

# Transparent Computation and Correlated Equilibrium\*

Sergei Izmalkov      Matt Lepinski      Silvio Micali      abhi shelat

## Abstract

Achieving correlated equilibrium is an important and extensively investigated problem at the intersection of many fields: in particular, game theory, cryptography and efficient algorithms. Thus far, however, perfectly rational solutions have been lacking, and the problem has been formulated with somewhat limited objectives. In this paper, we

- Provide a stronger and more general notion of correlated-equilibrium achievement; and
- Provide more rational solutions in this more demanding framework.

We obtain our game theoretic results by putting forward and exemplifying a stronger notion of secure computation. Traditionally, secure computation replaces a trusted party by multiple players computing on their separate shares of the data. In contrast, we directly replace a trusted party by a *transparent device*<sup>1</sup> that correctly and privately evaluates any function by performing only *public operations on unseen data*. To construct such devices, we substantially strengthen the ballot-box techniques of [ILM05]. We demonstrate the additional power of transparent computation by proving that the game-theoretic results of this paper are unachievable by traditionally secure protocols.

---

\*This material is based upon work supported by the National Science Foundation under Grant SES-0551244. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

<sup>1</sup>The authors would like to thank Michael Rabin for suggesting the term.

# 1 Game Theoretic Background and Correlated Equilibrium

**NORMAL-FORM GAMES.** In a normal-form game  $G$  with  $n$  players, each player  $i$  has (1) his own finite set of actions,  $A_i$ , and (2) his own utility function,  $u_i$ , mapping the action space  $A = A_1 \times \dots \times A_n$  into the real numbers. The action sets and the utility functions of  $G$  are *common knowledge*. The game is played in a single stage, without inter-player communication. Each player  $i$  can be thought of as being isolated in his own room, facing a panel of buttons: one for each action in  $A_i$ . In such a setting,  $i$  plays action  $a_i$  by pushing the corresponding button, simultaneously with the other players. Letting  $a = (a_1, \dots, a_n)$  be the resulting *outcome*, each player  $i$  receives  $u_i(a)$  as his payoff. A strategy  $\sigma_i$  for player  $i$  is a probability distribution over  $A_i$ .

**EXTENSIVE-FORM GAMES.** An extensive-form game is played in multiple stages, with the players acting one at a time. Such a game can be depicted as a tree. The root represents the start of the game; an internal node an intermediate stage of the game; and a leaf an ending of the game. To each leaf a vector of utilities of all the players is assigned. The game also specifies which player acts at which node, and the actions available to him. The children of a non-leaf node  $N$  correspond to the actions available to the player designated to act at node  $N$ . (Extensive-form games may also have a special player, *Nature*, that, when called to act, plays actions chosen according to fixed probability distributions commonly known to all players.)

An extensive-form game is of *perfect information* when all the players know the exact actions played so far, and thus the current node of the tree. For such games, a *strategy* of a player  $i$  consists of a (possibly probabilistic) function specifying the action to take at any node where  $i$  must act.

An extensive-form game is of *imperfect information* if the acting player is not perfectly informed about the actions played so far. Therefore, even though the tree structure of the game, the payoffs of the leaf nodes, and so on, continue to be common knowledge, the acting player no longer knows the exact node he is at. The information in his possession allows him to exclude many nodes, but is compatible with his being at one of several others. Effectively, the nodes of the tree are partitioned into so called *information sets*. Two nodes belong to the same information set of player  $i$  if  $i$  cannot distinguish between them. The information sets of different players are disjoint, and the game specifies the acting player at each information set.

**RATIONAL PLAY.** Solving a game means finding the ways in which rational players can play it. A solution to a normal-form game  $G$  is a Nash equilibrium. This is a profile  $\sigma = (\sigma_1, \dots, \sigma_n)$  of strategies that are *self-reinforcing*: that is, no player  $i$  has an incentive to deviate from his own strategy if he believes that all other players stick to their own strategies. Formally,  $\sigma$  is a Nash equilibrium if, for all players  $i$  and for all strategies  $\hat{\sigma}_i$ ,  $u_i(\sigma_i, \sigma_{-i}) \geq u_i(\hat{\sigma}_i, \sigma_{-i})$ .<sup>2</sup>

Nash equilibria could be defined syntactically in the same way for extensive-form games, but they would no longer be impeccably rational. In a normal-form game, the strategies of a Nash equilibrium  $\sigma$  are best responses to each other in a setting where all players act simultaneously. The players cannot see whether the others act according to their equilibrium strategies or deviate. In an extensive-form game, instead, players act over time, they may observe actions of the others inconsistent with equilibrium strategies and react accordingly. In a Nash equilibrium  $\sigma$ , however,  $\sigma_i$  *need not* be  $i$ 's best response if  $i$  notices that  $j$  deviated from  $\sigma_j$ . For example, a Nash equilibrium

---

<sup>2</sup>For comprehensive coverage of game-theoretic concepts see [OR97]. Following standard notation,  $\sigma_{-i}$  denotes the vector of strategies in  $\sigma$  for all players except  $i$ , and the utility function  $u_i$  evaluated on a vector of  $n$  strategies—rather than an outcome of  $n$  actions—refers to  $i$ 's *expected* utility, arising from the  $n$  underlying distributions.

concerning people properly standing in line may be supported by strategies such as “if you cut in front of the line, I will explode a grenade and kill both of us.” Clearly, such *empty threats* should not be taken too seriously. A rational player may reason “If I cut in front of you, then, when it is time for you to respond, you will act rationally and will not explode a grenade since you want to live. Thus, if I cut in front of you, I will receive a higher utility, so I will cut.”

The appropriate notion of rationality for extensive-form games is that of *sequential rationality*. Sequential rationality mandates that players act in their best interest in any situation that may possibly arise during play (and thus even at points where some players have already deviated from their equilibrium strategies). In extensive-form games with perfect information, sequential rationality is precisely captured by the notion of a *Subgame-Perfect Equilibrium*. A strategy profile  $\sigma$  is a subgame-perfect equilibrium if  $\sigma_i$  is the best response for player  $i$  at any node of the game tree where  $i$  acts, provided that in the future (from that node onward) all players stick to  $\sigma$ . Unfortunately, no equally compelling equilibrium notions are guaranteed to exist for extensive-form games of imperfect information. For such games the most popular notion by far is that of a *sequential equilibrium*.<sup>3</sup> Every game of imperfect information admits a sequential equilibrium. However, despite their name!, sequential equilibria are far from capturing sequential rationality: even Kreps himself concedes the artificiality of some of its aspects.

**CORRELATED EQUILIBRIUM.** As introduced by Aumann [Aum74], a *correlated equilibrium* for a normal-form game  $G$  is a probability distribution,  $E$ , over the action space  $A_1 \times \dots \times A_n$ , satisfying the following informal property. If —somehow!— (1) a profile of actions  $(a_1, \dots, a_n)$  is chosen according to  $E$ , and (2) each player  $i$  enters in possession of  $a_i$  as his “recommended” action (without gaining any other information about  $a_{-i}$ ) then, if the other players stick to their recommendations, no single player can improve his expected payoff by replacing his recommended action by a different one. Nash equilibria are thus special cases of correlated ones: a correlated equilibrium is a joint distribution over  $A_1 \times \dots \times A_n$ , while a Nash equilibrium is a product distribution.

**A FUNDAMENTAL QUESTION.** Correlated equilibria enjoy two crucial properties:

- They can be *efficiently computed* (see [Pap05]) —an open problem for Nash equilibria— and
- They offer to all players payoffs *at least equal and often much greater* than Nash equilibria.

For practical purposes, therefore, one might as well replace Nash equilibria by correlated ones. Correlated equilibria, however, have a main disadvantage over Nash ones: achieving them seems to require a lot of trust: in essence a trusted party  $T$  who first privately samples  $E$ 's distribution to get an  $n$ -tuple of recommended actions  $(a_1, \dots, a_n)$ , and then privately gives each  $a_i$  to his proper recipient without revealing him any other information. Thus, a natural question arises:

*Can correlated equilibrium be meaningfully achieved without trusted parties?*

In a sense, a positive answer would “practically bypass” the problem of computing Nash equilibria.

---

<sup>3</sup>As defined by Kreps and Wilson [KW82], sequential equilibria essentially assume a belief system  $\mu$  specifying, for each information set  $is$ , a probability distribution  $\mu_{is}$  over the nodes in  $is$ . (Such distributions can be interpreted as the “Bayesian implications” of the strategies the acting player believes have actually been used in reaching  $is$ .) Given such a  $\mu$ , one can compute expected payoffs from any strategy profile  $\sigma$  at any information set. A belief system should satisfy certain technical conditions aimed at preventing that  $\mu$  may be arbitrarily selected for the sole purpose of supporting a given strategy profile  $\sigma$ . Still, some degree of artificiality persists.

## 2 Problems of Traditional Correlated-Equilibrium Achievement

There are two traditional approaches to achieving correlated equilibrium. We improve both.

### 2.1 Correlating Devices

As originally envisaged by Aumann, for at least some correlated equilibria  $E$ , the trusted party might be replaceable by a physical correlating *device*  $D$ . The approach is certainly appealing: while an *individual* could be bribed to sample the players' actions incorrectly, or to leak to some players the recommended actions of other players, there is no such a thing as bribing a device.

#### Problems of Correlating Devices

Formally distilling the exact requirements that a correlating device  $D$  must satisfy appears quite difficult. We distinguish three main and separate difficulties:

- *Inspection*. Is it possible to inspect  $D$  so as to guarantee that it will work according to its specification?
- *Privacy (Given Successful Inspection)*. Can we prove that scrupulously following  $D$ 's specifications will not leak information about the players's recommended actions?
- *Universality*. Is there a single device  $D$  that works for all correlated equilibria  $E$ ?

General-purpose computers are certainly universal, but too complex to be truly inspectable: they are just too easy to manufacture so as to hide a malicious “malfunctioning.” In addition, being physical devices, they must obey the laws of physics, and Nature can have a say on whether they can operate in a private manner. Indeed, even honestly manufactured computers leak all kinds of physical observables —power consumption, electromagnetic radiation, etc.— that (contrary to most intuition!) can be and indeed *have been* exploited to recover the internal data on which they operate! (See [JKJ99, AARR02] for concrete example cases, and [MR04] and [GLMMR04] for a more general theory of security for physical computers.)

If digital devices are problematic, mechanical ones are even more so. For instance, assume that  $D$  consists of a box with a hole on the top, and that the probabilism necessary for  $D$  to sample the correlated distribution  $E$  is implemented by dropping a ball in the hole, and having it bounce through on a pyramid of pegs properly placed underneath. Then, inspecting such a  $D$  may be equally hard.<sup>4</sup> In addition, even if any malfunctioning could be ruled out, the action-reaction principle implies that  $D$  moves to “the right” if the ball falls to “the left” of the first peg, so that one could predict the first random bit (and indeed all other bits as well) used by  $D$ .<sup>5</sup> Finally, such  $D$  appears far from being universal, and its size is easily exponential in the binary description of  $E$ .<sup>6</sup> Even if each potential attack to the correctness or privacy of a correlating device could be adequately countered by a practical defence, such a patching process could prove endless. In sum, correlating devices have not been mathematically modeled, and a natural question arises:

*Can we distill a few and simple physical assumptions sufficient for proving the existence of correlating devices?*

---

<sup>4</sup>For instance, the pegs could be manufactured so that, after the number of trials expected in an inspection, they bend or dent even so slightly so as to make the last way to fall substantially more likely.

<sup>5</sup>Of course, anchoring  $D$  to the table, would just imply that the table would move to “the right,” and so on.

<sup>6</sup>Assuming that there are 10 players, each having 20 actions, then it is easy to define  $E$  so that it has a compact description, while the number of pegs of  $D$  is  $2^{100}$ .

## 2.2 Extended Games

A second —and much studied— approach to achieving correlated equilibrium was put forward by Barany [Bar92], utilizing the auxiliary concept of an extended game.

An extended game for a normal-form game  $G = (n, A, u)$  is a pair  $G' = (C, G)$ , where  $C$  is a communication protocol in which every player  $i$  outputs an action in  $A_i$ . Such a pair represents a game of extensive form played in two phases. In the first phase, the players execute  $C$  (so as to compute a profile of actions —allegedly to play them in  $G$ ); in the second phase, they play  $G$ , thus receiving the utilities of  $G$ . Thus, the first phase, commonly referred to as *cheap talk*, does not have utilities of its own (in game theoretic jargon it is a *game form*): the players of  $G'$  receive the utilities of  $G$  only in the second phase. Without loss of generality, player  $i$ 's strategy in  $G'$  coincides with his strategy in  $C$ . (Such strategy specifies which messages  $i$  should send in the cheap-talk phase, and, because playing the normal-form  $G$  involves no communication,  $i$  may as well choose which action to play in  $G$  as his last step of  $C$ .)

**Note:** any communication protocol  $C$ , and thus any extended game  $G'$ , crucially depends on the underlying communication model: i.e., the devices and infrastructure specifying how messages are exchanged.

In this background, achieving a correlated equilibrium  $E$  in a normal-form game  $G$  is defined as

*Constructing an **extended game**  $G'$  having an equilibrium  $E'$  with the same payoffs as  $E$ .*

Within this definitional framework, the main goals for achieving  $E$  in  $G$  consist of ensuring that:

- the equilibrium  $E'$  in  $G'$  is as strong as possible;
- $G'$  and  $E'$  exist for as many games  $G$  and equilibria  $E$  as possible; and (from a CS perspective)
- $G'$  is both easy to find and easy to play.

In the original construction of Barany,  $G$  was required to have 4 or more players, and  $E'$  consisted of an ordinary Nash equilibrium in  $G'$ . Dodis, Halevi and Rabin [DHR00] (see also [UV02]), assuming that the players are computationally bounded, provide cryptographic extended games  $G'$  that handle even two players, but also in their construction  $E'$  is just an ordinary Nash equilibrium (actually, a *computational* one). Ordinary Nash equilibria, as discussed, may not be sufficiently rational as solutions to the extended games. In fact, the aforementioned constructions rely on punishments of deviating players that are not rational to carry out.<sup>7</sup>

Extended games  $G'$  having equivalent sequential equilibria  $E'$  were indeed constructed by Ben-Porath [BP98] and Gerardi [Ger04]. Both constructions utilize a rich communication model in their cheap-talk phases (in particular, *simultaneous broadcast channels*<sup>8</sup>), and yet —as shown in [LMPS04]— require exponential time from the players of  $G'$  to run their prescribed strategies. Further, their constructions cannot handle an arbitrary number of players: Gerardi requires that  $G$  has 5 or more players, and Ben-Porath that  $G$  has 3 or more players.

ACCEPTABLE CORRELATED EQUILIBRIA. Finally note that all extended games must satisfy a technical requirement, at least in any communication model in which it is apparent that a player

---

<sup>7</sup>Both [Bar92] and [DHR00] specify that if  $i$  aborts then the other players are instructed to choose actions that “harm  $i$  to the maximum possible extent” (technically a *Min-Max punishment*). Such punishments, however, are empty threats: to harm  $i$ , another player  $j$  may have to harm himself too. Thus it may actually be rational for  $j$  *not* to punish  $i$ , if the latter deviates.

<sup>8</sup>Ordinary broadcast guarantees that the sender of a message and the sent message become common knowledge. A simultaneous broadcast channel guarantees that a plurality of players can broadcast messages simultaneously guaranteeing that no one can change his message midstream based on the messages he hears from the others.

has stopped communicating. If a player  $i$  publicly aborts the cheap talk of  $G'$ , then he will not be able to compute his recommended action  $a_i$ . Thus, even if the other players succeeded in computing *their own* recommended actions, it would be questionable for them to play them in  $G$ . (Indeed, by playing  $a_{-i}$ , the players in  $-i$  cannot in general expect correlated-equilibrium payoffs: because  $i$  is not aware of  $a_i$ , whatever strategy he chooses is independent of  $a_i$ , and so, generically, the player and the resulting payoffs of all the players will be different.) Unlike a protocol, however, a game must always go on, and thus  $G'$  must specify what else to do —i.e., which strategy profile  $\sigma = (\sigma_1, \dots, \sigma_n)$  to play in  $G$ — if a player  $i$  publicly refuse to cheap talk. Such an alternative play has to be rational as well, typically  $\sigma$  is specified to consist of a specific Nash of  $G$ , call it  $N_i$ . And for  $i$  to prefer cheap talking (and thus receiving his payoff in  $E$ ) to triggering the play of  $N_i$ ,  $i$ 's payoff in  $E$  must be greater than his payoff in  $N_i$ . Therefore, in any model in which aborting the communication becomes apparent, traditionally and rationally reaching correlated equilibrium  $E$  is possible only if  $E$  is *acceptable* —i.e., if, for all players  $j$ , there exists a Nash  $N_j$  such that  $j$ 's payoff in  $E$  exceed  $j$ 's payoff in  $N_j$ .<sup>9</sup>

### Problems of Extended Games

THE PROBLEM OF UNKNOWN BELIEFS. As discussed in Section 2, a sequential equilibrium  $E'$  presupposes that at each information set the acting player has certain beliefs about past actions of the others. To support  $E'$ , the existence of these beliefs is mandatory, but *not their reasonableness*. (In Ben-Porath's construction, all players must believe that the only way for a player to deviate from his equilibrium strategy consists of choosing actions at random. In Gerardi's construction, when it is clear that at least one of 3 players has deviated but it is not clear who, one must believe that the lexicographically smallest player has not deviated.) Note that, if the players of a *given* extensive-form game  $G'$  of imperfect information *actually have* specific beliefs  $B'$  supporting  $E'$ , then  $E'$  is meaningful, independent of the reasonableness of  $B'$ : given that the players do believe  $B'$ , it is irrational for them to deviate from  $E'$  when playing  $G'$ . However, when *designing* an extended game  $G'$  to reach a correlated equilibrium  $E$ , one does not know the *real* players (e.g., the specific people) who will be playing  $G'$ , let alone what beliefs they may have! Indeed, the designer is only given a normal-form game  $G$  and  $E$  —that is, a number of players  $n$ ,  $n$  actions sets,  $n$  utility functions, and a probability distribution over  $n$ -tuples of actions.

In sum, to be meaningful, an extensive-form construction for achieving correlated equilibrium must be (1) sequentially rational; and (2) must not rely on a particular specification of players' beliefs (besides those that are induced by the play itself).

THE PROBLEM OF ADDITIONAL EQUILIBRIA. Bárány's approach to achieving  $E$  in  $G$  is content with  $G'$  having an equilibrium  $E'$  equivalent to  $E$ , and accepts the fact that  $G'$  introduces additional and *artificial* equilibria (i.e., not originally in  $G$ , nor arising from the players's recommendations). Game theory tells us that rational play ends in an equilibrium, but not which equilibrium. Thus additional equilibria make it more questionable whether  $E'$  will be actually played.<sup>10</sup>

THE PROBLEM OF COALITIONS. As argued by [LMPS04], the extended games of [Bár92], [BP98], and [Ger04] enable two colluding players to totally and undetectably control the outcome. Worrying

---

<sup>9</sup>This technical requirement is explicit in Ben-Porath's construction. It does not appear in Gerardi's construction only because in it the players cannot tell whether a player has abandoned the cheap talk.

<sup>10</sup>[SCS03] show that even two games having identical dominant strategy equilibria but different ordinary Nash equilibria are in practice played quite differently.

about coalitions may seem illogical when trying to achieve correlated equilibrium. Indeed, such an equilibrium notion is defined only in terms of single-player deviations, and thus the players wishing to achieve it must be convinced that there are no coalitions. But then: what is the source of this conviction? If it is an axiom, so be it. But if it originates from a cost-benefit analysis (i.e., from the analysis that the cost of running a coalition is higher than the benefits that its members might obtain), then worrying about coalitions in extended games is very relevant. Consider the following

**Example:** Let  $G$  be a normal-form game with 10 players, where each player has two actions,  $a$  and  $b$ , and the payoff function is as follows: If all players play  $a$ , then the even players receive a utility of 1 and the odd players a utility of 0; if all players play  $b$ , then all odd players receive utility 1 and the even players utility 0; if exactly 5 players play  $a$ , then players 1 and 2 receive utility  $10^{10}$  and each other player receives utility  $-10^{10}$ ; in all other cases, all players receive utility -1. In such a game, let  $E$  be the correlated equilibrium giving probability  $1/2$  to each of the action profiles  $(a, \dots, a)$  and  $(b, \dots, b)$ .

In the example, the expected payoff of  $E$  is  $1/2$  for every player. Moreover, no coalition of players, of any size, has any incentive to form and deviate from their recommendations. However, in all extended games except [LMPS04], players 1 and 2 can collude so that the action profile computed during the cheap talk is  $(a, a, a, a, a, b, b, b, b, b)$ , so as to enrich themselves and impoverish everyone else. And no other player can suspect anything wrong based on his own recommendation!

### 3 Transparent and Verifiable Correlating Devices

We now define and construct a new and powerful class of devices. For concreteness, they operate on *ballots*, utilizing the following variant of the model of [ILM05]. (See Appendix B for details.)

#### 3.1 The Ballot-Box Model

**BALLOTS.** There are two types of ballots: *envelopes* and *super-envelopes*. An envelope contains an integer between 1 and 5. A super-envelope contains between 2 and 5 envelopes. Externally, all ballots of the same type are equal. Because it is slightly larger than an envelope, a super-envelope tightly packs the envelopes it contains, so that their order is maintained when the super-envelope is moved about. Unless inserted into the ballot box, each sealed ballot always stays in public view. Therefore, all players can associate with each ballot a common identifier, a positive integer  $j$ . All newly created ballots, as well as the ballots exiting the ballot box receive fresh identifiers.

**BALLOT OPERATIONS.** There are six *public operations*: (1) publicly write an integer between 1 and 5 on a piece of paper and put it into a new, empty envelope; (2) publicly insert a sequence of up to 5 envelopes into a new super-envelope; (3) publicly reorder a sequence of envelopes according to some permutation  $\pi$ ; (4) publicly open an envelope so that its content is revealed to all players; (5) publicly open a super-envelope to produce the envelopes it contains—in the same order in which they were inserted; (6) publicly place a sequence of up to 5 ballots (of the same type) in the ballot-box which returns them in random order. There also are two additional *private operations*: (7) privately read the content of a publicly chosen envelope and reseal it; and (8) privately re-order a sequence of up to 5 publicly chosen envelopes according to a privately chosen permutation  $\pi$ .

**INFORMATION.** Performing any operation generates a *public record*, that is, a piece of public information. For instance, when opening envelope  $j$ , the fact that the performed operation is

“opening”, and that the object of such opening is envelope  $j$ , and that the content of  $j$  is  $c_j$  become public knowledge. Only operations 7 and 8 generate both a public record (respectively, “envelope  $j$  has been read and resealed” and “envelope sequence  $S$  has been re-ordered”) and a *private record* available only to the device (respectively, the content  $c_j$  of envelope  $j$  and the permutation  $\pi$ ). Operation 6 generates no private information for the device or any potential observer: that is, while it is public knowledge that —say— ballots 2, 4, 6 and 8 have been randomized, no one knows by which permutation. The *public transcript*  $R$  (the *private transcript*  $r$ ) of a sequence of operations is the ordered concatenation of the public record (private record, possibly empty) of each operation.

### 3.2 Transparent Devices

**Definition 1:** A (ballot-box) *device* is a deterministic function  $D$  that, on input a public and a private transcript, returns either a (valid) operation or the special symbol  $END$ .

A device  $D$  is *transparent* if never returns a private operation.

An execution of  $D$  starts with the empty public transcript  $R^0$ , the empty private transcript  $r^0$ , and an empty set of ballots  $B^0$ , and generates a sequence of triplets  $S = (R^0, r^0, B^0), (R^1, r^1, B^1), \dots$  as follows. As long as  $D(R^i, r^i)$  returns an operation  $op^i$  *valid* for the current ballot set  $B^i$  (i.e.,  $op^i$  consists of opening an envelope  $j$  indeed existing in  $B^i$ ), then  $op^i$  is executed on  $B^i$  so as to generate (a new public record and thus the) next public transcript  $R^{i+1}$ , a (possibly empty private record and thus the) next private transcript  $r^{i+1}$ , and the (updated) next ballot set  $B^{i+1}$ .

REMARKS.

- *Generality.* The notion of a device is actually very general. Essentially, it is a mechanism reacting to all available information, whether public or internal.
- *Inspectability.* Transparent devices —unlike traditional computers— are remarkably “inspectable.” Indeed, the public transcript  $R$  unambiguously specifies which public operations are actually performed. Thus, every one sees that the operations performed are those intended.

### 3.3 Verifiable Devices

Let us now informally define a less intuitive but more stringent notion of a ballot-box device, whose existence immediately implies that of a transparent device. Below, we call a distribution  $E : S \rightarrow [0, 1]$  *finite* if  $E$  has finite support and  $E(s)$  has finite binary expansion for any  $s \in S$ .

**Definition 2:** Let  $D$  be a (ballot-box) device, and  $E : (\Sigma^*)^n \rightarrow [0, 1]$  be a finite distribution. We say that  $D$  is *verifiable* for  $E$  if there exists a fixed string  $R$ , an integer  $c$ , an encoding  $\mathcal{E}$ , and a probabilistic polynomial-time algorithm  $\mathcal{G}$  such that, for any execution with public transcript  $R$ , the following two properties hold:

- (1) The final set of ballots consists of  $cn$  envelopes containing an  $n$ -tuple  $\sigma = (\sigma_1, \dots, \sigma_n)$  distributed according to  $E$  —that is,  $\sigma_1$  is the string  $\mathcal{E}$ -encoded by the contents of the first  $c$  envelopes,  $\sigma_2$  the string  $\mathcal{E}$ -encoded by the next  $c$  envelopes, and so on.
- (2) The final private transcript is independent of  $\sigma$  and distributed according to  $\mathcal{G}()$  —that is,  $r$  is easy to generate on empty input.

REMARKS.

- *Correctness.* Any one can tell that a verifiable device  $D$  for a distribution  $E$  has executed “the right operations.” In fact, any one can see the public transcript generated by  $D$ , and, whenever it coincides with  $R$ , property (1) guarantees that a properly distributed  $n$ -tuple of strings is contained in the final envelopes.
- *Privacy.* After an execution of a verifiable  $D$ , no one can gain information about  $\sigma = (\sigma_1, \dots, \sigma_n)$ . To begin with,  $\sigma$  is “inside a sequence of envelopes” and thus not directly observable. In addition,  $D$  starts with an empty set of ballots and, by property (1), ends with an empty set of ballots—except for the  $cn$  envelopes containing  $\sigma$ . Therefore, there is no “left-over” envelope whose content might conceivably contain information about  $\sigma$ . Nor can  $D$  contain information about  $\sigma$  somewhere else. In fact, even assuming that  $D$  reliably stored every thing it sees, the most it could store is the final private transcript. But such transcript, by property (2), is *independent* of  $(\sigma_1, \dots, \sigma_n)$ .
- *Generality.* Notice that the above notion of verifiable device for  $E$  works for any distribution  $E$  and not just for correlated equilibrium ones.
- *From Verifiability to Transparency.* Every verifiable device  $D$  is easily converted to a transparent  $D'$ , because the distribution over  $D$ 's private transcripts is trivial to generate.<sup>11</sup> Essentially, it suffices to replace (a) each operation “read and reseal envelope  $j$ ” by the operation “publicly open envelope  $j$ ” —so as to make the public content  $c_j$ — immediately followed by the operation “publicly make a new envelope  $j$  with content  $c_j$ ,” and (b) “privately re-order a sequence on ballots  $S$  according to permutation  $p$ ” by “publicly reorder  $S$  according to  $p$ .” The so obtained  $D'$  enjoys equivalent correctness and privacy as  $D$ , except that it generates a *random* public transcript  $R$  rather than a *fixed*  $R$ . Of course, this does not matter from the secure-computation point of view, but, as we explain in the Section 4, it matters a bit from the game-theory one.

### 3.4 Main Result

**Theorem 1:** *For every finite distribution  $E$  over  $(\Sigma)^n$  there is a verifiable device  $D$  for  $E$ .*

In Appendix C we actually prove a very constructive version of Theorem 1. Informally, we prove the existence of a linear-time algorithm  $\mathcal{C}$  (for “compiler”) that, on input (a description of) a finite rational distribution  $E$ , outputs (a description of) a unambiguous verifiable device  $D$  for  $E$ . An immediate consequence of Theorem 1 is the following

**Corollary 1:** *For any finite distribution  $E$  over  $(\Sigma)^n$  there is a transparent device  $D$  for  $E$ .*

NEW PARADIGMS AND TECHNIQUES WITH OLD OPERATIONS. Ballot-box techniques have already been used in [ILM05] for securely evaluating an arbitrary function, but following the 20-year old paradigm of [GMW87]. (Roughly, in the ILM protocol the security of the joint computation was still obtained by sharing the “global state”  $S$  among the players, so that each player  $i$  knows and operates upon only his local *share*  $S_i$  of  $S$ , and knowledge of any proper subset of the shares does not suffice to know  $S$ .) Transparent computing instead demands that the data computed upon is never seen by any one, not even in part, so that ultimately the computation can be carried out by

<sup>11</sup>In our construction, for example, private transcripts consist of uniformly selected permutations in  $S_5$ , the symmetric group of 5 elements.

a single device without having to trust it. This is indeed a new security paradigm, and achieving it requires forging new techniques. Our operations, however, are not new: they are a subset of the ILM ones! And our target continues to be general computation: the same mix of “ands, nots and coin tosses.” The novelty here lies in (1) distilling new goals, and then in (2) assembling the same components in new ways so as to achieve the new goals. But the same is true for most notions of Secure Computation and techniques to achieve it.

### 3.5 Transparent (and Verifiable) Computation

In the final paper we shall construct both transparent and verifiable devices for *general computation*. That is, devices that work for arbitrary probabilistic functions with inputs (provided, in envelope form, by the players). This is the strongest form of secure computation achieved so far. In particular, it satisfies the information theoretic notion of security of [DM00] and thus automatically enjoys universal composability [Can01, PW00].

## 4 Perfect Achievement of Correlated Equilibrium

TWO TYPES OF TRUSTED PARTIES. We actually distinguish two main types of trusted parties  $T$  depending on  $T$ 's ability to deliver the recommendations to proper player after having correctly sampled an action profile  $(a_1, \dots, a_n)$  according to  $E$ . A *forceful*  $T$  can always deliver every  $a_i$  to its intended recipient  $i$  (i.e., without any cooperation from  $i$ ). A *respectful*  $T$  lacks this unilateral ability, and thus asks all players (e.g., in lexicographic order) whether they want to receive their recommendations.<sup>12</sup> If they all answer YES, then he privately delivers each  $a_j$  to the proper recipient. Else, if  $i$  is the first player to answer NO, then the players are forced to fall back to some *alternative*  $\mathcal{A}_i$ . For instance,  $\mathcal{A}_i$  may consist of (1) playing a pre-specified Nash equilibrium  $N_i$  in  $G$ ; (2) having  $T$  sample and “respectfully deliver” to the players new recommendations correlated so as to better punish  $i$ ; etc.<sup>13</sup> Note that  $G$ ,  $E$  and  $\mathcal{A}_1, \dots, \mathcal{A}_n$  exactly specify the power of any coalition  $C$ . Thus, coalition power with a respectful trusted party can be controlled by choosing proper alternatives  $\mathcal{A}_1, \dots, \mathcal{A}_n$ .

TWO TYPES OF VERIFIABLE DEVICES. For selecting  $(a_1, \dots, a_n)$  according to  $E$ ,  $T$  can be perfectly replaced by a verifiable device  $D$  for  $E$ . Such a  $D$  will produce, for each player  $i$ , a sequence of envelopes  $E_i$  whose contents encode  $a_i$ . But now, as for  $T$ , we face the task of delivering all these  $E_i$ . If a forceful physical way of delivering envelopes exists, it can be readily incorporated into  $D$ , thus making it perfectly equivalent to a forceful  $T$ . If there is no such way,  $D$  can be made perfectly equivalent to such a respectful  $T$  by (1) handing each envelope sequence  $E_i$  to the proper recipient  $i$ ; (2) having the players accept or reject the handed envelopes; and (3) adopting the same alternatives  $\mathcal{A}_1, \dots, \mathcal{A}_n$  envisaged for a respectful  $T$ .

PERFECT ACHIEVEMENT OF CORRELATED EQUILIBRIUM VIA VERIFIABLE DEVICES. Perfect equivalence between a trusted party  $T$  for  $E$  and its corresponding verifiable device  $D$  for  $E$

<sup>12</sup>From the players point of view, forcefulness can be thought as modeling the players' inability of publicly refusing to receive their recommendations, and respectfulness as modeling such an ability. In the extreme, a player  $i$  may reject a respectful delivery by publicly shooting  $T$  as he approaches  $i$  with an envelope containing  $a_i$ . In which case, as we discussed in Section 3, it is meaningless for the players in  $-i$  to play the sub-profile of recommendations  $a_{-i}$ .

<sup>13</sup>In the second example,  $\mathcal{A}_i$  recursively provides for additional alternatives if some other players refuse to receive such new recommendations.

essentially means that, for any player  $i$ , his strategies and payoffs in the ideal game  $G^T$  with  $T$  are isomorphic to his strategies and payoffs in the game  $G^D$  with  $D$ . For instance, in the case of forceful  $T$  and  $D$ , a strategy of  $i$  in  $G^T$  is a function with *one* input—the recommendation  $a_i$  received from  $T$ ; while a strategy of  $i$  in  $G^D$  has *two* inputs: the recommendation  $\sigma_i$  received by  $D$  and the public transcript  $R$  generated by  $D$ . However,  $R$  is always the same, and thus is a dummy input. (Of course, player  $i$  knows that the entire recommendation profile, whether  $a$  or  $\sigma$ , is distributed according to  $E$ .)

#### REMARKS

- *Strategic Preservation and Stronger Rationality.* Perfect achievement of correlated equilibrium exactly preserves all strategic properties of the trusted-party game. (In particular, it preserves the set of all equilibria—correlated or not; the power of any coalition; etc.) In addition, there are no “rationality weaknesses” nor dependence on “questionable beliefs.” In fact, no player has any opportunity to form beliefs about the prior actions of the other players: in  $G^D$ , the players act only in  $G$ , and thus they act simultaneously!
- *Transparency vs. Verifiability.* While transparent and verifiable devices are de facto equivalent from a secure-computation point of view, transparent devices cannot preserve all equilibria intact. In fact, in addition to the final envelopes  $E_1, \dots, E_n$ , a transparent device also generates a random (rather than a fixed) public transcript. And it is well known that, in the presence of a public random signal, rational play yields the payoffs not just of the Nash equilibria of  $G$ , but also of their convex combinations!

PERFECT ACHIEVEMENT OF CORRELATED EQUILIBRIUM VIA EXTENDED GAMES. Our perfect achievement of correlated equilibrium via verifiable devices immediately yields perfect achievement of correlated equilibrium via extended games. The only difference between a verifiable  $D$  and a fixed player performing  $D$ ’s operations is that the player may abort. Therefore, having Player 1 perform all the operations of  $D$ , and then respectfully deliver the envelopes  $E_i$  to the right player  $i$ , constitutes an extended game  $G'$  that perfectly achieves  $E$  for all possible alternatives  $\mathcal{A}_1, \dots, \mathcal{A}_n$ .

PURELY SEQUENTIAL EQUILIBRIA. Though the strategic preservation property of our extended games is crucial, it alone cannot “guarantee” that rational players will receive  $E$ ’s payoffs. In particular, because all strategic options of  $G$  are exactly preserved in  $G'$ ,  $E$  co-exists with the original Nash equilibria of  $G$ , making unclear which equilibrium will end up being played. However, as traditionally assumed in prior work (see the last paragraph of Section 2.2), whenever  $E$  is acceptable and  $\mathcal{A}_i$  is chosen to be a Nash of  $G$  that rewards player  $i$  less than  $E$ , then our extended games  $G'$  satisfy a property never previously achieved. Essentially, *honestly playing the prescribed strategies in  $G'$  is sequentially rational*. That is, for any player  $i$ , for all information sets at which  $i$  is called to act, and for all beliefs  $i$  may have about past play, it is in  $i$ ’s best interest to stick to his prescribed strategy. We call this new and strongest notion *Purely Sequential Equilibrium*, and formalize it in Appendix D. Let us emphasize that such a strong equilibrium cannot exist for most extensive-form games of imperfect information, but remarkably exists for all our extended games whenever  $E$  is acceptable. Finally, in Appendix E, we prove that all prior secure-computation protocols—including the ballot-box ones of [ILM05]—*cannot* yield extended games with a purely sequential equilibrium. Purely sequential equilibrium crucially depends on the crucial property attained in this paper: meaningfully computing without looking at the data.

## References

- [AARR02] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao and Pankaj Rohatgi. The EM side-channel(s). In *Proceedings of CHES*, 2002.
- [Aum74] Robert Aumann. Subjectivity and correlation in randomized strategies. *J. Math. Econ.*, 1:67–96, 1974.
- [BCIK03] Jozsef Balogh, János Csirk, Yuval Ishai and Eyal Kushilevitz. Private Computation Using a PEZ Dispenser. *Theoretical Computer Science*, 1:69–84, 2003.
- [Bár92] Imre Bárány. Fair distribution protocols or how the players replace fortune. *Mathematics of Operation Research*, 17:327–341, May 1992.
- [BAR86] David Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC1. In *Proceedings of STOC*, 1986.
- [BP98] Elchanan Ben-Porath. Correlation without mediation: Expanding the set of equilibria outcomes by “cheap” pre-play procedures. *Journal of Economic Theory*, 80:108–122, 1998.
- [Can01] Ran Canetti. Universally composable security. A new paradigm for cryptographic protocols. *Proceedings of FOCS*, 2001.
- [CGMA85] Benny Chor, Shafi Goldwasser, Silvio Micali and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proceedings of FOCS*, 1985.
- [DHR00] Yevgeniy Dodis, Shai Halevi, and Tal Rabin. A cryptographic solution to a game theoretic problem. In *Proceedings of CRYPTO*, volume 1880 of *LNCS*. Springer-Verlag, 2000.
- [DM00] Yevgeniy Dodis and Silvio Micali. Parallel Reducibility for Information-Theoretically Secure Computation. In *Proceedings of CRYPTO*, volume 1880 of *LNCS*. Springer-Verlag, 2000.
- [GLMMR04] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali and Tal Rabin. Tamper Proof Security: Theoretical Foundations for Security Against Hardware Tampering. In *Proceedings of TCC*, 2004.
- [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 Jonathan Katz. Rational secret sharing, revisited. Cryptography ePrint Archive, Report 2006/142, 2006. Available at: <http://eprint.iacr.org/>
- [GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game. In *Proceedings of STOC*, 1987.
- [HT04] Joseph Halpern and Vanessa Teague. Rational secret sharing and multiparty computation. In *Proceedings of STOC*, 2004.
- [JKJ99] Paul Kocher, Joshua Jaffe and Benjamin Jun. Differential Power Analysis. In *Proceedings of CRYPTO*, 1999.
- [KW82] David Kreps and Robert Wilson. Sequential equilibria. *Econometrica*, 50:863–894, 1982.

- [ILM05] Sergei Izmalkov, Matt Lepinski, and Silvio Micali. Rational secure function evaluation and ideal mechanism design. In *Proceedings of FOCS*, 2005.
- [LMPS04] Matt Lepinski, Silvio Micali, Chris Peikert, and Abhi Shelat. Completely fair SFE and coalition-safe cheap talk. In *Proceedings of PODC*, 2004.
- [MR04] Silvio Micali and Leonid Reyzin. Physically Observable Cryptography. In *Proceedings of TCC*, 2004.
- [OR97] Martin J. Osborne and Ariel Rubinstein. *Game Theory*. MIT Press, 1997.
- [Pap05] Christos Papadimitriou. Computing correlated equilibria in multiplayer games. In *Proceedings of STOC*, 2005.
- [PW00] Birgit Pfitzmann and Michael Waidner. Composition and Integrity Preservation of Secure Reactive Systems. In *Proceedings of CCS*, 2000.
- [SCS03] Tatsuyoshi Saijo, Timothy N. Cason, and Thomas Sjoström. Secure Implementation Experiments: Do Strategy-proof Mechanisms Really Work? Discussion papers 03012, Research Institute of Economy, Trade and Industry (RIETI), available at <http://ideas.repec.org/p/eti/dpaper/03012.html>.
- [UV02] Amparo Urbano and Jose E. Vila. Computational complexity and communication: Coordination in two-player games. *Econometrica*, 70(5):1893–1927, 2002.
- [Y86] Andrew Yao. Never published. The result is presented in Lindell and Pinkas (2004): “A proof of Yao’s protocol for secure two-party computation,” available at: [http://www.cs.biu.ac.il/lindell/abstracts/yao\\_abs.html](http://www.cs.biu.ac.il/lindell/abstracts/yao_abs.html).

## A Notation

We denote by  $\mathbb{R}^+$  the set of non-negative reals; by  $\Sigma$  the alphabet consisting of English letters, arabic numerals, and punctuation marks; by  $\Sigma^*$  the set of all finite strings over  $\Sigma$ ; by  $\perp$  a symbol not in  $\Sigma$ ; by  $\text{SYM}_k$  the group of permutations of  $k$  elements; by  $\mathcal{I}$  the identity permutation; by  $x := y$  the operation that assigns value  $y$  to variable  $x$ ; by  $\phi$  the empty string; and by  $\emptyset$  the empty set. If  $A$  is a set, by  $A^0$  we denote the empty set, and by  $A^k$  we denote the Cartesian product of a set  $A$  with itself  $k$  times. If  $x$  is a sequence, we denote by  $x_i$  the  $i$ th element of  $x$ . If  $x$  is a sequence of  $k$  integers, and  $m$  is an integer, by  $x + m$  we denote the sequence  $x_1 + m, \dots, x_k + m$ . If  $x$  and  $y$  are sequences, respectively of length  $j$  and  $k$ , we denote by  $x \circ y$  the sequence of  $j + k$  elements obtained by appending  $y$  to  $x$  (i.e., the  $i$ th element of  $x \circ y$  is  $x_i$  if  $i \leq j$ , and  $y_{i-j}$  otherwise). If  $x$  and  $y$  are strings, we denote the concatenation of  $x$  and  $y$  by either  $xy$  or  $x|y$ . If  $p$  is a permutation in  $\text{SYM}_n$ , then we denote the value  $p(i)$  by  $p_i$  and represent  $p$  as the string  $p_1|p_2|\dots|p_n$ .

## B The Ballot-box Model

### B.1 Ballot-box Operations

**Definition 1.** An *envelope* is a triple  $(j, c, 0)$ , where  $j$  a positive integer, and  $c$  a symbol of  $\Sigma$ . A *super-envelope* is a triple  $(j, c, L)$ , where both  $j$  and  $L$  are positive integers, and  $c \in \Sigma^L$ . A *ballot* is either an envelope or a super-envelope. If  $(j, c, L)$  is a ballot, we refer to  $j$  as its *identifier*, to  $c$  as its *content*, and to  $L$  as its *level*. (As we shall see,  $L$  represents the number of inner envelopes contained in a ballot.)

Let  $B$  be a set of ballots,  $j$  an integer, and  $j_1, \dots, j_m$  a sequence of integers. Then, by  $\text{cont}_B(j)$  we denote the symbol  $c \in \Sigma$ , if  $B$  has a single envelope with identifier  $j$  and content  $c$ , and the symbol  $\perp$  otherwise; and by  $\text{cont}_B(j_1, \dots, j_m)$  we denote the symbol  $\perp$  if  $\text{cont}_B(j_k) = \perp$  for some element  $j_k$ , and the string  $\text{cont}_B(j_1)|\dots|\text{cont}_B(j_m)$  otherwise.

A set of ballots  $B$  is *well-defined* if distinct ballots have distinct identifiers (i.e.,  $c = c'$  and  $l = l'$  whenever  $(j, c, l)$  and  $(j, c', l')$  belong to  $B$ ). If  $B$  is a well-defined set of ballots, by  $B_j$  or the expression *ballot  $j$*  we denote the unique ballot of  $B$  whose identifier is  $j$ , and by  $I_B$  the set of identifiers of  $B$ 's ballots. To emphasize that ballot  $j$  actually is an envelope (super-envelope) we may use the expression *envelope  $j$*  (*super-envelope  $j$* ).

**Definition 2.** A *global memory* consists of a triple  $(B, R, P)$ , where

- $B$  is a well defined set of ballots;
- $R$  is a sequence of strings in  $\Sigma^*$ ,  $R = R_1, R_2, \dots$ ; and
- $H$  is a sequence of strings in  $\Sigma^*$ .

We refer to  $B$  as the *ballot set*; to  $R$  as the *public record*; to each element of  $R$  as a *record*; and to  $P$  as the *private record*. The *empty global memory*, is the global memory for which the ballot set, the public record, and private record are all empty. We denote the set of all possible global memories by  $GM$ .

**Definition 3.** Ballot-box operations are functions from  $GM$  to  $GM$ . The subset of ballot-box operations possible at a given global memory  $gm$  is denoted by  $\mathcal{O}_{gm}$ . The operations in  $\mathcal{O}_{gm}$  are described below, grouped in 8 classes. For each  $a \in \mathcal{O}_{gm}$  we provide a formal identifier; an informal imperative (to facilitate the high-level description of our constructions); and a functional

specification. If  $gm = (B, R, P)$ , we actually specify  $a(gm)$  as a program acting on variables  $B$ ,  $R$ , and  $P$ , plus the auxiliary variable  $maxid$ , the maximum identifier in  $I_B$ .

1. (NEWENV,  $c$ ) —where  $c \in \Sigma$ .  
“Make a new envelope with public content  $c$ .”  
 $R := R \circ (\text{NEWENV}, c)$ ;  $maxid := maxid + 1$ ; and  $B := B \cup \{(maxid, c, 0)\}$ .
2. (OPENENV,  $j$ ) —where  $j$  is an envelope identifier in  $I_B$ .  
“Publicly open envelope  $j$  to reveal content  $cont_B(j)$ .”  
 $R := R \circ (\text{OPENENV}, j, cont_B(j))$  and  $B := B \setminus \{B_j\}$ .
3. (NEWSUPER,  $j_1, \dots, j_L$ ) —where  $j_1, \dots, j_L$  are distinct envelope identifiers in  $I_B$ .  
“Create a new super-envelope containing the envelopes  $j_1, \dots, j_L$ .”  
 $R := R \circ (\text{NEWSUPER}, j_1, \dots, j_L)$ ;  $maxid := maxid + 1$ ;  
 $B := B \cup \{(maxid, (cont_B(j_1), \dots, (cont_B(j_L)), L))\}$ ; and  $B := B \setminus \{B_{j_1}, \dots, B_{j_L}\}$ .
4. (OPENSUPER,  $j$ ) —where  $j \in I_B$  is the identifier of a super-envelope of level  $L$ .  
“Open super-envelope  $j$  to reveal the envelopes  $maxid + 1, \dots, maxid + L$ .”  
 $R := R \circ (\text{OPENSUPER}, j)$ ; letting  $cont_B(j) = (c_1, \dots, c_L)$ ,  
 $B := B \cup \{(maxid + 1, c_1, 0), \dots, (maxid + L, c_L, 0)\}$ ;  $maxid := maxid + L$ ; and  $B := B \setminus \{B_j\}$ .
5. (PUBLICPERMUTE,  $j_1, \dots, j_K, p$ ) —where  $p \in SYM_K$  and  $j_1, \dots, j_K \in I_B$  are distinct identifiers of ballots with the same level  $L$ .  
“Publicly permute  $j_1, \dots, j_K$  according to  $p$ .”  
 $R := R \circ (\text{PUBLICPERMUTE}, j_1, \dots, j_K, p)$ ;  $B := B \cup \{(maxid + 1, cont_B(j_{\pi(1)}), L), \dots, (maxid + K, cont_B(j_{\pi(K)}), L)\}$ ;  $B := B \setminus \{B_{j_1}, \dots, B_{j_K}\}$ ; and  $maxid := maxid + K$ .
6. (BALLOTBOX,  $j_1, \dots, j_K$ ) —where  $j_1, \dots, j_K \in I_B$  are distinct identifiers of ballots with the same level  $L$ .  
“Ballot-box  $j_1, \dots, j_K$ ”  
 $R := R \circ (\text{BALLOTBOX}, \sigma)$ ;  $p \leftarrow rand(SYM_K)$ ;  $B := B \cup \{(maxid + p(1), cont_B(j_1), L), \dots, (maxid + p(K), cont_B(j_K), L)\}$ ;  $B := B \setminus \{B_{j_1}, \dots, B_{j_K}\}$ ; and  $maxid := maxid + K$ .
7. (READRESEAL,  $j$ ) —where  $j$  is an envelope identifier in  $I_B$ .  
“Read and Reseal envelope  $j$ .”  
 $R := R \circ (\text{READRESEAL}, j)$ ;  $P = P \circ (\text{SecretOpen}, j, cont_B(j))$  and  $B := B \setminus \{B_j\}$ .
8. (SECRETPERMUTE,  $j_1, \dots, j_K, p$ ) — where  $p \in SYM_K$  and  $j_1, \dots, j_K \in I_B$  are distinct identifiers of ballots with the same level  $L$ .  
“Secretly permute ballots  $j_1, \dots, j_K$  according to  $p$ .”  
 $R := R \circ (\text{SECRETPERMUTE}, j_1, \dots, j_K)$ ;  $B := B \cup \{(maxid + 1, cont_B(j_{\pi(1)}), L), \dots, (maxid + K, cont_B(j_{\pi(K)}), L)\}$ ;  $B := B \setminus \{B_{j_1}, \dots, B_{j_K}\}$ ;  $maxid := maxid + K$ ; and  $P := P \circ (\text{SecretPermute}, p)$ .

REMARK. All ballot-box operations are deterministic functions, except for the BALLOTBOX operations.

## C Our Construction

### C.1 Overview

To prove our main result, we construct a verifiable device  $D$  for any finite probability distribution  $E$ . That is, we provide an efficient compilation procedure that on input a probability distribution  $E$ , produces a description of a verifiable device  $D$  for  $E$ .

REPRESENTING  $E$ . We choose to represent  $E$  as a combinatorial circuit (consisting of AND and NOT gates) which on input a random binary string samples from  $E$ . Clearly, there are other general methods for representing a probability distribution. (For example,  $E$  could alternatively be represented as a table of probabilities where the  $i$ th entry in the table is the probability that  $E$  assigns to the  $i$ th tuple of actions.) However, such methods provide a less compact representation and thus make the task of efficiency producing  $D$  strictly easier.

ENVELOPE ENCODINGS.

**Definition 4** ([ILM05]). Let  $B$  be a well-defined set of ballots and  $\sigma$  a vector mapping the index set  $\{1, 2, 3, 4, 5\}$  to identifiers of  $B$ 's envelopes. Then  $\sigma$  is an *envelope encoding* (of a 5-element permutation) if  $\text{cont}_B(\sigma_1) | \cdots | \text{cont}_B(\sigma_5) \in \text{SYM}_5$ . In such a case, we shall denote by  $\sigma$  the permutation  $\text{cont}_B(\sigma_1) | \cdots | \text{cont}_B(\sigma_5)$ .

Whenever  $\sigma$  is an envelope encoding,  $(\sigma_j, \sigma_j, 0) \in B$  for all  $j$ , and  $\sigma$  is tautologically the envelope encoding of  $\sigma$ . We reserve lower-case Greek letters to denote envelope encodings.

EVALUATING  $E$ 'S CIRCUIT. We follow the high-level approach of [ILM05]. That is, we represent the bit 0 as an envelope encoding of the permutation  $\mathcal{I}$  and we represent the bit 1 as an envelope encoding of the permutation  $a = 12453$ .<sup>14</sup> To obtain a sample from  $E$ , we must first generate an encoding of the random string to use as input for  $E$ 's circuit. We accomplish this, by providing a procedure *Coin* that produces an envelope encoding that represents a random bit.<sup>15</sup> Then, as described in [ILM05], we use the techniques of Barrington [BAR86] to reduce the evaluation of a general combinatorial circuit to the multiplication, inversion and cloning<sup>16</sup> of (envelope encodings of) permutations.

Our construction of a device for  $E$  is thus efficient (that is, linear in the size of  $E$ ). Indeed, we locally replace each gate in the circuit with a constant number of operations and then add a constant number of operations to generate each of the inputs of the circuit.

THE REMAINDER OF THIS SECTION. To complete the sketch of our construction, all that remains is to specify the procedures *Invert*, *Multiply*, *Clone* and *Coin*. For each procedure we provide a brief argument that the procedure is correct —that is, that it computes the correct output and

<sup>14</sup>That is, in the definition of a verifiable circuit, we take  $c = 5$  and the encoding  $\mathcal{E}$  maps 12345 to 0 and 12453 to 1.

<sup>15</sup>That is, with probability  $\frac{1}{2}$  *Coin* produces an envelope encoding of  $\mathcal{I}$  and with probability  $\frac{1}{2}$  *Coin* produces an envelope encoding of 12453.

<sup>16</sup>Here, *cloning* a permutation is the operation that takes an envelope encoding of permutation  $p$ , and produces two new envelope encodings of permutation  $p$ . A cloning procedure is necessary because the procedure for multiplying permutations “destroys” the permutations being multiplied.

produces a fixed public record—and that it is private—that is, it produces a private record which can be simulated on empty input.

## C.2 Specifying A Verifiable Device

A verifiable device  $D$  is *finite* if there exists a positive integer  $K$  such that any execution of  $D$  will produce  $END$  in at most  $K$  steps. We specify a finite verifiable device  $D$  as a list of  $K$  items, the  $k$ th of which fully describes the  $k$ th operation produced when executing  $D$ . In describing ballot-box operations, we use the informal imperatives provided in Appendix B. When the ballot-box operation  $op$  returned by  $D$  is a function of the current public record  $R$  and private record  $P$ , we specify  $op$  by a program that returns  $op$  on inputs  $R$  and  $P$ .

SHORTCUTS. For brevity and clarity, we often contract the description of several rounds into a single, conceptual round. For instance:

- If  $\sigma$  is an envelope encoding, by the conceptual round “For  $\ell = 1$  to 5, publicly open envelope  $\sigma_\ell$ ” we mean the following 5 rounds
  - “Publicly open envelope  $\sigma_1$ .
  - Publicly open envelope  $\sigma_2$ .
  - Publicly open envelope  $\sigma_3$ .
  - Publicly open envelope  $\sigma_4$ .
  - Publicly open envelope  $\sigma_5$ .”
- If  $p$  is a permutation in  $\text{SYM}_5$ , by the conceptual round “Create an envelope encoding  $\sigma$  of  $p$ ” we mean the following 5 rounds
  - “Create a new envelope  $\sigma_1$  with public content  $p_1$ .
  - Create a new envelope  $\sigma_2$  with public content  $p_2$ .
  - Create a new envelope  $\sigma_3$  with public content  $p_3$ .
  - Create a new envelope  $\sigma_4$  with public content  $p_4$ .
  - Create a new envelope  $\sigma_5$  with public content  $p_5$ . Then, set  $\sigma = \sigma_1, \dots, \sigma_5$ .”
- If  $\mathcal{D}$  is sub-device taking an envelope encoding as input and producing an envelope encoding as an output, by the conceptual round “Set  $\beta = \mathcal{P}(\alpha)$ ” we mean all of the rounds of an execution of  $\mathcal{D}$  when  $\mathcal{D}$ ’s input is  $\alpha$  and  $\mathcal{D}$ ’s output is  $\beta$ .

## C.3 Device Invert

### Device Invert

*Inputs:*  $\sigma$  — an envelope encoding of a permutation in  $S_5$ .

1. Create an envelope encoding  $\alpha$  of the identity permutation.
2. For  $\ell = 1$  to 5: Create a new super-envelope  $A_\ell$  containing the pair of envelopes  $(\alpha_\ell, \sigma_\ell)$ .
3. Ballot-box  $A_1, \dots, A_5$  to obtain  $A'_1, \dots, A'_5$ .
4. For  $\ell = 1$  to 5: Open super-envelope  $A'_\ell$  to expose envelope pair  $(\mu_\ell, \nu_\ell)$ .
5. For  $\ell = 1$  to 5: Read and re-seal  $\nu_\ell$  and denote its contents by  $\nu_\ell$ . Let  $\nu$  be the permutation that maps  $j$  to  $\nu_j$ .

6. For  $\ell = 1$  to 5: Create a new super-envelope  $B_\ell$  containing the pair of envelopes  $(\boldsymbol{\mu}_\ell, \boldsymbol{\nu}_\ell)$ .
7. Privately permute  $B_1, \dots, B_5$  according to  $\nu^{-1}$  to produce  $B'_1, \dots, B'_5$ .
8. For  $\ell = 1$  to 5: Open super-envelope  $B'_\ell$  to expose envelope pair  $(\boldsymbol{\rho}_\ell, \boldsymbol{\phi}_\ell)$ .
9. For  $\ell = 1$  to 5: Publicly open envelope  $\boldsymbol{\phi}_\ell$ . (Note that the revealed contents of  $\boldsymbol{\phi}_\ell$  is always the integer  $\ell$ .)

*Output:*  $\boldsymbol{\rho}$ .

**CORRECTNESS.** To establish the correctness of *Invert* we must show that  $\boldsymbol{\rho}$  is an envelope encoding of  $\sigma^{-1}$ . Because Step 2 binds together, in the same super-envelope  $A_\ell$ , the  $\ell$ th envelope of  $\boldsymbol{\alpha}$  and  $\boldsymbol{\sigma}$ , Step 3 applies the same, random and secret, permutation to both  $\boldsymbol{\alpha}$  and  $\boldsymbol{\sigma}$ . Letting  $x$  be this secret permutation, Step 4 “puts on the table” the envelope encodings  $\boldsymbol{\mu} = \mu_1, \dots, \mu_5$  and  $\boldsymbol{\nu} = \nu_1, \dots, \nu_5$  where  $\boldsymbol{\mu} = x\boldsymbol{\alpha} = x$  and  $\boldsymbol{\nu} = x\boldsymbol{\sigma}$ . At the end of Step 4, both  $\boldsymbol{\nu}$  and  $\boldsymbol{\mu}$  are totally secret. Step 5, however, privately reveals  $\boldsymbol{\nu}$  so that the device can compute  $\nu^{-1}$ . Because Step 6 binds together the  $\ell$ th envelopes of  $\boldsymbol{\mu}$  and  $\boldsymbol{\nu}$ , in Step 7 the permutation  $\nu^{-1}$  is applied to both  $\boldsymbol{\mu}$  and  $\boldsymbol{\nu}$ . Therefore,  $\hat{\boldsymbol{\phi}} = \nu^{-1}\boldsymbol{\nu} = \boldsymbol{\alpha}$ . This ensures that the revealed contents in Step 9 are constant, as required. Additionally,  $\boldsymbol{\rho} = \nu^{-1}\boldsymbol{\mu} = \sigma^{-1}x^{-1}x = \sigma^{-1}$ , as required.

**PRIVACY.** The private transcript produced by *Invert* is the following sequence:

(*ReadReseal*,  $\nu_1$ )  
 (*ReadReseal*,  $\nu_2$ )  
 (*ReadReseal*,  $\nu_3$ )  
 (*ReadReseal*,  $\nu_4$ )  
 (*ReadReseal*,  $\nu_5$ )  
 (*PrivatePermute*,  $\nu^{-1}$ )

Where  $\boldsymbol{\nu} = x\boldsymbol{\sigma}$  and  $x$  is a random permutation produced by the ballot box. Therefore,  $\boldsymbol{\nu}$  is a random permutation independent of  $\boldsymbol{\sigma}$ .

Let  $\mathcal{G}$  be the algorithm that on empty input selects a random permutation  $y \in \text{SYM}_5$  and outputs the following sequence:

(*ReadReseal*,  $y_1$ )  
 (*ReadReseal*,  $y_2$ )  
 (*ReadReseal*,  $y_3$ )  
 (*ReadReseal*,  $y_4$ )  
 (*ReadReseal*,  $y_5$ )  
 (*PrivatePermute*,  $y^{-1}$ )

Since  $y$  is a random element of  $\text{SYM}_5$ , the sequences output by  $\mathcal{G}()$  are distributed identically to the private transcripts produced by executing *Invert*.

## C.4 Device Multiply

### Device Multiply

*Inputs:*  $\boldsymbol{\sigma}$  and  $\boldsymbol{\tau}$  —each an envelope encoding of a permutation in  $\text{SYM}_5$ .

1. Set  $\alpha = \text{Invert}(\sigma)$ .
2. For  $\ell = 1$  to 5: Create a new super-envelope  $A_\ell$  containing the pair of envelopes  $(\tau_\ell, \alpha_\ell)$ .
3. Ballot-box  $A_1, \dots, A_5$  to obtain  $A'_1, \dots, A'_5$ .
4. For  $\ell = 1$  to 5: Open super-envelope  $A'_\ell$  to expose envelope pair  $(\mu_\ell, \nu_\ell)$ .
5. For  $\ell = 1$  to 5: Read and re-seal  $\nu_\ell$  and denote its contents by  $\nu_\ell$ . Let  $\nu$  be the permutation that maps  $j$  to  $\nu_j$ .
6. For  $\ell = 1$  to 5: Create a new super-envelope  $B_\ell$  containing the pair of envelopes  $(\mu_\ell, \nu_\ell)$ .
7. Privately permute  $B_1, \dots, B_5$  according to  $\nu^{-1}$  to produce  $B'_1, \dots, B'_5$ .
8. For  $\ell = 1$  to 5: Open super-envelope  $B'_\ell$  to expose envelope pair  $(\rho_\ell, \phi_\ell)$ .
9. For  $\ell = 1$  to 5: Publicly opens envelope  $\phi_\ell$ . (Note that the revealed contents of  $\phi_\ell$  is always the integer  $\ell$ .)

*Output:*  $\rho$ .

**CORRECTNESS.** To establish the correctness of *Multiply* we must show that  $\rho$  is an envelope encoding of  $\sigma\tau$ . The correctness of *Invert* implies that, at the completion of Step 1,  $\alpha$  is an envelope encoding of permutation  $\alpha = \sigma^{-1}$ . Because Step 2 binds together, in the same super-envelope  $A_\ell$ , the  $\ell$ th envelope of  $\tau$  and  $\alpha$ , Step 3 applies the same, random and secret, permutation to both  $\alpha$  and  $\sigma$ . Letting  $x$  be this secret permutation, Step 4 “puts on the table” the envelope encodings  $\mu = \mu_1, \dots, \mu_5$  and  $\nu = \nu_1, \dots, \nu_5$ , where  $\mu = x\tau$  and  $\nu = x\alpha = x\sigma^{-1}$ . At the end of Step 4, both  $\nu$  and  $\mu$  are totally secret. Step 5, however, privately reveals  $\nu$  so that the device can compute  $\nu^{-1}$ . Because Step 6 binds together the  $\ell$ th envelopes of  $\mu$  and  $\nu$ , in Step 7 the permutation  $\nu^{-1}$  is applied to both  $\mu$  and  $\nu$ . Therefore,  $\hat{\phi} = \nu^{-1}\nu = \mathcal{I}$ . This ensures that the revealed contents in Step 9 are constant, as required. Additionally,  $\rho = \nu^{-1}\mu = \sigma x^{-1}x\tau = \sigma\tau$ , as desired.

**PRIVACY.** The private transcript produced by *Multiply* is the following sequence:

(*ReadReseal*,  $\nu_1$ )  
 (*ReadReseal*,  $\nu_2$ )  
 (*ReadReseal*,  $\nu_3$ )  
 (*ReadReseal*,  $\nu_4$ )  
 (*ReadReseal*,  $\nu_5$ )  
 (*PrivatePermute*,  $\nu^{-1}$ )

Where  $\nu = x\sigma$  and  $x$  is a random permutation produced by the ballot box. Therefore,  $\nu$  is a random permutation independent of  $\sigma$ .

Let  $\mathcal{G}$  be the algorithm that on empty input selects a random permutation  $y \in \text{SYM}_5$  and outputs the following sequence:

(*ReadReseal*,  $y_1$ )  
 (*ReadReseal*,  $y_2$ )  
 (*ReadReseal*,  $y_3$ )

*(ReadReseal, y<sub>4</sub>)*

*(ReadReseal, y<sub>5</sub>)*

*(PrivatePermute, y<sup>-1</sup>)*

Since  $y$  is a random element of  $\text{SYM}_5$ , the sequences output by  $\mathcal{G}()$  are distributed identically to the private transcripts produced by executing *Multiply*.

## C.5 Device Clone

### Device Clone

*Inputs:*  $\sigma$  — an envelope encoding of a permutation in  $S_5$ .

1. Set  $\alpha = \text{Invert}(\sigma)$ .
2. Create an envelope encoding  $\beta$  of the identity permutation.
3. Create an envelope encoding  $\gamma$  of the identity permutation.
4. For  $\ell = 1$  to 5: Create a new super-envelope  $A_\ell$  containing the triple of envelopes  $(\beta_\ell, \gamma_\ell, \alpha_\ell)$ .
5. Ballot-box  $A_1, \dots, A_5$  to obtain  $A'_1, \dots, A'_5$ .
6. For  $\ell = 1$  to 5: Opens super-envelope  $A'_\ell$  to expose envelope triple  $(\mu_\ell, \nu_\ell, \eta_\ell)$ .
7. For  $\ell = 1$  to 5: Read and re-seal  $\eta_\ell$  and denotes its contents by  $\eta_\ell$ . Let  $\eta$  be the permutation that maps  $j$  to  $\eta_j$ .
8. For  $\ell = 1$  to 5: Create a new super-envelope  $B_\ell$  containing the triple of envelopes  $(\mu_\ell, \nu_\ell, \eta_\ell)$ .
9. Privately permute  $B_1, \dots, B_5$  according to  $\eta^{-1}$  to produce  $B'_1, \dots, B'_5$ .
10. For  $\ell = 1$  to 5: Open super-envelope  $B'_\ell$  to expose envelope triple  $(\rho_\ell, \phi_\ell, \psi_\ell)$ .
11. For  $\ell = 1$  to 5: Publicly open envelope  $\psi_\ell$ . (Note that the revealed contents of  $\psi_\ell$  is always the integer  $\ell$ .)

*Output:*  $\rho$  and  $\phi$ .

**CORRECTNESS.** To establish the correctness of *Clone* we must show that  $\rho$  and  $\phi$  are both envelope encodings of  $\sigma$ . The correctness of *Invert* implies that, at the completion of Step 1,  $\alpha$  is an envelope encoding of permutation  $\alpha = \sigma^{-1}$ . Because Step 4 binds together, in the same super-envelope  $A_\ell$ , the  $\ell$ th envelope of  $\alpha$ ,  $\beta$  and  $\gamma$ , Step 5 applies the same, random and secret permutation to  $\alpha$ ,  $\beta$  and  $\gamma$ . Letting  $x$  be this secret permutation, Step 6 “puts on the table” the envelope encodings  $\mu = \mu_1, \dots, \mu_5$ ,  $\nu = \nu_1, \dots, \nu_5$  and  $\eta = \eta_1, \dots, \eta_5$  where  $\mu = x\mathcal{I}$ ,  $\nu = x\mathcal{I}$  and  $\eta = x\sigma^{-1}$ . Step 7, however, privately reveals  $\eta$  to all players, so that the device can compute  $\eta^{-1}$ . Because Step 8 binds together the  $\ell$ th envelopes of  $\mu$ ,  $\nu$  and  $\eta$ , in Step 9 the permutation  $\nu^{-1}$  is applied to  $\mu$ ,  $\nu$  and  $\eta$ . Therefore,  $\hat{\psi} = \eta^{-1}\eta = \mathcal{I}$ . This ensures that the revealed contents in Step 11 are constant, as required. Additionally,  $\rho = \eta^{-1}\mu = \sigma x^{-1}x = \sigma$ , and  $\phi = \eta^{-1}\nu = \sigma x^{-1}x$ , as desired.

**PRIVACY.** The private transcript produced by *Clone* is the following sequence:

(*ReadReseal*,  $\eta_1$ )  
 (*ReadReseal*,  $\eta_2$ )  
 (*ReadReseal*,  $\eta_3$ )  
 (*ReadReseal*,  $\eta_4$ )  
 (*ReadReseal*,  $\eta_5$ )  
 (*PrivatePermute*,  $\eta^{-1}$ )

Where  $\eta = x\sigma$  and  $x$  is a random permutation produced by the ballot box. Therefore,  $\nu$  is a random permutation independent of  $\sigma$ .

Let  $\mathcal{G}$  be the algorithm that on empty input selects a random permutation  $y \in \text{SYM}_5$  and outputs the following sequence:

(*ReadReseal*,  $y_1$ )  
 (*ReadReseal*,  $y_2$ )  
 (*ReadReseal*,  $y_3$ )  
 (*ReadReseal*,  $y_4$ )  
 (*ReadReseal*,  $y_5$ )  
 (*PrivatePermute*,  $y^{-1}$ )

Since  $y$  is a random element of  $\text{SYM}_5$ , the sequences output by  $\mathcal{G}()$  are distributed identically to the private transcripts produced by executing *Clone*.

## C.6 Device Coin

### Device Coin

1. Create an envelope encoding  $\alpha$  of the identity permutation.
2. Create an envelope encoding  $\beta$  of permutation  $a = 12453$ .
3. Create a new super-envelope  $A$  containing envelopes  $\alpha_1, \dots, \alpha_5$ .
4. Create a new super envelope  $B$  containing envelopes  $\beta_1, \dots, \beta_5$ .
5. Ballot-box  $A$  and  $B$  to obtain super-envelopes  $C$  and  $D$ .
6. Open  $C$  to expose envelopes  $\gamma_1, \dots, \gamma_5$ . Set  $\gamma = \gamma_1, \dots, \gamma_5$ .
7. Open  $D$  to expose envelopes  $\nu_1, \dots, \nu_5$ .
8. Ballot-box  $\nu_1, \dots, \nu_5$  to obtain envelopes  $\mu_1, \dots, \mu_5$ .
9. For  $\ell := 1$  to  $5$ : Read and re-seal  $\mu_\ell$  and denote its contents by  $\mu_\ell$ . Let  $\mu$  be the permutation that maps  $j$  to  $\mu_j$ .
10. Privately permute  $\mu_1, \dots, \mu_5$  according to  $\mu^{-1}$ .
11. For  $\ell = 1$  to  $5$ : Publicly open envelope  $\phi_\ell$ . (Note that the revealed contents of  $\phi_\ell$  is always the integer  $\ell$ .)

*Output:* Sequence  $\gamma$ .

**CORRECTNESS.** To establish correctness, we must show that  $\gamma$  is the envelope encoding of a permutation that encodes a random bit  $b$ . That is,  $\gamma = \mathcal{I}$  with probability  $\frac{1}{2}$  and  $\gamma = a$  with probability  $\frac{1}{2}$ . At the end of Step 3,  $A$  contains an envelope encoding of the identity, and, at the end of Step 4,  $B$  contains an envelope encoding of permutation  $a$ . Therefore, at the end of Step 5,  $C$  contains either an envelope encoding of either  $\mathcal{I}$  or  $a$  (each with probability  $\frac{1}{2}$ ). Thus, at the end of Step 6,  $\gamma$  is the encoding of a random bit, as desired. The remaining steps ensure that no “extra” envelopes remain on the table and that the public record is a fixed constant. Step 8 ensures that  $\nu$  is a random permutation. Step 9 secretly reveals  $\nu$ . Step 10 obtains  $\mu$  by applying  $\nu^{-1}$  to  $\nu$ . Therefore,  $\mu = \mathcal{I}$  and so the contents revealed in Step 11 are a fixed constant, as required.

**PRIVACY.** The private transcript produced by *Coin* is the following sequence:

(*ReadReseal*,  $\nu_1$ )  
 (*ReadReseal*,  $\nu_2$ )  
 (*ReadReseal*,  $\nu_3$ )  
 (*ReadReseal*,  $\nu_4$ )  
 (*ReadReseal*,  $\nu_5$ )  
 (*PrivatePermute*,  $\nu^{-1}$ )

Where  $\nu$  is a random permutation produced by the ballot box. Therefore,  $\nu$  is a random permutation independent of  $\gamma$ .

Let  $\mathcal{G}$  be the algorithm that on empty input selects a random permutation  $y \in \text{SYM}_5$  and outputs the following sequence:

(*ReadReseal*,  $y_1$ )  
 (*ReadReseal*,  $y_2$ )  
 (*ReadReseal*,  $y_3$ )  
 (*ReadReseal*,  $y_4$ )  
 (*ReadReseal*,  $y_5$ )  
 (*PrivatePermute*,  $y^{-1}$ )

Since  $y$  is a random element of  $\text{SYM}_5$ , the sequences output by  $\mathcal{G}()$  are distributed identically to the private transcripts produced by executing *Coin*.

## D Pure Equilibria

Let  $\Gamma$  be an extensive-form game of imperfect information and  $IS$  an information set of  $\Gamma$ . By  $[IS]$  we denote the set of strategy profiles of  $\Gamma$  that reach  $IS$  with positive probability. If  $\sigma$  and  $\sigma'$  are strategy profiles of  $\Gamma$ ,  $\sigma' \in [IS]$ , we denote by  $u_i(\sigma' \circ \sigma \mid IS)$  the expected payoff of player  $i$  when all players follow strategies  $\sigma'$  until  $IS$  is reached, and  $\sigma$  afterwards.

**Definition 3.** We say that a strategy profile  $\sigma$  of  $\Gamma$  is a *pure equilibrium* if  $\forall$  information set  $IS$  of  $\Gamma$ ,  $\forall$  strategy profile  $\sigma' \in [IS]$ ,  $\forall$  player  $i$ , and  $\forall$  strategy  $\sigma''$ :

$$u_i(\sigma' \circ \sigma \mid IS) \geq u_i(\sigma' \circ (\sigma''_i, \sigma_{-i}) \mid IS),$$

where  $(\sigma''_i, \sigma_{-i})$  denotes the strategy profile obtained from  $\sigma$  by substituting  $\sigma_i$  with  $\sigma''_i$ .

Put simply, in a pure equilibrium it is best for the players to stick to their strategies at all information sets, *no matter what beliefs they may hold about past play*. In essence, therefore, pure equilibria extend the rationality of subgame-perfection to games of imperfect information.<sup>17</sup>

## E The “Purity Problem” of All Prior Protocols

All protocols for secure computation, including the ballot-box one of [ILM05], rely on the 20-year old paradigm of [GMW87]. Essentially, when securely computing a function  $F$ , at some point of the joint computation there exists a “global state”  $S$  which determines the eventual output(s). Such a global state, however, is in particular not known to any individual player. Rather each player  $i$  knows only his local *share*  $S_i$  of  $S$  which —by itself or together with a proper subset of other shares— does not provide any information about  $S$ . This security property is indeed very demanding, but pure equilibrium demands much more. In essence the problem is that, as long as a player knows his own local share of the global state, he may decide to abort whenever his share does not satisfy a given property  $\mathcal{P}$ . Thus, another player  $i$  may interpret the fact all other players have not aborted so far as an indication that their shares satisfy property  $\mathcal{P}$ . This indication, together with the knowledge of his own share, easily allows player  $i$  to form a belief about the global state and thus the final outcome. And if such a projected outcome is worse than the consequence of an abort, it will become rational for  $i$  to abort!<sup>18</sup> Let us now explicitly construct an example that illustrates this this scenario when two players try to reach correlated equilibrium via the ballot-box protocol of [ILM05].)

### Example

Consider the following two-player game  $G$  in which player 1 selects the row, and player 2 selects the column.

It is easy to verify that the distribution  $p$  assigning probability  $1/2$  to  $(U, C)$  and probability  $1/4$  each to  $(U, L)$  and  $(M, L)$  is a correlated equilibrium with payoffs  $(2.5, 2.5)$ . It is also easy to verify that  $(D, R)$  is a Nash Equilibrium with payoffs  $(1, 1)$ .

Notice that in this correlated equilibrium, whenever player 2 receives recommendation  $L$ , his payoff is 0. (This is balanced by the fact that his payoff is 5 when he receives recommendation  $C$ .) Therefore, although

	L	C	R
U	5, 0	0, 5	-5, -5
M	5, 0	-5, -5	-5, 5
D	-5, -5	5, -5	1, 1

Figure 1: Game  $G$

<sup>17</sup>Pure equilibria include additional distinct features. For instance, unlike sequential equilibria, they do not differentiate between information sets on or off the equilibrium path: all information sets are treated symmetrically. Accordingly, they *truly* tolerate any number of mistakes made in the past. Consider a player  $i$  who must act at an information set  $is$  on the path of a strategy profile  $\sigma$ . If  $\sigma$  is a sequential equilibrium, player  $i$  must put probability 0 on any node of  $is$  that is not reachable by  $\sigma$ , and therefore makes a best response only if all players have been rational so far and made no mistakes. (While this is sound in “theory,” it is shaky in “practice:” even computers make mistakes!) If  $\sigma$  were a pure equilibrium, instead, even on the equilibrium path beliefs are freely specified. In fact, player  $i$  wishes to stick to  $\sigma_i$  even when he is told the exact strategies used to reach  $is$ , and whether or not these strategies coincide with  $\sigma$ .

<sup>18</sup>Notice that this can occur even if  $E$  is acceptable and alternative Nash equilibrium  $N_i$  gives player  $i$  a payoff less than player  $i$ ’s payoff in  $E$ . Indeed,  $i$ ’s payoff in  $E$  is an *expectation* over all his possible recommendations and therefore there may very well be “bad” recommendations that yield *conditional* payoffs that are worse than those of  $N_i$ .

on average, the player 2 prefers the correlated equilibrium  $p$  to the Nash equilibrium  $(D, R)$ , if he knew that he would receive recommendation  $L$ , then he would have preferred the Nash. Without loss of generality, we shall encode recommendation  $L$  and  $U$  as binary string 00,  $C$  and  $M$  as string 01 and  $R$  and  $D$  as string 10.

Recall that in the ILM protocol, the players create a set of ballots  $B$  such that (for each  $i$ ): (1) the contents of ballots  $\Gamma_i^1 \subset B$  are known by player one and encode a string  $\gamma_i^1$ ; (2) the contents of ballots  $\Gamma_i^2 \subset B$  are known by player two and encode a string  $\gamma_i^2$ ; and (3) The (bitwise) XOR of  $\gamma_i^1$  and  $\gamma_i^2$  is the secret recommendation  $s_i$  of player  $i$  (i.e.,  $s_i = \gamma_i^1 \oplus \gamma_i^2$ ). In particular, this means that during the execution of the ILM protocol, player 1 and player 2 both know a share of player 2's recommendation.

A natural approach to achieving correlated equilibrium is for the players to execute ILM in order to compute and play recommendations from correlated equilibrium  $p$  in game  $G$ , and play Nash equilibrium  $(\eta_1, \eta_2) = (D, R)$  in the case of an abort. More formally, the prescribed strategy  $\sigma_i$  for player  $i$  is thus "Play ILM honestly to compute  $r_i$  and play  $r_i$  in  $G$ . If the other player aborts, play Nash strategy  $\eta_i$ ."

Unfortunately,  $\sigma = (\sigma_1, \sigma_2)$  is *not* a Pure Equilibrium. Consider the following alternative strategy  $\sigma'_1$  for player 1: "Play ILM honestly, except abort if the least significant bit (lsb) of  $\gamma_2^1$  is 1." Now consider an execution of profile  $\sigma' = (\sigma'_1, \sigma_2)$ . With probability 1/2, the execution reaches an information set  $is$  of player 2 in which the lsb of  $\gamma_2^2$  (i.e., player 2's share of his own value) is 0. In half the nodes of  $is$ , the lsb of  $\gamma_2^1 = 0$ , and in the other half, lsb of  $\gamma_2^1 = 1$ . However, because  $is$  was reached by playing  $\sigma'$  and because  $\sigma'$  instructs player 1 to abort whenever lsb of  $\gamma_2^1 = 1$ , then player 2 can deduce that  $\gamma_2^1$  must be 0! That is, since player 2 knows that  $\sigma'$  was played, then at information set  $is$ , player 2 knows that  $s_2 = \gamma_2^1 \oplus \gamma_2^2$  has lsb 0 and hence player 2 knows that his recommendation will be  $L$ . Therefore, letting  $\sigma''_2$  be the alternate strategy "Abort at information set  $is$ ", the payoff to player 2 for playing  $\sigma''_2$  at  $is$  is 1 and is *larger* than 0, which is the payoff to player 2 for playing  $\sigma_2$  at  $is$  given that  $is$  was reached via  $\sigma'$ . Thus,  $\sigma$  cannot be a Pure Equilibrium.

The above counter-example illustrates a general problem. If each player  $i$  (or some subset of the players) has a piece of information  $\gamma^i$  which is collectively sufficient to determine player  $j$ 's recommendation, then there cannot be a pure equilibrium. This is because one can easily construct a  $\sigma'_i$  which instructs  $i$  to abort for certain values of  $\gamma^i$ . If the end of the protocol has been reached, then player  $j$  can rightly infer information about his recommendation prior to the end of the protocol. In games such as the above  $G$ , this inference can give player  $j$  incentive to deviate because by doing so,  $j$  receives Nash equilibrium payoffs which just happen to be better (in this case) than those sampled from the correlated equilibrium.

As noted above, all prior proposals for achieving correlated equilibrium (and all existing secure computation protocols) succumb to a similar example because

*At some point in the protocol, the players collectively have enough information to compute some player  $j$ 's recommendation.*