Efficient Rational Secret Sharing in the Standard Communication Model

Georg Fuchsbauer* Jonathan Katz† Eric Levieil*
David Naccache*

Abstract

We propose a new methodology for rational secret sharing leading to various instantiations that are simple and efficient. In addition, we show a round-efficient way to eliminate the need for simultaneous channels even when secrets are chosen from an arbitrarily large domain. Our solution generates shares of *bounded* size, and may therefore be applied to the setting of rational secure computation as well.

1 Introduction

The classical problem of *t-out-of-n secret sharing* [Sha79, Bla79] involves a dealer D who distributes shares of a secret s to a group of n players $P_1, \ldots, P_n$ so that (1) any group of t or more players can reconstruct the secret without further involvement of the dealer, yet (2) any group of fewer than t players cannot recover the secret. Foe example, in Shamir's scheme [Sha79] the secret s lies in a finite field $\mathbb{F}$, with $|\mathbb{F}| > n$. The dealer chooses a random polynomial $f(x)$ of degree at most $t-1$ with $f(0) = s$, and gives each player P_i the "share" $f(i)$. To reconstruct the secret s, any t players simply broadcast their shares and interpolate the polynomial. On the other hand, any set of fewer than t players has no information about s given their shares.

The implicit assumption in the original formulation of the problem is that each party is either honest or corrupt, and honest parties are all willing to cooperate completely with each other when reconstruction of the secret is desired. Beginning with the work of Halpern and Teague [HT04], protocols for secret sharing and other cryptographic tasks have begun to be re-evaluated in a game-theoretic light (see [DR07, Kat08a] for an overview of work in this direction). In this context, parties are neither honest nor corrupt but are instead simply *rational* and are assumed (only) to act in their own self-interest.

Under natural assumptions regarding the utilities of the parties, standard secret-sharing schemes completely fail. For example, assume as in [HT04] that all players want to learn the secret above all else, but otherwise prefer that the fewest number of other players learn the secret. (Later, we will treat the utilities of the players more precisely.) For a set of t parties to reconstruct the secret in Shamir's scheme, each party is supposed to broadcast their share. It is easy to see, however, that each player is better off withholding their share no matter what the other players do. Consider P_1: if strictly fewer than $t-1$ other players reveal their shares to the rest of the group, then no one

*École Normale Supérieure, LIENS - CNRS - INRIA, Paris, France. Email: {georg.fuchsbauer, eric.levieil, david.naccache}@ens.fr

†University of Maryland, USA. Email: jkatz@cs.umd.edu. Work done while visiting ENS. Research supported by NSF CyberTrust grant #0830464 and NSF CAREER award #0447075.

learns the secret regardless of whether P_1 reveals his share or not. If more than $t-1$ other players reveal their shares, then everyone learns the secret and P_1's actions again have no effect. On the other hand, if *exactly* $t-1$ other players reveal their shares, then P_1 learns the secret (using his share) but can prevent other players from learning the secret by *not* publicly revealing his own share. As argued in [HT04], the result is that if all players are rational no one will broadcast their share and the secret will not be reconstructed.

More generally, let $t^* \geq t$ denote the number of players present during the secret-reconstruction phase. We observe the following about the game-theoretic equilibria of "standard" Shamir secret sharing (definitions of the relevant game-theoretic notions are given in Section 2):

- For any t, n, t^*, it is a Nash equilibrium for no one to reveal their share.
- If $t^* > t$, it is a Nash equilibrium for all t^* participating players to reveal their shares. However, as discussed above, broadcasting one's share is *weakly dominated* by not broadcasting anything; thus, the Nash equilibrium likely to be reached is the one in which no one reveals their share.
- If $t^* = t$, then having all t^* players reveal their shares is not even a Nash equilibrium.

A series of recent works [HT04, GK06, LT06, ADGH06, KN08a, KN08b] has focused on designing so-called *rational* secret-sharing protocols immune to the above problems. A protocol for rational secret sharing also follows from the more general results of Lepinski et al. [LMPS04, LMS05, IML05]. Each of these works has some or all of the following disadvantages:

On-line dealer. Halpern and Teague [HT04] introduced a general approach to solving the problem that has been followed in most subsequent work. Their solution, however, requires the continual involvement of the dealer (even after the initial shares have been distributed) to refresh parties' shares. The solution proposed by Halpern and Teague also applies only when $t, n \geq 3$.

Inefficiency. To eliminate the need for an on-line dealer, several researchers [GK06, LT06, ADGH06, KN08a] have suggested solutions that rely on multiple invocations of protocols for generic secure multi-party computation. Because the function being computed by these protocols is relatively complex, it is unclear whether *efficient* protocols with suitable functionality can be designed. The solutions of Lepinski et al. [LMPS04, LMS05, IML05], though following a different high-level approach, also rely on secure multi-party computation and are similarly inefficient.

Non-standard communication models. The solutions offered in [HT04, GK06, LT06, ADGH06] assume the existence of a *simultaneous broadcast channel* which implies that parties must decide on what value (if any) to broadcast in a given round before observing the values broadcast by other parties. Kol and Naor [KN08a] assume simultaneous broadcast for the initial protocol they propose, though they also show how to avoid this assumption at the cost of increasing the round complexity by a (multiplicative) factor linear in the size of the domain from which the secret is chosen. Besides the additional inefficiency resulting from this approach, this also means that their solution is not applicable for secrets of super-logarithmic length.

The solution of Lepinski et al. [LMPS04], and follow-up work by Izmalkov, Micali, and Lepinski [IML05], requires strong *physical* assumptions such as secure envelopes and ballot boxes. We remark that secure envelopes can be used to implement simultaneous broadcast (but not vice versa) and hence such physical assumptions represent a strictly stronger class of assumptions.

Recent work of Ong et al. [OPRV08] provides a solution in the standard communication model, but under the assumption that logarithmically many parties are completely honest and never deviate from the protocol. In our work, as in all work mentioned above, we do not impose this assumption.

Inapplicability to rational multi-party computation (MPC). A different secret sharing scheme of Kol and Naor [KN08b] avoids an on-line dealer, is efficient, and does not require simultaneous broadcast even when the secret is chosen from an arbitrarily large domain. In this scheme, however, the shares of the parties have *unbounded* length. While not a significant problem in its own right (indeed, the protocols in [HT04, GK06, LT06, ADGH06] have bounded-size shares but unbounded round complexity), it becomes an issue when applied to rational computation of general functions. A general approach [HT04, GK06, LT06, ADGH06] to rational computation of a function f (satisfying certain properties) is to first have the parties compute (using standard protocols for secure computation) a function f' that results in a secret sharing of the output of f, and then have the parties run a rational secret sharing protocol to reconstruct the output. As noted in [KN08a], this approach will not work when shares have unbounded size (since the communication used in computing f' is an upper bound on the size of the shares).

We remark that the round complexity of the Kol-Naor protocol mentioned above is linear in the number of parties.

1.1 Our Results

In this work we introduce a general framework for rational secret sharing without an on-line dealer. Our solution follows the same high-level idea as in previous work (see below), but implements this idea without resorting to generic secure multi-party computation. Our approach is therefore, arguably, simpler than prior solutions; more importantly, our new solution is very efficient in terms of both round complexity and required computation. We also show how our protocol can be applied in the standard communication model where all parties have access to a broadcast channel but rushing is allowed (i.e., simultaneous broadcast is *not* assumed). In this case, the round complexity of our protocol is independent of the domain of the secret as well as the number of parties (in contrast to [KN08a] and [KN08b], respectively). Since shares in our protocol have bounded size, our protocol can be used for rational computation of general functions as in previous work.

Of independent interest, we use here a different equilibrium notion than in most previous work. Specifically, we show that the reconstruction phase of our protocol forms a computational Nash equilibrium that is *stable with respect to trembles*. (This choice was motivated by the suggestion in [Kat08a].) We stress, though, that by following the same arguments as in prior work our scheme can also be shown to survive (a computational notion of) iterated deletion of weakly dominated strategies. We chose to use a new equilibrium notion since "surviving iterated deletion" is problematic in certain respects. See further discussion in Section 2.2.

In fairness, we also mention some disadvantages of our scheme. First, our protocol has a negligible probability of error. Second, the scheme handles only single-player deviations and is not resilient to coalitions of rational players; this problem is shared by the solutions of, e.g., [HT04, KN08b]. Finally, our solution offers only *computational* secrecy even before the reconstruction phase begins. That is, the share of even a single party determines the secret in an information-theoretic sense; all we guarantee is that any $t-1$ computationally bounded parties learn no information about the secret. In part for this reason, although our solution is a (computational) Nash equilibrium and can be shown to satisfy the solution concept used in [HT04, GK06, LT06, ADGH06], our protocol does not achieve the stronger equilibrium notions considered by Kol and Naor [KN08a, KN08b]. Our general feeling, however, is that the notions considered by Kol and Naor are "too strong" in the sense that they rule out many naturally appealing protocols; indeed, Kol and Naor themselves state [KN08b] that the equilibrium notion they use is "too restrictive" and should be considered sufficient but not necessary. Determining the "right" game-theoretic notions for rational secret

sharing is the subject of ongoing research [Kat08a, Kat08b].

Overview of our approach. We follow the same high-level approach as in [HT04, GK06, LT06, ADGH06, KN08a, KN08b]. Our reconstruction protocol proceeds in a sequence of "fake" iterations followed by a "real" iteration. Roughly speaking, these satisfy the following requirement:

- In the real iteration, everyone learns the secret (assuming everyone follows the protocol).
- In a fake iteration, no information about the secret is revealed.
- No party can tell, in advance, whether the next iteration will be real or fake.

The iteration number i^* of the real iteration is chosen according to a geometric distribution with parameter $\beta \in (0,1)$ (where β depends on the exact utilities of the players). The reconstruction mechanism is then as follows (for now, we assume synchronous broadcast): parties run each iteration until the real iteration is identified, at which point all parties output the secret. If some party fails to follow the protocol in any iteration, all parties abort. Intuitively, it is rational for P_i to follow the protocol as long as the expected gain of deviating (which occurs only if P_i aborts *exactly* in iteration i^*) is outweighed by the expected loss of deviating (which occurs if P_i aborts any time before iteration i^*).

In prior work of [GK06, LT06, ADGH06, KN08a], a secure multi-party computation was performed in each iteration to determine whether the given iteration will be real or fake. Instead we use the following idea, described in the 2-out-of-2 case (we omit some technical aspects of the protocol in order to focus on the main idea): The dealer D chooses i^* from the appropriate distribution *in advance*, at the time of sharing. The dealer then generates keys two key-pairs (vk_1, sk_1), (vk_2, sk_2) for a *verifiable random function* (VRFs) [MRV99] where vk represents a verification key and sk represents a secret key, and we denote by $\mathsf{VRF}_{sk}(x)$ the evaluation of the VRF on input x using secret key sk. The dealer gives the verification keys to both parties, gives sk_1 to P_1, and gives sk_2 to P_2. Furthermore, it gives $s_1 = s \oplus \mathsf{VRF}_{sk_2}(i^*)$ to P_1, and $s_2 = s \oplus \mathsf{VRF}_{sk_1}(i^*)$ to P_2. Each iteration consists of one message from each party: in iteration i, party P_1 sends $\mathsf{VRF}_{sk_1}(i)$ while P_2 sends $\mathsf{VRF}_{sk_2}(i)$. Observe that a fake iteration reveals nothing about the secret, in a computational sense. Furthermore, neither party can identify the real iteration in advance.

To complete the protocol, we need to provide a way for parties to identify the real iteration. As noted above, previous work [HT04, GK06, LT06, ADGH06, KN08a] allows parties to identify the real iteration *as soon as it occurs*. This suffices when simultaneous broadcast is available, since each party must decide on its next-iteration message before it learns whether the iteration is real or not. When simultaneous channels are not available, however, this approach does *not* work since it is vulnerable to an obvious rushing strategy. Kol and Naor [KN08a, KN08b] give two different ways to circumvent this problem, but as mentioned earlier the first applies only for domains of polynomial size (and yields round complexity linear in the domain size), while the second yields round complexity linear in the number of parties and requires shares of unbounded size.

Motivated by recent work on fairness (in the malicious setting) [GHKL08, GK08], we suggest the following, new approach: delay the signal indicating whether a given iteration is real or fake until the *following* iteration. As before, a party cannot risk aborting until it is sure that the real iteration has occurred; by the time it is sure of this fact, however, the real iteration is over and all parties can reconstruct the secret. This eliminates the need for simultaneous channels, while increasing the (expected) round complexity by only a single round.

In our description above, the protocol relies on VRFs. We show that, in fact, trapdoor permutations suffice. The general approach outlined above can also be extended to the case of t-out-of-n secret sharing, for any t, n.

1.2 Outline of the Paper

Section 2 provides appropriate definitions, and in particular describes the game-theoretic concepts used in this paper. In Section 3 we describe the solution as outlined in the previous section, in the 2-out-of-2 setting and based on VRFs. We then explain how to extend the protocol to the general t-out-of-n case, and also show how the protocol can be based on trapdoor permutations.

2 Modeling and Definitions

We first describe our model for secret sharing, followed by definitions of *computational Nash equilibrium* and *resistance to trembles*. We also provide a definition of verifiable random functions (VRFs). We denote the security parameter by k, and let PPT stand for "probabilistic polynomial time."

2.1 Secret Sharing

A *t-out-of-n secret-sharing scheme for domain $\mathcal{S}$* (with $|\mathcal{S}| > 1$) is a scheme carried out by a dealer D and a set of n parties $P_1, \ldots, P_n$ in two phases. In the first phase (the *sharing phase*), a secret $s \in \mathcal{S}$ is chosen and given to the dealer. Based on this secret and a security parameter 1^k, the dealer generates shares $s_1, \ldots, s_n$ and gives s_i to player P_i. In the second phase (the *reconstruction phase*), some set of $t^* \geq t$ parties carry out some protocol that will allow them to jointly reconstruct s. We impose the following requirements:

Secrecy: The shares of any $t-1$ parties reveal nothing about s, in a computational sense. Formally, for any secrets $s_0, s_1 \in \mathcal{S}$ and any indices $i_1, \ldots, i_{t-1}$ the following distributions are computationally indistinguishable:

$$\left\{(s_1, \ldots, s_n) \leftarrow D(s_0) : (s_{i_1}, \ldots, s_{i_{t-1}})\right\} \quad \text{and} \quad \left\{(s_1, \ldots, s_n) \leftarrow D(s_1) : (s_{i_1}, \ldots, s_{i_{t-1}})\right\}.$$

Correctness: For any set of $t^* \geq t$ parties who run the reconstruction phase honestly, the correct secret s will be reconstructed with all but negligible probability (in k).

The above definition views parties as either malicious or honest. We now incorporate utilities into our discussion, and treat each party as simply *rational*.

2.2 Rational Secret Sharing

Given some set of $t^* \geq t$ parties (numbered $P_1, \ldots, P_{t^*}$ for convenience), let the outcome o of the reconstruction phase be a binary vector of length t^* with $o_i = 1$ iff the output of P_i is equal to the initial secret s (i.e., P_i "learned the secret"). We stress that in contrast to prior work, we consider a party to have learned the secret s if and only if it outputs s, and do not care whether that party "actually" learned the secret or not. In particular, in our model a party who outputs a random value in the domain $\mathcal{S}$ without running the reconstruction phase at all ends up "learning" the secret with probability $1/|\mathcal{S}|$. We model the problem this way for at least two reasons:

1. Our formulation allows us to assign utility to learning *partial* information about the secret, something that is not reflected in prior formulations. In particular, partial information that increases the probability with which a party can output the correct secret will increase the expected utility of that party.

2. It is difficult, in general, to formally model what it means for a party to "really" learn the secret, especially when considering arbitrary protocols and behaviors. In contrast, in our definition it is easy to tell whether a player learns the secret by just looking at their output. Our notion also appears better suited for a computational setting, where a party might "know" the secret from an information-theoretic point of view, yet be unable to output it.

Let $\mathsf{num}(o) \stackrel{\text{def}}{=} \sum_i o_i$; i.e., $\mathsf{num}(o)$ is simply the number of players who learn the secret. Let μ_i denote the utility of player P_i for the outcome o. Following [HT04] and all subsequent work in this area, we make the following assumptions about the utility functions of the players:

- $o_i > o'_i \Rightarrow \mu_i(o) > \mu_i(o')$.
- If $o_i = o'_i$, then $\mathsf{num}(o) < \mathsf{num}(o') \Rightarrow \mu_i(o) > \mu_i(o')$.

That is, player P_i first prefers outcomes in which he learns the secret; otherwise, player P_i prefers strategies in which the fewest number of other players learn the secret. For the purposes of our analysis, we distinguish only three types of outcomes, described from the point of view of P_i:

1. If o is an outcome in which P_i learns the secret and no other player does, then $\mu_i(o) \stackrel{\text{def}}{=} U^+$.
2. If o is an outcome in which P_i learns the secret and at least one other player does also, then $\mu_i(o) \stackrel{\text{def}}{=} U$.
3. If o is an outcome in which P_i does not learn the secret, then $\mu_i(o) \stackrel{\text{def}}{=} U^-$.

Clearly, our conditions impose $U^+ > U > U^-$. Define

$$U_{\text{random}} \stackrel{\text{def}}{=} \frac{1}{|\mathcal{S}|} \cdot U^+ + \left(1 - \frac{1}{|\mathcal{S}|}\right) \cdot U^- ; \tag{1}$$

this is the expected utility of a party who outputs a random guess for the secret (assuming other parties terminate without any output, or with the wrong output). When dealing with a t-out-of-n secret-sharing scheme we also assume $U > U_{\text{random}}$; otherwise, players have essentially no incentive to run the reconstruction phase at all.

Given a vector of strategies $\vec{\sigma}$ for the parties active in the reconstruction phase, we let $U_i(\vec{\sigma})$ denote the expected utility of P_i as a function of the security parameter k. The expectation is taken over the initial choice of s (which we will always assume to be uniform), the dealer's randomness, and the randomness of the players' strategies. Following standard game-theoretic notation, define $\vec{\sigma}_{-i} \stackrel{\text{def}}{=} (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_{t^*})$ and $(\sigma'_i, \vec{\sigma}_{-i}) \stackrel{\text{def}}{=} (\sigma_1, \ldots, \sigma_{i-1}, \sigma'_i, \sigma_{i+1}, \ldots, \sigma_n)$; that is, $(\sigma'_i, \vec{\sigma}_{-i})$ denotes the strategy vector $\vec{\sigma}$ with P_i's strategy changed to σ'_i. We now define the most basic equilibrium notion for secret sharing.

Definition 1 Let Π be a t-out-of-n secret-sharing scheme, where σ_i denotes the prescribed actions of party P_i in the reconstruction phase. Π induces a *computational Nash equilibrium* if for any $t^* \geq t$, any set of t^* distinct indices $I = \{i_1, \ldots, i_{t^*}\}$, any $i \in I$, and any PPT strategy σ'_i, there exists a negligible function ϵ such that

$$U_i(\sigma'_i, \vec{\sigma}_{-i}) \leq U_i(\vec{\sigma}) + \epsilon(k)$$

(where $\vec{\sigma} = (\sigma_{i_1}, \ldots, \sigma_{i_{t^*}})$). $\diamondsuit$

As discussed extensively in prior work (see [Kat08a]), a Nash equilibrium is generally considered "too weak" in the sense that it may still be rational for parties to deviate from their prescribed actions. In most previous work starting with [HT04], the more restrictive notion of a *Nash equilibrium surviving iterated deletion of weakly dominated strategies* was used instead [GK06, LT06, ADGH06]. This notion is not fully satisfactory, either [KN08b, Kat08a]; typical proofs that protocols meet this property proceed by showing that *no* strategies are deleted, and Kol and Naor [KN08b] give an example of a protocol that is intuitively "bad" but which nevertheless can be shown to satisfy the notion. Kol and Naor suggest other concepts, including that of a *Nash equilibrium surviving backward induction* [KN08a] and a *strict Nash equilibrium* [KN08b]. These notions seem too restrictive to be useful in a computational setting, where (informally) there is always a negligible chance of breaking whatever computational assumption the protocol is based on. Because of these problems (and motivated by [Kat08a]), we use the notion of *stability with respect to trembles*.[1] Our definition expands and makes precise the suggestion of [Kat08a].

We consider the setting where strategies refer to interactive Turing machines executing some protocol. Two strategies σ_i, σ_i' are said to *yield equivalent play with respect to* $\vec{\sigma}_{-i}$ if the distribution of messages sent in an execution of $(\sigma_i, \vec{\sigma}_{-i})$ is identical to the distribution of messages sent in an execution of $(\sigma_i', \vec{\sigma}_{-i})$. We stress that σ_i, σ_i' may differ in their local computation and local outputs, as well as in the messages they send when interacting with some other set of strategies $\vec{\sigma}'_{-i}$. We say that strategy σ_i' is *δ-close to* strategy σ_i if there exists a PPT strategy $\hat{\sigma}_i$ such that σ_i' can be expressed in the following form: with probability $1-\delta$, follow σ_i; otherwise, follow $\hat{\sigma}_i$. (In other words, σ_i' differs from σ_i with probability at most δ.) Given a strategy vector $\vec{\sigma}$, we say that $\vec{\sigma}'$ *is δ-close to* $\vec{\sigma}$ if with probability $1-\delta$ *all* parties play according to $\vec{\sigma}$, while with probability δ the parties can behave arbitrarily.

Definition 2 Let Π be a t-out-of-n secret-sharing scheme, where σ_i denotes the prescribed actions of party P_i in the reconstruction phase. Π induces a *computational Nash equilibrium stable with respect to trembles* if

1. Π induces a computational Nash equilibrium;
2. There is a constant $\delta > 0$ (independent of the security parameter) such that for any $t^* \geq t$, any set of t^* distinct indices $I = \{i_1, \ldots, i_{t^*}\}$, any $i \in I$, any vector of PPT strategies $\vec{\rho}_{-i}$ that is δ-close to $\vec{\sigma}_{-i}$, and any PPT strategy ρ_i, there exists a PPT strategy σ_i' such that
   (a) There is a negligible function ϵ such that $U_i(\rho_i, \vec{\rho}_{-i}) \leq U_i(\sigma_i', \vec{\rho}_{-i}) + \epsilon(k)$.
   (b) σ_i' and ρ_i yield equivalent play with respect to $\vec{\sigma}_{-i}$. ♢

Intuitively, stability with respect to trembles means that even if a party P_i believes that the other parties might play some other strategy with probability δ (but will follow the protocol the rest of the time), there is still no better strategy for P_i than to outwardly follow the protocol (while perhaps performing some additional local computation). Moreover, if Π induces a computational Nash equilibrium it means that any additional local computation performed by P_i will not help as long as other parties follow the protocol.

[1] We do not claim that this notion is free of problems; in fact, the notion as defined here is probably too strong in that it rules out some protocols that are intuitively "good". Since we are showing a positive result, we did not consider any weaker variants of the definition.

2.3 Verifiable Random Functions (VRFs)

A VRF is a keyed function whose output is "random-looking" but can still be verified as correct given an associated proof along with a public key. The notion was introduced by Micali, Rabin, and Vadhan [MRV99], and various constructions in the standard model are known [MRV99, Dod02, Lys02, DY05, CL07]. The definition that follows is stronger than the "standard" definition in that it includes an extra uniqueness requirement, but the construction of, e.g., [DY05] achieves it. (Also, we use VRFs only as a stepping stone to our construction based on trapdoor permutations.)

Definition 3 A *verifiable random function* (VRF) with range $\mathcal{R} = \{\mathcal{R}_k\}$ is a tuple of probabilistic polynomial-time algorithms (Gen, Eval, Prove, Vrfy) such that the following hold:

Correctness: For all k, any (pk, sk) output by $\mathsf{Gen}(1^k)$, the algorithm Eval_{sk} maps k-bit inputs to the set $\mathcal{R}_k$. Furthermore, for any $x \in \{0,1\}^k$ we have $\mathsf{Vrfy}_{pk}(x, \mathsf{Eval}_{sk}(x), \mathsf{Prove}_{sk}(x)) = 1$.

Verifiability: For all (pk, sk) output by $\mathsf{Gen}(1^k)$, there does not exist a tuple (x, y, y', π, π') with $y \neq y'$ and $\mathsf{Vrfy}_{pk}(x, y, \pi) = 1 = \mathsf{Vrfy}_{pk}(x, y', \pi')$.

Unique proofs: For all (pk, sk) output by $\mathsf{Gen}(1^k)$, there does not exist a tuple (x, y, π, π') with $\pi \neq \pi'$ and $\mathsf{Vrfy}_{pk}(x, y, \pi) = 1 = \mathsf{Vrfy}_{pk}(x, y, \pi')$.

Pseudorandomness: Consider the following experiment:

1. Generate $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ and give pk to A.
2. A adaptively queries a sequence of strings $x_1, \ldots \in \{0,1\}^k$ and is given $y_i = \mathsf{Eval}_{sk}(x_i)$ and $\pi_i = \mathsf{Prove}_{sk}(x_i)$ in response to each such query x_i.
3. A outputs a string $x \in \{0,1\}^k$ subject to the restriction $x \notin \{x_1, \ldots\}$.
4. A random bit $b \leftarrow \{0,1\}$ is chosen. If $b = 0$ then A is given $y = \mathsf{Eval}_{sk}(x)$; if $b = 1$ then A is given a random $y \leftarrow \mathcal{R}_k$.
5. A makes more queries as in step 2, as long as none of these queries is equal to x.
6. At the end of the experiment, A outputs a bit b'. We say A *succeeds* if $b' = b$.

We require that for any probabilistic polynomial-time adversary A, there exists a negligible function ϵ such that the success probability of A is at most $\frac{1}{2} + \epsilon(k)$. ♢

3 2-out-of-2 Rational Secret Sharing using VRFs

We now describe our protocol Π for the case of 2-out-of-2 secret sharing, and prove that Π induces a computational Nash equilibrium stable with respect to trembles. Let $\mathcal{S}$ be the domain of the secret, and assume $\mathcal{S} = \{0,1\}^\ell$ where ℓ may depend on k. Let (Gen, Eval, Prove, Vrfy) be a VRF with domain $\{0,1\}^\ell$, and let (Gen', Eval', Prove', Vrfy') be a VRF with domain $\{0,1\}^k$. The protocol is defined as follows:

Sharing phase: Let s denote the secret. The dealer chooses an integer $i^* \in \{1, \ldots\}$ according to a geometric distribution with parameter β,[2] where β is a constant that depends on the players' utilities (and is independent of the security parameter); we discuss how to set β below. We assume

[2] That is, the dealer flips a biased coin whose probability of landing heads is β and sets i^* to be the number of coin flips until the first head occurs.

Reconstruction phase

At the outset of this phase, P_1 chooses $\bar{s}^{(0)}$ uniformly from $\mathcal{S}$ and P_2 chooses $s^{(0)}$ the same way. Then in each iteration i, the parties do the following:

(P_2 sends message to P_1:) Party P_2 computes $\bar{y}_i := \mathsf{Eval}_{sk_2}(i)$, $\bar{\pi}_i := \mathsf{Prove}_{sk_2}(i)$ and $\bar{y}'_i := \mathsf{Eval}'_{sk'_2}(i)$, $\bar{\pi}'_i := \mathsf{Prove}'_{sk'_2}(i)$. It sends $(\bar{y}_i, \bar{\pi}_i, \bar{y}'_i, \bar{\pi}'_i)$ to P_1.

(P_1 receives message from P_2:) Party P_1 receives $(\bar{y}_i, \bar{\pi}_i, \bar{y}'_i, \bar{\pi}'_i)$ from P_2. If P_2 does not send anything, or if $\mathsf{Vrfy}_{pk_1}(i, \bar{y}_i, \bar{\pi}_i) = 0$ or $\mathsf{Vrfy}'_{pk'_1}(i, \bar{y}'_i, \bar{\pi}'_i) = 0$, then P_1 outputs $\bar{s}^{(i-1)}$ and terminates.

(P_1 sends message to P_2:) Party P_1 computes $y_i := \mathsf{Eval}_{sk_1}(i)$, $\pi_i := \mathsf{Prove}_{sk_1}(i)$ and $y'_i := \mathsf{Eval}'_{sk'_1}(i)$, $\pi'_i := \mathsf{Prove}'_{sk'_1}(i)$. It sends $(y_i, \pi_i, y'_i, \pi'_i)$ to P_2.

If $\mathsf{signal}_1 = \bar{y}'_i$ then P_1 outputs $\bar{s}^{(i-1)}$ and terminates. Otherwise, it sets $\bar{s}^{(i)} := \mathsf{share}_1 \oplus \bar{y}_i$ and continues.

(P_2 receives message from P_1:) Party P_2 receives $(y_i, \pi_i, y'_i, \pi'_i)$ from P_1. If P_1 does not send anything, or if $\mathsf{Vrfy}_{pk_1}(i, y_i, \pi_i) = 0$ or $\mathsf{Vrfy}'_{pk'_1}(i, y'_i, \pi'_i) = 0$, then P_2 outputs $s^{(i-1)}$ and terminates.

If $\mathsf{signal}_2 = y'_i$ then P_2 outputs $s^{(i-1)}$ terminates. Otherwise, it sets $s^{(i)} := \mathsf{share}_2 \oplus y_i$ and continues.

Figure 1: The reconstruction phase of secret-sharing protocol Π.

$i^* < 2^k - 1$ since this occurs with all but negligible probability. (Technically, if $i^* \geq 2^k - 1$ then the dealer can just give a special error message to each party.)

The dealer generates $(pk_1, sk_1), (pk_2, sk_2) \leftarrow \mathsf{Gen}(1^k)$ and $(pk'_1, sk'_1), (pk'_2, sk'_2) \leftarrow \mathsf{Gen}'(1^k)$, and then computes:

- $\mathsf{share}_1 := \mathsf{Eval}_{sk_2}(i^*) \oplus s$ and $\mathsf{share}_2 := \mathsf{Eval}_{sk_1}(i^*) \oplus s$;
- $\mathsf{signal}_1 := \mathsf{Eval}'_{sk'_2}(i^* + 1)$ and $\mathsf{signal}_2 := \mathsf{Eval}'_{sk'_1}(i^* + 1)$.

Finally, the dealer gives to P_1 the values $(sk_1, sk'_1, pk_2, pk'_2, \mathsf{share}_1, \mathsf{signal}_1)$, and gives to P_2 the values $(sk_2, sk'_2, pk_1, pk'_1, \mathsf{share}_2, \mathsf{signal}_2)$.

Reconstruction phase: A high-level overview of the protocol was given in Section 1.1, and so we jump right in to the formal specification here. The reconstruction phase proceeds in a series of iterations, where each iteration consists of one message sent by each party. Although these messages could be sent at the same time (since they do not depend on each other), we do not want to assume synchronous communication and we therefore simply require P_2 to go first in each iteration.

Before the formal proof, we give some intuition as to why the reconstruction phase of Π is a computational Nash equilibrium for appropriate choice of β. Assume P_2 follows the protocol, and consider possible deviations by P_1. (Deviations by P_2 can be analyzed symmetrically. In fact, that case is easier to analyze since P_2 goes first in every iteration.) There are essentially two things P_1 can do: it can abort in iteration $i = i^* + 1$ (i.e., as soon as it receives $\bar{y}'_i = \mathsf{signal}_1$), or it can abort in some iteration $i < i^* + 1$. In the first case, when P_1 aborts in iteration $i^* + 1$, party P_1 "knows" that it learned the dealer's secret in the preceding iteration (that is, in iteration i^*) and can thus output the correct secret; however, P_2 will output $s^{(i^*)} = s$ and so gets credit for "learning" the secret as well. (Here we rely on the verifiability property of the VRF, in that P_1 cannot change y_{i^*} without being detected.) Thus, in this case, P_1 does not increase its utility beyond what it would achieve by following the protocol. In the second case, where P_1 aborts in some iteration $i < i^* + 1$, the issue is that P_1 does not know whether $i = i^*$ or not. The best strategy it can adopt is to

output $\bar{s}^{(i)}$ and hope that $i = i^*$. The expected utility that P_1 obtains by following this strategy can be estimated as follows:

- The probability that P_1 aborts in iteration $i = i^*$ is roughly β.
- When $i < i^*$, player P_1 has "no information" about s and so the best it can do is guess. The expected utility of P_1 in this case is thus at most U_{random} (cf. Equation (1)).
- U^+ is always an upper bound for the utility of P_1, even when $i = i^*$.

Putting everything together, the expected utility of P_1 following this strategy is at most

$$\beta \cdot U^+ + (1 - \beta) \cdot U_{\text{random}} .$$

Since $U_{\text{random}} < U$ by assumption, it is possible to set β so that the entire expression above is strictly less than U; in that case, P_1 has no incentive to deviate. We make this intuitive argument precise in the proof of the following theorem.

Theorem 1 *Let β be such that $U > \beta \cdot U^+ + (1 - \beta) \cdot U_{\text{random}}$. Then protocol Π, using this value of β, is a 2-out-of-2 secret-sharing scheme that induces a computational Nash equilibrium stable with respect to trembles.*

Proof We must first show that Π is a valid secret-sharing scheme. The secrecy property will follow from the proof that the reconstruction phase is a computational Nash equilibrium: if secrecy did not hold then computing the secret locally and not participating in the reconstruction phase at all would be a profitable deviation. We therefore focus on correctness. Assuming both parties run the protocol honestly, there are only three events that would prevent the correct secret from being reconstructed: either $i^* \geq 2^k - 1$ (which occurs with negligible probability), or $\mathsf{signal}_1 = \mathsf{Eval}'_{sk'_2}(i)$ or $\mathsf{signal}_2 = \mathsf{Eval}'_{sk'_1}(i)$ for some $i < i^* + 1$. Security of the VRF easily implies that the latter two events happen with only negligible probability.

We next prove that Π induces a computational Nash equilibrium. Assume P_2 follows the strategy σ_2 as prescribed by the protocol, and let σ'_1 denote any probabilistic polynomial-time strategy followed by P_1. (The other case, where P_1 follows the protocol and we look at deviations by P_2, follows similarly with an even simpler proof.) In a given execution of the reconstruction phase, let i denote the iteration, if any, in which P_1 aborts; if P_1 never aborts then we set $i = \infty$. Let early be the event that $i < i^*$; let exact be the event that $i = i^*$; and let late be the event that $i > i^*$. Let $\mathsf{correct}$ be the event that P_1 outputs the correct secret s. We will consider the probabilities of these events in two experiments: the experiment defined by running the actual secret-sharing scheme, and a second experiment where P_1 is given $\mathsf{share}_1, \mathsf{signal}_1$ chosen uniformly at random from the appropriate ranges. Probabilities in the first experiment will be denoted by $\Pr_{\text{real}}[\cdot]$, and probabilities in the second experiment will be denoted by $\Pr_{\text{ideal}}[\cdot]$.

We have

$$\begin{aligned} & U_1(\sigma'_1, \sigma_2) \qquad (2) \\ & \leq \; U^+ \cdot \Pr_{\text{real}}[\mathsf{exact}] + U^+ \cdot \Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] + U^- \cdot \Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] + U \cdot \Pr_{\text{real}}[\mathsf{late}], \end{aligned}$$

using the fact (as discussed in the intuition preceding the theorem) that whenever late occurs P_2 outputs the correct secret s. Since when both parties follow the protocol P_1 gets utility U, we need to show that there exists a negligible function ϵ such that $U_1(\sigma'_1, \sigma_2) \leq U + \epsilon(k)$. This is proved by the following claims.

Claim 1 *There exists a negligible function ϵ such that*

$$\begin{aligned}
|\Pr_{\text{real}}[\mathsf{exact}] - \Pr_{\text{ideal}}[\mathsf{exact}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\mathsf{late}] - \Pr_{\text{ideal}}[\mathsf{late}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\mathsf{correct} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early}]| &\leq \epsilon(k) \\
|\Pr_{\text{real}}[\overline{\mathsf{correct}} \wedge \mathsf{early}] - \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early}]| &\leq \epsilon(k).
\end{aligned}$$

Proof This follows easily from the pseudorandomness of the VRFs. ■

Let abort be the event that P_1 aborts before iteration i^*+1 (so $\mathsf{abort} = \mathsf{exact} \vee \mathsf{early}$). Define

$$\begin{aligned}
U_{\text{ideal}} \stackrel{\text{def}}{=}\ & U^+ \cdot \Pr_{\text{ideal}}[\mathsf{exact} \wedge \mathsf{abort}] + U^+ \cdot \Pr_{\text{ideal}}[\mathsf{correct} \wedge \mathsf{early} \wedge \mathsf{abort}] \\
& + U^- \cdot \Pr_{\text{ideal}}[\overline{\mathsf{correct}} \wedge \mathsf{early} \wedge \mathsf{abort}] + U \cdot \Pr_{\text{ideal}}[\mathsf{late}].
\end{aligned}$$

Claim 1 shows that $|U_1(\sigma_1', \sigma_2) - U_{\text{ideal}}| \leq \epsilon(k)$ for some negligible function ϵ. It remains to bound U_{ideal}. Information-theoretic arguments show that

$$\begin{aligned}
\Pr_{\text{ideal}}[\mathsf{exact} \mid \mathsf{abort}] &= \beta \\
\Pr_{\text{ideal}}[\mathsf{correct} \mid \mathsf{early}] &= \frac{1}{|\mathcal{S}|};
\end{aligned}$$

therefore,

$$\begin{aligned}
U_{\text{ideal}} &= U^+ \cdot \left(\beta \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + \frac{1}{|\mathcal{S}|} \cdot (1-\beta) \cdot \Pr_{\text{ideal}}[\mathsf{abort}]\right) \\
&\quad + U^- \cdot \left(1 - \frac{1}{|\mathcal{S}|}\right)(1-\beta) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] + U \cdot (1 - \Pr_{\text{ideal}}[\mathsf{abort}]) \\
&= U + \left(U^+ \cdot \left(\beta + \frac{1}{|\mathcal{S}|} \cdot (1-\beta)\right) + U^- \cdot \left(1 - \frac{1}{|\mathcal{S}|}\right)(1-\beta) - U\right) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] \\
&= U + \left(\beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}} - U\right) \cdot \Pr_{\text{ideal}}[\mathsf{abort}] \qquad (3) \\
&\leq U
\end{aligned}$$

using the fact that $\beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}} - U < 0$. This completes the proof that Π induces a computational Nash equilibrium.

To complete the proof, we show that Π induces a computational Nash equilibrium stable with respect to trembles. This part of the proof relies heavily on the uniqueness property of the VRFs. In particular, the uniqueness property implies that in each iteration each party has a *unique* message it can send that will not be treated as an abort. This rules out any way for the parties to "signal" to each other without being detected.

Let δ be a constant that we will fix later. Let ρ_2 denote any PPT strategy for P_2 that is δ-close to σ_2, and let ρ_1 be an arbitrary PPT strategy for P_1. We show the existence of a PPT strategy σ_1' satisfying the requirements of Definition 2. (Once again we focus on deviations by P_1 but the case of P_2 is analogous.)

Say an iteration-i message $(\bar{y}_i, \bar{\pi}_i, \bar{y}_i', \bar{\pi}_i')$ from P_2 *represents an abort* if P_2 does not send anything, or if $\mathsf{Vrfy}_{pk_1}(i, \bar{y}_i, \bar{\pi}_i) = 0$ or $\mathsf{Vrfy}'_{pk_1'}(i, \bar{y}_i', \bar{\pi}_i') = 0$. Say P_2 *aborts* if it sends a message representing an abort. (These notions are defined analogously for P_1.) Strategy σ_1' is defined as follows:

1. Given input $(sk_1, sk_1', pk_2, pk_2', \mathsf{share}_1, \mathsf{signal}_1)$, run ρ_1 on this input. Set $\mathsf{aborted} := 0$.

2. In each iteration i:

   (a) Receive the iteration-i message m_i from P_2. If P_2 aborts, then set $\mathsf{aborted} := 1$.

   (b) Give m_i to ρ_1 and get in response some message m_i'.

   (c) If $\mathsf{aborted} = 1$ then forward m_i' to P_2; otherwise, compute the response $(y_i, \pi_i, y_i', \pi_i')$ as prescribed by Π and send that to P_2.

3. If $\mathsf{aborted} = 0$ then output whatever is prescribed by Π; otherwise, output whatever ρ_1 outputs.

When σ_1' interacts with σ_2, then $\mathsf{aborted}$ is never set to 1; thus, σ_1' and ρ_1 yield equivalent play with respect to σ_2. It remains to show that $U_1(\rho_1, \rho_2) \leq U_1(\sigma_1', \rho_2) + \epsilon(k)$ for some negligible function ϵ. Let $\hat{\rho}_2$ denote the "residual" strategy of ρ_2 (i.e., the strategy it follows with probability δ). In an interaction involving ρ_1, let abort denote the event that ρ_1 aborts before P_2 aborts, and let $p_{\text{abort}}(\sigma)$ be the probability of this event when ρ_1 is interacting with the strategy σ. We first claim that the only "advantage" of playing ρ_1 rather than σ_1' arises due to ρ_1 being the first to abort.

Claim 2 $U_1(\rho_1, \hat{\rho}_2) - U_1(\sigma_1', \hat{\rho}_2) \leq p_{\text{abort}}(\hat{\rho}_2) \cdot (U^+ - U^-)$.

Proof Note that abort is well-defined in the interaction of σ_1' with $\hat{\rho}_2$ since σ_1' runs a copy of ρ_1 as a sub-routine. When abort does *not* occur, there are two possibilities: neither ρ_1 nor P_2 ever aborts, or P_2 aborts first. We consider these in turn:

- When neither ρ_1 nor P_2 aborts, the output of P_2 is unchanged whether P_1 is running σ_1' or ρ_1. Furthermore, the output of P_1 when running σ_1' is equal to the correct secret. Thus, the utility of P_1 when running σ_1' is at least the utility of P_1 when running ρ_1.
- If P_2 aborts first, the outputs of both P_1 and P_2 will be identical regardless of whether P_1 runs σ_1' or ρ_1; this follows because as soon as P_2 aborts, strategy σ_1' "switches" to playing strategy ρ_1.

So, the utility obtained by playing σ_1' can only possibly be less than the utility obtained by playing ρ_1 when abort occurs. The maximum difference in the utilities in this case is $U^+ - U^-$. ■

The next claim shows that abort occurs at least as often when ρ_1 interacts with σ_2 as when ρ_1 interacts with $\hat{\rho}_2$.

Claim 3 $p_{\text{abort}}(\sigma_2) \geq p_{\text{abort}}(\hat{\rho}_2)$.

Proof To see this, consider some view of ρ_1 on which it aborts first when interacting with $\hat{\rho}_2$. (The view includes both the information d_1 given to ρ_1 by the dealer as well as the messages from P_2.) Since ρ_1 aborts first and, in every iteration, there is a *unique* non-aborting message that P_2 can send, it follows that ρ_1 will also abort when interacting with σ_2 (who never aborts first) whenever ρ_1 is given d_1 from the dealer. The claim follows. ■

Define $U^* \stackrel{\text{def}}{=} \beta \cdot U^+ + (1-\beta) \cdot U_{\text{random}}$ and recall that $U^* < U$ by assumption. Now,

$$\begin{aligned} U_1(\rho_1, \rho_2) &= (1-\delta) \cdot U_1(\rho_1, \sigma_2) + \delta \cdot U_1(\rho_1, \hat{\rho}_2) \\ &\leq (1-\delta) \cdot \big(U + (U^* - U) \cdot p_{\text{abort}}(\sigma_2)\big) + \delta \cdot U_1(\rho_1, \hat{\rho}_2) + \epsilon(k) \end{aligned}$$

(where ϵ is a negligible function), using Equation (3). Also,

$$\begin{aligned} U_1(\sigma_1', \rho_2) &= (1-\delta) \cdot U_1(\sigma_1', \sigma_2) + \delta \cdot U_1(\sigma_1', \hat{\rho}_2) \\ &= (1-\delta) \cdot U + \delta \cdot U_1(\sigma_1', \hat{\rho}_2)\,. \end{aligned}$$

It follows that

$$\begin{aligned} & U_1(\rho_1, \rho_2) - U_1(\sigma_1', \rho_2) \\ &= (1-\delta) \cdot (U^* - U) \cdot p_{\text{abort}}(\sigma_2) + \delta \cdot \big(U_1(\rho_1, \hat{\rho}_2) - U_1(\sigma_1', \hat{\rho}_2)\big) + \epsilon(k) \\ &\leq (1-\delta) \cdot (U^* - U) \cdot p_{\text{abort}}(\hat{\rho}_2) + \delta \cdot (U^+ - U^-) \cdot p_{\text{abort}}(\hat{\rho}_2) + \epsilon(k)\,, \end{aligned}$$

using Claims 2 and 3. Since $U^* - U$ is strictly negative, $\delta > 0$ can be found for which the above expression is negligible for k large enough. This completes the proof. ■

3.1 Using Trapdoor Permutations Instead of VRFs

The protocol from the previous section can be adapted rather easily to use trapdoor permutations rather than VRFs. The key observation is that the VRFs in the previous protocol were used only in a very specific way: the VRF was applied to inputs *in order*. What we can do instead is to use a trapdoor permutation f with associated hardcore bit h to "instantiate" a VRF in our scheme in the following way: The "public key" will be a description of f and a random element y in the domain of f; the secret key will be the trapdoor enabling inversion of f. In iteration i, the "evaluation" of the VRF on "input" i will be the ℓ-bit sequence $h(f^{-(i-1)\ell-1}(y)), h(f^{-(i-1)\ell-2}(y)), \ldots, h(f^{-(i-1)\ell-\ell}(y))$ and the "proof" will be $\pi_i = f^{-(i-1)\ell-\ell}(y)$. Verification can be done with respect to the original point y, but can also be done in time independent of i by using the previous proof π_{i-1} (namely, by checking that $f^\ell(\pi_i) = \pi_{i-1}$).

The key point is that the essential properties we need still hold: the properties of verifiability and uniqueness of proofs are easy to see, and pseudorandomness still holds but with resect to a modified game where the adversary queries $\mathsf{Eval}_{sk}(1), \ldots, \mathsf{Eval}_{sk}(i)$ and then has to guess whether it is given $\mathsf{Eval}_{sk}(i+1)$ or a random string. We omit further details.

3.2 Extension to the t-out-of-n Case

To adapt our approach to the t-out-of-n case, we proceed as follows. (We revert back to using VRFs for notational simplicity, but this extension can be instantiated using trapdoor permutations as well.) Each party P_i will be associated with two keys for the VRF, denoted (pk_i, sk_i) and (pk_i', sk_i') as before. As in the case of Shamir secret sharing, the dealer chooses a random degree-$(t-1)$ polynomial G such that $G(0) = s$. The dealer chooses an iteration r^* according to a geometric distribution as before, and then gives the following information to each player P_i:

- Public keys of all other parties: $\{(pk_j, pk_j')\}_{j \neq i}$.
- Its own secret keys: sk_i, sk_i'.
- Its own share $g_i = G(i)$ and "blinded" shares of all other parties: $\{g_{i,j} = G(j) \oplus \mathsf{Eval}_{sk_j}(r^*)\}_{j \neq i}$.
- A set of "signals": $\{\mathsf{signal}_{i,j} = \mathsf{Eval}'_{sk_j'}(r^*+1)\}_{j \neq i}$.

Computational hiding holds, since any set of $t-1$ parties $P_{i_1}, \ldots, P_{i_{t-1}}$ learns (in a computational sense) only the values $G(i_1), \ldots, G(i_{t-1})$.

Given a set of $t^* \geq t$ active players, the reconstruction phase proceeds as follows: Only the first t parties (arranged lexicographically) take part, and the rest of the parties simply observe the communication over a broadcast channel. Let I^* denote the indices of all t^* parties, and let I denote the indices of the t lexicographically first parties. Then, in each iteration r:

- If any party in $I^* \setminus I$ speaks, then each party P_i terminates and outputs $s_i^{(r-1)}$ (or a random element of $\mathcal{S}$ if $r = 1$).
- Each party P_i, for $i \in I$, broadcasts $y_i^{(r)} = \mathsf{Eval}_{sk_i}(r)$ and $\bar{y}_i^{(r)} = \mathsf{Eval}'_{sk'_i}(r)$, along with the respective proofs $\mathsf{Prove}_{sk_i}(r), \mathsf{Prove}'_{sk'_i}(r)$. Incorrect proofs are treated as an abort.
- Each P_i does as follows:
  - If any other party P_j (for $j \in I$) broadcast a value $\bar{y}_j^{(r)}$ such that $\mathsf{signal}_{i,j} = \bar{y}_j^{(r)}$, then P_i outputs $s_i^{(r-1)}$ and terminates.
  - If any party P_j (for $j \in I$) aborted, then P_i outputs $s_i^{(r-1)}$ and terminates.
- If P_i has not terminated, then it computes $s_i^{(r)}$ as follows: set $g_\ell := g_{i,\ell} \oplus y_\ell^{(r-1)}$ for all $\ell \neq i$. Interpolate a degree-$(t-1)$ polynomial through the t points $\{g_i\}_{i=1}^t$, and let $s_i^{(r)}$ be the constant term of this polynomial.

We omit a proof that, by setting the parameter β appropriately, this protocol induces a computational Nash equilibrium stable with respect to trembles.

References

[ADGH06] I. Abraham, D. Dolev, R. Gonen, and J. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *ACM Conference on Principles of Distributed Computing (PODC) 2006*, pages 53–62. ACM Press, 2006.

[Bla79] G.R. Blakley. Safeguarding cryptographic keys. *National Computer Conference*, 48:313–317, 1979.

[CL07] M. Chase and A. Lysyanskaya. Simulatable VRFs with applications to multi-theorem NIZK. In *Advances in Cryptology — Crypto 2007*. Springer, 2007.

[Dod02] Y. Dodis. Efficient constructions of (distributed) verifiable random functions. In *Public-Key Cryptography (PKC) 2002*, pages 1–17. Springer, 2002.

[DR07] Y. Dodis and T. Rabin. Cryptography and game theory. In N. Nisan, T. Roughgarden, É. Tardos, and V.V. Vazirani, editors, *Algorithmic Game Theory*. Cambridge University Press, 2007.

[DY05] Y. Dodis and A. Yampolskiy. A verifiable random function with short proofs and keys. In *Public Key Cryptography (PKC 2005)*, pages 416–431. Springer, 2005.

[GHKL08] D. Gordon, C. Hazay, J. Katz, and Y. Lindell. Complete fairness in secure two-party computation. In *40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 413–422. ACM Press, 2008.

[GK06] S.D. Gordon and J. Katz. Rational secret sharing, revisited. In *Security and Cryptography for Networks (SCN)*, pages 229–241. Springer, 2006.

[GK08] D. Gordon and J. Katz. Partial fairness in secure two-party computation, 2008. Available at `http://eprint.iacr.org/2008/206`.

[HT04] Joseph Y. Halpern and Vanessa Teague. Rational secret sharing and multiparty computation. In *36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 623–632. ACM Press, 2004.

[IML05] Sergei Izmalkov, Silvio Micali, and Matt Lepinski. Rational secure computation and ideal mechanism design. In *FOCS*, pages 585–595. IEEE Computer Society, 2005.

[Kat08a] J. Katz. Bridging game theory and cryptography: Recent results and future directions. In *Theory of Cryptography Conference (TCC)*, pages 251–272. Springer, 2008.

[Kat08b] J. Katz. Ruminations on defining rational MPC, 2008. Talk given at SSoRC, Bertinoro, Italy. Slides available at `http://www.daimi.au.dk/~jbn/SSoRC2008/program`.

[KN08a] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *Theory of Cryptography Conference (TCC)*, pages 320–339. Springer, 2008.

[KN08b] G. Kol and M. Naor. Games for exchanging information. In *40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 423–432. ACM Press, 2008.

[LMPS04] M. Lepinski, S. Micali, C. Peikert, and A. Shelat. Completely fair SFE and coalition-safe cheap talk. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 1–10. ACM Press, 2004.

[LMS05] M. Lepinski, S. Micali, and A. Shelat. Collusion-free protocols. In *37th ACM Symposium in Theory of Computing (STOC)*, pages 543–552. ACM Press, 2005.

[LT06] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behavior in multiparty computation. In *Advances in Cryptology — Crypto 2006*, pages 180–197. Springer, 2006.

[Lys02] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In *Advances in Cryptology — Crypto 2002*, pages 597–612. Springer, 2002.

[MRV99] S. Micali, M. Rabin, and S. Vadhan. Verifiable random functions. In *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 120–130. IEEE, 1999.

[OPRV08] S. J. Ong, D. Parkes, A. Rosen, and S. Vadhan. Fairness with an honest minority and a rational majority, 2008. Available at `http://eprint.iacr.org/2008/097`.

[Sha79] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.