1

Cryptography and Game Theory

Yevgeniy Dodis, NYU and Tal Rabin, IBM

Abstract

The Cryptographic and Game Theory worlds seem to have an intersection in that they both deal with an interaction between mutually distrustful parties which has some end result. In the cryptographic setting the multiparty interaction takes the shape of a set of parties communicating for the purpose of evaluating a function on their inputs, where each party receives at the end some output of the computation. In the game theoretic setting parties interact in a game which guarantees some payoff for the participants according to their joint actions of all the parties, while the parties wish to maximize their own payoff. In the past few years the relationship between these two areas has been investigated with the hope of having cross fertilization and synergy. In this chapter we describe the two areas, the similarities and differences, and some of the new results stemming from their interaction.

The first and second section will describe the cryptographic and the game theory settings (respectively). In the third section we contrast the two settings, and in the last sections we detail some of the existing results.

1.1 Cryptographic Notions and Settings

Cryptography is a vast subject requiring its own book. Therefore, in the following we will only give a high-level overview of the problem of *Multi-Party Computation* (MPC), ignoring most of the lower-level details and concentrating only on aspects relevant to Game Theory.

MPC deals with the following problem. There are $n \geq 2$ parties $P_1, ..., P_n$ where party P_i holds input t_i, $1 \leq i \leq n$, and they wish to compute together a function $s = f(t_1, ..., t_n)$ on their inputs. The goal is that each party will learn the output of the function, s, yet with the restriction that P_i will not learn any additional information about the input of the other parties aside from what can be deduced from the pair (t_i, s). Clearly it is the secrecy restriction which adds complexity to the problem, as without it each party could announce its input to all other parties, and each party would locally compute the value of the function. Thus, the goal of MPC is to achieve the following two properties at the *same time*: correctness of the computation and privacy preservation of the inputs.

Let us exemplify the problem. Assume that there are n parties which wish to participate in an auction for some item. The party P_1 is the auctioneer. Parties $P_2, ..., P_n$ each have their personal bid for the item, i.e. bidder P_i has his bid t_i. For simplicity we assume that $t_i \neq t_j$, $\forall i, j$. They want to figure out the highest bid and its bidder while not revealing any additional information about the bids of the losers. Thus, they wish to compute the function $(val, i) = f(\cdot, t_2, \ldots, t_n)$ where $val = \max(t_2, \ldots, t_n)$ and $i = \text{argmax}(t_2, \ldots, t_n)$. Thus, everybody learns who the winner is, and what is the winning bid, but no other information is leaked.

Two generalizations. The following two generalizations of the above scenario are often useful.

(i) *Probabilistic functions.* Here the value of the function depends on some random string r chosen according to some distribution: $s = f(t_1, \ldots, t_n; r)$. For example, in the auction above we can solve the case when some of the bids could be equal by choosing the winner at random. An example of this is the coin-flipping functionality, which takes no inputs, and outputs an unbiased random bit. Notice, it is crucial that the value r is not controlled by any of the parties, but is somehow jointly generated during the computation.

(ii) *Multi-Output Functions.* It is not mandatory that there be a single output of the function. More generally there could be a unique output for each party, i.e. $(s_1, ..., s_n) = f(t_1, ..., t_n)$. In this case, only party P_i learns the output s_i, and no other party learns any information about the other parties input and outputs aside from what can be derived from its own input and output. For example, in the auction above there seems to be little reason to inform all the losing parties of the winner and the winning bid. Thus, we would rather compute a function $(s_1 = (val, i), 0, \ldots, 0, s_i = 1, 0, \ldots, 0) = f(\cdot, t_2 ... t_n)$, where (val, i) is defined as above. The only two parties to receive an output are the auctioneer P_1 and the party P_i who won the auction.

The Parties. One of the most interesting aspects of MPC is to reach the objective of computing the function value, but under the assumption that some of the parties may deviate from the protocol. In Cryptography, the parties are usually divided into two types: *honest* and *faulty*. An honest party follows the protocol without any deviation. Otherwise, the party is considered to be faulty. The faulty behavior can exemplify itself in a wide range of possibilities. The most benign faulty behavior is where the parties follow the protocol, yet try to learn as much as possible about the inputs of the other parties. These parties are called *honest-but-curious* (or *semi-honest*). At the other end of the spectrum, the parties may deviate from the prescribed protocol in any way that they desire, with the goal of either influencing the computed output value in some way, or of learning as much as possible about the inputs of the other parties. These parties are called *malicious*.

We envision an adversary $\mathcal{A}$ who controls all the faulty parties and can coordinate their actions. Thus, in a sense we assume that the faulty parties are working together and can exert the most knowledge and influence over the computation out of this collusion. The adversary can corrupt any number of parties out of the n participating parties. Yet, in order to be able to achieve a solution to the problem, in many cases we would need to limit the number of corrupted parties. We call this limit a threshold k, indicating that that the protocol remains secure as long as the number of corrupted parties is at most k.

1.1.1 Security of Multiparty Computations

We are ready to formulate the idea of what it means to *securely* compute a given function f. Assume that there exists a *trusted party* who privately receives the inputs of all the participating parties, calculates the output value s and then transmits this value to each one of the parties.† This process clearly computes the correct output of f, and also does not enable the participating parties to learn any additional information about the inputs of others. We call this model the *ideal model*. The security of MPC then states that a protocol is secure if its execution satisfies the following: 1. the honest parties compute the same (correct) outputs as they would in the ideal model; and 2. the protocol does not expose more information than a comparable execution with the trusted party, in the ideal model.

Intuitively, this is explained in the following way. The adversary's interaction with the parties (on a vector of inputs) in the protocol generates a *transcript*. This transcript is a random variable which includes the outputs of all the honest parties, which is needed to ensure correctness as explained below, and the output of the adversary $\mathcal{A}$. The latter output, without loss of generality, includes all the information that the adversary learned, including

† Note, that in the case of a probabilistic function the trusted party will choose r according to the specified distribution and use it in the computation. Similarly, for multi-output functions the trusted party will only give each party its own output.

its inputs, private state, all the messages sent by the honest parties to $\mathcal{A}$, and, depending on the model (see later discussion on the communication model), maybe even include more information, such as public messages that the honest parties exchanged. If we show that *exactly* the same transcript distribution‡ can be generated when interacting with the trusted party in the ideal model, then we are guaranteed that no information is leaked from the computation via the execution of the protocol, as we know that the ideal process does not expose any information about the inputs. In some sense what is said here is that if the transcript can be generated given only the output of the computation, then the transcript itself does not expose any information, as anyone can generate it by themselves knowing only the output. More formally,

Definition 1.1 Let f be a function on n inputs and let π be a protocol which computes the function f. Given an adversary $\mathcal{A}$ which controls some set of parties, we define $\mathrm{REAL}_{\mathcal{A},\pi}(t)$ to be the sequence of outputs of honest parties resulting from the execution of π on input vector t under the attack of $\mathcal{A}$, in addition to the output of $\mathcal{A}$. Similarly, given an adversary $\mathcal{A}'$ which controls a set of parties, we define $\mathrm{IDEAL}_{\mathcal{A}',f}(t)$ to be the sequence of outputs of honest parties computed by the trusted party in the ideal model on input vector t, in addition to the output of $\mathcal{A}'$. We say that π *securely computes* f if, for every adversary $\mathcal{A}$ as above, there exists an adversary $\mathcal{A}'$, which controls the same parties in the ideal model, such that, on any input vector t, we have that the distribution of $\mathrm{REAL}_{\mathcal{A},\pi}(t)$ is "indistinguishable" from the distribution of $\mathrm{IDEAL}_{\mathcal{A}',f}(t)$ (where the term "indistinguishable will be explained later).

Intuitively, the task of the ideal adversary $\mathcal{A}'$ is to generate (almost) the same output as $\mathcal{A}$ generates in the real execution (referred to also as the real model). Thus, the attacker $\mathcal{A}'$ is often called the *simulator* (of $\mathcal{A}$). Also note that the above definition guarantees correctness of the protocol. Indeed, the transcript value generated in the ideal model, $\mathrm{IDEAL}_{\mathcal{A}',f}(t)$, also includes the outputs of the honest parties (even though we do not give these outputs to $\mathcal{A}'$), which we know were correctly computed by the trusted party. Thus, the real transcript $\mathrm{REAL}_{\mathcal{A},\pi}(t)$ should also include correct outputs of the honest parties in the real model.

THE INPUTS OF THE FAULTY PARTIES. We assumed that every party P_i has an input t_i which it enters into computation. However, if P_i is faulty, nothing stops P_i from changing t_i into some t_i'. Thus, the notion of "correct"

‡ The requirement that the transcript distribution be exactly the same will be relaxed later on.

input is only defined for honest parties. However, the "effective" input of a faulty party P_i could be defined as the value t'_i that the simulator $\mathcal{A}'$ (which we assume exists for any real model $\mathcal{A}$) gives to the trusted party in the ideal model. Indeed, since the outputs of honest parties looks the same in both models, for all effective purposes P_i must have "contributed" the same input t'_i in the real model.

Another possible misbehavior of P_i, even in the ideal model, might be a refusal to give any input at all to the trusted party. This can be handled in a variety of ways, ranging from aborting the entire computation to simply assigning t_i some "default value". For concreteness, we assume that the domain of f includes a special symbol $\perp$ indicating this refusal to give the input, so that it is well defined how f should be computed on such missing inputs. What this requires is that in any real protocol we detect when a party does not enter its input and deal with it exactly in the same manner as if the party would input $\perp$ in the ideal model.

Variations on Output Delivery. In the above definition of security it is implicitly assumed that all honest parties receive the output of the computation. This is achieved by stating that $\mathrm{IDEAL}_{\mathcal{A}',f}(t)$ includes the outputs of all honest parties. We therefore say that our current definition *guarantees output delivery*.

A more relaxed property than output delivery is *fairness*. If fairness is achieved, then this means that if at least one (even faulty!) party learns its outputs, then all (honest) parties eventually do too. A bit more formally, we allow the ideal model adversary $\mathcal{A}'$ to instruct the trusted party not to compute any of the outputs. In this case, in the ideal model either all the parties learn the output, or none do. Since the $\mathcal{A}$'s transcript is indistinguishable from $\mathcal{A}'$'s this guarantees that the same fairness guarantee must hold in the real model as well.

Yet, a further relaxation of the definition of security is to only provide *correctness and privacy*. This means that faulty parties can learn their outputs, and prevent the honest parties from learning theirs. Yet, at the same time the protocol will still guarantee that: 1. if an honest party receives an output, then this is the correct value, and 2. the privacy of the inputs and outputs of the honest parties is preserved.

Variations on the Model. The basic security notions introduced above are universal and model-independent. However, specific implementations crucially depend on spelling out precisely the model where the computation will be carried out. In particular, the following issues must be specified:

(i) *The parties.* As mentioned above, the faulty parties could be honest-but-curious or malicious, and there is usually an upper bound k on the number of parties which the adversary can corrupt.

(ii) *Computational Assumptions.* We distinguish between the computational setting and the information theoretic setting. In the information theoretic model we assume that the adversary is unlimited in its computing powers. In this case the term "indistinguishable" in Definition 1.1 is formalized by either requiring the two transcript distributions to be identical (so called *perfect security*) or, at least, statistically close in their variation distance (so called *statistical security*). On the other hand, in the computational setting we restrict the power of the adversary (as well as that of the honest parties). A bit more precisely, we assume that the corresponding MPC problem is parameterized by the *security parameter* λ, in which case (a) all the computation and communication shall be done in time polynomial in λ; and (b) the misbehavior strategies of the faulty parties are also restricted to be run in time polynomial in λ. Further, the term "indistinguishability" in Definition 1.1 is formalized by *computational indistinguishability*: two distribution ensembles $\{X_\lambda\}_\lambda$ and $\{Y_\lambda\}_\lambda$ are said to be computationally indistinguishable, if for any polynomial-time distinguisher D, the quantity ϵ, defined as $|Pr[D(X_\lambda) = 1] - Pr[D(Y_\lambda) = 1]|$, is a "negligible" function of λ. This means that for any $j > 0$ and all sufficiently large λ, ϵ eventually becomes smaller than λ^{-j}.

This modeling of computationally bounded parties enables us to build secure MPC protocols depending on plausible computational assumptions, such as the hardness of factoring large integers, etc.

(iii) *Communication Assumptions.* The two common communication assumptions are the existence of a *secure channel* and the existence of a *broadcast* channel. Secure channels assume that every pair of parties P_i and P_j are connected via a an authenticated, private channel. A broadcast channel is a channel with the following properties: if a party P_i (honest or faulty) broadcasts a message m, then m is correctly received by all the parties (who are also sure the message came from P_i). In particular, if an honest party receives m, then it knows that every other honest party also received m.

A different communication assumptions is the existence of *envelopes*. An envelope (in its most general definition) guarantees the following properties: that a value m can be stored inside the envelope, it will be held without exposure for a given period of time and then the value m will be revealed without modification. A *ballot box* is an enhancement of the envelope setting which also provides a random shuffling mechanism of the envelopes.

These are of course idealized assumptions which allow for a clean description of a protocol, as they separate the communication issues from the computational ones. These idealized assumptions may be realized by a physical mechanisms, but in some settings such mechanisms may not be available. Then it is important to address the question *if and under what circumstances* we can remove a given communication assumption. For example, we know that the assumption of a secure channel can be substituted with a protocol, but under the introduction of a computational assumption and a public key infrastructure. In general, the details of these substitutions are delicate and need to be done with care.

1.1.2 Existing Results for Multiparty Computation

Since the introduction of the MPC problem in the beginning of the 80's, the work in this area has been extensive. We will only state, without proofs, a few representative results from the huge literature in this area.

Theorem 1.2 *Secure MPC protocols withstanding coalitions of up to k malicious parties (controlled by an attacker $\mathcal{A}$) exist in the following cases:*

(i) *assuming $\mathcal{A}$ is computationally bounded, secure channels, and a broadcast channel (and a certain cryptographic assumption, implied for example, by the hardness of factoring, is true), then:*
(a) for $k < n/2$ with output delivery.
(b) for $k < n$ with correctness and privacy.
(c) additionally assuming envelopes, for $k < n$ with fairness.

(ii) *assuming $\mathcal{A}$ is computationally unbounded:*
(a) assuming secure channels, then for $k < n/3$ with output delivery.
(b) assuming secure and broadcast channels, then for $k < n/2$ with output delivery (but with an arbitrarily small probability of error).
(c) assuming envelopes, ballot-box and a broadcast channel, then for $k < n$ with output delivery.

Structure of Current MPC Protocols. A common design structure of many MPC protocols proceeds in three stages: commitment to the inputs, computation of the function on the committed inputs, revealing of the output. Below we describe these stages at a high level, assuming for simplicity that the faulty parties are honest-but-curious.

In the first stage the parties commit to their inputs, this is done by utilizing the first phase of a two-phased primitive called *secret sharing*. The first phase of a (k, n)-secret sharing scheme is the *sharing phase*. A dealer, D, who holds some secret z, computes n shares $z_1, \ldots, z_n$ of z and gives the share z_i to party P_i. The second phase is the *reconstruction phase,* which we describe here and utilize later. For the reconstruction the parties broadcast their shares in order to recover z. Informally, such secret sharing schemes satisfy the following two properties: (1) k, or fewer, shares do not reveal any information about z; but (2) any $k+1$ or more shares enable one to recover z. Thus, up to k colluding parties learn no information about z after the sharing stage, while the presence of at least $k+1$ honest parties allows one to recover the secret in the reconstruction phase (assuming, for now, that no incorrect shares are given).

The classical secret sharing scheme satisfying these properties is the Shamir secret sharing scheme. Here we assume that the value z lies in some finite field F of cardinality greater than n (such as the field of integers modulo a prime $p > n$). The dealer D chooses a random polynomial g of degree k with the only constraint that the free coefficient of g is z. Thus, $z = g(0)$. Then, if $\alpha_1 \ldots \alpha_n$ are arbitrary but agreed in advance non-zero elements of F, the shares of party P_i is computed as $z_i = g(\alpha_i)$. It is now easy to observe that any $k+1$ shares z_i are enough to interpolate the polynomial g and compute $g(0) = z$. Furthermore, any set of k shares is independent of z. This is easy to see as for any value $z' \in F$ there exists a $(k+1)$st share such that with the given set of k shares they interpolate a polynomial g' where $g'(0) = z'$, in a sense making any value of the secret equally likely. Thus, properties (1) and (2) stated above are satisfied.

To summarize, the first stage of the MPC is achieved by having each party P_i invoke the first part of the secret sharing process as the dealer D of its input t_i, and distribute the correct shares of t_i to each party P_j. If f is probabilistic, the players additionally run a special protocol at the end of which a (k, n)-secret sharing of a *random and secret* value r is computed.

In the second stage the parties compute the function f. This is done by evaluating the pre-agreed upon arithmetic circuit representing f over F, which is composed of addition, scalar-multiplication and multiplication gates. The computation proceeds by evaluating the gates one by one. We inductively assume that the inputs to the gates are shared in the manner described above in the secret sharing scheme, and we guarantee that the output of the gate will preserve the same representation. This step forms the heart of most MPC protocols. The computation of the addition and scalar-multiplication gates are typically pretty straightforward and does not require communication (e.g., for the Shamir secret sharing scheme the parties locally simply add or multiply by the scalar their input shares), but is considerably more involved for the multiplication gate and requires communication. For our purposes we will not need the details of the computation mechanism, simply assuming that this computation on shares is possible will suffice. Therefore, we can assume that at the end of the second stage the parties have a valid secret sharing of the required output(s) of the function f. The most crucial observation is that no additional information is leaked throughout this stage, since all the values are always shared through a (k, n)-secret sharing scheme.

Finally, in the last stage the parties need to compute their individual outputs of the function. As we have inductively maintained the property that the output of each gate is in the secret sharing representation, then the same it true for the output gate of f. Thus, to let the parties learn the output s which is the value of the function, the parties simply run the reconstruction phase of the secret sharing scheme (as described above), by having each party broadcast its share of s.

1.2 Game Theory Notions and Settings

Strategic Games. We start by recalling the basic concept of (one-shot) strategic games with complete information. Such a game $G = (I, (S_i)_{i \in I}, (u_i)_{i \in I})$ is given by a set I of n parties $P_1 \dots P_n$, a set of actions S_i for each party P_i, and a set of real-valued utility functions $u_i : S \to R$ for each party P_i, where $S = S_1 \times \dots \times S_n$. The parties move simultaneously, each choosing an action $s_i \in S_i$. The *payoff* (or utility) of party P_i is $u_i(s_1, \dots, s_n)$. The (probabilistic) algorithm x_i that tells party P_i which action to take is called its *strategy* and a tuple of strategies $x = (x_1, \dots, x_n)$ is called a *strategy profile*. Notationally, if we let $\Delta(B)$ denote the set of probability distributions over a finite set B, we have $x_i \in \Delta(S_i)$ and $x \in \Delta(S)$. We denote by s_{-i} (or x_{-i}) the strategy profile of all parties except the party P_i (whom we sometimes also denote P_{-i}). Finally, we naturally extend the utility functions to strategy profiles, by defining $u_i(x)$ as the expectation of $u_i(s)$, where $s = (s_1, \dots, s_n)$ is chosen according to x.

Game Theory assumes that each party is *selfish and rational*, i.e. only cares about maximizing its (expected) payoff. As a result, we are interested in strategy profiles that are *self-enforcing*. In other words, even knowing the strategy of the other parties, each party still has no incentive to deviate from its own strategy. Such a strategy profile is called an *equilibrium*.

Nash equilibrium. This equilibrium notion corresponds to a strategy profile in which parties' strategies are *independent*. More precisely, the induced distribution x over the pairs of actions must be a product distribution; i.e., $x \in \Delta(S_1) \times \ldots \times \Delta(S_n)$.

Definition 1.3 A *Nash equilibrium* (NE) of a game G is an independent strategy profile $(x_1^*, \ldots, x_n^*)$, s.t. $\forall P_i \in I$, $s_i \in S_i \Rightarrow u_i(x_i^*, x_{-i}^*) \geq u_i(s_i, x_{-i}^*)$.

In other words, for every party P_i, following its prescribed strategy x_i^* is an optimal response to the other parties' prescribed strategies x_{-i}^*. Nash's famous result in game theory states that every finite strategic game has at least one NE.

Correlated equilibrium. Nash equilibrium is quite a natural and appealing notion, since parties can follow their strategies independently of each other. However, we will see that it is often possible to achieve considerably higher expected payoffs if one allows general, *correlated* strategies $x \in \Delta(S)$ (as opposed to $\Delta(S_1) \times \ldots \times \Delta(S_n)$). To make sense of such a correlated profile when the game is implemented, we augment the basic setting of strategic (one-shot) games by introducing a trusted party M called a *mediator*. M will sample a tuple of correlated actions $s = (s_1 \ldots s_n)$ according to x, and then *privately* recommend the action s_i to party P_i.† In particular, after receiving its recommendation s_i each party P_i will have a conditional distribution, denoted $(x_{-i} \mid s_i)$, regarding the strategy profile recommended to the other parties P_{-i}, after which P_i can decide if following s_i is indeed the best course of action assuming the remaining parties follow $x_{-i} \mid s_i$. In a *correlated equilibrium*, the answer to this question is positive: it is in the interest of all the parties to follow the recommendations of the mediator.

Definition 1.4 A *correlated equilibrium* (CE) is a strategy profile $x^* \in \Delta(S)$, such that for any $P_i \in I$, $s_i \in S_i$ and any action s_i^* in the support of x_i^*, we have $u_i(s_i^*, (x_{-i}^* \mid s_i^*)) \geq u_i(s_i, (x_{-i}^* \mid s_i^*))$.

Correlated equilibria always exist and form a convex set which necessarily includes the convex hull of Nash equilibria (where the latter is defined to

† More generally, the mediator can tell some other information to P_i, in addition to the recommended strategy s_i. However, it turns out this extra freedom will not enrich the class of correlated equilibria we define below. Intuitively, this is because the player will not want to deviate no matter what this extra information might be, so we might as well not provide it to begin with.

consist of all convex combinations of Nash equilibria; notice, such equilibria can be implemented by the mediator by flipping a biased coin telling which Nash equilibria to recommend to players). However, by utilizing correlated (but not necessarily identical) recommendations, one can achieve correlated equilibria with equilibrium payoffs *outside* (and often significantly better!) anything in the convex hull of Nash equilibria payoffs. This is demonstrated in the following simple example which is known as the "game of chicken".

Game of Chicken. This 2×2 game is shown in the table below.

	C	D
C	4,4	1,5
D	5,1	0,0

"Chicken"

	C	D
C	1/4	1/4
D	1/4	1/4

Mixed Nash x^3

	C	D
C	0	1/2
D	1/2	0

Public Coin $\bar{x}$

	C	D
C	1/3	1/3
D	1/3	0

Correlated x^*

Each party can either "dare" (D) or "chicken out" (C). The combination (D, D) has a devastating effect on both parties (payoffs $[0, 0]$), (C, C) is quite good (payoffs $[4, 4]$), while each party would ideally prefer to dare while the other chickens-out (giving him 5 and the opponent 1). Although the "wisest" pair of actions is (C, C), this is not a NE, since both parties are willing to deviate to D (believing that the other party will stay at C). The game is easily seen to have three NEs: $x^1 = (D, C)$, $x^2 = (C, D)$ and $x^3 = (\frac{1}{2} \cdot D + \frac{1}{2} \cdot C,\ \frac{1}{2} \cdot D + \frac{1}{2} \cdot C)$. The respective NE payoffs are $[5, 1]$, $[1, 5]$ and $[\frac{5}{2}, \frac{5}{2}]$. We see that the first two NEs are asymmetric, while the last mixed NE has small payoffs, since the mutually undesirable outcome (D, D) happens with non-zero probability $\frac{1}{4}$ in the product distribution. The best symmetric strategy profile in the convex hull of the NEs is $\bar{x} = \frac{1}{2}x^1 + \frac{1}{2}x^2 = (\frac{1}{2}(C, D) + \frac{1}{2}(D, C))$, yielding payoffs $[3, 3]$. On the other hand, the profile $x^* = (\frac{1}{3}(C, D) + \frac{1}{3}(D, C) + \frac{1}{3}(C, C))$ is a CE, yielding payoffs $[3\frac{1}{3}, 3\frac{1}{3}]$ outside the convex hull of the NEs.

To briefly see that x^* is a CE, consider the row party P_1 (same works for the column party P_2). If P_1 is recommended to play C, its expected payoff is $\frac{1}{2} \cdot 4 + \frac{1}{2} \cdot 1 = \frac{5}{2}$ since, conditioned on $s_1^* = C$, party P_2 is recommended to play C and D with probability $\frac{1}{2}$ each. If P_1 switched to D, its expected payoff would still be $\frac{1}{2} \cdot 5 + \frac{1}{2} \cdot 0 = \frac{5}{2}$, making P_1 reluctant to switch. Similarly, if party P_1 is recommended D, it knows that P_2 plays C (as (D, D) is never played in x^*), so its payoff is 5. Since this is the maximum payoff of the game, P_1 would not benefit by switching to C in this case. Thus, we indeed have a correlated equilibrium, where each party's payoff is $\frac{1}{3}(1 + 5 + 4) = 3\frac{1}{3}$, as claimed.

Interestingly, we also point out that mediators do not necessarily increase the parties' payoffs. In fact, carefully designed mediators can force the parties into considerably *worse* payoffs than what is possible in the unmediated game. For example, in the "game of chicken" it is easy to see that the profile $x^{**} = (\frac{1}{3}(C,D) + \frac{1}{3}(D,C) + \frac{1}{3}(D,D))$ is also a CE, yielding payoffs $[2,2]$ which are less than the worst Nash Equilibrium $x^3 = (\frac{1}{4}(C,D) + \frac{1}{4}(D,C) + \frac{1}{4}(C,C) + \frac{1}{4}(D,D))$ (the latter giving payoffs $[\frac{5}{2},\frac{5}{2}]$). Thus, the mediator can generally expand the set of equilibria in all directions. Intuitively, though, we are usually interested to implement only mediators, who *improve* the payoffs of most, or even all of the parties.

Games with Incomplete Information. So far we discussed strategic games where parties knew the utilities of other parties. In games with incomplete information each party has a private type $t_i \in T_i$, where the joint vector $t = (t_1, \ldots, t_n)$ is assumed to be drawn from some publicly known distribution. The point of such type, t_i, is that it affects the utility function of party P_i: namely, the utility u_i not only depends on the actions $s_1 \ldots s_n$, but also on the private type t_i of party P_i, or, in even more general games, on the entire type vector t of *all* the parties. With this in mind, generalizing the notion of Nash equilibrium to such games is straightforward. (The resulting Nash equilibrium is also called *Bayesian*.)

Mediated games generalize to the typed setting, in which parties have to send their types to the mediator M before receiving the joint recommendation. Depending on the received type vector t, the mediator samples a correlated strategy profile s and gives each party its recommended action s_i, as before. We remark that the expected canonical strategy of party P_i is to honestly report its type t_i to M, and then follow the recommended action s_i. However, P_i can deviate from the protocol in two ways: 1. send a wrong type t'_i or not send a type at all to M, as well as 2. decide to change the recommended action from s_i to some s'_i.† As a mediator may receive faulty types, a fully defined sampling strategy for the mediator should specify the joint distribution x for every type $t = (t_1 \ldots t_n)$, even outside the support of the joint type distribution. Formally, x^t should be defined for every $t \in \prod_i (T_i \cup \{\perp\})$, where $\perp$ is a special symbol indicating an invalid type. (In particular, games of complete information can be seen as a special case where all $t_i = \perp$ and each party "refused" to report its type.) With this in mind, the generalization of CE to games with incomplete information is straightforward.

Aborting the Game. We assume that the parties will always play the

† Although not relevant in the sequel, we mention the famous *revelation principle* for such games. It states that from the perspective of characterizing correlated equilibria of such games, we can safely assume that each party will honestly report its type t_i to M, and only then consider the question whether or not to follow the recommendation of M.

game by choosing an action $s_i \in S_i$ and getting an appropriate payoff $u_i(s)$. Of course, we can always model refusal to play by introducing a special action $\perp$ into the strategy space, and defining the explicit utilities corresponding to such actions. In other words, to allow for abort, we must explicitly specify the penalty for aborting the computation. Indeed, many games effectively guarantee participation by assigning very low payoff to actions equivalent to aborting the computation. However, this is not a requirement; in fact, many games do not even have the abort action as parts of their action spaces. To summarize, aborting is not something which is inherent to games, although it could be modeled within the game, if required.

EXTENDED GAMES. So far we only considered strategic games, where parties move in "one-shot" (possibly with the help of the mediator). Of course, these games are special cases of much more general *extensive form* games (with complete or incomplete information), where a party can move in many rounds and whose payoffs depend on the entire run of the game. In our setting we will be interested only in a special class of such extensive form games, which we call *(strategic) games extended by cheap-talk*, or, in brief, *extended games.*

An extended game G^* is always induced by a basic strategic game G (of either complete or incomplete information), and has the following form. In the *cheap-talk (or preamble) phase*, parties follow some *protocol* by exchanging messages in some appropriate communication model. This communication model can vary depending on the exact setting we consider. But once the setting is agreed upon, the format of the cheap talk phase is well defined. After the preamble, the *game phase* will start and the parties simply play the original game G. In particular, the payoffs of the extended game are exactly the payoff that the parties get in G (and this explains why the preamble phase is called "cheap talk").

Correspondingly, the strategy x_i of party P_i in the extended game consists of its strategy in the cheap talk phase, followed by the choice of an action s_i that P_i will play in G. Just like in strategic games, we assume that the game phase must always go on. Namely, aborting the game phase will be modeled inside G, but only if necessary. However, the parties can always abort the preamble phase of the extended game, and prematurely decide to move on to the game phase. Thus, a valid strategy profile for the extended game *must* include instructions of which action to play if some other party refuses to follow its strategy, or, more generally, deviates from the protocol instructions during the cheap talk phase (with abort being a special case of such misbehavior).

Nash Equilibrium of Extended Games. With this in mind, (Bayesian) Nash equilibrium for extended games is defined as before: a tuple of independent strategic $x_1^* \dots x_n^*$ such that for every P_i and for every alternative strategy x_i, we have $u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*)$. We remark, however, that Nash equilibrium is known to be too liberal for extensive form games, as it allows for "unreasonable" strategy profiles to satisfy the definition of NE. For example, it allows for equilibrium strategies containing so called "empty threats" and has other subtle deficiencies. Nevertheless, in order to keep our presentation simple we will primarily restrict ourselves to the basic concept of NE when talking about extended games.

Collusions. All the discussion so far assumed the traditional *non-cooperative* setting, where agents are assumed not to form collusions. In contrast, *cooperative game theory* tries to model reasonable equilibrium concepts arising in scenarios where agents are allowed to form collusions. Unfortunately, in many games no equilibria resistant to coalitional deviations may even exist! However, when they do, such equilibria seem to be more stable than conventional non-cooperative equilibria. However, traditional game-theoretic treatment of such equilibria are fairly weak. We will come back to this issue in Section 1.4.1, where we provide the definition of an equilibrium which we think is the most appropriate for our setting and has been influenced by the MPC setting.

1.3 Contrasting MPC and Games

As we can see, MPC and games share several common characteristics. In both cases an important problem is to compute some function $(s_1 \dots s_n) = f(t_1 \dots t_n; r)$ in a private manner. However, there are some key differences summarized in Figure 1.3, making the translation from MPC to Games and vice versa a promising but non-obvious task.

Incentives and Rationality. Game theory is critically built on incentives. Though it may not necessarily explain why parties participate in a game, once they do, they have a very well defined incentive. Specifically, players are assumed to be *rational* and only care about maximizing their utility. Moreover, rationality is common knowledge: parties are not only rational, but know that other parties are rational and utilize this knowledge when making their strategic decisions. In contrast, the incentives in the MPC setting remain external to the computation, and the reason the computation actually ends with a correct and meaningful output comes from the assumption on the parties. Specifically, in the MPC setting one as-

Issue	*Cryptography*	*Game Theory*
Incentive	Outside the Model	Payoff
Players	Totally Honest or Malicious	Always Rational
Solution Drivers	Secure Protocol	Equilibrium
Privacy	Goal	Means
Trusted Party	In the Ideal Model	In the Actual Game
Punishing Cheaters	Outside the Model	Central Part
Early Stopping	Possible	The Game Goes On!
Deviations	Usually Efficient	Usually Unbounded
k-collusions	Tolerate "large" k	Usually only $k = 1$

sumes that there exist two diametrically opposite kinds of parties: *totally honest* and *arbitrarily malicious*. The honest parties are simply assumed to blindly follow their instructions, without any formally defined rational behind this. On the other hand, malicious parties might behave in a completely irrational manner, and a good MPC protocol has to protect against such "unreasonable" behavior. Thus, the settings are somewhat incomparable in general. On the one hand, the MPC setting may be harder as it has to protect against completely unexplained behavior of the malicious parties (even if such behaviors would be irrational had the parties had the utilities defined). On the other hand, the Game Theory setting could be harder as it does not have the benefit of assuming that some of the parties (i.e., the honest parties) blindly follow the protocol. However, we remark that this latter benefit disappears for the basic notions of Nash and correlated equilibria, since there one always assumes that the other parties follow the protocol when considering whether or not to deviate. For such basic concepts, we will indeed see in Section 1.4.2 that the MPC setting is more powerful.

Privacy and Solution Drivers. In the cryptographic setting the objective is to achieve a secure protocol, as defined in Definition 1.1. In particular, the main task is to eliminate the trusted party in a private and resilient way. While in the game theory setting the goal is to achieve "stability" by means of some appropriate equilibrium. In particular, the existence of the mediator is just another "parameter setting" resulting in a more desirable, but harder to implement equilibrium concept. Moreover, the privacy constraint on the mediator is merely a technical way to justify a much richer class of "explainable" rational behaviors. Thus, in the MPC setting privacy is the *goal* while in the game theory setting it is a *means to an end*.

"CRIME AND PUNISHMENT". We also notice that studying deviations from the prescribed strategy is an important part of both the cryptographic and the game-theoretic setting. However, there are several key differences.

In cryptography, the goal is to compute the function, while achieving some security guarantees in spite of the deviations of the faulty parties. Most protocols also enable the participating parties to detect which party has deviated from the protocol. Yet, even when exposed, in many instances no action is taken against the faulty party. Yet, when an action, such as removal from the computation, is taken, this is not in an attempt to punish the party, but rather to enable the protocol to reach its final goal of computing the function. In contrast, in the game-theoretic setting it is crucial to specify exactly how the misbehavior will be dealt with by the other parties. In particular, one typical approach is to design reaction strategies which will negatively affect the payoffs of the misbehaving party(s). By rationality, this *ensures* that it is in no player's self-interest to deviate from the prescribed strategy.

We already commented on a particular misbehavior when a party refuses to participate in a given protocol/strategy. This is called *early stopping*. In the MPC setting, there is nothing one can do about this problem, since it is possible in the ideal model as well. In the Game Theory setting, however, we already pointed out that one always assumes that "the game goes on". I.e., if one wishes, it is possible to model stopping by an explicit action with explicit payoffs, but the formal game is always assumed to be played. Thus, if we use MPC inside a game-theoretic protocol, we will have to argue — from the game-theoretic point of view — what should happen when a given party aborts the MPC.

EFFICIENCY. Most game-theoretic literature places no computational limitations on the efficiency of a party when deciding whether or not to deviate. In contrast, a significant part of cryptographic protocol literature is designed to only withstand computationally bounded adversaries (either because the task at hand is otherwise impossible, or much less efficient). We will see how to incorporate such efficiency considerations into game theory in Section 1.4.1.

COLLUSIONS. Finally, we comment again on the issue of collusions. Most game-theoretic literature considers non-cooperative setting, which corresponds to collusions of size $k = 1$. In contrast, in the MPC setting the case $k = 1$ is usually straightforward, and a lot of effort is made in order to make the maximum collusion threshold as much as possible. Indeed, in most MPC settings one can tolerate at least a linear fraction of colluding parties, and sometimes even a collusion of all but one party.

1.4 Cryptographic Influences on Game Theory

In this section we discuss how the techniques and notions from MPC and cryptography can be used in Game Theory. We start by presenting the notions of computational and k-resilient equilibria, which were directly influenced by cryptography. We then proceed by describing how to use appropriate MPC protocols and replace the mediator implementing a given CE by a "payoff-equivalent" cheap-talk phase in a variety of contexts.

1.4.1 New Notions

COMPUTATIONAL EQUILIBRIUM. Drawing from the cryptographic world, we consider settings where parties participating in the extended game are computationally bounded and we define the notion of *computational equilibriums*. In this case we only have to protect against *efficient* misbehavior strategies x_i. A bit more precisely, we will assume that the basic game G has constant size. However, when designing the preamble phase of the extended game, we can parameterize it by the security parameter λ, in which case (a) all the computation and communication shall be done in time polynomial in λ; and (b) the misbehavior strategies x_i are also restricted to be run in time polynomial in λ.

The preamble phase will be designed under the assumption of the existence of a computationally hard problem. However, this introduces a negligible probability (see Section 1.1.1) that within x_i the attacker might break (say, by luck) the underlying hard problem, and thus might get considerably higher payoff than by following the equilibrium strategy x_i^*. Of course, this can improve this party's expected payoff by at most a negligible amount (since the parameters of G, including the highest payoff, are assumed constant with respect to λ), so we must make an assumption that the party will not bother to deviate if its payoffs will only increase by a negligible amount. This gives rise to the notion of *computational Nash equilibrium*: a tuple of independent strategies $x_1^* \ldots x_n^*$ where each strategy is *efficient* in λ such that for every P_i and for every alternative *efficient* in λ strategy x_i, we have $u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*) - \epsilon$, where ϵ is a negligible function of λ.

k-Resiliency. As we mentioned, the Game Theory world introduced several flavors of cooperative equilibria concepts. Yet, for our purposes here, we define a stronger type of such an equilibrium, called a *resilient* (Nash or Correlated) equilibrium. Being a very strong notion of an equilibrium, it may not exist in most games. Yet, we chose to present it since it will exist in the "Game Theory-MPC" setting, where we will use MPC protocols in several game-theoretic scenarios. The possibility of realizing such strong equilibria using MPC shows the strength of the cryptographic techniques. Furthermore, with minor modifications, most of the results we present later in the chapter extend to weaker kinds of cooperative equilibria, such as various flavors of a more well known *coalition-proof equilibrium.*†.

Informally, resilient equilibrium requires protection against all coalitional deviations which strictly benefit even *one* of its colluding parties. Thus, no such deviation will be justifiable to *any* member of the coalition, meaning that the equilibrium strategies are very stable.

To define this more formally, we need to extend our notation to handle coalitions. Given a strategic game $G = (I, (S_i)_{P_i \in I}, (u_i)_{P_i \in I})$ and a coalition $C \subseteq I$, we denote $S_C = \prod_{P_i \in C} S_i$. Then a joint strategy x_C of C is an element of $\Delta(S_C)$. Given a strategy profile x^*, we will also denote by x^*_C its "projection" onto C, and by x^*_{-C} its "projection" on $I \backslash C$. Then we say that an independent strategy profile $(x^*_1, \ldots, x^*_n)$ is a *k-resilient Nash Equilibrium of G*, if for all coalitions C of cardinality at most k, all deviation strategies $x_C \in \Delta(S_C)$, and *all* members $P_i \in C$, we have $u_i(x^*_C, x^*_{-C}) \geq u_i(x_C, x^*_{-C})$.

The notion of *k-resilient correlated equilibrium* $x \in \Delta(S)$ is defined similarly, although here we can have two variants. In the *ex ante* variant, members of C are allowed to collude only *before* receiving their actions form the mediator: namely, a deviation strategy will tell each member of the coalition how to change its recommended action, but this would be done without knowledge of the recommendations to the other members of the coalition. In the more powerful *interim* variant, the members of the coalition will see the entire recommended action vector s^*_C and then can attempt to jointly change it to some s_C. Clearly, *ex ante* correlated equilibria are more abundant than *interim* equilibria. For example, it is easy to construct games where already 2-resilient *ex ante* CEs achieve higher payoffs than 2-resilient *interim* equilibria, and even games where the former correlated equilibria exist and the latter do not! This is true because the *ex ante* setting makes a strong restriction that coalitions cannot form after the mediator gave its

† Informally, these equilibria only prevent deviations benefiting *all* members of the coalition, while resilient equilibria also prevent deviations benefiting even *a single* such member.

recommended actions. Thus, unless stated otherwise, k-resilient CE will refer to the *interim* scenario.

Finally, we mention that one can naturally generalize the above notions to games with incomplete information, and also define (usual or computational) k-resilient Nash equilibria of extended games. On the other hand, it is quite non-obvious how to properly define stronger computational equilibria, such as subgame-perfect, sequential, etc. equilibria. Indeed, such definitions are quite subtle, and are not well understood at the current stage.

1.4.2 Removing the Mediator in Correlated Equilibrium

The natural question which can be asked is whether the mediator can be removed in the game theory setting, simulating it with a multiparty computation. The motivation for this is clear, as the presence of the mediator significantly expands the number of equilibria in strategic form games; yet, the existence of such a mediator is a very strong and often unrealizable assumption.

Recall that in any correlated equilibrium x of a strategic game G (with imperfect information, for the sake of generality), the mediator samples a tuple of recommended action $(s_1, ..., s_n)$ according to the appropriate distribution based on the types of the parties. This can be considered as the mediator computing some probabilistic function $(s_1, ..., s_n) = f(t_1 \dots t_n; r)$. We define the following extended game G^* of G by substituting the mediator with an MPC and ask whether the extended game is a (potentially computational) Nash equilibrium.

(i) In the preamble stage, the parties run an "appropriate" MPC protocol† to compute the profile $(s_1 \dots s_n)$. Some additional actions may be needed (see below).

(ii) Once the preamble stage is finished, party P_i holds a recommended action s_i, which it uses in the game G.

Meta-Theorem. Under "appropriate" conditions, the above strategies form a (potentially computational) Nash equilibrium of the extended game G^*, which achieves the same expected payoffs for all the parties as the corresponding correlated equilibrium of the original game G.‡

As we discussed in Section 1.3, there are several differences between the

† Where the type of the protocol depends on the particular communication model and the capabilities of the parties.

‡ Note that the converse (every NE of G^* can be achieved by a CE of G) is true as well.

MPC and the game theory settings. Not surprisingly, we will have to resolve these differences before validating the meta-theorem above. To make matters a bit more precise, we assume that

- x is an interim k-resilient correlated equilibrium† of G that we are trying to simulate. $k = 1$ (i.e., no collusions) will be the main special case.
- the MPC protocol computing x is cryptographically secure against coalitions of up to k malicious parties. This means the protocol is at least correct and private, and we will comment about its "output delivery" guarantees later.
- The objective is to achieve a (possibly computational) k-resilient Nash equilibrium x^* of G^* with the same payoffs as x.

Now the only indeterminant in the definition of G^* is to specify the behavior of the parties in case the MPC computation fails for some reason.

Using MPC with Guaranteed Output Delivery. Recall that there exist MPC protocols (in various models) which guarantee output delivery for various resiliencies k. Namely, the malicious parties cannot cause the honest parties not to receive their output. The only thing they can do is to choose their inputs arbitrarily (where a special input $\perp$ indicates they refuse to provide the input). But since this is allowed in the mediated game as well, and k-resilient equilibrium ensures the irrationality of such behavior (assuming the remaining $(n - k)$ parties follow the protocol), we know the parties will contribute their proper types and our Meta-theorem is validated:

Theorem 1.5 *If x is a k-resilient CE of G specified by a function f, and π is an MPC protocol (with output delivery) securely computing f against a coalition of up to k computationally unbounded/bounded parties, then running π in the preamble step (and using any strategy to select a move in case some misbehavior occurs) yields a k-resilient regular/computational NE of the extended game G^*, achieving the same payoffs as x.*

Using Fair MPC. In some instances (e.g. part i.c of Theorem 1.2) we cannot guarantee output delivery, but can still achieve fairness. Recall, this means that if at least one party P_i obtains its correct output s_i, then all parties do. However, it might be possible for misbehaving parties to cause everybody to abort or complete the protocol without an output.

In the case where the protocol terminates successfully, we are exactly in the same situation as if the protocol had output delivery, and the same

† As we already remarked, the techniques presented here easily extend to weaker coalitional equilibria concepts.

analysis applies. In the other case, we assume that the protocol enables detection of faulty behavior and that it is observed that one of the parties (for simplicity, assume it is P_n) deviated from the protocol. As the protocol is fair, the aborting deviation must have occurred before any party has any information about their output. The simplest solution is to restart the computation of x from scratch with all parties. The technical problem with this solution is that it effectively allows (a coalition containing) P_n to mount a denial of service attack, by misbehaving in every MPC iteration causing the preamble to run forever.

Instead, to make the extended game always finite, we follow a slightly more sophisticated punishment strategy. We restart the preamble without P_n, and let the $(n-1)$ remaining parties run a new MPC to compute the $(n-1)$-input function f' on the remaining parties' inputs and a default value $\perp$ for P_n: $f'(t_1, \ldots, t_{n-1}; r) = f(t_1, \ldots, t_{n-1}, \perp; r)$. Notice, in this new MPC n is replaced by $n-1$ and k replaced by $k-1$ (as P_n is faulty), which means that the ratio $\frac{k-1}{n-1} < \frac{k}{n}$, and, thus, f' can still be securely computed in the same setting as f. Also notice that P_n does not participate in this MPC, and will have to decide by itself (or with the help of other colluding members) which action to play in the actual game phase. In contrast, parties $P_1 \ldots P_{n-1}$ are instructed to follow the recommendations they get when computing f', if f' completes. If not, then another party (say, P_{n-1}) must have aborted this MPC, in which case we reiterate the same process of excluding P_{n-1}, and so on. Thus, at some point we have that the process will end, as there is a finite number n of parties and we eliminate (at least) one in each iteration.

Next, we argue that the resulting strategy profile x^* forms a k-resilient Nash equilibrium of G^*. To see this, the fairness of the MPC step clearly ensures that the only effective misbehavior of a coalition of size $|C|$ is to declare invalid types $\perp$ for some of its members, while changing the real type for others. In this case, their reluctance to do so follows from the fact that such misbehavior is allowed in the mediated game as well. And since we assumed that the strategy profile x is a k-resilient correlated equilibrium of G, it is irrational for the members of the coalition to deviate in this way.

Using Correct and Private MPC: Case $k = 1$. To see the difficulties which arise from the use of an MPC without fairness, consider two parties playing the game of chicken (see Section 1.2), and trying to implement the correlated equilibrium x having payoffs 10/3 for both parties. If party P_2 receives its move before P_1 does, and the move is C, then P_2 knows that its expected payoff is $\frac{5}{2} < \frac{10}{3}$. In contrast, if the move is D, it knows its payoff is $5 > \frac{10}{3}$. Thus, P_2 may decide to abort the game if (and only if) its recommendation is C. And even if P_1 "punishes" P_2 by playing its "conditional" strategy $x_1 = \frac{2}{3} \cdot C + \frac{1}{3} \cdot D$, responding with D still gives P_2 expected payoff of $\frac{2}{3} \cdot 5 = \frac{10}{3}$, making its overall expected payoff strictly greater than $\frac{10}{3}$.

Nevertheless, we show that one can still use unfair (yet private and correct) MPC protocols in an important special case of the problem. Specifically, we concentrate on the usual coalition-free case $k = 1$, and also restrict our attention to games with complete information (i.e., no types). In this case, we show that if some party P_i deviates in the MPC stage (perhaps by aborting the computation based on its recommended action), the remaining parties P_{-i} can sufficiently punish P_i to discourage such an action. Let the *min-max* value v_i for party P_i denote the worst payoff that players P_{-i} can *jointly* enforce on P_i: namely, $v_i = \min_{z_{-i} \in \Delta(S_{-i})} \max_{s_i \in S_i} u_i(s_i, z_{-i})$.

Claim 1.6 *For any correlated equilibrium x of G, any P_i and any action s_i' for P_i in the support of x_i, $Exp(u_i(s) \mid s_i = s_i') \geq v_i$.*

Proof Notice, since x is a CE, s_i' is the best response of P_i to the profile $\bar{x}_{-i}$ defined as x_{-i} conditioned on $s_i = s_i'$. Thus, the payoff P_i gets in this case is what others would force on P_i by playing $\bar{x}_{-i}$, which is at least as large as what others could have selected by choosing the *worst* profile z_{-i}. $\square$

Now, in case P_i would (unfairly) abort the MPC step, we will instruct the other parties P_{-i} to punish P_i to its min-max value v_i. More specifically, parties P_{-i} should play the correlated strategy z_{-i} which would force P_i into getting at most v_i. Notice, however, since this strategy is correlated, they would need to run another MPC protocol to implement z_{-i}.† By the above claim, *irrespective of the recommendation s_i* that P_i learned, the corresponding payoff of P_i can only go down by aborting the MPC. Therefore, it is in P_i's interests not to abort the computation after learning s_i.

We notice that the above punishment strategy does not straightforwardly generalize to more advanced settings. For example, in case of coalitions it could be that the min-max punishment for P_1 tremendously benefits another colluding party P_2 (who poses honest and instructs P_1 to abort the computation to get high benefits for itself). Also, in the case of incomplete information, it is not clear how to even define the min-max punishment, since the parties do not even know the precise utility of P_i!

† Notice, there are no dishonest parties left, so any MPC protocol for the honest-but-curious case would work.

1.4.3 Stronger Equilibria

So far we only talked about plain Nash equilibria of the extended game G^*. As we already commented briefly, Nash equilibria are usually too weak to capture extensive-form games. Therefore, an interesting (and still developing!) direction in recent research is to ensure much stronger and more stable equilibria which would simulate correlated equilibria of the original game.

Eliminating Empty Threats. One weakness of the Nash equilibrium is that it allows for the so called empty threats. Consider, for example, the min-max punishment strategy used above. In some games, punishing a misbehaving party to its min-max value is actually very damaging for the punishers as well. Thus, the threat to punish the misbehaving party to the min-max value is not credible in such cases, despite being a NE. In this case, eliminating such an empty threat could be done by modifying the punishment strategy to playing the worst Nash equilibrium of G for P_i (in terms of P_i's payoff) when P_i is caught cheating. Unlike the min-max punishment, this is no longer an empty threat because it is an equilibrium of G. However, it does limit (although slightly) the class of correlated equilibria one can simulate, as one can only achieve a payoff vector which is at least as large as the worst Nash equilibrium for each player. Additionally, formally defining such so called subgame-perfect or sequential equilibria has not yet been done in the computational setting, where most MPC protocols are analyzed.

Ex Ante Correlated Equilibria. So far we only talked about simulating interim correlated equilibria, where colluding parties can base their actions after seeing all their recommendations. Another interesting direction is that of simulating *ex ante* correlated equilibria, where colluding parties can only communicate prior to contacting the mediator. To implement this physical restriction in real life we need to design *collusion-free protocols*, where one has to ensure that no subliminal communication (a.k.a. *steganography*) is possible. This is a very difficult problem. Indeed, most cryptographic protocols need randomness (or entropy), and it is known that entropy almost always implies steganography. In fact, it turns out that, in order to build such protocols, one needs some physical assumptions in the real model as well. On a positive side, it is known that envelopes (and a broadcast channel) are enough for building a class of collusion-free protocols sufficient to simulate *ex ante* correlated equilibria without the mediator.

Iterated Deletion of Weakly Dominated Strategies. In Section 1.5.2 we will study a pretty general class of "function evaluation games", where the objective is to achieve Nash equilibrium which survives so called *iterated deletion of weakly dominated strategies* (see Section 1.5.2).

STRATEGIC AND PRIVACY EQUIVALENCE. The strongest recent results regarding removing the mediator is to ensure (polynomially efficient) "real-life" simulation which guarantees an extremely strong property called *strategic and privacy equivalence*. Intuitively, it implies that our simulation gives exactly the same power in the real model as in the ideal model. As such, it precisely preserves all different types of equilibria of the original game (e.g., without introducing *new*, unexpected equilibria in the extended game, which we allowed so far), does not require the knowledge of the utility functions or an a-priori type distribution (which most of the other results above do), does not give any extra power to arbitrary coalitions, preserves privacy of the players types as much as in the ideal model, and has other attractive properties. Not surprisingly, strategic and privacy equivalence is very difficult to achieve, and requires some physical assumptions in the real model as well. The best known result is an extension of the MPC result ii.c in Theorem 1.2, and shows how to implement strategic and privacy equivalence assuming a broadcast channel, envelopes and a ballot-box.

To summarize, MPC techniques are promising in replacing the mediator by cheap talk in a variety of situations. However, more work has to be done in trying to achieve stronger kinds of equilibria using weaker assumptions.

1.5 Game Theoretic Influences on Cryptography

The influence of Game Theory on Multiparty Computation has exemplified itself in modeling multiparty computation with a game-theoretic flavor by introducing *rational* parties with some natural utility functions into the computation. Once this is done, two main areas of investigation are as follows. First, we try to characterize the class of functions where it is in parties' *selfish interest* to report their true inputs to the computation. We call such functions *non-cooperatively computable* (NCC). Second, we can ask to what extent the *existing* MPC protocols (used to compute NCC functions) form an appropriate equilibrium for the extended game, where we remove the trusted mediator by cheap talk computing the same function. As we will see, the answer will depend on the strength of the equilibrium we desire (and, of course, on the natural utilities we assign to the "function evaluation game" defined below). Furthermore, issues arising in the MPC "honest vs. malicious" setting also hold in the Game Theory "rational" setting, further providing a synergy between these two fields.

1.5.1 Non-Cooperatively Computable Functions

In order to "rationalize" the process of securely evaluating a given function f, we first need to define an appropriate *function evaluation game*. For concreteness, we concentrate on single-output functions $f(t_1 \dots t_n)$, although the results easily generalize to the n-output case. We also assume that each input t_i matters (i.e., for some t_{-i} the value of f is not yet determined without t_i).

FUNCTION EVALUATION GAME. We assume the parties' types t_i are their inputs to f (which are selected according to some probability distribution D having full support). The action of each party P_i is its guess about the output s^* of f. The key question, however, is how to define the utilities of the parties. Now, there are several natural cryptographic considerations which might weight into the definition of party P_i's utility:

- *Correctness.* Each P_i wishes to compute f correctly.
- *Exclusivity.* Each P_i prefers others parties P_j not to learn the value of f correctly.
- *Privacy.* Each P_i wishes to leak as little as possible about its input t_i to the other parties.
- *Voyeurism.* Each P_i wishes to learn as much as possible about the other parties' inputs.

Not surprisingly, one can have many different definitions for a cryptographically motivated utility function of party P_i. In turn, different definitions would lead to different results. For concreteness, we will restrict ourselves to one of the simplest and, arguably, most natural choices. Specifically, we will only consider correctness and exclusivity, and will value correctness over exclusivity. However, other choices might also be interesting in various situations, so our choice here is certainly with a loss of generality.

A bit more formally, recall that the utility u_i of party P_i depends on the true type vector t of all the parties, and the parties' actions $s_1 \dots s_n$. Notice, the true type vector t determines the correct function value $s^* = f(t)$, and parties' actions determine the boolean vector $\mathsf{correct} = (\mathsf{correct}_1, \dots, \mathsf{correct}_n)$, where $\mathsf{correct}_i = 1$ if an only if $s_i = s^*$. In our specific choice of the utility function we will assume that the utilities of each party only depend on the boolean vector $\mathsf{correct}$: namely, which of the parties learned the output and which did not. Therefore, we will write $u_i(\mathsf{correct})$ to denote the utility of party P_i. Now, rather than assigning somewhat arbitrary numbers to capture correctness and exclusivity, we only state the minimal constraints that imply these properties. Then, the correctness constraint states that

$u_i(\mathsf{correct}) > u_i(\mathsf{correct}')$, whenever $\mathsf{correct}_i = 1$ and $\mathsf{correct}'_i = 0$. Similarly, exclusivity constraint states that if (a) $\mathsf{correct}_i = \mathsf{correct}'_i$, (b) for all $j \neq i$ we have $\mathsf{correct}_j \leq \mathsf{correct}'_j$, while (c) for some j actually $\mathsf{correct}_j = 0$ and $\mathsf{correct}'_j = 1$, then $u_i(\mathsf{correct}) > u_i(\mathsf{correct}')$. Namely, provided P_i has the same success in learning the output, it prefers as few parties as possible to be successful.

Non-Cooperatively Computable Functions. Having defined the function evaluation game, we can now ask what are the equilibria of this game. In this case, Nash equilibria are not very interesting, since parties typically have too little information to be successful with any nontrivial probability. On the other hand, it is very interesting to study correlated equilibria of this game. Namely, parties give their inputs t_i to the mediator M, who then recommends an action s_i^* for each party. Given that each party is trying to compute the value of the function f, it is natural to consider "canonical" mediator strategy: namely, that of evaluating the function f on the reported type vector t, and simply recommending each party to "guess" the resulting function value $s^* = f(t)$. Now, we can ask the question of characterizing the class of functions f for which this canonical strategy is indeed a correlated equilibrium of the function evaluation game. To make this precise, though, we also need to define the actions of the mediator if some party gives a wrong type to the mediator. Although several options are possible, here we will assume that the mediator will send an error message to all the parties and let them decide by themselves what to play.

Definition 1.7 We say that a function f is *non-cooperatively computable* (NCC) with respect to utility functions $\{u_i\}$ (and a specific input distribution D) if the above canonical mediated strategy is a correlated equilibrium of the function evaluation game. Namely, it is in the parties' selfish interest to honestly report their true inputs to the mediator.

We illustrate this definition by giving two classes of functions which are never NCC. Let us say that a function f is *dominated* if there exists an index i and an input t_i which determine the value of f irrespective of the other inputs t_{-i}. Clearly, for such an input t_i it is not in the interest of P_i to submit t_i to the mediator, as P_i is assured of $\mathsf{correct}_i = 1$ even without the help of M, while every other party is not (for at least some of its inputs). Thus, dominated functions cannot be NCC. For another example, a function f is *reversible* if for some index i and some input t_i, there exists another input t'_i and a function g, such that (a) for all other parties' inputs t_{-i} we have $g(f(t'_i, t_{-i}), t_i) = f(t_i, t_{-i})$, and (b) for some other parties' inputs t_{-i} we have

$f(t'_i, t_{-i}) \neq f(t_i, t_{-i})$. Namely, property (a) states that there is no risk in terms of correctness for P_i to report t'_i instead of t_i, while property (b) states that at least sometimes P_i will be rewarded by higher exclusivity. A simple example of such (boolean) function is the parity function: negating one's input always negates the outcome, but still in a manner easily correctable by negating the output back. Clearly, reversible functions are also not NCC.

In general, depending on the exact utilities and the input distribution D, other functions might also be non-NCC. However, if we assume that the risk of losing correctness is *always* too great to be tempted by higher exclusivity, it turns out that these two classes are the *only* non-NCC functions. (And, thus, most functions, like majority, are NCC.) More precisely, assume that the utilities and the input distribution D are such that for all vectors $\mathsf{correct}, \mathsf{correct}', \mathsf{correct}''$ satisfying $\mathsf{correct}_i = \mathsf{correct}'_i = 1$, $\mathsf{correct}''_i = 0$, we have $u_i(\mathsf{correct}) > (1-\epsilon)u_i(\mathsf{correct}') + \epsilon u_i(\mathsf{correct}'')$, where ϵ is the smallest probability in D. Namely, if by deviating from the canonical strategy there is even a minuscule chance of P_i not learning the value of f correctly, this loss will always exceed any potential gain caused by many other parties not learning the outcome as well. In this case we can show

Theorem 1.8 *Under the above assumption, a function f is NCC if and only if it is not dominated and not reversible.*†

Collusions. So far we concentrated on the case of no collusions; i.e. $k = 1$. However, one can also define (a much smaller class of) *k-Non-Cooperatively Computable* (k-NCC) functions, for which no coalition of up to k parties has any incentive to deviate from the canonical strategy of reporting their true types. One can also characterize k-NCC functions under appropriate assumptions regarding the utilities and the input distribution D.

1.5.2 Rational Multiparty Computation

Assume a given function f is k-NCC, so it is in the parties' own interest to contribute their inputs in the ideal model. We now ask the same question as in Section 1.4: can we replace the mediator computing f by a corresponding MPC protocol for f? Notice, by doing so the parties effectively run the cryptographic MPC protocol for computing f. Thus, a positive answer would imply that a given MPC protocol π not only securely computes f from a cryptographic point of view, but also from a game-theoretic, rational

† In fact, under our assumption that each party's input matters in some cases and D has full support, it is easy to see that every dominated function is also reversible.

point of view! Fortunately, since the function evaluation game is just a particular game, Theorem 1.5 immediately implies

Theorem 1.9 *If f is a k-NCC function (w.r.t. to some utilities and input distribution) and π is a MPC protocol securely computing f against a coalition of up to k computationally unbounded/bounded parties, then π is a k-resilient regular/computational Nash equilibrium for computing f in the corresponding extended game.*

From a positive perspective, this result shows that for the goal of achieving just a Nash equilibrium, current MPC protocols can be explained in rational terms, as long as the parties are willing to compute f in the ideal model. From a negative perspective, the latter constraint non-trivially limits the class of functions f which can be rationally explained, and it is an interesting open problem how to rationalize MPC even for non-NCC functions, for which the cryptographic definition still makes perfect sense.

Stronger Equilibria. As another drawback, we already mentioned that the notion of Nash equilibrium is really too weak to capture the rationality of extensive-form processes, such as multiparty computation protocols. Thus, an important direction is to try achieving stronger kinds of equilibria explaining current MPC protocols, or, alternatively, design robust enough MPC protocols which would achieve such equilibria. In Section 1.4.3 we briefly touched on several *general* results in this direction (which clearly still apply to the special case of the function evaluation games). Here we will instead concentrate on the *specifics* of computing the function under the correctness and exclusivity preferences defined in the previous section, and will study a specific refinement of the Nash equilibrium natural for these utility functions.

To motivate our choice, let us see a particular problem with current MPC protocols. Recall, such protocols typically consist of three stages; in the first two stages the parties enter their inputs and compute the secret sharing of the output of f, while the last stage consists of the opening of the appropriate output shares. Now we claim that the strategy of not sending out the output shares is always at least as good, and *sometimes better* than the strategy of sending the output shares. Indeed, consider any party P_i. The correctness of output recovery for P_i is not affected by whether or not P_i sent his own share, irrespective of the behavior of the other parties. Yet, not sending the share to others might, in some cases, prevent others from reconstructing their outputs, resulting in higher exclusivity for P_i. True, *along the Nash equilibrium path* of Theorem 1.9, such cases where the share of P_i was critical did not exhibit themselves. Still, in reality it seems that there is no incentive for any party to send out their shares, since this is never better, and *sometimes worse* than not sending the shares. This motivates the following definition.

Definition 1.10 We say that a strategy $s \in S_i$ is *weakly dominated* by $s' \in S_i$ with respect to S_{-i} if (a) there exists $s_{-i} \in S_{-i}$ such that $u_i(s, s_{-i}) < u_i(s', s_{-i})$ and (b) for all strategies $s'_{-i} \in S_{-i}$ we have that $u_i(s, s'_{-i}) \leq u_i(s', s'_{-i})$. We define *iterated deletion of weakly dominated strategies* (IDoWDS) as the following process. Let $\mathbf{DOM}_i(S_1, ..., S_n)$ denote the set of strategies in S_i that are weakly dominated with respect to S_{-i}. Let $S_i^0 = S_i$ and for $j \geq 1$ define S_i^j inductively as $S_i^j = S_i^{j-1} \backslash \mathbf{DOM}_i(S_1^{j-1}, \ldots, S_n^{j-1})$ and let $S_i^\infty = \bigcap_{j \geq 1} S_i^j$. Finally, we say that a Nash equilibrium $(x_1, \ldots x_n)$ *survives IDoWDS*, if each $x_i \in \Delta(S_i^\infty)$.
k-resilient Nash equilibria surviving IDoWDS are defined similarly.†

Now, the above discussion implies that the k-resilient Nash equilibrium from Theorem 1.9 does not survive IDoWDS. On a positive side, the only reason for that was because the basic secret sharing scheme where the parties are instructed to blindly open their shares does not survive IDoWDS. It turns out that the moment we fix the secret sharing scheme to survive IDoWDS, the resulting Nash equilibrium for the function evaluation game will survive IDoWDS too, and Theorem 1.9 can be extended to Nash equilibrium surviving IDoWDS. Therefore, we will only treat the latter, more concise problem. We remark, however, that although a Nash equilibrium surviving IDoWDS is better than plain Nash equilibrium, it is still a rather weak concept. For example, it still allows for "empty threats", and has other undesirable properties. Thus, stronger equilibria are still very desirable to achieve.

Rational Secret Sharing. Recall, in the (k, n)-secret sharing problem the parties are given (random valid) shares $z_1 \ldots z_n$ of some secret z, such that any k shares leak no information about z, while any $k+1$ or more shares reveal z. We can define the secret sharing game, where the objective of each party is to guess the value of z, and where we assume that parties' utilities satisfy the correctness and exclusivity constraints defined earlier. In the extended game corresponding to the secret sharing game, the parties can perform some computation before guessing the value of the secret. For our

† We notice that, in general, it matters in which order one removes the weakly dominated strategies. The specific order chosen above seems natural, however, and will not affect the results we present below.

communication model, we assume that it is strong enough to perform generic multiparty computation, since this will be the case in the application to the function evaluation game. (On the other hand, we will only need MPC with correctness and privacy, and not necessarily fairness.) Additionally, if not already present, we also assume the existence of a *simultaneous broadcast channel*, where at each round all parties can simultaneously announce some message, after which they atomically receive the messages of all the other parties. Our goal is to build a preamble protocol for which the outcome of all the parties learning the secret z will be a k-resilient Nash equilibrium for the extended game which survives IDoWDS.

As we observed already, the natural 1-round preamble protocol where each party is supposed to simply broadcast its share does not survive IDoWDS. In fact, a simple backward induction argument shows that any preamble protocol having an a-priori fixed number of simultaneous broadcast rounds (and no other physical assumptions, such as envelopes and ballot-boxes) cannot enable the parties to rationally learn the secret and survive IDoWDS. Luckily, it turns out that we can have *probabilistic* protocols with no fixed upper bound on the number of rounds, but which have a constant *expected* number of rounds until each party learns the secret. We sketch the simplest such protocol below. W.l.o.g. we assume that the domain of the secret sharing scheme is large enough to deter random guessing of z, and also includes a special value denoted $\perp$, such that z is guaranteed to be different from $\perp$.

Let $\alpha \in (0,1)$ be a number specified shortly. At each iteration $r \geq 1$, the parties do the following two steps:

(i) Run a MPC protocol on inputs z_i which computes the following probabilistic functionality. With probability α, compute fresh and random (k,n)-secret sharing $z'_1 \ldots z'_n$ of z, where party P_i learns z'_i. Otherwise, with probability $1-\alpha$ compute a random (k,n)-secret sharing $z'_1 \ldots z'_n$ of $\perp$, where party P_i learns z'_i.†

(ii) All parties P_i simultaneously broadcasts z'_i to other parties.

(iii) If either the MPC protocol fails for even one party, or even one party failed to broadcast the value z'_i, all parties are instructed to abort.

(iv) Each party tries to recover some value z' from the shares received by the other parties. If the recovery fails, or at least one share is inconsistent with the final value z', the party aborts the preamble. Otherwise, if $z' = \perp$ the parties proceed to the next iteration, while

† This protocol is typically pretty efficient for the popular Shamir's secret sharing scheme.

in case $z' \neq \perp$ the parties stop the preamble and output z' as their guess for z.

Notice, by the privacy of the MPC step, no coalition C of up to k parties knows if the value z' is equal to z or $\perp$. Thus, in case this coalition chooses not to broadcast their shares, they will only learn the value z (while punishing all the other parties) with probability α, and not learn the value z forever with probability $1-\alpha$. Thus, if α is small enough (depending on the particular utilities), the risk of not learning the secret will outweigh the gain of achieving higher exclusivity. Also, it is easy to see that no strategy of the above protocol is weakly dominated by another strategy, so the above Nash equilibrium survives IDoWDS.

The above protocol works for *any* k. However, it runs in expected $O(1/\alpha)$ iterations, which is constant, but depends on the specific utilities of the parties (and the value k). Somewhat more sophisticated protocols are known to work for not too large k, but have expected number of iterations which is independent of the utilities. These results are summarized without further details below.

Theorem 1.11 *Assume the parties utilities satisfy correctness over exclusivity properties for the (k, n)-secret sharing game. Then there exists k-resilient Nash equilibria for the extended game which survive IDoWDS and run in expected constant number of iterations r, where*

- *$k < n$, but r depends on the specific utilities.*
- *$k < n/2$, r is fixed, but the parties still need to know a certain parameter depending on the specific utilities.*
- *$k < n/3$, r is fixed, and no other information about the utilities is needed.*

1.6 Conclusions

As we have seen, the settings of MPC in cryptography and correlated equilibrium in game theory have many similarities, as well as many differences. Existing results so far started to explore these connections, but much works remains to be done. For example, can we use some flavors of MPC to remove the mediator, while achieving very strong types of Nash equilibria, but with more realistic physical and other set-up assumptions? Or can we use game theory to ?rationalize? MPC protocols for non-NCC functions (such as parity), or to explain other popular cryptographic tasks such as commitment or zero-knowledge proofs? Additionally, so far ?rationalizing? MPC using game theory resulted only in more sophisticated protocols. Are there natural instances where assuming rationality will simplify the design of cryptographic tasks?

1.7 Notes

The multiparty computation problem (Section 1.1) was introduced in Yao [Yao82]. The basic definitional and construction approaches were introduced in Goldreich, Micali and Wigderson [GMW87], in particular the paradigm of a real/ideal execution. In Section 1.1.1 we follow the definitional framework of Canetti [Can00], which is based on the works of Goldwasser and Levin, Micali and Rogaway, and Beaver [GL90, MR91, Bea91] (respectively). The results mentioned in Theorem 1.2 are from the following: parts i.a and i.b from Goldreich, Micali and Wigderson [GMW87], part i.c from Lepinski, Micali, Peikert and shelat [LMPS04], part ii.a from Ben-Or, Goldwasser and Wigderson [BGW88] and Chaum, Crepeau and Damgard [CCD88], part ii.b from Rabin and Ben-Or [RB89] and Beaver [Bea91], part ii.c from Izmalkov, Lepinski and Micali [ILM05]. The secret Sharing protocol presented is Shamir's Secret Sharing [Sha79]. The notion of indistinguishability was introduced in Goldwasser and Micali [GM84]. For a more formal and in depth discussion on multiparty computations see Goldreich [Gol04].

In Section 1.2 we present the classical results of Nash [Nas51] and Aumann [Aum74] for Nash and correlated equilibrium (respectively). The extension of correlated equilibrium to games with incomplete information is due to Forges [For86]. The notion of extended games is from Barany [Bar92]. For a broader game theory background, see the book by Osborne and Rubinstein [OR99].

The comparison discussion between Game Theory and Cryptography, as it appears in Section 1.3, was initiated in Dodis, Halevi and Rabin [DHR00] and later expanded by Feigebaum and Shenker [FS02]; yet here we further expand on these points. The related discussion was also carried out in many other works, such as [Bar92, LMPS04, ILM05, ADGH06].

The notion of computational equilibrium which appears in Section 1.4.1 was introduced in [DHR00]. The work of Urbano and Vila [UV02, UV04] also deals with the computational model, but does not explicitly define this notion. The importance of tolerating collusions was first addressed in our setting by [FS02]. For the k-resilient equilibrium we chose the formulation of Abraham, Dolev, Gonen and Halpern [ADGH06], as we felt it best suited our presentation. For other related formulations, see the references in [ADGH06], and also a recent work of Lysyanskaya and Triandopoulos [LT06]. The results which appear in Section 1.4.2 appear in the following. Theorem 1.5 follows by combining results such as [DHR00, Bar92, BP98, Ger04, UV02, UV04] and [ADGH06]. The result for using fair MPC appears in [LMPS04]. The introduction of a min-max punishment to deal with unfair MPC in the attempt to remove the mediator appears in [DHR00]. For some efficiency improvements to the protocol of [DHR00], see the works of Teague [Tea04] and Attalah et al. [ABFL06]. The results which appear in Section 1.4.2 appear in the following. The worst equilibrium punishment technique was first applied to unmediated games by Ben-Porath [BP98]. The notion of collusion free protocols which is used to implement ex ante equilibria is from the work of Lepinski, Micali and shelat [LMas05]. The result of achieving strategic and privacy equivalence under physical assumptions is from [ILM05].

The non-cooperative computation formulation and some discussion used in Section 1.5.1 are introduced (for $k = 1$) by Shoham and Tennenholtz [ST05], and expanded by McGrew, Porter and Shoham [MPS03]. Theorem 1.8 is also from [ST05], while the formulation of "correctness followed by exclusivity" utilities is from Halpern and Teague [HT04]. The results in Section 1.5.2 appear as follows. The introduction of rational secret sharing surviving IDowDS and the impossibility result of reaching it in a fixed number of rounds are from [HT04]. The protocol for rational secret sharing we present appears in [ADGH06] and (for $k = 1$) by Gordon and Katz [GK06]. Yet, a more complicated and less general solution along these lines appeared first (for $k = 1$) in [HT04]. Theorem 1.11 is from [ADGH06]. For a different, but related "mixed MPC" model, see [LT06].

Acknowledgments. We would like to thank the following people for extensive discussions, explanations and general advice: Ittai Abraham, Ran Canetti, Hugo Krawczyk, Matt Lepinski, Anna Lysyanskaya, Silvio Micali, abhi shelat, Nikos Triandopoulos and special thanks to our co-author Shai Halevi.

Bibliography

[ABFL06] Mikhail Atallah, Marina Blanton, Keith Frikken, and Jiangtao Li. Efficient Correlated Action Selection. In *Financial Cryptography*, Lecture Notes in Computer Science. Springer, 2006.

[ADGH06] Ittai Abraham, Danny Dolev, Rica Gonen, and Joe Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In *PODC '06*, pages 53–62. ACM Press, 2006.

[Aum74] R. Aumann. Subjectivity and Correlation in Randomized Strategies. *Journal of Math. Econ.*, 1:67–96, 1974.

[Bar92] I. Barany. Fair Distribution Protocols or How the Players Replace Fortune. *Mathematics of Operations Research*, 17(2):327–341, 1992.

[Bea91] Donald Beaver. Secure multiparty protocols and zero-knowledge proof systems tolerating a faulty minority. *J. Cryptology*, 4(2):75–122, 1991.

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness Theorems for Noncryptographic Fault-Tolerant Distributed Computations. In *Proc. 20th STOC* [sto88], pages 1–10.

[BP98] Elchanan Ben-Porath. Correlation without Mediation: Expanding the Set of Equilibrium Outcomes by "Cheap" Pre-play Procedures. *Journal of Economic Theory*, 80(1):108–122, 1998.

[Can00] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000. Available at eprint.iacr.org/1998/018.

[CCD88] D. Chaum, C. Crepeau, and I. Damgard. Multiparty Unconditionally Secure Protocols. In *Proc. 20th STOC* [sto88], pages 11–19.

[DHR00] Y. Dodis, S. Halevi, and T. Rabin. A Cryptographic Solution to a Game Theoretic Problem. In *Crypto 2000*, pages 112–130, 2000. LNCS No. 1880.

[For86] Francoise M Forges. An approach to communication equilibria. *Econometrica*, 54(6):1375–85, 1986.

[FS02] Joan Feigenbaum and Scott Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 1–13. ACM Press, 2002.

[Ger04] Dino Gerardi. Unmediated Communication in Games with Complete and Incomplete Information. *Journal of Economic Theory*, 114:104,131, 2004.

[GK06] S. Dov Gordon and J. Katz. Rational secret sharing, revisited. In *5th Conference on Security and Cryptography for Networks*, 2006. Updated version available at http://eprint.iacr.org/2006/142.

[GL90] S. Goldwasser and L. Levin. Fair computation of general functions in presence of immoral majority. In *Crypto '90*, pages 77–93, 1990. LNCS No. 537.

[GM84] S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, April 1984.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game. In *Proc. 19th STOC*, pages 218–229. ACM, 1987.

[Gol04] Oded Goldreich. *Foundations of Cryptography: Volume 2*. Cambridge University Press, 2004. Preliminary version *http://philby.ucsd.edu/cryptolib.html/*.

[HT04] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *Proc. of 36th STOC*, pages 623–632. ACM Press, 2004.

[ILM05] Sergei Izmalkov, Matt Lepinski, and Silvio Micali. Rational secure computation and ideal mechanism design. In *Proc. of 46^{th} FOCS*, pages 585–595, 2005.

[LMas05] Matt Lepinksi, Silvio Micali, and abhi shelat. Collusion-free protocols. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 543–552. ACM Press, 2005.

[LMPS04] Matt Lepinski, Silvio Micali, Chris Peikert, and Abhi Shelat. Completely fair sfe and coalition-safe cheap talk. In *PODC '04: Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pages 1–10. ACM Press, 2004.

[LT06] Anna Lysyanskaya and Nikos Triandopoulos. Rationality and adversarial

Behavior in Multi-Party Computation . In *Crypto 2006*, 2006.

[MPS03] Robert McGrew, Ryan Porter, and Yoav Shoham. Towards a general theory of non-cooperative computation (extended abstract). In *TARK IX*, 2003.

[MR91] S. Micali and P. Rogaway. Secure Computation. In *Crypto '91*, pages 392–404, 1991. LNCS No. 576.

[Nas51] John Nash. Non-Cooperative Games. *Annals of Mathematics*, 54:286–295, 1951.

[OR99] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory.* MIT Press, 1999.

[RB89] T. Rabin and M. Ben-Or. Verifiable Secret Sharing and Multiparty Protocols with Honest Majority. In *Proc.* 21*st STOC*, pages 73–85. ACM, 1989.

[Sha79] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22:612–613, 1979.

[ST05] Yoav Shoham and Moshe Tennenholtz. Non-cooperative computation: Boolean functions with correctness and exclusivity. *Theor. Comput. Sci.*, 343(1-2):97–113, 2005.

[sto88] *Proc.* 20*th STOC.* ACM, 1988.

[Tea04] V. Teague. Selecting Correlated Random Actions. In *Financial Cryptography*, volume 3110 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2004.

[UV02] Amparo Urbano and Jose E. Vila. Computational Complexity and Communication: Coordination in Two-Player Games. *Econometrica*, 70(5):1893–1927, 2002.

[UV04] Amparo Urbano and Jose E. Vila. Computationally Restricted Unmediated Talk under Incomplete Information. *Economic Theory*, 23:283–320, 2004.

[Yao82] A.C. Yao. Protocols for secure computations. In Proceedings of FOCS'82, pages 160–164, Chicago, 1982. IEEE.