On the Fairness of Finite Boolean Functions

Nikolaos Makriyannis
Departament de Tecnologies de la Informació i les Comunicacions
Universitat Pompeu Fabra
Email: nikolaos.makriyannis@upf.edu

***Abstract*—Let f be a finite two-input Boolean function. Assume that two parties, P_1 and P_2, holding inputs x and y respectively, wish to compute $f(x, y)$ by means of a secure two-party protocol. We say that the parties compute f *with complete fairness* (or that f is *fair*) if, whenever one of the parties learns the desired output, then both of them do.**

In [1], Cleve showed that in the presence of an adversary, it is impossible for two parties to agree on a random bit, provided that the output of each party is always well defined. The implication is that completely fair coin-tossing protocols do not exist in the presence of a dishonest majority and, furthermore, functions that can be used for coin-tossing (like XOR) are not computable with complete fairness. Since then there has been limited investigation on the subject. In [2] however, the authors show that complete fairness is in fact possible for certain functions. In particular, they design a protocol that computes Yao's millionaire problem with complete fairness and they even show that there exist functions with embedded XORs that are computable with complete fairness.

In this paper, we generalize Cleve's original result and provide a new infinite family of finite two-input Boolean functions that are not computable with complete fairness.

I. Introduction

The concept of fairness goes back to the early days of modern cryptography. It first appears in relation to digital signature exchange circa 1980 [3]. Quickly, researchers were investigating fairness (and related notions) in the more general two/multi-party computation framework. Regarding *complete fairness*, Cleve's paper [1], along with the negative result it implies, seemed to have discouraged the cryptographic community from further exploring the subject. The first positive result appears two decades later, when the authors of [2] show that complete fairness can in fact be achieved for certain functions.

In this paper we show a new negative result. Namely, we will be proving the following claim.

Claim 1: Let $f : X \times Y \to \{0, 1\}$ be a finite function such that for some $p \in (0, 1)$, for all $(x, y) \in X \times Y$,

$$\Pr[f(x, \cdot) = 0] = \Pr[f(\cdot, y) = 0] = p.$$

Then, in the presence of a dishonest majority, f is *not* computable with complete fairness.

In fact, the claim follows by generalising Cleve's result. We show that two parties cannot agree on a single bit no matter what the desired probability distribution of that bit is. To this end, our work is organized in the following way:

- Section II introduces notation and necessary definitions.
- In Section III we generalize Cleve's result for coin-tossing by proving that general bit selection schemes are not fair.
- In Section IV we deduce that finite Boolean functions that can be used for bit selection schemes are not fair.
- Section V provides some examples of functions that are not computable with complete fairness.
- The final section contains a conclusion and a brief discussion about future work.

II. Preliminaries

We begin with some notation and definitions. At the end of this section we introduce the concept of fairness using the standard ideal/real model approach from [4]. Note that most of the material in this section can be found in [2].

Let $n \in \mathbf{N}$ denote the security parameter. A function $\mu(\cdot)$ is *negligible* if it vanishes faster than any (positive) inverse-polynomial. A distribution ensemble $X = \{X(a, n)\}_{a \in \mathcal{D}_n, n \in \mathbf{N}}$ is an infinite sequence of random variables indexed by $\mathcal{D}_n$ and n. Two distribution ensembles, X and Y, are *computationally indistinguishable* if for every non-uniform polynomial-time algorithm D, there exists a negligible function μ 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 $X \stackrel{c}{\equiv} Y$ when the ensembles are computationally indistinguishable and $X \stackrel{s}{\equiv} Y$ when they are statistically close.

A *two-party functionality* $\mathcal{F} = \{f_n\}_{n \in \mathbf{N}}$, is a sequence of random processes such that each f_n maps pairs of inputs to pairs of random variables (one for each party). The domain of f_n is denoted $X_n \times Y_n$ and the output (f_n^1, f_n^2). A *two-party protocol* Π for computing a functionality $\mathcal{F}$, is a polynomial-time protocol such that on inputs $x \in X_n$ and $y \in Y_n$, the joint distribution of the outputs of any honest execution of Π is statistically close to $f_n(x, y) = (f_n^1(x, y), f_n^2(x, y))$.

In particular, suppose that $f_n^1(x, y) = f_n^2(x, y) = \ell$, where ℓ is a single bit, and let a and b denote the output from an execution of Π, of P_1 and P_2 respectively. Motivated by the

definitions of ϵ-*consistency* and *bias* in [1], for fixed $p \in (0,1)$, define the following:

Definition 1: We say that protocol Π is p-*consistent* if there exists $\epsilon > 0$ such that $\Pr[a = b] \geq \max\{p, 1-p\} + \epsilon$.

Definition 2: Define the p-*bias towards* 0 of ℓ to be $\Pr[\ell = 0] - p$. Similarly, define the p-*bias towards* 1 of ℓ to be $\Pr[\ell = 1] - 1 + p$. Let the p-*bias* of ℓ be the modulus of any of the two aforementioned values.

Let $\mathcal{F} = \{f_n\}_{n \in \mathbf{N}}$ be a two-party functionality. Assume there are two parties, P_1 and P_2, holding the same value 1^n, as well as (private) inputs $x \in X_n$ and $y \in Y_n$ respectively. Furthermore, assume there exists an adversary $\mathcal{S}$ that corrupts one of the parties. Computing $\mathcal{F}$ in the *ideal model* amounts to the following procedure:

1) P_1 holds 1^n and x, P_2 holds 1^n and y, adversary $\mathcal{S}$ receives an auxiliary input z.
2) The honest party sends its input to a trusted party $\mathcal{T}$ whereas the corrupted party sends any value of $\mathcal{S}$'s choice. Write (x', y') for the pair sent to $\mathcal{T}$. (We assume that communication between $\mathcal{T}$ and the parties is private.)
3) If $x' \notin X_n$ or $y' \notin Y_n$, then the trusted party reassigns the corresponding value to some default valid input and sends $f_n^1(x', y'; r)$ to P_1 and $f_n^2(x', y'; r)$ to P_2, where r is chosen uniformly at random.
4) The honest party outputs whatever $\mathcal{T}$ sent him, the corrupted party outputs nothing and $\mathcal{S}$ outputs an arbitrary (probabilistic polynomial-time computable) function of its view.

Define $\mathsf{VIEW}^{\mathsf{ideal}}_{\mathcal{F},\mathcal{S}(z)}(x,y,n)$ and $\mathsf{OUT}^{\mathsf{ideal}}_{\mathcal{F},\mathcal{S}(z)}(x,y,n)$ to be the random variables consisting of the *view* of adversary $\mathcal{S}$ and the output of the honest party, respectively. Similarly, for any two-party protocol Π for computing $\mathcal{F}$, let $\mathcal{A}$ be an adversary in the real model corrupting one of the parties and define $\mathsf{VIEW}^{\mathsf{real}}_{\Pi,\mathcal{A}(z)}(x,y,n)$ and $\mathsf{OUT}^{\mathsf{real}}_{\Pi,\mathcal{A}(z)}(x,y,n)$ to be the random variables consisting of the *view* of adversary $\mathcal{A}$ and the output of the honest party, respectively. Finally, let $\Omega^{\mathsf{real}}_{\Pi,\mathcal{A}(z)}(x,y,n)$ and $\Omega^{\mathsf{ideal}}_{\mathcal{F},\mathcal{S}(z)}(x,y,n)$ denote the random variables consisting of the adversary's view and the honest party's output in the real and ideal model respectively.

Definition 3: Using the notation above, we say that Π *computes* $\mathcal{F}$ *with complete fairness* if for every non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ in the real model, there exists a non-uniform probabilistic polynomial-time adversary $\mathcal{S}$ in the ideal model such that:

$$\left\{\Omega^{\mathsf{real}}_{\Pi,\mathcal{A}(z)}(x,y,n)\right\}_{x,y,z,n} \stackrel{c}{\equiv} \left\{\Omega^{\mathsf{ideal}}_{\mathcal{F},\mathcal{S}(z)}(x,y,n)\right\}_{x,y,z,n}. \quad (1)$$

To give some intuition on the above definition, note that the ideal model corresponds to the best case scenario in terms of privacy and security. Thus, informally, what equation (1) actually says is that the adversary has access to the same information in both worlds and, furthermore, the adversary's influence on the honest party's output is the same in both worlds. In other words, an execution of a protocol that computes a functionality with complete fairness, essentially amounts to computing the functionality by means of a trusted party in the ideal model.

III. Fairness of Bit Selection Schemes

In this section, we prove that *bit selection schemes* are not computable with complete fairness. In particular, using the protocol of [1], we show that if two parties wish to compute a (common) single bit according to some predetermined probability distribution, then there exists an adversary that can influence the honest party's output in a way that cannot be reproduced in the ideal model. Note that Cleve's original result is a special case of what follows with $p = 1/2$.

Formally, suppose that parties P_1 and P_2 wish to compute a single bit ℓ such that $\Pr[\ell = 0] = p \in (0,1)$. To this end, they execute Π, an $r(n)$-round protocol, where $r(n)$ is bounded by some polynomial. Without loss of generality, we describe Π as follows:

- Before the protocol starts, P_1 and P_2 compute backup bits a_1 and b_0, respectively.
- At round i, party P_1 sends a message (string of bits) to P_2 who computes a backup bit b_i. Party P_2 then sends a message to P_1 who computes a backup bit a_{i+1}.
- At the end of the protocol, or if communication fails, parties output the last bit they successfully constructed.

We refer to such a protocol as a *bit selection scheme*.

Theorem 1: Suppose that Π is a p-consistent two-party bit selection scheme. Then there exists an adversary with a strategy such that the p-bias of the honest party's output is at least

$$\frac{\epsilon}{4r(n)+1},$$

for some $\epsilon > 0$.

Proof: We define $4r(n)+1$ adversaries

$$\begin{array}{cccccccc} \overline{\mathcal{A}}, & \mathcal{A}^1_{1,0}, & \ldots & \mathcal{A}^1_{r(n),0}, & \mathcal{A}^1_{1,1}, & \ldots & \mathcal{A}^1_{r(n),1}, \\ & \mathcal{A}^2_{1,0}, & \ldots & \mathcal{A}^2_{r(n),0}, & \mathcal{A}^2_{1,1}, & \ldots & \mathcal{A}^2_{r(n),1} \end{array}$$

with the following quitting strategies:

- Adversary $\overline{\mathcal{A}}$ instructs P_1 to quit immediately (no information is exchanged).
- Adversary $\mathcal{A}^j_{i,\ell}$ instructs P_j to proceed normally until round $i-1$. At round i, if the corrupted party's backup bit (a_i or b_i) is equal to ℓ then proceed to the next round and quit. Otherwise quit at round i.

Next, let $\overline{\delta}$ and $\delta^j_{i,\ell}$ denote the p-bias of the honest party's output under the attack of $\overline{\mathcal{A}}$ and $\mathcal{A}^j_{i,\ell}$, respectively. From the definition of p-bias, we deduce the following lower bounds:

$$\begin{aligned} \overline{\delta} &\geq \max\{\Pr[b_0 = 0], \Pr[b_0 = 1]\} - \max\{p, 1-p\}, \\ \delta^1_{i,0} &\geq \Pr[a_i = 0 \wedge b_i = 0] + \Pr[a_i = 1 \wedge b_{i-1} = 0] - p, \\ \delta^1_{i,1} &\geq \Pr[a_i = 1 \wedge b_i = 1] + \Pr[a_i = 0 \wedge b_{i-1} = 1] - 1 + p, \\ \delta^2_{i,0} &\geq \Pr[b_i = 0 \wedge a_{i+1} = 0] + \Pr[b_i = 1 \wedge a_i = 0] - p, \\ \delta^2_{i,1} &\geq \Pr[b_i = 1 \wedge a_{i+1} = 1] + \Pr[b_i = 0 \wedge a_i = 1] - 1 + p. \end{aligned}$$

Let Δ denote the average of the above values. After simplification, we deduce that

$$(4r(n)+1)\Delta \geq \overline{\delta} + \Pr[b_0 \neq a_1] + \Pr[b_{r(n)} = a_{r(n)+1}] - 1. \tag{2}$$

Knowing that a_1 and b_0 are independent variables (no information is exchanged), notice that

$$\begin{aligned}\Pr[b_0 \neq a_1] &= 1 - \Pr[b_0 = a_1] \\ &= 1 - \Pr[a_1 = 0]\Pr[b_0 = 0] \\ &\qquad - \Pr[a_1 = 1]\Pr[b_0 = 1] \\ &\geq 1 - \max\{\Pr[b_0 = 0], \Pr[b_0 = 1]\}.\end{aligned}$$

Plug the above expression into equation (2), use the p-consistency of Π and deduce that there exists $\epsilon > 0$ such that

$$\Delta \geq \frac{\epsilon}{4r(n)+1}.$$

■

We are now going to show how the previous theorem relates to the notion of fairness. From now on the security parameter as well as auxiliary inputs will be implicit in the discussion. Consider the following *ideal* protocol:

1) The honest party sends an arbitrary value to trusted party $\mathcal{T}$. The corrupted party sends a value of adversary $\mathcal{A}$'s choice.
2) After receiving the messages, $\mathcal{T}$ computes a bit ℓ such that $\Pr[\ell = 0] = p$ and sends ℓ to both parties.
3) The honest party outputs ℓ, the corrupted party outputs nothing and $\mathcal{A}$ outputs an arbitrary (probabilistic polynomial-time computable) function of its view.

We see that no matter what the adversary does, the p-bias of ℓ is always 0. Hence, by Theorem 1, for any p-consistent real-world protocol that emulates the above scheme, there exists a quitting strategy such that the p-bias of the honest party's output is non-negligible. We conclude that bit selection schemes are *not* fair.

Now, let $f : X \times Y \to \{0,1\}$ be a finite function with the following property: For all $(x,y) \in X \times Y$,

$$\Pr[f(x,\cdot) = 0] = \Pr[f(\cdot,y) = 0] = p. \tag{3}$$

For a function f satisfying equation (3), define the *ideal** *model for computing* f to consist in the following procedure:

1) The honest party sends a value chosen uniformly at random from its input domain. The corrupted party sends a value of adversary $\mathcal{A}$'s choice.
2) After receiving the inputs, say x' and y' (if the values are not in the appropriate domains reassign), $\mathcal{T}$ sends $f(x',y')$ to both parties.
3) The honest party outputs whatever $\mathcal{T}$ sent him, the corrupted party outputs nothing and $\mathcal{A}$ outputs an arbitrary (probabilistic polynomial-time computable) function of its view.

Assume P_1 is honest (the case where P_2 is honest is identical) and denote a its output. Then

$$\begin{aligned}\Pr[a = 0] &= \Pr[f(x,y) = 0] \\ &= \sum_{z \in X} \Pr[f(z,y) = 0] \cdot \Pr[x = z] \\ &= \sum_{z \in X} \frac{p}{|X|} = p.\end{aligned}$$

Once again, the p-bias of the honest party's output is trivial. Noting that the *ideal** model for computing f is a variation of the ideal model for bit selection schemes, we deduce that, provided the honest party chooses his input randomly, functions that satisfy expression (3) are *not* computable with complete fairness.

IV. Fairness of Finite Boolean Functions

In this section we show that, as functionalities, finite two-input Boolean functions satisfying equation (3) are *not* computable with complete fairness i.e. even when the parties' inputs are predetermined. Assume the contrary and let Π be a two-party protocol for computing f with complete fairness. In other words, for every $(x,y) \in X \times Y$, for every adversary $\mathcal{A}$ in the real model, there exists an adversary $\mathcal{S}$ in the ideal model such that

$$\left\{\Omega^{\mathsf{real}}_{\Pi,\mathcal{A}}(x,y)\right\}_{x,y} \overset{c}{\equiv} \left\{\Omega^{\mathsf{ideal}}_{f,\mathcal{S}}(x,y)\right\}_{x,y},$$

where the Ω's are the probability distributions from Definition 3. Define *real* protocol Π^* to be the same as Π except that before engaging into any kind of computation or communication, the parties are instructed to choose an input uniformly at random from their input domains. We claim that Π^* is now a *completely fair bit selection scheme*.

Indeed, let $\mathcal{A}$ and $\mathcal{S}$ denote adversaries in the real and ideal* model respectively. Furthermore, let $\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}$ and $\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}$ denote the random variables consisting of the adversary's view and the honest party's output in the real and ideal* model respectively. We write $\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}(x,y)$ and $\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}(x,y)$ to denote that P_1 and P_2 chose inputs x and y respectively. Finally, let D be a non-uniform polynomial-time algorithm. Then

$$\begin{aligned}&|\Pr[D(\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}) = 1] - \Pr[D(\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}) = 1]| \\ &= \Big| \sum_{(x,y)\in X\times Y} \Pr[(x,y)]\big(\Pr[D(\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}(x,y)) = 1] \\ &\qquad - \Pr[D(\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}(x,y)) = 1]\big)\Big| \\ &\leq \sum_{(x,y)\in X\times Y} \frac{1}{|X|\cdot|Y|}\Big|\Pr[D(\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}(x,y)) = 1] \\ &\qquad - \Pr[D(\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}(x,y)) = 1]\Big| \end{aligned} \tag{4}$$

Since we assume that P_1 and P_2 hold inputs x and y respectively, variables $\Omega^{\mathsf{real}}_{\Pi^*,\mathcal{A}}(x,y)$ and $\Omega^{\mathsf{ideal}^*}_{f,\mathcal{S}}(x,y)$ boil down to $\Omega^{\mathsf{real}}_{\Pi,\mathcal{A}}(x,y)$ and $\Omega^{\mathsf{ideal}}_{f,\mathcal{S}}(x,y)$. Consequently, using the assumption that Π is fair, there exists an adversary $\mathcal{S}$ in the ideal model such that the expression

$$|\Pr[D(\Omega^{\mathsf{real}}_{\Pi,\mathcal{A}}(x,y)) = 1] - \Pr[D(\Omega^{\mathsf{ideal}}_{f,\mathcal{S}}(x,y)) = 1]|$$

is upper-bounded by a negligible function. Thus, after simplification, expression (4) admits the same bound. Hence Π^* is a completely fair bit selection scheme and the claim is now proven.

Theorem 2: Let $f : X \times Y \to \{0,1\}$ be a finite function such that for some $p \in (0,1)$, for all $(x,y) \in X \times Y$,

$$\Pr[f(x,\cdot) = 0] = \Pr[f(\cdot,y) = 0] = p.$$

Then f is *not* computable with complete fairness.

V. Examples

We provide a few examples of functions that satisfy the hypothesis of the previous theorem. First, a small lemma.

Lemma 1: Let $f : X \times Y \to \{0,1\}$ be a finite function such that for all $(x,y) \in X \times Y$,

$$\Pr[f(x,\cdot) = 0] = p$$
$$\Pr[f(\cdot,y) = 0] = q$$

Then

$$p = q.$$

Proof:

$$\begin{aligned}\Pr[f(x,y) = 0] &= \sum_{z \in X} \Pr[f(z,y) = 0]\Pr[x = z] \\ &= \sum_{z \in X} \frac{p}{|X|} = p, \\ &= \sum_{z \in Y} \Pr[f(x,z) = 0]\Pr[y = z] \\ &= \sum_{z \in Y} \frac{q}{|Y|} = q.\end{aligned}$$

■

Without loss of generality, suppose that $X = \{1,\ldots,k_X\}$ and $Y = \{1,\ldots,k_Y\}$. We describe f as a $k_X \times k_Y$ logical matrix, where entry (i,j) corresponds to value $f(i,j)$. Using Lemma 1 and Theorem 2, we deduce that if the rows (resp. columns) of a given Boolean matrix have equal weight, then the associated function is *not* computable with complete fairness. In particular, the XOR function which has matrix representation

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

is not computable with complete fairness. More generally, any function whose associated matrix is a permutation matrix is not computable with complete fairness. Indeed, if P is a $k \times k$ permutation matrix, then $\Pr[P_{i,\cdot} = 1] = \Pr[P_{\cdot,j} = 1] = 1/k$. Furthermore, the translation of any permutation matrix by the all-1 matrix yields a matrix P' with the following property:

$$\Pr[P'_{i,\cdot} = 1] = \Pr[P'_{\cdot,j} = 1] = \frac{k-1}{k},$$

hence functions associated with such matrices are also not computable with complete fairness. Finally, we provide a family of matrices whose associated functions can be used as coin-tossing functionalities.

Let $k = 2k'$, where k' is odd, and consider vectors v_1, $v_2 \in \mathbf{F}_2^k$ such that

$$v_1(i) = \begin{cases} 1 & \text{if } i \leq k', \\ 0 & \text{otherwise.} \end{cases}$$

$$v_2(i) = \begin{cases} 1 & \text{if } i \leq k'-1 \text{ or } i = k'+1, \\ 0 & \text{otherwise.} \end{cases}$$

Furthermore, let $v_j^{(s)}$ be the vector obtained by shifting v_j by two positions downwards, s times. In particular, after one shift, entry i of $v_j = v_j^{(0)}$ becomes entry $i+2$ modulo $2k'$ of $v_j^{(1)}$. Finally, define $C_k \in \mathbf{F}_2^{k \times k}$ to be the matrix whose columns are alternately $v_1^{(s)}$ and $v_2^{(s)}$ starting with $s = 0$ and ending at $s = k'-1$. In particular, for $k = 6$, we have

$$C_6 = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

It is not hard to see that since k' is odd, the columns/rows of C_k have weight equal to $k' = k/2$. Thus, the associated functions can be used as coin-tossing functionalities and hence, are *not* computable with complete fairness.

VI. Conclusion and Future Work

In this paper, we showed that *finite* functions $f : X \times Y \to \{0,1\}$ satisfying

$$\forall (x,y) \in X \times Y, \quad \Pr[f(x,\cdot) = 0] = \Pr[f(\cdot,y) = 0] = p,$$

for some $p \in (0,1)$, are *not* computable with complete fairness. We generalized Cleve's original result and we have thus shown that, in the presence of a dishonest majority, there are many more finite two-input Boolean functions which are not fair. Previously, the theorem was only known to be true for $p = 1/2$.

The ultimate goal would be to classify all finite-two input Boolean functions with respect to fairness i.e. find a necessary and sufficient condition for a given function to be computable with complete fairness.

In [2], the authors show that if each party has exactly two possible inputs, then there are essentially two possible functions (up to fairness). These are the XOR operator and the OR operator. They show that OR is fair and since XOR is not, they obtain a classification of all such functions up to fairness. Interestingly, using our negative result and the positive result of [2], we can also classify up to fairness all-but-one functions where each of the parties have exactly three possible inputs.

Still, we are very far from classifying all finite two-input Boolean functions, let alone multiple-input non-deterministic functionalities in general. Future steps towards this goal include:

1) Designing new simulation strategies and showing that the protocol of [2] can be used to compute more functions with complete fairness.

2) Designing new two-party protocols for computing finite Boolean functions and proving fairness under the real/ideal model.
3) Extending the impossibility result of the present paper to more functions.

It would also be interesting to explore these questions by considering additional notions like partial fairness (see [5]) or rational fairness (see [6], [7]).

REFERENCES

[1] R. Cleve, "Limits on the security of coin flips when half the processors are faulty (extended abstract)," in *STOC*, J. Hartmanis, Ed. ACM, 1986, pp. 364–369.

[2] S. D. Gordon, C. Hazay, J. Katz, and Y. Lindell, "Complete fairness in secure two-party computation," *J. ACM*, vol. 58, no. 6, p. 24, 2011.

[3] S. Even and Y. Yacobi, "Relations among public key signature systems," Technion Israel Institute of Technology, Computer Science Department, Tech. Rep. 175, 1980.

[4] O. Goldreich, *Foundations of Cryptography: Basic Applications*, ser. Foundations of Cryptography. Cambridge University Press, 2004.

[5] S. D. Gordon and J. Katz, "Partial fairness in secure two-party computation," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, H. Gilbert, Ed., vol. 6110. Springer, 2010, pp. 157–176.

[6] A. Groce and J. Katz, "Fair computation with rational players," Cryptology ePrint Archive, Report 2011/396, 2011, http://eprint.iacr.org/.

[7] G. Asharov, R. Canetti, and C. Hazay, "Towards a game theoretic view of secure computation," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed., vol. 6632. Springer, 2011, pp. 426–445.

[8] R. Canetti, "Security and composition of multi-party cryptographic protocols," *JOURNAL OF CRYPTOLOGY*, vol. 13, p. 2000, 1998.