random Digits. *J. Res. Nat. Bur. Stand.*, 12:36–38, 1951.

[73] A. C. Yao. How to generate and exchange secrets. In *FOCS*, pages 162–167, 1986.

[74] Günter M. Ziegler. *Lectures on 0/1-Polytopes.* 2000.

# Appendices

Appendix **A**

# Security-with-Abort

**Inputs:** Each party $P_i$ holds $1^n$ and $x_i \in X_i$. The adversary is given an auxiliary input $z \in \{0,1\}^*$. The trusted party $\mathcal{T}$ has no input.

**Parties send inputs:** Each honest party sends its input to $\mathcal{T}$, each corrupted party sends a value of the adversary's choice. Write $(x'_1, \ldots, x'_m)$ for the tuple of inputs received by $\mathcal{T}$.

**The trusted party performs computation:** If any $x'_i$ is not in the appropriate domain (or was not sent at all), then $\mathcal{T}$ reassigns the aberrant input to some default value. Write $(x'_1, \ldots, x'_m)$ for the tuple of inputs after (possible) reassignment. The trusted party then chooses a random string $r$ and computes $f(x'_1, \ldots, x'_m; r)$.

**Trusted party sends outputs to corrupted parties:** Each corrupted party $P_i \in B$ receives $f_i(x'_1, \ldots, x'_m; r)$.

**Adversary decides whether to abort:** The adversary instructs the trusted party to either *abort* or *continue.*

**Trusted party sends outputs to honest parties:** Each honest party $P_i \notin B$ receives $\perp$ or $f_i(x'_1, \ldots, x'_m; r)$, depending on the adversary's choice in the previous step.

**Outputs:** Each honest party outputs whatever $\mathcal{T}$ sent him, the corrupted parties output nothing, and the adversary outputs a probabilistic polynomial-time function of its view.

Figure A.1: The Ideal Model with Abort.

In this model, the simulator (ideal-world adversary) decides who receives output. The honest parties' outputs and the adversary's view in the model-with-abort are denoted $(\mathrm{Out}^{\mathsf{IM\ abort}}_{\mathcal{S}(z),f}, \mathrm{View}^{\mathsf{IM\ abort}}_{\mathcal{S}(z),f})$.

**Definition A.1.** We say that $\Pi$ computes $f$ *securely with abort* tolerating coalitions of size at most $t$ if for every non-uniform polynomial time adversary $\mathcal{A}$, there exists a non-uniform polynomial time adversary $\mathcal{S}$ (called the simulator) controlling $B$ in the ideal model such that

$$\left\{\left(\mathrm{Out}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}, \mathrm{View}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}\right)(1^n, x_1, \ldots, x_m)\right\}_{\substack{(x_1,\ldots,x_m)\in X,\\ z\in\{0,1\}^*, n\in\mathbb{N}}} \overset{c}{\equiv} \left\{\left(\mathrm{Out}^{\mathsf{IM\ abort}}_{\mathcal{S}(z),f}, \mathrm{View}^{\mathsf{IM\ abort}}_{\mathcal{S}(z),f}\right)(1^n, x_1, \ldots, x_m)\right\}_{\substack{(x_1,\ldots,x_m)\in X,\\ z\in\{0,1\}^*, n\in\mathbb{N}}}.$$

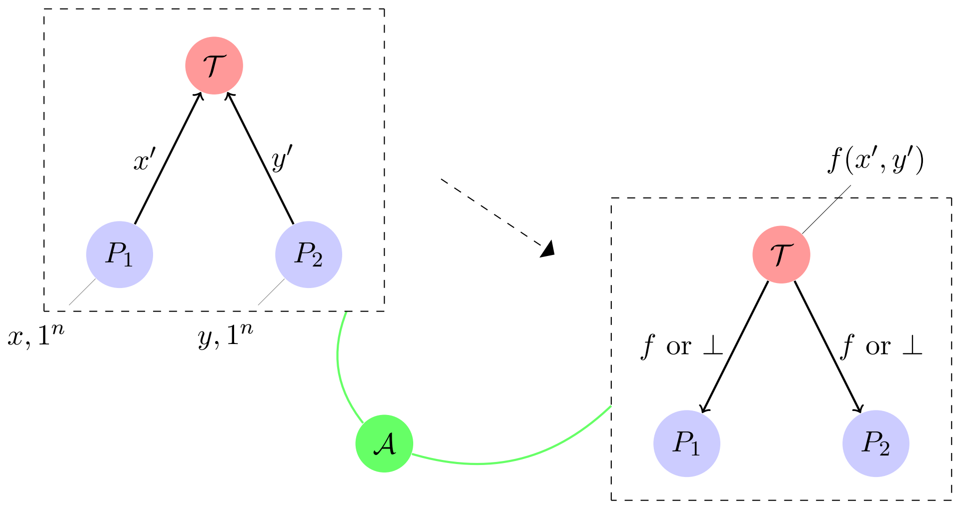

Figure A.2: Visual Illustration of the Two-Party Secure-with-Abort Model.

Appendix **B**

# Dealer Model to Plain Model Transformation

The first step is to transform the dealer from an interactive player that is invoked multiple times, to an offline player that is only invoked once at the beginning. Define message authentication code $(\texttt{Gen}, \texttt{Mac}, \texttt{Vrfy})$ consisting of three algorithms such that

- $\texttt{Gen}$ takes as input $1^n$ and generates a key $k$,
- $\texttt{Mac}$ takes as inputs a message $m$ and a key $k$, and generates a tag $t = \texttt{Mac}_k(m)$,
- $\texttt{Vrfy}$ takes as inputs a message $m$, a tag $t$ and a key $k$, and outputs a bit $b = \texttt{Vrfy}_k(m\,;\,t)$ .

In the offline dealer model, the parties are instructed to reconstruct bits $a_i$ and $b_i$ by interacting with one another. Thus, instead of receiving $a_1 \ldots a_r$ (for $P_1$) and $b_1 \ldots b_r$ (for $P_2$) in a sequence or $r$ iterations, parties receive shares $\{a_i^{(1)}, b_i^{(1)}\}_{i=1\ldots r}$ and $\{a_i^{(2)}, b_i^{(2)}\}_{i=1\ldots r}$ respectively in a single invocation of the dealer. As the notation suggests, $a_i^j$ and $b_i^j$ are shares of a 2-out-of-2 sharing scheme of $a_i$ and $b_i$ respectively.

Now, for $i = 1 \ldots r$, party $P_2$ is instructed to send $a_i^{(2)}$ to $P_1$, followed by $P_1$ who is instructed to send $b_i^{(1)}$ to $P_2$. To prevent parties from lying, these shares are signed using a secure message authentication code described above. See Figure B.1 for a full description of the offline dealer model.

**Inputs:** Party $P_1$ (resp. $P_2$) holds $1^n$ and $x \in X$ (resp. $y \in Y$). The dealer $\mathcal{D}$ has no input.

**Parties compute backup outputs:** $P_1$ (resp. $P_2$) is instructed to compute $a_0$ (resp. $b_0$).

**Parties send inputs:** Parties are instructed to send their inputs to the dealer. Write $(x', y')$ for the inputs received by $\mathcal{D}$.

**The dealer performs computation:** The dealer computes bits $a_1, \dots, a_r$ and $b_1, \dots, b_r$, where $r \in \mathbb{N}$ depends on the security parameter. The dealer generates random bits $a_1^{(1)}, \dots, a_r^{(1)}, b_1^{(1)}, \dots, b_r^{(1)}$, and computes $a_i^{(2)} = a_i^{(1)} \oplus a_i$ and $b_i^{(2)} = b_i^{(1)} \oplus b_i$, for every $i \in \{1, \dots, r\}$. The dealer generates keys $k^a$ and $k^b$ and computes tags $t_i^{(2)} = \mathtt{MAC}_{k^a}\left(a_i^{(2)} \middle\| i\right)$ and $t_i^{(1)} = \mathtt{MAC}_{k^b}\left(b_i^{(1)} \middle\| i\right)$, for every $i \in \{1, \dots, r\}$.

**The dealer replies:**

1. Party $P_1$ receives $a_1^{(1)}, \dots, a_r^{(1)}, b_1^{(1)}, \dots, b_r^{(1)}$ from the dealer, as well as key $k^a$ and tags $t_1^{(1)}, \dots, t_r^{(1)}$. Then, $P_1$ instructs $\mathcal{D}$ to *abort* or *continue*. In the first case, the dealer sends $\perp$ to both parties and halts. Otherwise go to next step.
2. Party $P_2$ receives $a_1^{(2)}, \dots, a_r^{(2)}, b_1^{(2)}, \dots, b_r^{(2)}$ from the dealer, as well as key $k^b$ and tags $t_1^{(2)}, \dots, t_r^{(2)}$.

**Parties exchange information:** For $i = 1 \dots r$

1. Party $P_2$ sends $(a_i^{(2)}, t_i^{(2)})$ to $P_1$. If $\mathtt{Vrfy}_{k^a}\left(a_i^{(2)} \,\|\, i\,;\, t_i^{(2)}\right) = 1$, party $P_1$ sets $a_i = a_i^{(1)} \oplus a_i^{(2)}$ and proceeds to the next step. In any other case (early abort or failed verification), $P_1$ halts and outputs the last $a_{i-1}$.
2. Party $P_1$ sends $(b_i^{(1)}, t_i^{(1)})$ to $P_1$. If $\mathtt{Vrfy}_{k^b}\left(b_i^{(1)} \,\|\, i\,;\, t_i^{(1)}\right) = 1$, party $P_1$ sets $b_i = b_i^{(1)} \oplus b_i^{(2)}$ and proceeds to the next step. In any other case (early abort or failed verification), $P_2$ halts and outputs $b_{i-1}$.
3. If $i = r$, parties output $a_r$ and $b_r$ respectively.

Figure B.1: The Offline Dealer Model

Appendix **C**

# Playing with the Parameters of GHKL

Recall that, prior to round $i^*$, the backup outputs of $P_2$ are distributed according to $\mathbf{q} = M^T \cdot \mathbf{1}_\ell/\ell$. That is by choosing an input for $P_1$ uniformly at random. For some functions, the uniform distribution is problematic. For example, GHKL is not fully secure for the function described by the matrix

$$M = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} .$$

However, it can be shown that the protocol is fully secure if $\mathbf{q} = M^T \cdot (\mathbf{e}_2 + \mathbf{1}_7)/8$. That is by choosing an input for $P_1$ according to the more elaborate distribution encoded by $(\mathbf{e}_2 + \mathbf{1}_7)/8$. We investigate how to find relevant distributions for arbitrary functions.

Fix probability vector $\mathbf{u}$ and redefine $\mathbf{q} = M^T \cdot \mathbf{u}$ in protocol GHKL. Using the same simulator as Gordon et al. we deduce the generalized protocol is fully secure if[1] for some $\alpha \in (0, 1)$, and all $x \in \{1, \ldots, \ell\}$, there exist probability vectors $\mathbf{x}_x^{(0)}$ and $\mathbf{x}_x^{(1)}$ such that

$$\begin{aligned} M^T \cdot \mathbf{x}_x^{(0)} &= \mathbf{q} + \lambda_x^{(0)} \cdot ((\mathbf{1}_k - \mathsf{row}_x) * \mathbf{q}) \\ M^T \cdot \mathbf{x}_x^{(1)} &= \mathbf{q} + \lambda_x^{(1)} \cdot (\mathsf{row}_x * (\mathbf{1}_k - \mathbf{q})) \ , \end{aligned} \tag{C.1}$$

[1] We stress that the conditions depend on the simulator.

where $\lambda_x^{(a)} \in \mathbb{R}$ depends on $\alpha$ and $M$.

**Proposition C.1.** *The following statements are equivalent.*

- $\exists \alpha \in (0,1)$, $\forall x \in X$, $\exists$ *probability vectors* $\mathbf{x}_x^{(0)}$, $\mathbf{x}_x^{(1)}$ *satisfying Equation* (C.1).
- *There exists a strictly positive probability vector* $\mathbf{u}$ *such that* $\mathbf{v} \mapsto \mathbf{q} * \mathbf{v}$ *is an endomorphism of* $\mathcal{H}(M^T)$.

*Proof.* ($\Uparrow$) If $Q$ is an endomorphism of $\mathcal{H}(M^T)$ and $\mathbf{u}$ is strictly positive, then relevant vectors can be found by adjusting the parameter of $i^*$, exactly like in the proof from Chapter 3. ($\Downarrow$) Define $\mu = \{ j \mid \mathbf{u}(j) = 0\}$. Obviously, $\mathbf{v} \mapsto \mathbf{v} * \mathbf{q}$ is an endomorphism of $\mathcal{H}(M^T)$. We show that there exists $\mathbf{u}_0 \in \ker(M^T)$ such that $\sum_j \mathbf{u}_0(j) = 0$ and $\mathbf{u}_0(i) > 0$, for every $i \in \mu$. From Equation (C.1), deduce that there exist $\mathbf{a}_x^{(0)}$ and $\mathbf{a}_x^{(1)}$ such that

$$\begin{aligned} M^T \cdot \mathbf{a}_x^{(0)} &= (\mathbf{1}_k - \mathsf{row}_x) * \mathbf{q} \\ M^T \cdot \mathbf{a}_x^{(1)} &= \mathsf{row}_x * (\mathbf{1}_k - \mathbf{q}) \ , \end{aligned}$$

where $\sum_j \mathbf{a}_x^{(a)}(j) = 0$ and $\mathbf{a}_x^{(a)}(i) \geq 0$, for every $i \in \mu$. Moving a few terms around, deduce that

$$\begin{aligned} M^T \cdot \mathbf{a}_x^{(0)} - M^T \cdot \mathbf{u} &= \mathsf{row}_x * \mathbf{q} \\ M^T \cdot \mathbf{a}_x^{(1)} + M^T \cdot \mathbf{e}_x &= -\mathsf{row}_x * \mathbf{q} \ . \end{aligned}$$

Define $\mathbf{u}_0 = \mathbf{a}_x^{(0)} - \mathbf{u} + \mathbf{a}_x^{(1)} + \mathbf{e}_x$ and notice that $\mathbf{u}_0 \in \ker(M^T)$, $\sum_j \mathbf{u}_0(j) = 0$, $\mathbf{u}_0(x) > 0$, and $\mathbf{u}_0(x') \geq 0$, for every $x' \in \mu$. Repeat the process for every $x \in \mu$ and conclude. □

Let $P_0$ denote the orthogonal projection onto the kernel of $M$, and write $Q_i$ for the diagonal matrix $Q_i(j,j) = \mathsf{row}_i(j)$.

**Theorem C.2.** *If the non-monochromatic columns of* $M$ *do not contain* $\mathbf{1}_\ell$ *in their linear span and the equation below admits a strictly positive solution, then GHKL computes* $f$ *with full security.*

$$\begin{pmatrix} P_0 \cdot Q_1 \\ \vdots \\ P_0 \cdot Q_\ell \end{pmatrix} M^T \begin{pmatrix} r_1 \\ \vdots \\ r_\ell \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \ . \tag{C.2}$$

*Proof.* Assuming the non-monochromatic columns of $M$ do not contain $\mathbf{1}_\ell$ in their linear span, we analyse the conditions under which $\mathbf{v} \mapsto \left(M^T \cdot \mathbf{u}\right) * \mathbf{v}$ is an endomorphism of $\mathrm{im}(M^T)$, for strictly positive $\mathbf{u}$. Write $\mathbf{u} = (r_1, \ldots, r_\ell)^T$ and note that $Q = \sum_i r_i Q_i$. It holds that $Q$ defines an endomorphism of $\mathrm{im}(M^T)$ if and only if

$$\forall j \in \{1, \ldots, \ell\}, \qquad \left(\sum_i r_i Q_i\right) \mathsf{row}_j^T \in \mathrm{im}(M^T) \ . \tag{C.3}$$

Since the matrices are all diagonal, and thus commute, we deduce that

$$\begin{aligned}\left(\sum_i r_i Q_i\right) \mathsf{row}_j &= \left(\sum_i r_i Q_i\right) Q_j \cdot \mathbf{1}_\ell = Q_j \left(\sum_i r_i \mathsf{row}_i\right) \\ &= Q_j \left(\sum_i r_i \mathsf{row}_i\right) = Q_j \cdot M^T (r_1, \ldots, r_\ell)^T \ .\end{aligned}$$

Consequently, (C.3) holds if and only if (C.2) admits a strictly positive solution, in which case the probability vector is obtained by normalization. □

Appendix D

# FairTwoPartySpecial: Proof of Security

We follow the proof from the symmetric case, with two important modifications. First, we need two distinct simulations for $P_1$, depending on the choice of input. In particular, if $\mathcal{A}$ hands $x_1$ or $x_2$ to $\mathcal{S}$, then the simulation follows the symmetric case exactly. If $\mathcal{A}$ hands $x_3$ or $x_4$, then we simulate $P_1$ as if he were $P_2$ in the symmetric case. For the second player, we claim that simulating $P_2$ as if he were $P_1$ in Protocol FairTwoParty$_\sigma$ results in the correct simulator. We begin our analysis with a corrupt $P_1$, and we assume that the adversary hands either $x_1$ or $x_2$ to the simulator.

Let $\mathbf{q} = M^T \cdot \mathbf{1}_\ell$ and $\mathbf{p} = M \cdot \mathbf{1}_k$. We compare the distribution of $(a_i, \mathsf{out}_2)$ in the two worlds given that the adversary aborts in round $i \leq i^*$. We compute vectors $\mathbf{c}_x^{(0)}$ and $\mathbf{c}_x^{(1)}$ where $q_x^{(a)}(y)$ is the desired probability that the output of $P_2$ in the simulation (given that the adversary has aborted in round $i < i^*$) is 1 when the input of $P_1$ is $x$, the input of $P_2$ is $y$ and $a_i = a$. Next, assuming the adversary hands $x \in \{x_1, x_2\}$, we consider the four possible values of $(a_i, \mathsf{out}_2)$, we deduce that

$$\begin{aligned}
\mathbf{c}_x^{(0)}(y) &= \mathbf{q}(y) + \frac{\alpha}{(1-\alpha)(1-\mathbf{p}(x))}(1 - f_1(x,y))(\mathbf{q}(y) - f_2(x,y)) \\
\mathbf{c}_x^{(1)}(y) &= \mathbf{q}(y) + \frac{\alpha}{(1-\alpha)\mathbf{p}(x)} f_1(x,y)(\mathbf{q}(y) - f_2(x,y)) \ ,
\end{aligned}$$

and conclude that $M^{(*,1)T}\mathbf{x}_x^{(a)} = \mathbf{c}_x^{(a)}$ for the following vectors $\mathbf{x}_x^{(a)}$:

$$\begin{aligned}
\mathbf{x}_{x_1}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{p}(x_1))} 1/2 \cdot (0,1,-1,0)^T \\
\mathbf{x}_{x_1}^{(1)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{p}(x_1)} 1/4 \cdot (-3,-1,3,1)^T \\
\mathbf{x}_{x_2}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{p}(x_2))} 1/4 \cdot (1,-1,-1,1)^T \\
\mathbf{x}_{x_2}^{(1)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{p}(x_2)} 1/2 \cdot (0,-1,1,0)^T.
\end{aligned}$$

It remains to show that for some $\alpha \in (0,1)$ the above vectors become probability vectors - they already sum to 1, we just need them to be positive. A straightforward calculation reveals that any $\alpha \leq 1/9$ will do.

**Corrupt** $P_2$**.** As mentioned above, we construct a simulator $\mathcal{S}$ given a black-box to $\mathcal{A}$ that is completely analogous to $P_1$'s simulator in protocol FAIRTWOPARTY$_\sigma$. Namely, say that $\mathcal{A}$ hands $y \in Y$ to $\mathcal{S}$ for the computation of $f$. The simulator chooses $i^*$ according to a geometric distribution with parameter $\alpha$ and • for $i = 0, \ldots, i^* - 1$, the simulator hands $b_i = f(\widetilde{x}^{(i)}, y)$ to $\mathcal{A}$, where $\widetilde{x}^{(i)} \in_U X$. If $\mathcal{A}$ decides to quit, $\mathcal{S}$ sends $y_0$ according to distribution $\mathbf{y}_y^{(b_i)}$ (to be defined below), outputs $(b_0, \ldots, b_i)$ and halts. • for $i = i^*$, the simulator sends $y$ to the trusted party, receives $b = f_2(x,y)$ and hands $b_{i^*} = b$ to $\mathcal{A}$. If $\mathcal{A}$ decides to quit, $\mathcal{S}$ outputs $(b_0, \ldots, b_{i^*})$ and halts. • for $i = i^* + 1, \ldots, r$, the simulator hands $b_i = b$ to $\mathcal{A}$. If $\mathcal{A}$ decides to quit, $\mathcal{S}$ outputs $(b_0, \ldots, b_i)$ and halts. • If $\mathcal{A}$ is still active at the end, $\mathcal{S}$ outputs $(b_0, \ldots, b_r)$ and halts.

For reasons mentioned in the proof of Theorem 5.9, we only need to compare the distributions $(\text{View}_{\mathcal{A}}, \mathsf{out}_1)$ in the real and ideal worlds given that the adversary aborts in a round $1 \leq i \leq i^*$. Thus, in the rest of the proof we let $i$ be the round in which the adversary aborts and assume that $1 \leq i \leq i^*$. The view of the adversary in both worlds is $b_0, \ldots, b_i$. Notice that in both worlds $b_1, \ldots, b_{i-1}$ are equally distributed and are independent of $(b_i, \mathsf{out}_1)$. Thus, we only compare the distribution of $(b_i, \mathsf{out}_1)$ in both worlds. Now, for every $x \in X$, $y \in Y$, and $b \in \{0,1\}$, define $\mathbf{d}_y^{(b)} \in \mathbb{R}^\ell$ such that $\mathbf{d}_y^{(b)}(x) = \Pr[f_1(x, y_0) = 1]$, where $y_0$ is chosen according to the distribution $\mathbf{y}_y^{(b)}$. Consider the probability that $(b_i, \mathsf{out}_1)$ is equal to $(0,0)$

and $(0, 1)$, and deduce that

$$\mathbf{d}_{y_1}^{(0)} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_1))} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1/2 \end{pmatrix},$$

$$\mathbf{d}_{y_2}^{(0)} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_2))} \begin{pmatrix} 0 \\ 0 \\ 1/2 \\ 0 \end{pmatrix},$$

and

$$\mathbf{d}_{y_3}^{(0)} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_3))} \begin{pmatrix} 0 \\ 0 \\ 1/2 \\ 0 \end{pmatrix},$$

$$\mathbf{d}_{y_4}^{(0)} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_4))} \begin{pmatrix} 0 \\ 0 \\ -1/2 \\ 0 \end{pmatrix}.$$

It follows that $M^{(1,*)}\mathbf{y}_y^{(0)} = \mathbf{d}_y^{(0)}$ for the following vectors:

$$\begin{aligned}
\mathbf{y}_{y_1}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_1))} 1/2 \cdot (0,1,-1,0)^T \\
\mathbf{y}_{y_2}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_2))} 1/2 \cdot (1,0,-1,0)^T \\
\mathbf{y}_{y_3}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_3))} 1/2 \cdot (1,0,-1,0)^T \\
\mathbf{y}_{y_4}^{(0)} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)(1-\mathbf{q}(y_4))} 1/2 \cdot (-1,0,1,0)^T.
\end{aligned}$$

To obtain probability vectors, any $\alpha \leq 9$ will do. Similarly, consider the probability that $(b_i, \mathsf{out}_1)$ is equal to $(1,0)$ and $(1,1)$, and deduce that

$$\mathbf{d}^{(1)}_{y_1} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_1)} \begin{pmatrix} 0 \\ 0 \\ -1/2 \\ 0 \end{pmatrix} \quad ,$$

$$\mathbf{d}^{(1)}_{y_2} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_2)} \begin{pmatrix} 0 \\ 0 \\ 0 \\ -1/2 \end{pmatrix} \quad ,$$

and

$$\mathbf{d}^{(1)}_{y_3} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_3)} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1/2 \end{pmatrix} \quad ,$$

$$\mathbf{d}^{(1)}_{y_4} = \begin{pmatrix} 3/4 \\ 1/4 \\ 1/2 \\ 1/2 \end{pmatrix} + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_4)} \begin{pmatrix} 0 \\ 0 \\ 0 \\ -1/2 \end{pmatrix} \quad .$$

It follows that $M^{(1,*)}\mathbf{y}^{(1)}_y = \mathbf{d}^{(1)}_y$ for the following vectors:

$$\begin{aligned}
\mathbf{y}^{(1)}_{y_1} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_1)} 1/2 \cdot (-1,0,1,0)^T \\
\mathbf{y}^{(1)}_{y_2} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_2)} 1/2 \cdot (0,-1,1,0)^T \\
\mathbf{y}^{(1)}_{y_3} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_3)} 1/2 \cdot (0,1,-1,0)^T \\
\mathbf{y}^{(1)}_{y_4} &= 1/4 \cdot (1,1,1,1)^T + \frac{\alpha}{(1-\alpha)\mathbf{q}(y_4)} 1/2 \cdot (0,-1,1,0)^T.
\end{aligned}$$

To obtain probability vectors, any $\alpha \leq 9$ will do. $\square$

Appendix E

# The Leaky Model

As the name suggests, in the leaky model, the honest party's output is leaked to the adversary.

**Inputs:** Parties $P_1$, $P_2$ hold $1^n$, and $x \in X$ and $y \in Y$ respectively. The adversary is given an auxiliary input $z \in \{0,1\}^*$. The trusted party $\mathcal{T}$ has no input.

**Parties send inputs:** The honest party sends its input to $\mathcal{T}$, the corrupted party sends a value of the adversary's choice. Write $(x', y')$ for the couple of inputs received by $\mathcal{T}$.

**The trusted party performs computation:** If either $x'$ or $y'$ is not in the appropriate domain (or was not sent at all), then $\mathcal{T}$ reassigns the aberrant input to some default value. Write $(x', y')$ for the pair of inputs after (possible) reassignment. The trusted party then chooses a random string $r$ and computes $f(x', y'; r)$.

**Trusted party sends outputs:** Each party $P_i$ receives $f_i(x', y'; r)$.

**Trusted party leaks inputs:** $\mathcal{T}$ hands the honest party's input ($x'$ or $y'$) to $\mathcal{A}$.

**Outputs:** Each honest party outputs whatever $\mathcal{T}$ sent him, the corrupted parties output nothing, and the adversary outputs a probabilistic polynomial-time function of its view.

Figure E.1: The Ideal Model with Leakage.

Notice that the ideal model satisfies the security requirements of – correctness – independence of inputs – fairness, but **not** privacy. Let $\mathcal{S}$ be an adversary in the ideal model (also called the simulator) given auxiliary input $z$, and write $(\mathrm{Out}^{\mathsf{Leak}}_{\mathcal{S}(z),f}, \mathrm{View}^{\mathsf{Leak}}_{\mathcal{S}(z),f})(1^n, x, y)$, for the honest party's output and the simulator's output in the leaky ideal model.

**Definition E.1.** Let $\Pi$ be a protocol for computing $f$. We say that $\Pi$ computes $f$ *with fairness* if for every non-uniform polynomial time adversary $\mathcal{A}$ controlling party $P_i$ in the real model, there exists a non-uniform polynomial time adversary $\mathcal{S}$ controlling $P_i$ in the ideal model such that

$$\left\{\left(\mathrm{Out}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}, \mathrm{View}^{\mathsf{Real}}_{\mathcal{A}(z),\Pi}\right)(1^n, x, y)\right\}_{\substack{(x,y)\in X\times Y,\\ z\in\{0,1\}^*, n\in\mathbb{N}}}$$
$$\overset{c}{\equiv} \left\{\left(\mathrm{Out}^{\mathsf{Leak}}_{\mathcal{S}(z),f}, \mathrm{View}^{\mathsf{Leak}}_{\mathcal{S}(z),f}\right)(1^n, x, y)\right\}_{\substack{(x,y)\in X\times Y,\\ z\in\{0,1\}^*, n\in\mathbb{N}}}.$$

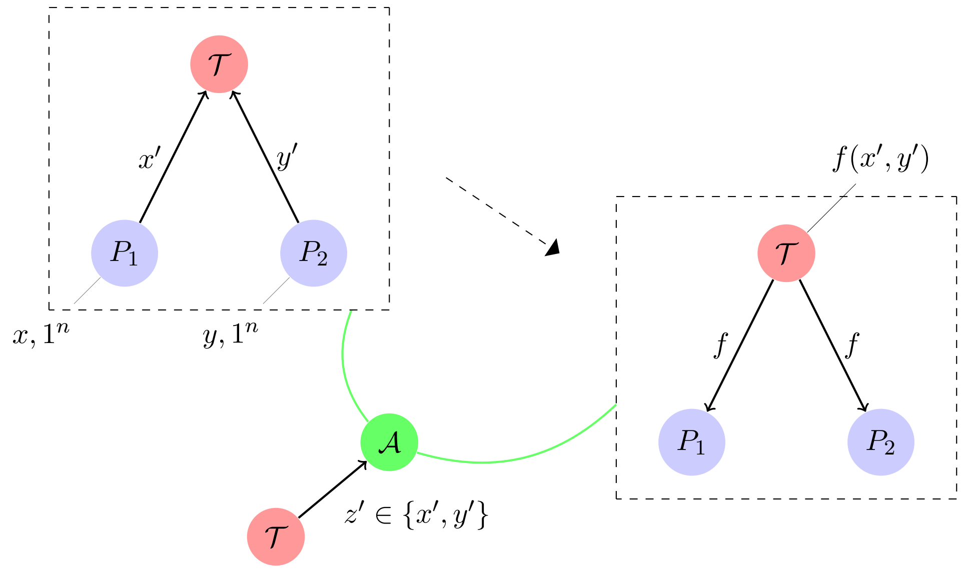

Figure E.2: Visual illustration of the Two-Party Leaky Model.

Appendix **F**

# Direct Proof of Proposition 8.15

For a corrupt $P_1$, we show the protocol is secure with respect to the leaky model – Appendix E p. 164. The difference between this model and the fully secure one is that the trusted party leaks the honest party's output to the adversary. The simulation for $P_1$ is identical to the simulation of protocol SecSamp2Fair with one major difference. In order to construct $a_{i^*-1}$ in the ideal model, the simulator uses the input leaked by the trusted party.

Write $\mathbf{p}$ and $\mathbf{q}^T$ for $M^{(1,*)}\mathbf{1}_k$ and $\mathbf{1}_\ell^T M^{(*,1)}$. Define $\mathbf{p}'_{x_0} \in \mathbb{R}^k$ such that $\mathbf{p}'_{x_0}(y_0) = \Pr\left[a_{i^*-1} = 1 \,|\, (x,y) = (x_0, y_0)\right]$ where $x$ and $y$ denote the inputs that the dealer receives from the parties. Observe that $\mathbf{p}'_{x_2} = \mathbf{p}'_{x_3} = \mathbf{1}_5 - \mathbf{p}'_{x_5} = \mathbf{1}_5$,

$$\mathbf{p}'_{x_1} = \frac{1}{2} \cdot (1,2,1,2,1)^T, \quad \mathbf{p}'_{x_4} = (1,0,1,0,1)^T \ .$$

Next, for an adversary quitting at round $i$, let $a_i$ denote the last bit handed to the adversary and let $\mathsf{out}_2$ denote the honest party's output. Write $(a_i, \mathsf{out}_2)^{\mathsf{R}}$ and $(a_i, \mathsf{out}_2)^{\mathsf{Id}}$ for the pair in the relevant model. For reasons encountered in many other proofs, it suffices to compare the probability distributions of $(a_i, \mathsf{out}_2)^{\mathsf{R}}$ and $(a_i, \mathsf{out}_2)^{\mathsf{Id}}$, assuming $i^* - 1$ has not been surpassed. So, in the real model:

$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{R}} = (0,0)\right] = (1-\gamma)(1-\mathbf{p}(x))(1-\mathbf{q}(y)) + \gamma(1-\mathbf{p}'_x(y))/2$$
$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{R}} = (0,1)\right] = (1-\gamma)(1-\mathbf{p}(x))\mathbf{q}(y) + \gamma(1-\mathbf{p}'_x(y))/2$$

and

$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{R}} = (1,0)\right] = (1-\gamma)\mathbf{p}(x)(1-\mathbf{q}(y)) + \gamma\mathbf{p}'_x(y)/2$$
$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{R}} = (1,1)\right] = (1-\gamma)\mathbf{p}(x)\mathbf{q}(y) + \gamma\mathbf{p}'_x(y)/2\ .$$

Turning to the ideal model, write $\mathbf{c}_x^{(\alpha)}(y)$ for the distribution of the honest party's output. Again, assuming $i^* - 1$ has not been surpassed:

$$\begin{aligned}\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{Id}} = (0,0)\right] &= (1-\gamma)(1-\mathbf{c}_x^{(0)}(y))(1-\mathbf{p}(x)) \\ &\quad + \gamma(1-f_2(x,y))(1-\mathbf{p}'_x(y)) \\ \Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{Id}} = (0,1)\right] &= (1-\gamma)\mathbf{c}_x^{(0)}(y)(1-\mathbf{p}(x)) + \gamma f_2(x,y)(1-\mathbf{p}'_x(y))\ ,\end{aligned}$$

and

$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{Id}} = (1,0)\right] = (1-\gamma)(1-\mathbf{c}_x^{(1)}(y))\mathbf{p}(x) + \gamma(1-f_2(x,y))\mathbf{p}'_x(y)$$
$$\Pr\left[(a_i, \mathsf{out}_2)^{\mathsf{Id}} = (1,1)\right] = (1-\gamma)\mathbf{c}_x^{(1)}(y)\mathbf{p}(x) + \gamma f_2(x,y)\mathbf{p}'_x(y)\ .$$

Thus

$$\begin{aligned}\mathbf{c}_x^{(0)}(y) &= \mathbf{q}(y) + \frac{\gamma(1-\mathbf{p}'_x(y))(1/2 - f_2(x,y))}{(1-\gamma)(1-\mathbf{p}(x))} \\ \mathbf{c}_x^{(1)}(y) &= \mathbf{q}(y) + \frac{\gamma\mathbf{p}'_x(y)(1/2 - f_2(x,y))}{(1-\gamma)\mathbf{p}(x)}\ .\end{aligned}$$

In matrix form:

$$\begin{aligned}\mathbf{c}_x^{(0)} &= \mathbf{q} + \lambda_{\gamma,x}^{(0)} \cdot (1-\mathbf{p}'_x) * \left(\mathbf{1}_5/2 - M^{(*,1)T}\mathbf{e}_x\right) \\ \mathbf{c}_x^{(1)} &= \mathbf{q} + \lambda_{\gamma,x}^{(1)} \cdot \mathbf{p}'_x * \left(\mathbf{1}_5/2 - M^{(*,1)T}\mathbf{e}_x\right)\end{aligned}$$

where $\lambda_{\gamma,x}^{(\alpha)} = \gamma \cdot (1-\gamma)^{-1} \cdot (1 - \alpha + (-1)^{\alpha+1}\mathbf{p}(x))^{-1}$. Finally, note that $\langle \mathbf{L}_2 \rangle = \langle (1,1,0,1,1)^T \rangle$ and deduce that, for every $\mathbf{y} \in \langle \mathbf{L}_2 \rangle$,

$$\left((\mathbf{1}_5 - \mathbf{p}'_x) * \left(\mathbf{1}_5/2 - M^{(*,1)T}\mathbf{e}_x\right)\right)^T \cdot \mathbf{y} = 0$$
$$\left(\mathbf{p}'_x * \left(\mathbf{1}_5/2 - M^{(*,1)T}\mathbf{e}_x\right)\right)^T \cdot \mathbf{y} = 0\ .$$

Conclude using the fundamental theorem of Linear Algebra.