A Full Characterization of Functions that Imply Fair Coin Tossing and Ramifications to Fairness

Gilad Asharov **Bar-Ilan University**

Yehuda Lindell Bar-Ilan University

Tal Rabin IBM Research

Bar-Ilan University

IBM

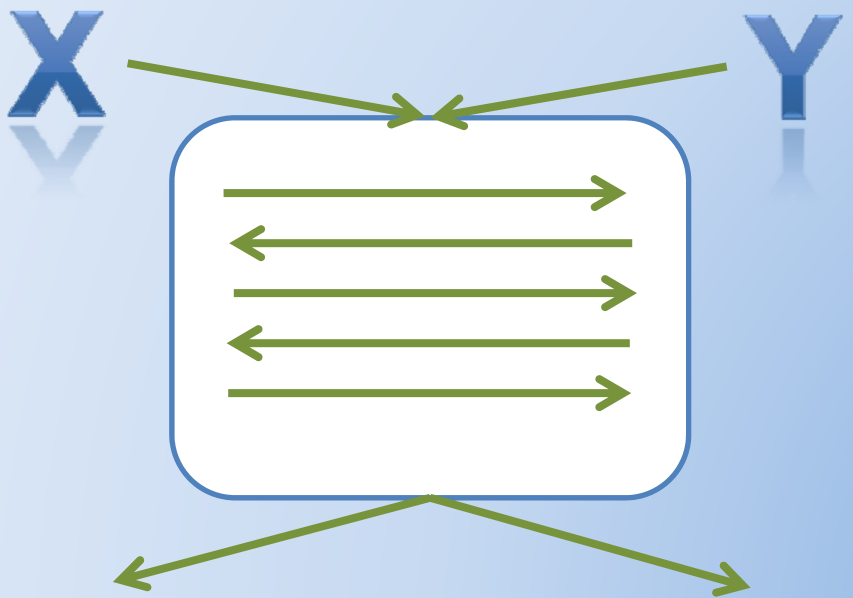
TCC 2013

Secure Multiparty Computation

- A set of parties with private inputs wish to compute some joint function of their inputs
- Parties wish to preserve some security properties. E.g., privacy and correctness
- Security must be preserved in the face of adversarial behavior by some of the participants, or by an external party

Fairness

- *The adversary* receives an output **if and only if** *the honest party* receives an output
  - In some sense, parties receive outputs **simultaneously**



Coin-Tossing

- The coin-tossing functionality:

$$f(\lambda, \lambda) = (U, U)$$

(U is the uniform distribution over {0,1})

  - both parties **agree** on the same uniform bit
  - **no** party can **bias** the result
- In 1986, Cleve showed that it is ***impossible*** to construct a fair **coin-tossing** protocol
  - Intuitively, no simultaneous exchange, so one party always has more information about the result, and can abort and bias the result

YL1

Fairness for Other Functionalities

- Gordon, Hazay, Katz and Lindell [STOC08] showed that there exist some **non-trivial** functions that can be computed with **complete fairness**!
  - Any protocol with no embedded XOR (essentially the less-than functionality)
  - Some specific functionalities with embedded XOR

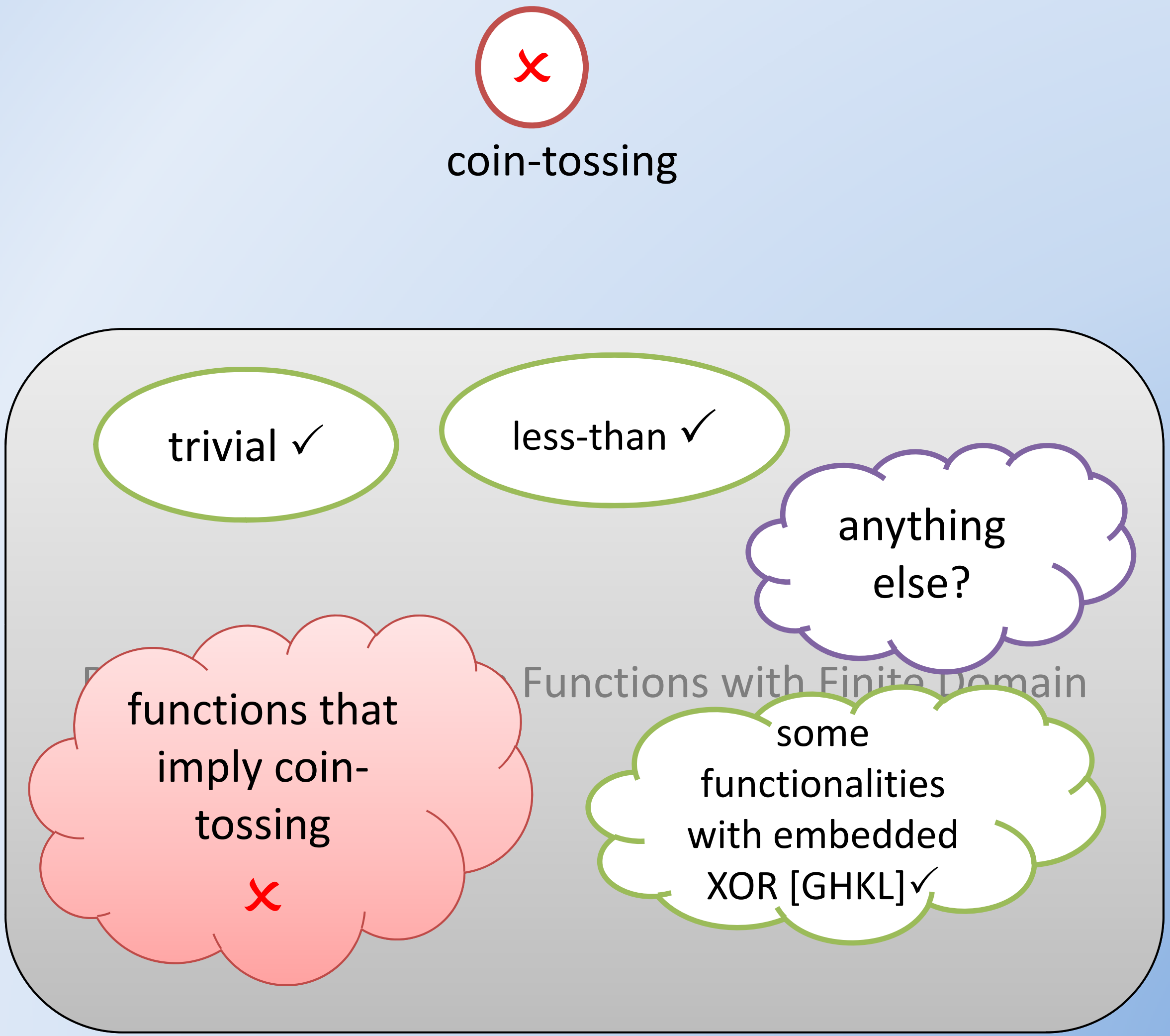
Characterizing Fairness

- **A fundamental question:**

  What functions can and cannot be securely computed with complete fairness?

- The only known impossibility result for fairness today is still that of Cleve

What Do We Know About the World?



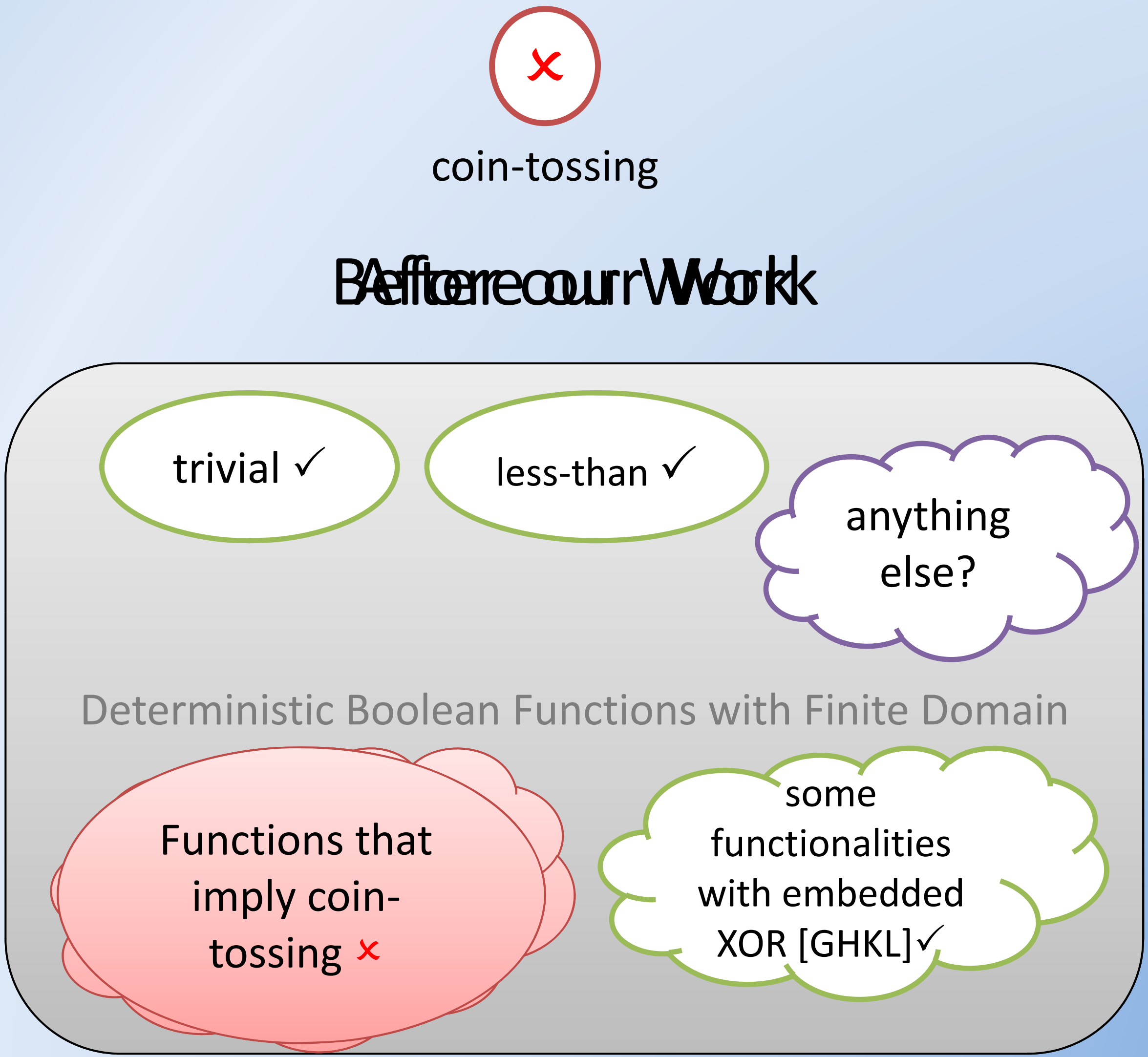
Characterizing Fairness

- Which Boolean functions with finite domain can be computed with complete fairness?
- Can we characterize the functions via a ***property*** such that:
  - If the function **satisfies** the property:
    it **can be** computed fairly
  - If the function **does not satisfy** the property:
    it **cannot be** computed fairly

Our Main Result

- We give a simple ***property*** (a criterion) such that
- If the function satisfies the property –
  it **implies** coin-tossing
  - Thus, it ***cannot*** be computed with complete fairness
- If the function does not satisfy the property –
  it **does not imply** coin-tossing*
  - We know exactly what Cleve's impossibility rules out
  - Proving impossibility for other functions requires a **new** proof (cannot be reduced to Cleve)

What Do We Know About the World?



What About This Function?

	y_1	y_2
x_1	**0**	**1**
x_2	**1**	**0**
x_3	**1**	**1**

What About This Function?

	y_1	y_2	y_3
x_1	**0**	**1**	**1**
x_2	**1**	**0**	**0**
x_3	**1**	**1**	**0**

What About This Function?

	y_1	y_2	y_3
x_1	**0**	**1**	**1**
x_2	**1**	**0**	**0**
x_3	**1**	**1**	**0**

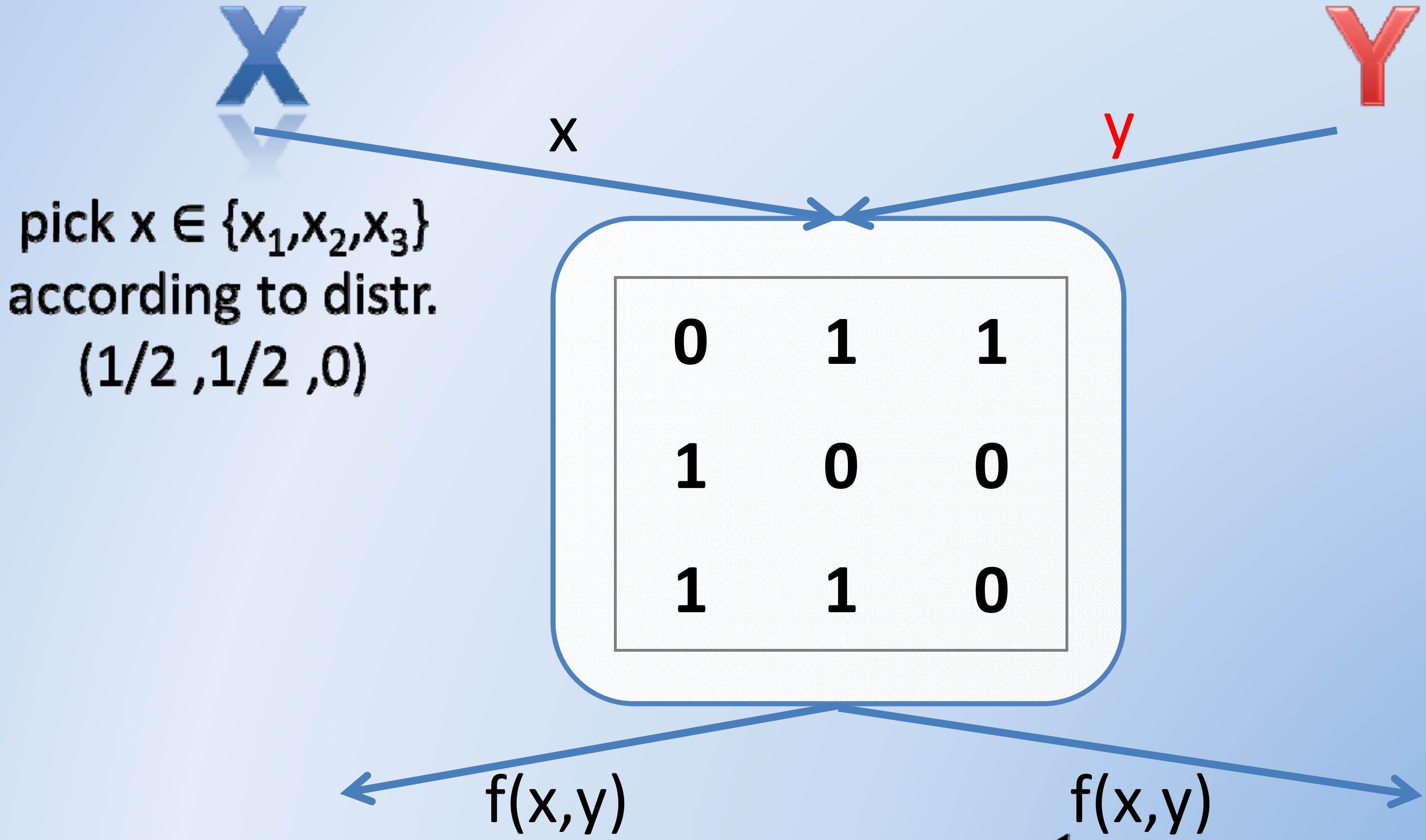
What About This Function?

		y_1	y_2	y_3
1/2	x_1	0	1	1
1/2	x_2	1	0	0
0	x_3	1	1	0

What About This Function?

		y_1	y_2	y_3
1/2	x_1	**0**	**1**	**1**
1/2	x_2	**1**	**0**	**0**
0	x_3	1	1	0
		1/2	1/2	1/2

The Protocol – Malicious Y



$$\Pr[f(x,y) = 1] = \frac{1}{2}$$

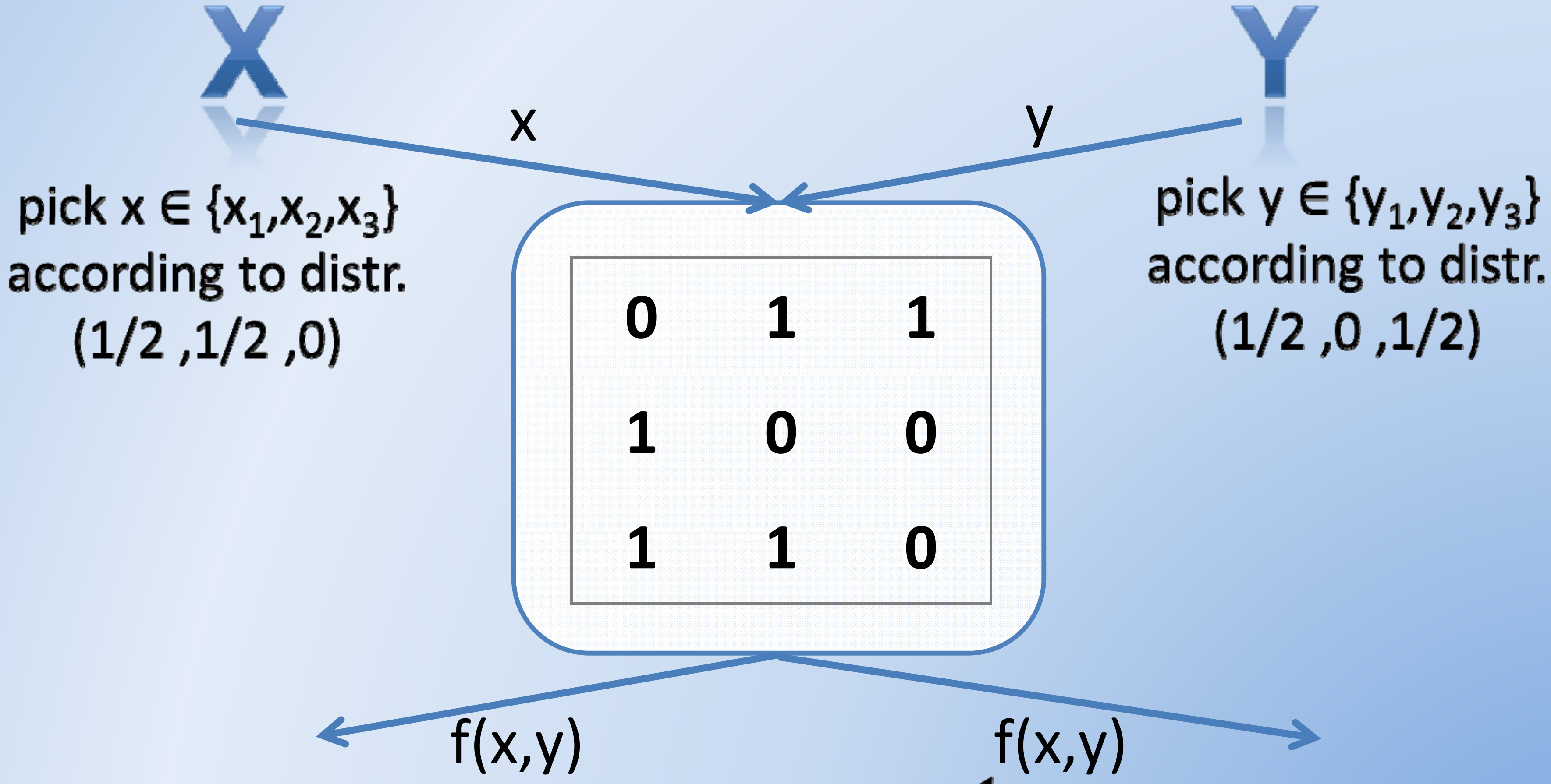
What About Party Y?

	y_1	y_2	y_3
x_1	**0**	**1**	**1**
x_2	**1**	**0**	**0**
x_3	**1**	**1**	**0**

What About Party Y?

		y_1	y_2	y_3
		1/2	0	1/2
1/2	x_1	0	1	1
1/2	x_2	1	0	0
1/2	x_3	1	1	0

The Overall Protocol



$$\Pr[f(x,y) = 1] = \frac{1}{2}$$

Another Point of View

	y_1	y_2	y_3
x_1	**0**	**1**	**1**
x_2	**1**	**0**	**0**
x_3	**1**	**1**	**0**

Another Point of View

$$\begin{pmatrix} p_1 & p_2 & p_3 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix}$$

distribution over
the inputs of **X**

distribution over
the inputs of **Y**

$$= \Pr[output = 1]$$

Another Point of View

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \frac{1}{2}$$

$$\begin{pmatrix} p_1 & p_2 & p_3 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1/2 \\ 0 \\ 1/2 \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & p_3 \end{pmatrix} \begin{pmatrix} 1/2 \\ 1/2 \\ 1/2 \end{pmatrix} = \frac{1}{2}$$

Generalizing the Above: Definitions

f is δ_1-left balanced

if there exists a probability vector $\boldsymbol{p} = (p_1, \dots, p_m)$, $0 \le \delta_1 \le 1$
such that: $\boldsymbol{p} \cdot M_f = \delta_1 \cdot \mathbf{1}_\ell$

f is δ_2-right balanced

if there exists a probability vector $\boldsymbol{q} = (q_1, \dots, q_\ell)$, $0 \le \delta_2 \le 1$
such that: $M_f \cdot \boldsymbol{q}^T = \delta_2 \cdot \mathbf{1}_m^T$

f is δ-balanced

if f is δ-left balanced and δ-right balanced

Our Main Theorem

- If f is δ-balanced for some $0 < \delta < 1$, then it **implies** coin-tossing
- If f is not δ-balanced *for any* $0 < \delta < 1$, then it **does not imply** coin-tossing*

Implying Coin-Tossing

Theorem

If f is δ-balanced for some $0 < \delta < 1$, then it **implies** coin-tossing

Proof:

f is $\boldsymbol{\delta}$-balanced $\Rightarrow$ coin tossing for δ-coin

Apply **von-Neumann**'s method to toss a fair-coin

The Impossibility Result

Theorem

If f is not δ-balanced for any $0 < \delta < 1$, then it **does not imply** coin tossing*

- We show that there does not exist a fair coin-tossing protocol in the f-hybrid model
  - For any coin-tossing protocol in the f-hybrid model, there exists an (inefficient) adversary that can bias the result
- Unlike Cleve – the parties have some **simultaneous exchange**. Thus, a completely different argument is needed

Impossibility in the OT-hybrid model

- The adversary is *inefficient*
  - It computes the distributions over all possible random coins of an honest **X**
  - This computation can be approximated given an $\mathcal{NP}$-oracle
- We do not know how to construct an *efficient* adversary
- Impossibility still holds if the parties have an ideal OT
  - Embedded OR implies OT [Kilian 91]
  - A function that doesn't contain an embedded OR is 1/2-balanced

Impossibility of a Single Invocation Non-Left-Balanced Function

if f is δ-left balanced and δ-right balanced

if there exists a probability vector $\boldsymbol{p} = (p_1, \ldots, p_m)$, $0 \le \delta_1 \le$ such that: $\boldsymbol{p} \cdot M_f = \delta_1 \cdot \mathbf{1}_\ell$

f is δ_1-left balance

if there exists a probability vector $\boldsymbol{q} = (q_1, \ldots, q_\ell)$, $0 \le \delta_2 \le 1$ such that: $M_f \cdot \boldsymbol{q}^T = \delta_2 \cdot \mathbf{1}_m^T$

f is δ_2-right balanced s.t. f is δ-balanced

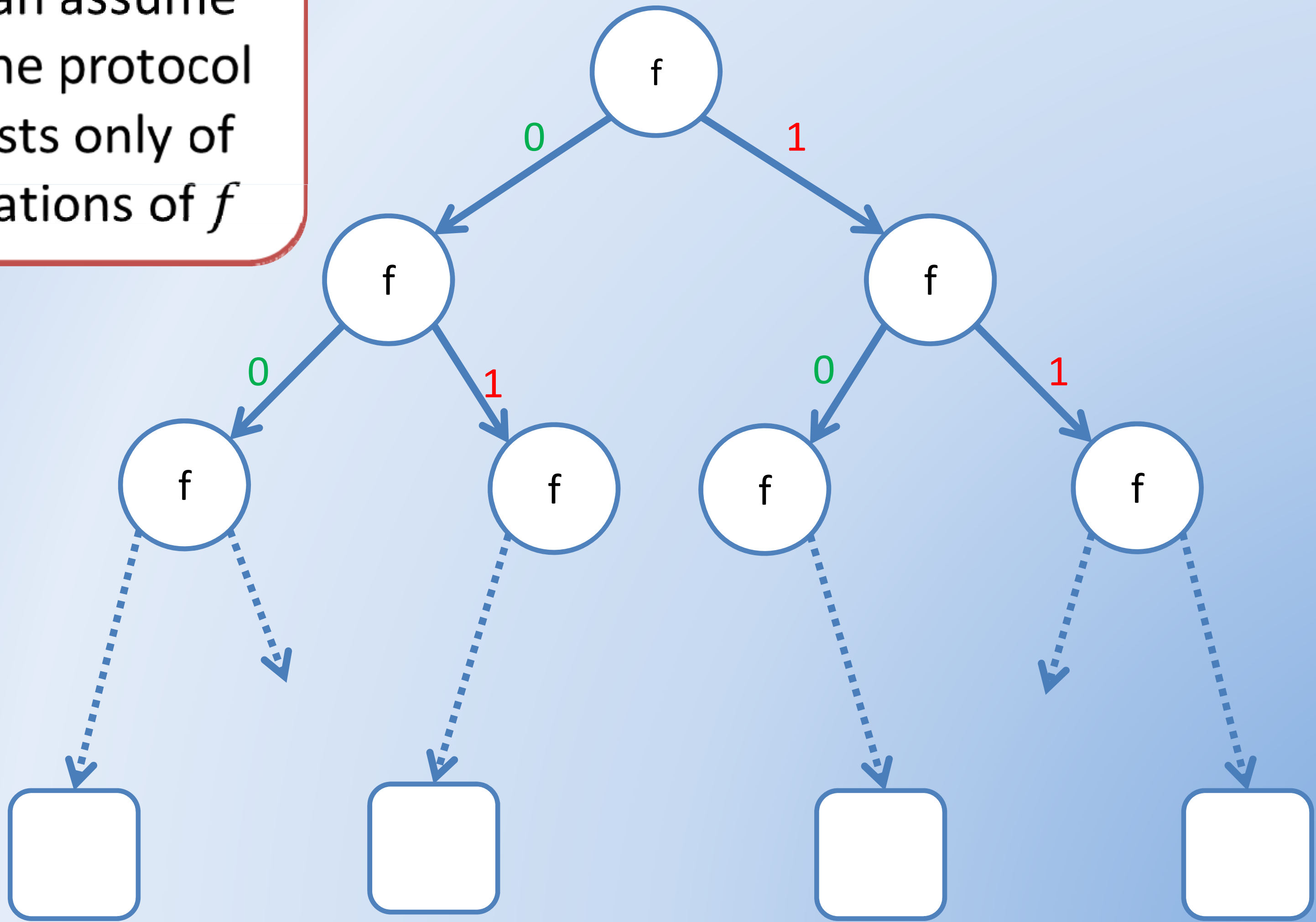
$$\boldsymbol{p} \cdot M_f \cdot \boldsymbol{e}_i^T = \delta_i$$

$$\boldsymbol{p} \cdot M_f \cdot \boldsymbol{e}_j^T = \delta_j$$

A malicious **Y** can always bias the probability to get 1 in a single invocation!

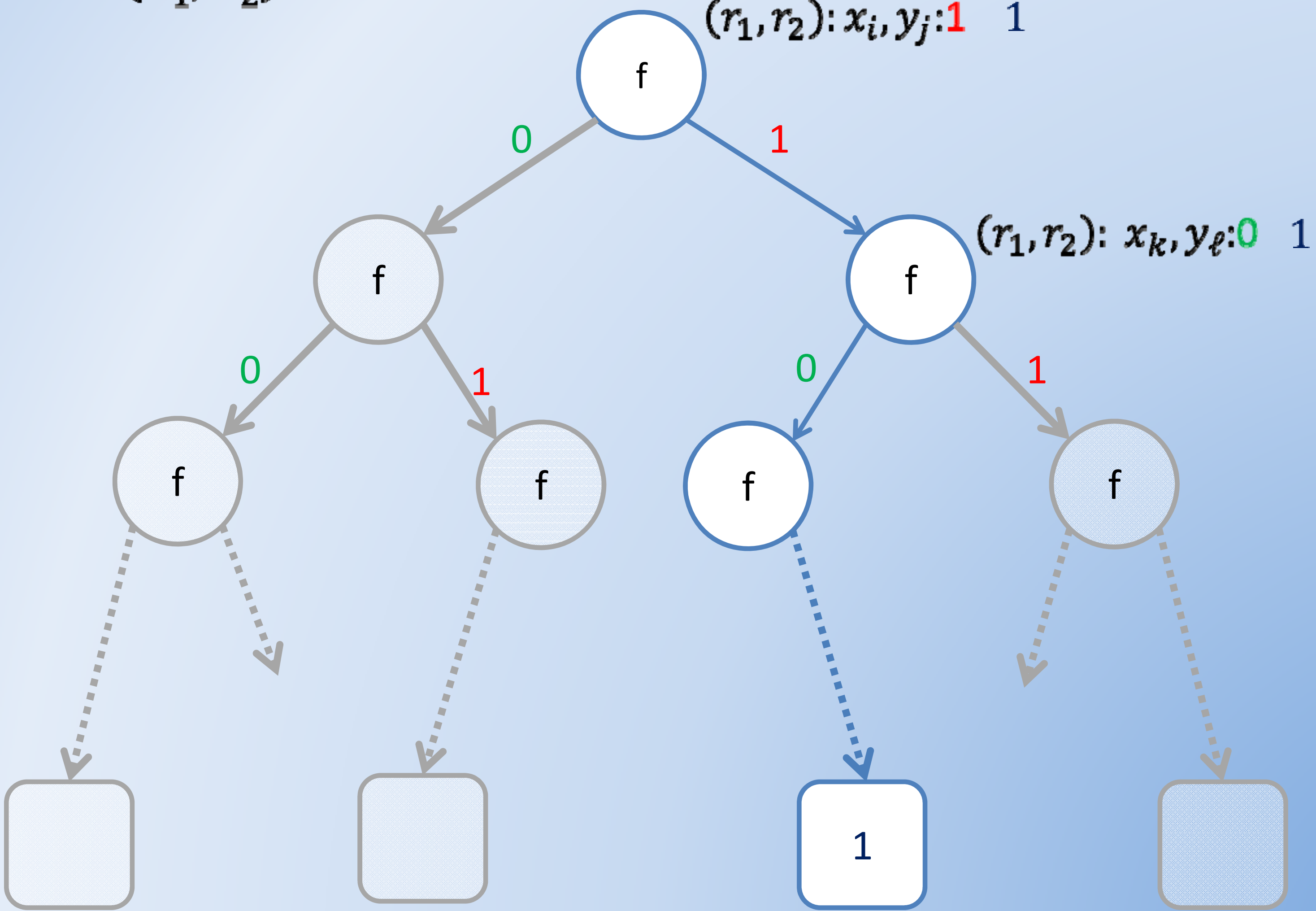
The Protocol Transcript Tree

We can assume that the protocol consists only of invocations of f

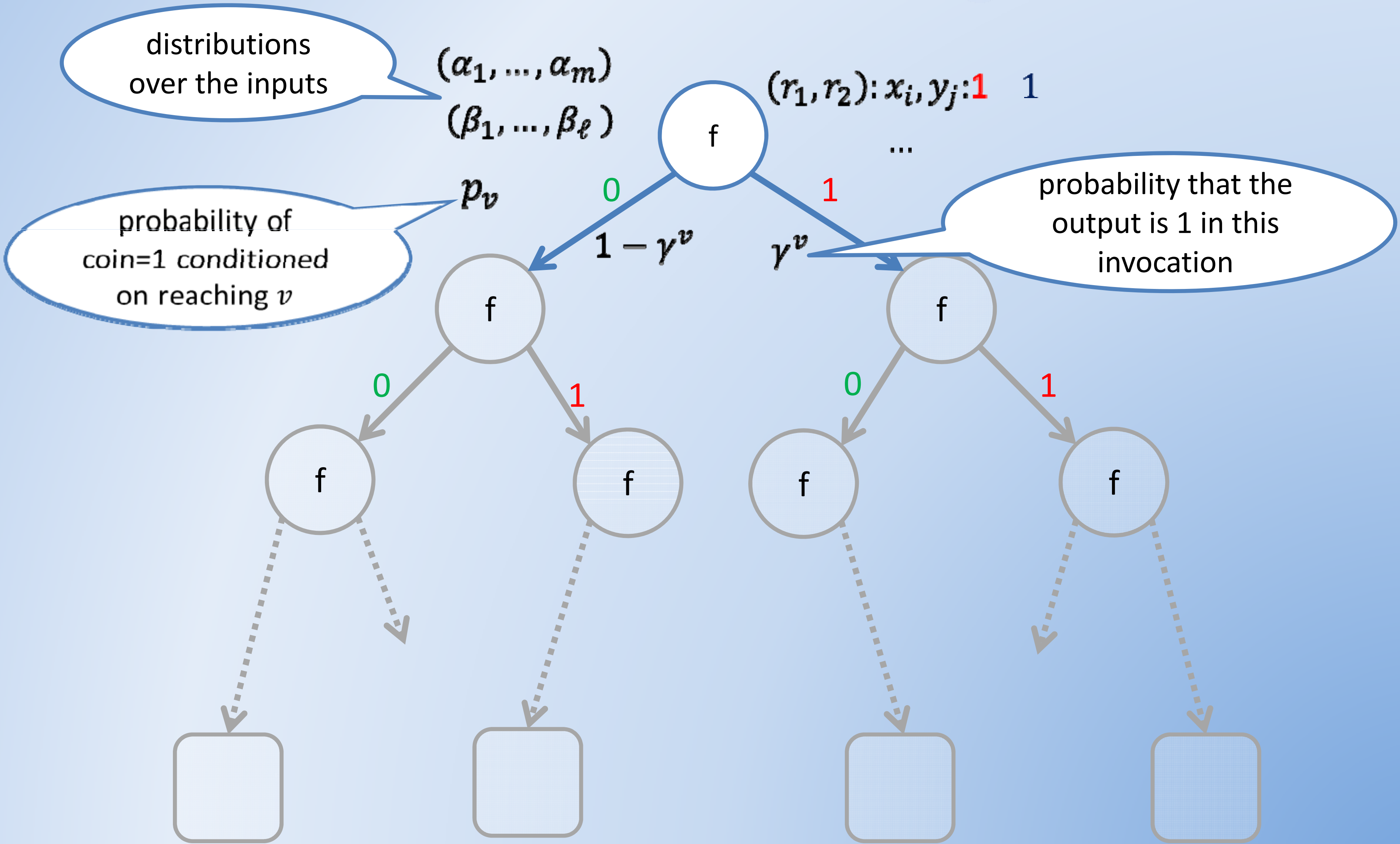


The Protocol Transcript Tree

(r_1, r_2)

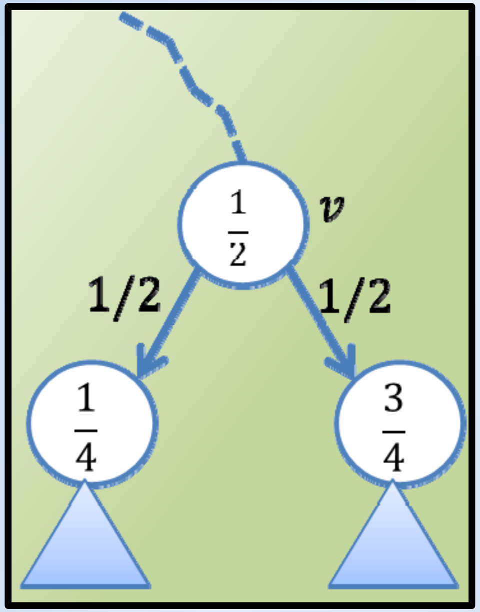


The Protocol Transcript Tree

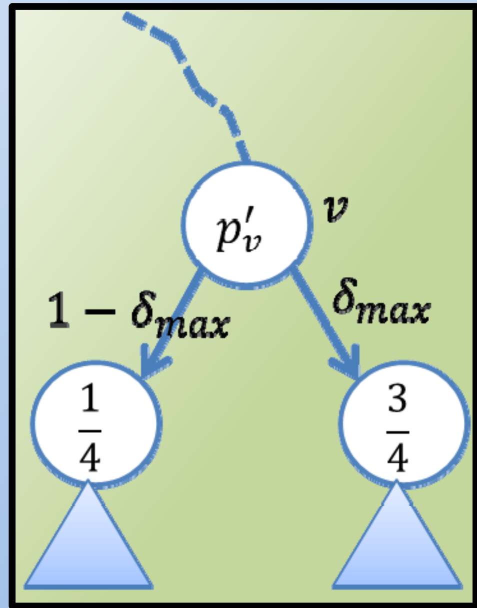


Attacking the Protocol

The adversary acts honestly but searches for a "jump" between probabilities in parent and children



instead using $\boldsymbol{\beta^v}$, use $\boldsymbol{e_i}$ or $\boldsymbol{e_j}$



We show that in any execution, such a "jump" exists

In The Paper

- We also study the case of a ***fail-stop*** adversary
  - Follows the protocol specifications but may abort prematurely
- Unclear how to model fail-stop in the ideal world
  - Is the simulator allowed to change the corrupted party's input?
- We consider two possible definitions and study fairness in *both* cases

Conclusion

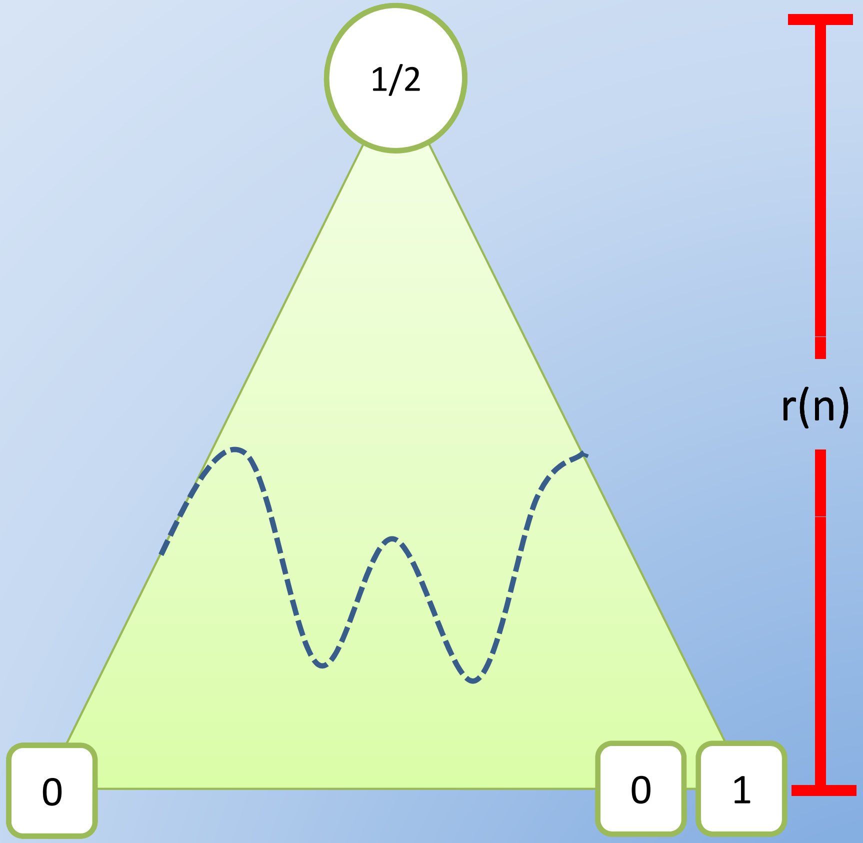
- We give a simple property (a criterion) s.t.:
  - If the function **satisfies** the property, it **implies** coin-tossing
  - If the function **does not satisfy** the property, it **does not imply** coin-tossing
- We consider the same question for **fail-stop** adversary
- This is an important step forward towards understanding fair secure computation

Thank You!!

An Execution

Every path (execution) from root to leaf has such a "jump":

- The probability in the root is 1/2
- The probability in each leaf is either 0 or 1



End of Proof Sketch