PPAD is as Hard as LWE and Iterated Squaring

Nir Bitansky[1], Arka Rai Choudhuri[2], Justin Holmgren[3], Chethan Kamath[1], Alex Lombardi[4], Omer Paneth[1], and Ron D. Rothblum[5]

[1] Tel Aviv University
nirbitan@tau.ac.il,ckamath@protonmail.com,omerpa@tauex.tau.ac.il
[2] UC Berkeley arkarc@berkeley.edu
[3] NTT Research justin.holmgren@ntt-research.com
[4] MIT alexlombardi@alum.mit.edu
[5] Technion rothblum@cs.technion.ac.il

Abstract. One of the most fundamental results in game theory is that every finite strategic game has a Nash equilibrium, an assignment of (randomized) strategies to players with the stability property that no individual player can benefit from deviating from the assigned strategy. It is not known how to efficiently *compute* such a Nash equilibrium — the computational complexity of this task is characterized by the class **PPAD**, but the relation of **PPAD** to other problems and well-known complexity classes is not precisely understood. In recent years there has been mounting evidence, based on cryptographic tools and techniques, showing the hardness of **PPAD**.

We continue this line of research by showing that **PPAD** is as hard as *learning with errors* (LWE) and the *iterated squaring* (IS) problem, two standard problems in cryptography. Our work improves over prior hardness results that relied either on (1) sub-exponential assumptions, or (2) relied on "obfustopia," which can currently be based on a particular combination of three assumptions. Our work additionally establishes *public-coin* hardness for **PPAD** (computational hardness for a publicly sampleable distribution of instances) that seems out of reach of the obfustopia approach.

Following the work of Choudhuri et al. (STOC 2019) and subsequent works, our hardness result is obtained by constructing an *unambiguous and incrementally-updateable* succinct non-interactive argument for IS, whose soundness relies on polynomial hardness of LWE. The result also implies a verifiable delay function *with unique proofs*, which may be of independent interest.

1 Introduction

The concept of a Nash equilibrium is fundamental to the modern understanding of *games*: given a description of payoffs as a function of k player strategies (which take value in a finite domain), what are a collection of strategy *distributions* that cannot be locally improved? It is not a priori clear that such mixed strategies should exist, but the seminal work of Nash [46] shows that they do. In

the language of modern computational complexity, this implies that Nash equilibrium is a *total search problem*, a search problem such that every instance of the problem is guaranteed to have a solution. It turns out that computing (arbitrarily good approximate) solutions to this problem is in fact in the complexity class **TFNP** [45], the class of total search problems with *efficient verification*. In fact, it is *complete* for its subclass called **PPAD** [14, 21, 48], for which the existence of solution is guaranteed via "*p*olynomial *p*arity *a*rgument on *d*irected graphs". Thus, understanding the computational complexity of **PPAD** exactly corresponds to understanding the complexity of computing a Nash equilibrium.

Despite many decades of attention, we do not currently have polynomial-time algorithms for Nash (or any **PPAD**-complete problem); indeed, it is widely believed that **PPAD** is computationally intractable. Understanding to what extent this is the case, and why, has been a major line of research at the intersection of game theory, computational complexity, and (perhaps surprisingly) *cryptography*. In our work, we further explore this connection to cryptography and prove new hardness results for **PPAD** under cryptographic assumptions.

Prior work. Before describing our results, we summarize the state of affairs prior to our work. The goal of this line of work is to prove theorems of the form "if **PPAD** can be solved in polynomial-time, then standard cryptography is broken." The usual notion of "cryptography is broken" is that there is a probabilistic polynomial-time (PPT) algorithm solving a problem fundamental to cryptography with non-negligible advantage or success probability. As we will see, prior work, which fall into the two categories described below, falls somewhat short of achieving this ideal.

- **Specialized Proof Systems**: Starting from [15], there has been a sequence of works obtaining hardness in **PPAD** by building *unambiguous, incremental, succinct non-interactive arguments* [15, 16, 23, 39, 41, 43], which in turn implies the hardness of **PPAD**. These works build such proof systems (and therefby establish hardness of **PPAD**) based on (1) the hardness of breaking the Fiat-Shamir heuristic [15, 16, 23], (2) the subexponential hardness of both iterated squaring (IS) and learning with errors (LWE) [43], (3) the subexponential hardness only of LWE [39], or (4) the superpolynomial hardness of a problem about bilinear groups [41] along with the exponential-time hypothesis (ETH).
  Unfortunately, none of these results achieve what we required above: a polynomial-time reduction from breaking cryptography (in polynomial time) to **PPAD**. In particular, these results leave open the possibility that there is a polynomial-time algorithm for **PPAD** and yet *all* of these problems are hard in the standard cryptographic sense.
- **Obfustopia:** Another sequence of works [4, 29, 34] show that **PPAD** is hard in "obfustopia", which is a world where indistinguishability obfuscation [3, 28] and functional encryption [7, 47] exist. Unlike the previous approach, this line of work *is* capable of relying on polynomial hardness: in particular, [29] showed that if **PPAD** is easy, then functional encryption cannot exist.

Combined with the groundbreaking results of [36, 37], this in turn would imply that one of three seemingly hard problems[6] in cryptography must be easy.
While the results of [36,37] are based on well-founded assumptions, they have received less scrutiny than other cryptographic assumptions. Even more fundamentally, we do not want to base the hardness of such a central complexity class such as **PPAD** only on the conjunction of three specific hardness assumptions.

In our work, we ask whether it is possible for the *first* line of work – basing **PPAD**-hardness on unambiguous proof systems – to rely on standard, polynomial-time hardness assumptions.

1.1 Our Results

Our first result shows that (average-case) **PPAD** hardness follows from the polynomial-time hardness of iterated squaring in RSA groups and LWE. In fact, as showed in [34], the same techniques imply hardness in the sub-class **CLS** $\subseteq$ **PPAD** introduced in [22]. We further strengthen these hardness results to the subclass **UEOPL** $\subseteq$ **CLS**, which is one of the lowest known sub-classes of **TFNP** [24].

Theorem 1 (Following Theorem 3 and Corollary 3, informally stated). *If there exists a PPT algorithm that solves* **PPAD** *with non-negligible probability, then there exists a PPT algorithm that breaks either IS in RSA groups or LWE with non-negligible probability.*

Slightly more formally, for a complete problem P in **PPAD**, we construct a distribution $\mathcal{D}$ on instances of P with the following property: if there is a polynomial-time algorithm A such that $\mathsf{A}(x)$ is a solution to $P(x)$ with non-negligible probability when sampling $x \leftarrow \mathcal{D}$, then there is a polynomial-time algorithm B that solves IS or solves LWE with non-negligible probability.

Public-coin **PPAD** *hardness.* Our hardness result is actually slightly stronger than what is achieved by the obfustopia reductions. We show *public-coin* hardness of P: there is a sampling algorithm for $\mathcal{D}$ such that the existence of such a B is guaranteed *even if* A is given the random coins used in sampling x. To our knowledge, this is the first hardness result for publicly sampleable distributions in **PPAD**. Moreover, previous hardness results that were based on polynomially falsifiable assumptions [4, 29, 34] seem inherently limited to secret-coin hardness because their instance distributions contain obfuscated circuits (or functional encryption ciphertexts that simulate the functionality of an obfuscated circuit). We remark that our public-coin hardness result may be somewhat surprising because the IS problem in an RSA modulus does not itself have a public-coin sampler.

[6] The problems, roughly, are to break an SXDH assumption, to break a large-field LPN assumption, and to break a low-depth PRG.

Unique VDFs from standard assumptions. Our techniques also yield new results for verifiable delay functions (VDFs) [6]. We construct VDFs with *unique* proofs, which we call unique VDFs, based on the standard LWE assumption and the standard sequential hardness assumption regarding IS.

Theorem 2 (informally stated). *If IS in RSA group is sequentially-hard and LWE is polynomially-hard, then there exists a unique VDF.*

Ours is the first construction of unique VDFs that is based on a polynomial hardness assumption. Recently, Freitag, Pass and Sirkin [27], constructed VDFs from polynomial hardness of LWE and *any* sequentially-hard function, but it does not satisfy uniqueness. We view it as an interesting question whether such VDFs have applications in cryptography.

The building block. Along the way (as in previous work) we construct an unambiguous, incremental, succinct non-interactive argument system for IS. This serves as the building block for all our results stated above. The soundness of our argument system is based on LWE, and is established by instantiating the Fiat-Shamir heuristic applied to a variant of Pietrzak's interactive proof system for IS. We also formulate an abstract protocol template (that we call "outline-and-batch" protocols) that generically implies **PPAD**-hardness and captures essentially all existing results as well as our new protocol.

1.2 Technical Overview

Toward the construction of hard **PPAD** instances, we resort to a common paradigm in the literature, that of constructing *mergeable and unambiguous proofs* [15,16,23,39,41,43]. In this paradigm, we consider some underlying computation:

$$x_1 \to x_2 \to \cdots \to x_T \ ,$$

where each step $x_t \to x_{t+1}$, $1 \le t < T$, is computable in fixed polynomial time, but computing the last state x_T cannot be done efficiently for large (super-polynomial) T. For concreteness, the reader may think of iterated squaring over the RSA group $\mathbb{Z}_N^\times$ where, for a (randomly sampled) $g \in \mathbb{Z}_N^\times$, $x_t := g^{2^t} \bmod N$; note that computing $x_t \to x_{t+1}$ can be carried out by one modular squaring, but computing x_T for a large T is believed to be infeasible [53]. For $1 \le t < t' \le T$, the corresponding proof system should allow computing (non-interactive) proofs $\pi_{t\to t'}$ for statements of the form $x_t \to x_{t'}$ (i.e., the state $x_{t'}$ is reachable from state x_t) and should satisfy the following requirements:

1. **Soundness:** it should be computationally hard to prove false statements.
2. **Unambiguity:** for any (true) statement $x_t \to x_{t'}$, it should be computationally hard to find any accepting proof $\pi^*_{t\to t'}$ other than the "prescribed" proof $\pi_{t\to t'}$ computed by the efficient merging process.
3. **Recursive proof-merging:** given d proofs $\pi_{1\to t}, \pi_{t\to 2t}, \ldots, \pi_{(d-1)t\to dt}$, for statements

$$x_1 \to x_t, x_t \to x_{2t}, \ldots, x_{(d-1)t} \to x_{dt},$$

computing a proof $\pi_{1\to dt}$ for the statement $x_1 \to x_{dt}$, where $d \in \mathbb{N}$ is some fixed merging parameter, can be efficiently reduced to computing a single proof $\pi'_{1\to t}$ for some related statement $x'_1 \to x'_t$. In other words, the d proofs for statements of "size" t can be merged into a proof for a statement of "size" dt via a recursive call to compute an additional (related) proof of size t. In the concrete example of iterated squaring, the "size" of the statement corresponds to the number of modular squaring operations required to go from $x_t := g^{2^t}$ to $x_{t'} := g^{2^{t'}}$.

Mergeable, Unambigous Proofs from Iterated Squaring and Fiat-Shamir. As mentioned, the mergeable unambiguous proofs paradigm has by now several instantiations in the literature. Focusing on obtaining a polynomial reduction, we consider one particular instantiation, based on Pietrzak's protocol for the iterated squaring (IS) problem [51]. The protocol is a public-coin interactive proof for statements of the form "g^{2^T} equals h modulo N", where N is a public modulus whose factorization is known to neither the prover nor the verifier – we denote such a statement by $g \xrightarrow{T} h$. At the heart of Pietrzak's protocol is a technique for reducing a statement $g \xrightarrow{T} h$ to a related, new statement $g' \xrightarrow{T/2} h'$ that is half the size.[7] This is done by having the (honest) prover specify an integer μ such that the intermediate statements $g \xrightarrow{T/2} \mu$ and $\mu \xrightarrow{T/2} h$ hold (i.e., μ is the "midpoint"), and then having the verifier *reduce* these two statements into one, using its random challenge r as follows:

$$g' := g^r \mu \bmod N \text{ and } h' := \mu^r h \bmod N.$$

The above, "halving sub-protocol" is repeated for $\log(T)$ rounds, at the end of which the verifier ends up with a statement of the form $g'' \xrightarrow{2} h''$, which it can, itself, check by modular squaring. To make this proof system non-interactive, previous works turn to the Fiat-Shamir paradigm [25] of applying an appropriate hash function to the statement to derive the verifier's challenge.

Instantiating Fiat-Shamir. Since Pietrzak's protocol has statistical soundness, the above approach already yields hard **PPAD** instances in the random oracle model [16,23]. Our focus is of course on obtaining a construction without random oracles. Indeed, a recent surge of results has successfully instantiated Fiat-Shamir without random oracles in various scenarios [8, 11, 12, 17, 18, 19, 20, 32, 33, 35, 38, 39, 42, 43, 50]. This has, in fact, also yielded hard **PPAD** instances, but so far none based on polynomial hardness assumptions. Especially relevant to us is the work of Lombardi and Vaikuntanathan [43] who instantiate the Fiat-Shamir transform for Pietrzak's protocol, based on *sub-exponential* hardness of LWE. At a high level, the sub-exponential loss in [43] comes from the difficulty of computing (or successfully guessing) the so called *bad verifier challenges* in the protocol — the precise quantitative complexity of this task turns out to crucially

[7] Throughout this section, we assume for simplicity that the time parameter T is a power of 2.

affect Fiat-Shamir instantiability. For the particular case of Pietrzak's protocol, a verifier challenge is bad if either of the intermediate statements $g \xrightarrow{T/2} \mu$ or $\mu \xrightarrow{T/2} h$ is false, but the new randomized statement $g' \xrightarrow{T/2} h'$ happens to be true. As a part of the soundness argument, it was demonstrated in [51] that the set of bad verifier challenges consists of at most a few elements, but it turns out that computing them (even given the factorization of N) requires solving an intractable discrete-log problem (see, e.g., [43] for a discussion).

Reduced Challenge Space and Soundness Amplification. A first observation toward eliminating the subexponential loss is that bad challenges in Pietrzak's protocol are *efficiently verifiable* given the factorization of N. In particular, there is a straight-forward modification of Pietrzak's protocol that uses a polynomial size challenge space, which makes it trivial to find the bad challenges by enumerating and testing every possibility (which can be done efficiently given the above observation). However, this modification causes the protocol to have inverse polynomial soundness error, so the resulting protocol cannot be made interactive via Fiat-Shamir.

A natural attempt to resolve this is to repeat the small-challenge protocol many times in parallel to reduce the soundness error. Indeed parallel repetition reduces the soundness error and importantly, using a recent work of Holmgren, Lombardi and Rothblum [33], we can even instantiate Fiat-Shamir for such a protocol based on (polynomially secure) LWE.[8] Their instantiation essentially works for any parallel-repeated three-message proof, as long as the bad challenges in each individual copy of the protocol are efficiently verifiable. (It also works for protocols with more rounds provided a certain round-by-round soundness requirement that is satisfied by Pietrzak's protocol, further discussed below). It turns out, however, that this approach falls short of our goal. The issue is that *the resulting proofs are not unambiguous*. As noted in [52], while parallel repetition amplifies soundness, it *does not* amplify unambiguity. The reason is that a cheating prover that breaks unambiguity in a *single* copy of the base protocol (out of many), can in particular obtain two accepting proofs for the same statement, breaking the unambiguity of the whole protocol.

Amplifying Unambiguity. As described above, while parallel repetition has the desired effect on soundness, unambiguity suffers from a *single point of failure*: that is, it suffices to cheat in a single copy of the base protocol without affecting the other copies. Instead we would like to start with a protocol that morally still works with many copies (as in parallel repetition) but mixes these together so that any deviations propagate across the entire protocol. Indeed such a protocol was constructed by Block et al. [5], who construct an interactive proof system for IS for a completely different purpose than considered here.[9] Specifically, for

[8] In fact, this yields a (non-unique) VDF based on the standard hardness of IS and LWE. However, this is subsumed by the result from [27] mentioned in Section 1.1. In Section 1.2, we will construct *unique* VDF from same assumptions.

[9] The goal in [5] (also see [30]) was to construct a statistically-sound protocol that works for IS in *arbitrary* groups. In comparison, Pietrzak's protocol is statistically-

an arbitrary group $\mathbb{G}$, they consider λ (possibly-identical) statements

$$\left(g_1 \xrightarrow{T} h_1, \cdots, g_\lambda \xrightarrow{T} h_\lambda\right), \tag{1}$$

where (in the honest case) $h_i = g_i^{2^T}$ over $\mathbb{G}$ for all $i \in [1, \lambda]$. As in Pietrzak's protocol, the prover sends over a tuple of midpoints $(\mu_1, \cdots, \mu_\lambda)$, for claimed values $\mu_i = g_i^{2^{T/2}}$. This results in 2λ intermediate statements of the form

$$\left(g_1 \xrightarrow{T/2} \mu_1, \mu_1 \xrightarrow{T/2} h_1, \cdots, g_\lambda \xrightarrow{T/2} \mu_\lambda, \mu_\lambda \xrightarrow{T/2} h_\lambda\right),$$

which we rewrite as

$$\left(\widetilde{g}_1 \xrightarrow{T/2} \widetilde{h}_1, \cdots, \widetilde{g}_{2\lambda} \xrightarrow{T/2} \widetilde{h}_{2\lambda}\right). \tag{2}$$

To recurse, λ new statements are derived by a 2λ-to-λ (batch) reduction, where the i-th new statement $g'_i \xrightarrow{T/2} h'_i$ is constructed by choosing a random subset S_i of the 2λ statements as follows:

$$g'_i = \prod_{j \in S_i} \widetilde{g}_j \text{ and } h'_i = \prod_{j \in S_i} \widetilde{h}_j. \tag{3}$$

Even if a single original statement in Eq. (1) is false, it was shown in [5] that each new statement in Eq. (3) is true with probability at most $1/2$ (over the choice of S_i) : intuitively, in the (worst) case that the j^*-th statement is the only false statement in Eq. (1), then it is included in Eq. (4) with probability $1/2$, rendering the new statement false. Since there are λ new statements, constructed using *independent* random subsets, the soundness of the resulting protocol is $1/2^\lambda$. Unambiguity amplifies in an identical manner: a cheating prover deviating from the prescribed honest prover strategy affects each new statement with probability roughly $1/2$, "propagating" false statements and, as a result, circumventing the issue of a single point of failure we had previously discussed. By recursing, as above, $\log(T)$ times, the statement reduces to a statement which can be efficiently checked by the verifier.

Applying [33]. Now that we have solved the issues with unambiguity in the interactive protocol, we would like to make it non-interactive in the common reference string (CRS) model by applying the Fiat-Shamir transform. Briefly, the Fiat-Shamir transform for any *public-coin* interactive proof is defined with respect to some hash function family $\mathcal{H}$, where a single hash function H sampled from this family is set to be the CRS. The round-collapse is due to the fact that the verifier's message for each round is simply computed to be the output of H applied to the transcript of the protocol up to that point. The security of the instantiated transform relies on correlation intractability of hash functions for

sound only in groups that are guaranteed to have no low-order elements, e.g., in the group of *signed quadratic residues* [26,31].

bad challenges [13]. This is, in particular, true for random oracles when the bad challenges are "sparse".

As already stated, the Fiat-Shamir transform has been successfully instantiated based on standard assumptions for several protocols. Of particular interest to our work is a recent work of Holmgren, Lombardi and Rothblum [33]. We illustrate their idea directly for the [5] protocol. Consider the 2λ intermediate statements from Eq. (2) and let $j^* \in [1, 2\lambda]$ be an index such that $\widetilde{g}_{j^*}^{2^{T/2}} \neq \widetilde{h}_{j^*}$. This can occur either due to the fact that one of the initial λ statements was incorrect, or a cheating prover deviated from the prescribed prover strategy. Then, the i-th new statement $g'_i \xrightarrow{T/2} h'_i$ is true if and only if

$$\prod_{j \in S_i} (\widetilde{g}_j)^{2^{T/2}} = \prod_{j \in S_i} \widetilde{h}_j. \tag{4}$$

Recall that the above only happens with probability at most $1/2$ and, consequently, the probability that at least one of the λ new statements is false is $1 - 1/2^{\lambda}$. We can now define the set of bad challenges, i.e., the *bad set* $\mathcal{B}$, that results in *all* λ new statements to be true. To be precise,

$$\mathcal{B} = \mathcal{B}_{(\tilde{g}_1, \cdots, \tilde{g}_{2\lambda}), (\tilde{h}_1, \cdots, \tilde{h}_{2\lambda}), T/2} := \Big\{ S_1, \cdots, S_\lambda \subseteq [1, 2\lambda] \;\Big|\; \prod_{j \in S_i} (\tilde{g}_j)^{2^{T/2}} = \prod_{j \in S_i} \tilde{h}_j \text{ for all } i \in [1, \lambda] \Big\}.$$

Note that $\mathcal{B}$[10] can be represented as the product of λ sets, i.e.

$$\mathcal{B} = \mathcal{B}_1 \times \cdots \times \mathcal{B}_\lambda, \tag{5}$$

where each

$$\mathcal{B}_i := \{ S_i \subseteq [1, 2\lambda] \mid \prod_{j \in S_i} (\widetilde{g}_j)^{2^{T/2}} = \prod_{j \in S_i} \widetilde{h}_j \}. \tag{6}$$

This *product structure* of $\mathcal{B}$ shown in Eq. (5) is crucial for us to invoke [33] who show, assuming polynomial hardness of LWE, that there exists a hash function family $\mathcal{H}$ such that the Fiat-Shamir transform is sound whenever $\mathcal{B}$ is a product set such that each $\mathcal{B}_i$ is *efficiently verifiable.*[11] Here the set $\mathcal{B}_i$ is said to be efficiently verifiable if there is a polynomial-sized circuit C that on input $((\tilde{g}_1, \cdots, \tilde{g}_{2\lambda}), (\tilde{h}_1, \cdots, \tilde{h}_{2\lambda}), i, S_i)$ that decides whether $S_i \in \mathcal{B}_i$. In our setting, C needs to check whether Eq. (4) holds, which can be done efficiently if C could compute the product $\prod_{j \in S_i} (\widetilde{g}_j)^{2^{T/2}}$ in Eq. (6), even for super-polynomial T. This is possible, for instance, in any group of the form $\mathbb{Z}_N^\times$ (including RSA groups) if C has a *trapdoor*, viz., the factorization of the modulus N, hardcoded in its description: C can first compute the intermediate value $e := 2^{T/2} \bmod \phi(N)$ using

[10] We drop the subscript for the $\mathcal{B}$ set for clarity when the subscript is clear from the context.

[11] We refer the reader to the technical section for full details on invoking [33].

the trapdoor and then compute g^e mod N by a single modular exponentiation. Thus, as long as we work in groups where one can efficiently verify each $\mathcal{B}_i$ (with the help of a trapdoor), the Fiat-Shamir transform applied to the [5] protocol is a secure non-interactive argument in the CRS model.

Additionally, in order for the resulting non-interactive argument to preserve properties of the multi-round unambiguous interactive proof, the protocol needs to satisfy the stronger property [10, 43] of *unambiguous round-by-round soundness*. In the technical section, we show that the soundness and unambiguity discussion of [5] earlier easily extend to satisfy this property.

Application to Unique VDFs Having constructed an unambiguous (succinct) non-interactive argument system for IS, we essentially immediately obtain a VDF family with *unique* proofs based on (1) the polynomial hardness of LWE, and (2) the assumption that IS is an inherently sequential function. The only detail that needs to be verified is that the computational complexity of the prover is $T \cdot (1 + o(1))$ for T sequential squarings. This can be proved following an analogous argument in [51]: after applying $T + 1$ sequential squaring operations

$$g_0 = g, g_1 = g^2, \ldots, g_T = g^{2^T},$$

it is possible to compute all prover messages with $\mathsf{poly}(\lambda) \cdot \sqrt{T}$ additional group operations as follows.

- Compute all prover messages from round $\frac{1}{2}\log(T)$ onwards with the naive prover algorithm, incurring an additive computational overhead of $\mathsf{poly}(\lambda) \cdot \sqrt{T}$, and
- Compute all prover messages in the first $\frac{1}{2}\log(T)$ rounds by storing $\sqrt{T}$ of the computed g_is, where each prover message is computed a product-combinations of a (pre-determined) subset of these stored values. This incurs a *total* additive overhead of $\mathsf{poly}(\lambda) \cdot \sqrt{T}$.

Remark 1 (Comparison to the [43] VDF). The [43] VDF uses complexity leveraging in a way so that the honest prover is only efficient (relative to the squaring computation) when the squaring parameter T is subexponentially large in the description of the RSA modulus. Relatedly, the protocol then only achieves a slightly superpolynomial gap between the complexity of the honest prover and the complexity of the cheating provers ruled out by soundness. In contrast, our construction does not require complexity leveraging, resulting in a VDF with far more standard efficiency parameters.

Applications to PPAD-Hardness For establishing hardness of **PPAD**, we have to show that the non-interactive argument obtained above satisfies the third requirement, i.e., recursive proof-merging. The two sets of intermediate statements from Eq. (2) can be succinctly denoted as

$$(g_1, \cdots, g_\lambda) \xrightarrow{T/2} (\mu_1, \cdots, \mu_\lambda) \text{ and } (\mu_1, \cdots, \mu_\lambda) \xrightarrow{T/2} (h_1, \cdots, h_\lambda) \tag{7}$$

with corresponding [5] proofs

$$\pi((g_1, \cdots, g_\lambda) \xrightarrow{T/2} (\mu_1, \cdots, \mu_\lambda)) \text{ and } \pi((\mu_1, \cdots, \mu_\lambda) \xrightarrow{T/2} (h_1, \cdots, h_\lambda)).$$

The proof for $(g_1, \cdots, g_\lambda) \xrightarrow{T} (h_1, \cdots, h_\lambda)$ can be computed as

$$\pi((g_1, \cdots, g_\lambda) \xrightarrow{T} (h_1, \cdots, h_\lambda)) \coloneqq \left((\mu_1, \cdots, \mu_\lambda), \pi((g'_1, \cdots, g'_\lambda) \xrightarrow{T/2} (h'_1, \cdots, h'_\lambda))\right)$$

where $(g'_1, \cdots, g'_\lambda) \xrightarrow{T/2} (h'_1, \cdots, h'_\lambda)$ is derived via the 2λ-to-λ (batch) reduction from the statements in Eq. (7). Furthermore, the proof

$$\pi((g'_1, \cdots, g'_\lambda) \xrightarrow{T/2} (h'_1, \cdots, h'_\lambda))$$

is generated by recursing on the statement $(g'_1, \cdots, g'_\lambda) \xrightarrow{T/2} (h'_1, \cdots, h'_\lambda)$ to compute its proof. Since the reduction is efficient, the non-interactive argument satisfies recursive proof merging as desired. As shown in [15], this actually implies hardness of the sub-class **CLS** $\subseteq$ **PPAD**. We strengthen this result further to show hardness in **UEOPL** $\subseteq$ **CLS**, one of the lowest-lying sub-classes of **TFNP** [24].

Remark 2 (Abstract protocol). While we have limited our discussion specifically to the case of IS, in the technical sections (Sections 3 and 5) we describe an abstract protocol template that we call "outline and batch." We show that any problem family admitting a *downward self-reduction* and a *(randomized) batching reduction* (reducing k' instances of the problem to sufficiently fewer $k < k'$ instances) admits an unambiguous and incremental non-interactive argument system that suffices for our hardness results. We refer the reader to the technical sections for details.

Obtaining Public-Coin Hardness in PPAD. Finally, we discuss how to obtain hard distributions of **PPAD** instances that are *publicly samplable* under the same computational assumptions as before: the polynomial hardness of LWE and IS over RSA group. It is a priori unclear why one should expect to obtain public-coin hardness under these assumptions, since we don't know a public-coin algorithm for sampling an RSA modulus! Nevertheless, we obtain the result via the following two ideas.

First, we observe that our Fiat-Shamir hash function $\mathcal{H}$ can be sampled from a public-coin distribution. In [33], the hash functions have a *computationally pseudorandom* (and private-coin) description, but they can be switched to uniformly random because even the adaptive unambiguous soundness of the protocol considered in our work is an efficiently verifiable property (given the group order as a trapdoor). Put another way, the adaptive soundness of the protocol follows from an efficiently falsifiable form of correlation intractability, which is thus preserved under computational indistinguishability.

The more serious issue is how to handle the *group* (and group element) description. We handle this by working over $\mathbb{Z}_N^\times$ for a *different* value of N (rather

than an RSA modulus). A naive idea would be to work over a *uniformly random modulus* N; unfortunately, the squaring problem mod a uniformly random N is not hard, because N will be prime with inverse polynomial probability (by the prime number theorem), in which case the group order $\phi(N) = N-1$ is efficiently computable. Our actual solution is as follows: consider $N = N_1 \cdot \ldots \cdot N_{\mathsf{poly}(\lambda)}$ for a sufficiently large $\mathsf{poly}(\lambda)$, where all integers N_i are public and uniformly random in the range $[1, 2^\lambda]$. First of all, we note that our techniques for constructing hard **PPAD** instances from iterated squaring apply to this choice of modulus as well: all that is required is that there is a way to efficiently sample (necessarily using *secret coins*) the squaring problem description along with a trapdoor containing the group order $|\mathbb{Z}_N^\times| = \phi(N)$ (this is captured by our generic construction). This is possible using efficient algorithms for generating random factored integers [2, 40].

This tells us that public-coin hardness in **PPAD** follows from the hardness of LWE along with the polynomial hardness of IS modulo N (given the coins for sampling the IS instance). To complete the proof, we show that this follows from the polynomial hardness of (secret-coin) IS in an RSA modulus. We prove this by a direct reduction that embeds an RSA modulus IS problem instance into a public-coin instance of this new IS problem; crucially, we use the fact that with all but negligible probability over $N_1, \ldots, N_{\mathsf{poly}(\lambda)}$, at least one N_i is an RSA modulus.

1.3 Organisation

We state definitions and provide background relevant to the paper in Section 2. In Section 3, we describe the abstract 'outline-and-batch' protocol, prove its unambiguous soundness and explain how existing protocols fit this abstraction. In Section 4, we describe the unambiguous non-interactive argument for IS that forms the basis of the results in this work. Hardness of the class **PPAD** is shown in Section 5 by constructing hard instances of RSVL using the results from Section 4. Due to a lack of space, we defer some details to the full version of the paper.

2 Preliminaries

Notation. First, we list the notation that will be used throughout this paper.

1. For $a, b \in \mathbb{N}$, $a < b$, by $[a, b]$ we denote the sequence of integers $\{a, a+1, \cdots, b\}$.
2. For an alphabet Σ and $n \in \mathbb{N}$, we write Σ^n, $\Sigma^{<n}$ and $\Sigma^{\leq n}$ to denote, respectively, strings over Σ with length equal to, less than, and less than or equal to n. We use ε to denote the empty string. For strings a and b we use ab to denote string concatenation.
3. Vectors and tuples are in bold face. We parse a vector or a tuple $\boldsymbol{x} \in \mathcal{X}^k$ as $\boldsymbol{x} =: (x_0, \cdots, x_{k-1})$; $\boldsymbol{x}$ is said to be a k-vector. A subscripted vector $\boldsymbol{x}_v \in \mathcal{X}^k$ is parsed as $\boldsymbol{x}_v =: (x_{v,0}, \cdots, x_{v,k-1})$.

4. For $x \in \mathcal{X}$, $y \in \mathcal{Y}$ and a function $f : \mathcal{X} \to \mathcal{Y}$, we write $x \xrightarrow{f} y$ to denote the (true or false) *statement* "y equals $f(x)$". Sometimes, when the context is clear, we will simplify the notation: e.g., for $h := g^{2^T} \bmod N$, we simply write $g \xrightarrow{T} h$ to denote
$$g \xrightarrow{(\cdot)^{2^T} \bmod N} h.$$
We extend this notation to vectors: for $\boldsymbol{x} \in \mathcal{X}^k$ and $\boldsymbol{y} \in \mathcal{Y}^k$, we define $\boldsymbol{f} = f^k : \mathcal{X}^k \to \mathcal{Y}^k$ as $(f(x_0), \cdots, f(x_{k-1}))$ and therefore $\boldsymbol{x} \xrightarrow{\boldsymbol{f}} \boldsymbol{y}$ denotes statement that $x_i \xrightarrow{f} y_i$ for all $i \in [0, k-1]$.
5. For a statement x, we denote $\pi(x)$ to denote a non-interactive proof for x. For example, for x, y and f as in Item 4, we write $\pi(x \xrightarrow{f} y)$ to denote a non-interactive proof for the statement $x \xrightarrow{f} y$.
6. For $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, we write $y := \mathsf{A}(x)$ (resp., $y \leftarrow \mathsf{A}(x)$) to denote the execution of a deterministic (resp., randomised) algorithm A on input x to output y. For $k \in \mathbb{N}$, vectors $\boldsymbol{x} \in \mathcal{X}^k$ and $\boldsymbol{y} \in \mathcal{Y}^k$, we denote repeated *parallel* execution of A by $\boldsymbol{y} := \mathbf{A}(\boldsymbol{x})$, i.e., $y_i := \mathsf{A}(x_i)$ for all $i \in [0, k-1]$.

2.1 Search Problems, TFNP, and Reductions

We define below search problems, and the relevant complexity classes needed for our work. We start by defining search problems.

Definition 1 (Search Problems [1]). *A* search problem *is a relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$. Let $\mathcal{R}(x)$ denote $\{y : (x,y) \in \mathcal{R}\}$. A function $f : \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$ is said to* solve *$\mathcal{R}$ if for every $x \in \{0,1\}^*$ satisfying $\mathcal{R}(x) \neq \emptyset$, it holds that $f(x) \in \mathcal{R}(x)$; and for all other x, $f(x) = \perp$.*

Definition 2 (Total Relations). *A relation $\mathcal{R}$ is said to be* total *if for all $x \in \{0,1\}^*$, there exists y such that $(x,y) \in \mathcal{R}$.*

Definition 3 (Polynomially Balanced). *A relation $\mathcal{R}$ is said to be* polynomially balanced *if there is a polynomial p such that for any strings $x, y \in \{0,1\}^*$, if $(x,y) \in \mathcal{R}$ then $|y| \leq p(|x|)$.*

Definition 4 (FNP). *The complexity class* **FNP** *consists of all polynomially balanced search problems $\mathcal{R}$ for which there is a polynomial-time algorithm that on input (x,y) outputs whether or not $(x,y) \in \mathcal{R}$.*

Definition 5 (TFNP). *The complexity class* **TFNP** *consists of all total search problems in* **FNP**.

For further discussion of relevant sub-classes of **TFNP**, we refer the reader to the full version of this work.

Definition 6 (Reductions). *If P and Q are search problems, a* randomized Karp reduction from P to Q with error $\epsilon(\cdot)$ *is a pair of p.p.t. machines (M, N) such*

that if f is a function that solves Q, then for any $x \in \{0,1\}^n$ with $P(x) \neq \emptyset$, we have

$$\Pr\left[(x,y) \in P\right] \geq 1 - \epsilon(n)$$

when sampling $x' \leftarrow \mathsf{M}(x)$, $y \leftarrow \mathsf{N}(f(x'))$.

Next, we consider the search problem RelaxedSinkOfVerifiableLine (RSVL), which is relevant to the main result of this paper. We point out that RSVL *not* a total problem since, looking ahead, there is no way to syntactically guarantee that the successor and verifier circuits are well-behaved (see Remark 3).

Definition 7 ([15]). RelaxedSinkOfVerifiableLine *(*RSVL*)*

- Instance.
  1. *Boolean circuit* $\mathsf{S} : \{0,1\}^m \to \{0,1\}^m$
  2. *Boolean circuit* $\mathsf{V} : \{0,1\}^m \times [0, 2^m - 1] \to \{\mathsf{accept}, \mathsf{reject}\}$
  3. *Integer* $L \in [0, 2^m - 1]$
  4. *String* $v_0 \in \{0,1\}^m$
- Promise. *For every* $v \in \{0,1\}^m$ *and* $i \in [0, 2^m - 1]$, $\mathsf{V}(v,i) = 1$ if $i \leq L$ *and* $v = \mathsf{S}^i(v_0)$.
- Solution. *One of the following:*
  1. **The sink:** *a vertex* $v \in \{0,1\}^m$ *such that* $\mathsf{V}(v, L) = 1$*; or*
  2. **False positive:** *a pair* $(v,i) \in \{0,1\}^m \times [0, 2^m - 1]$ *such that* $v \neq \mathsf{S}^i(v_0)$ *and* $\mathsf{V}(v,i) = 1$.

Remark 3. It seems likely that RSVL is not in **FNP**, let alone in **PPAD**. Specifically, checking that a pair (v,i) constitutes a false positive is difficult because i may be super-polynomial in the instance size.

Nevertheless, [15] constructed a (randomized) reduction from RSVL to EOML (which is a search problem complete for **CLS** $\subseteq$ **PPAD**) with error that is inversely polynomially bounded away from 1. This error is somewhat large, and allows for the possibility EOML is "slightly" easier than RSVL.

Still, the reduction suffices for establishing the standard cryptographic hardness of EOML (i.e. that no polynomially bounded algorithm can succeed with *any* non-negligible probability) based on analogous hardness for RSVL. In turn, we establish the latter hardness based on LWE (Assumption 4) and the iterated squaring assumption (Assumption 9).

Theorem 3 ([15]). *There is a randomized Karp reduction from* RSVL *to* EOML *with error probability* $\epsilon(n) = 1 - n^{-O(1)}$.

2.2 Learning with Errors

The following standard preliminaries about the Learning with Errors (LWE) problem are based on [43, 49].

Definition 8 (LWE Distribution). *For any* $\mathbf{s} \in \mathbb{Z}_q^n$ *and any distribution* $\chi \subseteq \mathbb{Z}_q$*, the* LWE distribution $A_{\mathbf{s},\chi} \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ *is sampled by choosing* $\mathbf{a} \in \mathbb{Z}_q^n$ *uniformly at random, sampling* $e \leftarrow \chi$*, and outputting* $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e)$*.*

Assumption 4 (Decision LWE)**.** Let $m = m(n) \geq 1$, $q = q(n) \geq 2$ be integers, and let $\chi(n)$ be a probability distribution on $\mathbb{Z}_{q(n)}$. The $\mathsf{LWE}_{n,m,q,\chi}$ problem, parameterized by n, is to distinguish whether $m(n)$ independent samples are drawn from $A_{\mathbf{s},\chi}$ (for $\mathbf{s}$ that is sampled uniformly at random) or are drawn from the uniform distribution. The hardness assumption is that is hard for $\mathsf{poly}(n)$-sized adversaries to decide the $\mathsf{LWE}_{n,m,q,\chi}$ problem.

2.3 Correlation-Intractable Hash Families

The following preliminaries are partially taken from [33, 43].

Definition 9 (Hash family). *For a pair of efficiently-computable functions* $(n(\cdot), m(\cdot))$*, a* hash family *with input length* n *and output length* m *is a collection* $\mathcal{H} = \{H_\lambda : \{0,1\}^{s(\lambda)} \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{m(\lambda)}\}_{\lambda \in \mathbb{N}}$ *of keyed hash functions, along with a pair of p.p.t. algorithms:*

- $\mathcal{H}.\mathsf{Gen}(1^\lambda)$ *outputs a hash key* $k \in \{0,1\}^{s(\lambda)}$*.*
- $\mathcal{H}.\mathsf{Hash}(k, x)$ *computes the function* $H_\lambda(k, x)$*. We may use the notation* $H(k, x)$ *to denote hash evaluation when the hash family is clear from context.*

As in prior works [11, 50] we consider the security notion of correlation intractability [13] for single-input relations and its restriction to (single-input) functions.

Definition 10 (Correlation Intractability). *For a given relation ensemble* $\mathcal{R} = \{\mathcal{R}_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}\}$*, a hash family* $\mathcal{H} = \{H_\lambda : \{0,1\}^{s(\lambda)} \times \{0,1\}^{n(\lambda)} \to \{0,1\}^{m(\lambda)}\}$ *is said to be* $\mathcal{R}$-correlation intractable with security (s, δ) *if for every* s*-size* $\mathsf{A} = \{\mathsf{A}_\lambda\}$*,*

$$\Pr_{\substack{k \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda) \\ x \leftarrow \mathsf{A}(k)}} \Big[\big(x, H(k,x)\big) \in \mathcal{R} \Big] = O(\delta(\lambda)).$$

We say that $\mathcal{H}$ *is* $\mathcal{R}$-correlation intractable with security δ *if it is* (λ^c, δ)*-correlation intractable for all* $c > 1$*. Finally, we say that* $\mathcal{H}$ *is* $\mathcal{R}$-correlation intractable *if it is* $(\lambda^c, 1/\lambda^c)$*-correlation intractable for all* $c > 1$*.*

We will use the recent result of [33] on correlation intractability for *product relations*.

Definition 11 (Product Relation). *We say that* $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}^t$ *is a* product relation *if for every* $x \in \mathcal{X}$*, the set* $\mathcal{R}(x) = \{y : (x, y) \in \mathcal{R}\} \subseteq \mathcal{Y}^t$ *has a decomposition*

$$\mathcal{R}(x) := \mathcal{B}_1(x) \times \mathcal{B}_2(x) \times \ldots \mathcal{B}_t(x)$$

(where each $\mathcal{B}_i(x)$ *is a subset of* $\mathcal{Y}$*). We say that such an* $\mathcal{R}$ *is* efficiently product verifiable *if for some such choice of* $\mathcal{B}_i$*, there is a poly-size circuit* $\mathsf{C}(x, i, y_i)$ *that decides whether* $y_i \in \mathcal{B}_i(x)$*.*

Theorem 5 ([33]). *Assume the hardness of* LWE*. Then, for every size bound* $S(\lambda) = \mathsf{poly}(\lambda)$*, input length* $n(\lambda)$*, and output length* $m(\lambda) \cdot t(\lambda)$ *such that* $t(\lambda) \geq \lambda^{\Omega(1)}$*, there exists a correlation intractable hash family* $\mathcal{H}$ *for product relations* $\mathcal{R}$ *that are (1) product verifiable by size* $S(\lambda)$ *circuits, and (2) sparse in the sense that for every* x, i*, we have that* $|\mathcal{B}_i(x)| \leq \frac{1}{2} \cdot 2^{m(\lambda)}$*.*

Remark 4. In [33], hash function keys have a computationally pseudorandom distribution. However, for the purposes of Theorem 5, hash function keys may be taken to be uniformly random strings (by invoking the pseudorandomness property), because the security property in Theorem 5 is efficiently falsifiable.

2.4 Interactive Proofs and the Fiat-Shamir Heuristic

The following preliminaries are partially taken from [33,43]. We begin by recalling the definitions of interactive proofs and arguments.

Definition 12 (Interactive proof and argument system). *An* interactive proof *(resp.,* interactive argument*) for a promise problem* $\mathcal{L} = (\mathcal{L}_{\text{YES}}, \mathcal{L}_{\text{NO}})$ *is a pair* (P, V) *of interactive algorithms satisfying:*

- **Completeness.** *For any* $x \in \mathcal{L}_{\text{YES}}$*, when* P *and* V *interact on common input* x*, the verifier* V *outputs* 1 *with probability* 1*.*
- **Soundness.** *For any* $x \in \mathcal{L}_{\text{NO}} \cap \{0,1\}^n$ *and any* unbounded *(resp.,* polynomial-time*) interactive* P^**, when* P^* *and* $\mathsf{V}(x)$ *interact, the probability that* V *outputs* 1 *is a negligible function of* n*.*

The protocol is public-coin *if each of* V*'s messages is an independent uniformly random string of some length (and the verifier's decision to accept or reject does not use any secret state). In this setting, we will denote prover messages by* $(\alpha_1, \ldots, \alpha_\ell)$ *and verifier messages by* $(\beta_1, \ldots, \beta_{\ell-1})$ *in a* $2\ell - 1$*-round protocol.*

Definition 13 (Non-interactive argument system). *A* non-interactive *argument scheme (in the CRS model) for a promise problem* $\mathcal{L} = (\mathcal{L}_{\text{YES}}, \mathcal{L}_{\text{NO}})$ *is a triple* $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *of* non-interactive *algorithms with the following properties:*

- $\mathsf{Setup}(1^n)$ *outputs a common reference string* CRS*.*
- $\mathsf{P}(\mathsf{CRS}, x)$ *outputs a proof* π*.*
- $\mathsf{V}(\mathsf{CRS}, x, \pi)$ *outputs a bit* $b \in \{0,1\}$

It satisfies the notions of completeness and (computational) soundness as above.

We next define the notion of *unambiguous soundness* [52]. For non-interactive arguments, the soundness notion we consider is *adaptive* in that we allow the prover P^* to adaptively choose the statement x after seeing the CRS.

Definition 14 (Unambiguous Soundness [15, 52]). *A public-coin interactive proof system Π is* unambiguously sound *if (1) it is sound, and (2) for every $x \in \mathcal{L}$ and every (complete) collection of verifier messages $(\beta_1, \ldots, \beta_{\ell-1})$, there exists a distinguished proof $\pi^*(x, \beta_1, \ldots, \beta_{\ell-1})$ such that the following soundness condition holds: For all $x \in \mathcal{L}$ and all cheating provers P^*, the probability that the transcript $\langle \mathsf{P}^*(x), \mathsf{V}(x) \rangle$ contains a proof π such that $\mathsf{V}(x, \pi) = 1$ and $\pi \neq \pi^*(x, \beta_1, \ldots, \beta_{\ell-1})$ is negligible.*

Definition 15 (Adaptive Unambiguous Soundness).
A non-interactive argument system $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ is adaptively unambiguously sound *against (uniform or non-uniform) time T adversaries if for all instances $x \in \mathcal{L}$ and all common reference strings CRS, there exists a "distinguished proof" $\pi^*(\mathsf{CRS}, x)$ such that the following soundness condition holds: For all time T cheating provers P^*, the probability that $\mathsf{P}^*(\mathsf{CRS}) = (x, \pi)$ where $\mathsf{V}(x, \pi) = 1$ and* either *$x \notin \mathcal{L}$ or $\pi \neq \pi^*(\mathsf{CRS}, x)$ is negligible.*

Our results proceed by constructing (unambiguously sound) interactive proof systems and compiling them into non-interactive argument systems using the Fiat-Shamir transform, which we describe next.

Definition 16 (Fiat-Shamir Transform). *Let Π denote a public coin interactive proof (or argument) system Π that has ℓ prover messages and $\ell - 1$ verifier messages of length $m = m(\lambda)$. Then, for a hash family*

$$\mathcal{H} = \left\{ \left\{ H_k : \{0,1\}^* \to \{0,1\}^{m(\lambda)} \right\}_{k \in \{0,1\}^\lambda} \right\}_\lambda,$$

we define the Fiat-Shamir non-interactive protocol $\Pi_{\mathrm{FS},\mathcal{H}} = (\mathsf{Setup}, \mathsf{P}_{\mathrm{FS}}, \mathsf{V}_{\mathrm{FS}})$ as follows:

- $\mathsf{Setup}(1^\lambda)$*: sample a hash key $k \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda)$.*
- $\mathsf{P}_{\mathrm{FS}}(x)$*: for $i \in \{1, \ldots, \ell\}$, recursively compute the following pairs (α_i, β_i):*
  - *Compute $\alpha_i = \mathsf{P}(\tau_i)$ for $\tau_i = (x, \alpha_1, \beta_1, \ldots, \alpha_{i-1}, \beta_{i-1})$.*
  - *Compute $\beta_i = H_k(\tau_{i-1}, \alpha_i)$.*

  Then, $\mathsf{P}_{\mathrm{FS}}(x)$ outputs $\pi = (\alpha_1, \beta_1, \ldots, \alpha_\ell)$.
- $\mathsf{V}_{\mathrm{FS}}(\mathsf{CRS}, x, \pi)$ *parses $\pi = (\alpha_1, \beta_1, \ldots, \alpha_\ell)$ and verifies that:*
  - *$\beta_i = H_k(\tau_{i-1}, \alpha_i)$ for all $1 \leq i \leq \ell - 1$, and*
  - *$\mathsf{V}(x, \pi) = 1$.*

We note the following facts about $\Pi_{\mathrm{FS},\mathcal{H}}$

1. *The honest prover complexity of $\Pi_{\mathrm{FS},\mathcal{H}}$ is equal to the honest prover complexity of Π with an additive overhead of computing $\ell - 1$ hash values.*
2. *The verifier complexity of $\Pi_{\mathrm{FS},\mathcal{H}}$ is equal to the verifier complexity of Π with the same hashing additive overhead.*
3. *The protocol $\Pi_{\mathrm{FS},\mathcal{H}}$ is not necessarily sound, even if Π is sound and $\mathcal{H}$ is a "strong cryptographic hash function". As we will discuss later, soundness* is *guaranteed when Π satisfies what is called "round-by-round soundness", defined next.*

Round-by-Round (Unambiguous) Soundness and Fiat-Shamir. Following [10,11,15,43], we consider the notion of round-by-round (unambiguous) soundness to capture a particular kind of soundness analysis for super-constant round interactive proofs. For these proof systems, it has been shown that correlation intractability for an appropriate relation suffices for a hash family to instantiate the Fiat-Shamir heuristic for unambiguously round-by-round sound interactive proofs.

Definition 17 (Unambiguous Round-by-Round Soundness [10,15,43]). *Let $\Pi = (\mathsf{P}, \mathsf{V})$ be a $2\ell - 1$-message public coin interactive proof system for a language $\mathcal{L}$.*

We say that Π has unambiguous round-by-round soundness error $\epsilon(\cdot)$ *if there exist functions* $(\mathsf{State}, \mathsf{NextMsg})$ *with the following syntax.*

- State *is a deterministic (not necessarily efficiently computable) function that takes as input an instance x and a transcript prefix τ and outputs either* accept *or* reject.
- $\mathsf{NextMsg}$ *is a deterministic (not necessarily efficiently computable) function that takes as input an instance x and a transcript prefix τ and outputs a (possibly aborting) prover message $\alpha \in \{0,1\}^* \cup \{\bot\}$.*

We additionally require that the following properties hold.

1. *If $x \notin \mathcal{L}$, then $\mathsf{State}(x, \emptyset) =$ reject, where $\emptyset$ denotes the empty transcript.*
2. *If $\mathsf{State}(x, \tau) =$ reject for a transcript prefix τ, then $\mathsf{NextMsg}(x, \tau) = \bot$. That is, $\mathsf{NextMsg}(x, \tau)$ is only defined on accepting states.*
3. *For every input x and partial transcript $\tau = \tau_i$, then for every potential prover message $\alpha_{i+1} \neq \mathsf{NextMsg}(x, \tau)$, it holds that*

$$\Pr_{\beta_{i+1}}\left[\mathsf{State}\big(x, \tau|\alpha_{i+1}|\beta_{i+1}\big) = \mathsf{accept}\right] \leq \epsilon(n)$$

4. *For any* full[12] *transcript τ, if $\mathsf{State}(x, \tau) =$ reject then $\mathsf{V}(x, \tau) = 0$.*

We say that Π is unambiguously round-by-round sound *if it has unambiguous round-by-round soundness error ϵ for some $\epsilon(n) = \mathsf{negl}(n)$.*

Next, we restate the result that specific forms of correlation intractability suffice to instantiate the Fiat-Shamir transform for protocols satisfying unambiguous round-by-round soundness.

Theorem 6 ([10,43]). *Suppose that $\Pi = (\mathsf{P}, \mathsf{V})$ is a $2\ell-1$-message public-coin interactive proof for a language $\mathcal{L}$ with perfect completeness and unambiguous round-by-round soundness with corresponding functions* $(\mathsf{State}, \mathsf{NextMsg})$*. Let $\mathcal{X}_n$ denote the set of partial transcripts (including the input and all messages sent) and let $\mathcal{Y}_n$ denote the set of verifier messages when Π is executed on an input of length n.*

[12] By a full transcript, we mean a transcript for which the verifier halts.

Finally, define the relation ensemble $\mathcal{R} = \mathcal{R}_{\mathsf{State},\mathsf{NextMsg}}$ *as follows:*

$$\mathcal{R}^{(n)}_{\mathsf{State},\mathsf{NextMsg}} := \left\{ \left((x,\tau|\alpha),\beta\right) : \begin{array}{c} x \in \{0,1\}^n, \\ \alpha \neq \mathsf{NextMsg}(x,\tau) \\ \textit{and} \\ \mathsf{State}(x,\tau|\alpha|\beta) = \mathsf{accept} \end{array} \right\}.$$

If a hash family $\mathcal{H} = \{\mathcal{H}_n : \mathcal{X}_n \to \mathcal{Y}_n\}$ *is correlation intractable for* $\mathcal{R}$*, then the round-reduced protocol* $\Pi_{\mathrm{FS},\mathcal{H}}$ *is an adaptively unambiguously sound argument system for* $\mathcal{L}$*.*

3 The Outline-and-Batch Protocol

For $\mathcal{X}, \mathcal{Y} \subseteq \{0,1\}^*$, let $f : \mathcal{X} \to \mathcal{Y}$ be a function and $\|\cdot\| : \mathcal{X} \to \mathbb{N}$ denote a "size measure" for inputs to f. Let $\mathcal{X}_n$ denote $\{x \in \mathcal{X} : \|x\| = n\}$, and let $f_n : \mathcal{X}_n \to \mathcal{Y}$ denote the restriction of f to $\mathcal{X}_n$. Recall from Section 2 that $f^k_n : \mathcal{X}^k_n \to \mathcal{Y}^k$ denotes the function mapping $(x_1, \ldots, x_k)$ to $(f_n(x_1), \ldots, f_n(x_k))$.

Definition 18 (Downwards self-reduction). *A* downwards self-reduction *for* f *is a deterministic oracle algorithm* D *such that for any* n *and* $x \in \mathcal{X}_n$*,* $\mathsf{D}^{f_{n-1}}(x) = f(x)$*. If on input* x*,* D *queries* $q_1, \ldots, q_d$*, then we say that* D *is a* d*-query downwards self-reduction and we refer to* $((q_1, f(q_1)), \ldots, (q_k, f(q_d)))$ *as an* outline *of the evaluation of* f *on* x*.*

Definition 19 (Batching reduction). *A* k'-to-k batching reduction *for* f *with* soundness error ϵ *is a probabilistic algorithm* B *that on input* $\{(x'_i, y'_i) \in \mathcal{X}_n \times \mathcal{Y}\}_{i\in[1,k']}$ *outputs* $\{(x_i, y_i) \in \mathcal{X}_n \times \mathcal{Y}\}_{i\in[1,k]}$ *such that:*

- *(Completeness) If* $y'_i = f(x'_i)$ *for all* $i \in [1, k']$*, then with probability* 1*,* $y_i = f(x_i)$ *for all* $i \in [1, k]$*.*
- *(Soundness) If* $y'_i \neq f(x'_i)$ *for some* $i \in [1, k']$*, then with all but* ϵ *probability over the randomness of* B*,* $y_i \neq f(x_i)$ *for some* $i \in [1, k]$*.*

We remark that it may be useful to consider batching reductions that are interactive*, but for our purposes, non-interactive batching reductions suffice for our instantiations. We leave discussion of abstract interactive batching reductions to future work.*

Theorem 7. *If* f *has a* d*-query downwards self reduction* D *and a* dk*-to-*k*-batching reduction* B *with error* ϵ*, then there is a public-coin interactive proof for the language*

$$\mathcal{L}^k_{f_n} := \{((x_1, \cdots, x_n), (y_1, , \cdots, y_n)) \in \mathcal{X}^k \times \mathcal{Y}^k : f_n(x_i) = y_i \textit{ for all } i \in [1, k]\}$$

with $n-1$ *rounds of interaction and with unambiguous round-by-round soundness error* ϵ*.*

Remark 5. We remark that the hypotheses of Theorem 7 can be relaxed to only require completeness and soundness for B when applied to inputs that correspond to the queries of an evaluation of D. This relaxation captures the classical sumcheck protocol [44] as a special case.

Proof of Theorem 7. The prover P and verifier V both take as input a statement $((x_1, \ldots, x_k), (y_1, \ldots, y_k)) \in \mathcal{X}_n^k \times \mathcal{Y}_n^k$, and the protocol is defined recursively.

Base Case: If $n = 1$, then no messages are sent (P does nothing), and V accepts only if $f_1(x_i) = y_i$ for all $i \in [1, k]$.

Recursive Case: If $n > 1$, then:

1. P computes $\mathsf{D}^{f_{n-1}}(x_i)$ for each $i \in [1, k]$, recording the queries made by D and answering queries according to f_{n-1}. Then P sends all k corresponding d-tuples of query-answer pairs to V. Let $((\tilde{x}'_1, \tilde{y}'_1), \ldots, (\tilde{x}'_{dk}, \tilde{y}'_{dk}))$ denote the concatenation of all k d-tuples of query-answer pairs received by V.
2. When V receives k d-tuples of query-answer pairs, V checks for each i that the i-th tuple is consistent with an execution of D[13] on input x_i (if not, then V rejects). V then samples randomness r for B and sends it to P.
3. Let $((\tilde{x}''_1, \tilde{y}''_1), \ldots, (\tilde{x}''_k, \tilde{y}''_k))$ denote $\mathsf{B}((\tilde{x}'_1, \tilde{y}'_1), \ldots, (\tilde{x}'_{dk}, \tilde{y}'_{dk}); r)$. P and V recursively invoke the interactive proof for f^k on $((\tilde{x}'_1, \ldots, \tilde{x}'_k), (\tilde{y}'_1, \ldots, \tilde{y}'_k))$.

We next describe how to give Π the structure of an unambiguous round-by-round sound protocol.

- At any step in the recursion, we have "current inputs" $x_1, \ldots, x_k$ as well as outputs $y_1, \ldots, y_k$ claimed in the previous recursive step. At this execution point, we define State to be accept if and only if $f(x_i) = y_i$ for all i.
- After Step 1 in a recursive call, we define State to be accept iff $\mathcal{V}$ has not rejected and $f(\tilde{x}') = \tilde{y}'$ for every pair $(\tilde{x}', \tilde{y}')$ in the lists sent by the prover.

We define $\mathsf{NextMsg}(\tau)$ to be the output of the honest prover algorithm in the description of the recursion above, which means to:

- Compute B on its previous message and the verifier's challenge r, and
- Compute the downwards self-reduction on the resulting tuple of inputs.

Given this description of $(\mathsf{State}, \mathsf{NextMsg})$, unambiguous round-by-round soundness follows from the correctness of the downwards self-reduction and the soundness of the batching reduction. □

Finally, we discuss instantiating the Fiat-Shamir transform for the protocol Π in Theorem 7 by appealing to Theorem 6. Since the round-by-round State function is fairly simple for our protocol Π (in that it does not depend on the entire protocol history), we can rely on correlation intractability for relations with a fairly simple description. By invoking Theorem 6, we obtain the following corollary.

[13] That is, V emulates an execution of D on each x_i, checking that for every j, the jth oracle call in the sequence of k executions is to $\tilde{x}'_j$; it then uses $\tilde{y}'_j$ as the oracle's output in its emulation.

Corollary 1. *Under the hypotheses of Theorem 7 and additionally assuming the existence of a hash family $\mathcal{H}$ that is correlation intractable for the following relation $\mathcal{R}$:*

$$\mathcal{R}^{(n)}_{\mathsf{State},\mathsf{NextMsg}} := \left\{ \left(\alpha = (n, \tilde{x}'_1, \tilde{y}'_1, \ldots, \tilde{x}'_{dk}, \tilde{y}'_{dk}), r\right) : \begin{array}{c} \tilde{y}'_j \neq f_{n-1}(\tilde{x}'_j) \text{ for some } j \\ \text{and} \\ \tilde{y}''_i = f_{n-1}(\tilde{x}''_i) \text{ for all } i \end{array} \right\},$$

where $\{(\tilde{x}''_i, \tilde{y}''_i)\}_{i\in[1,k]}$ is the output of B on input $\{(\tilde{x}'_j, \tilde{y}'_j)\}_{j\in[1,dk]}$ and random coins r, there is a non-interactive argument system for $\mathcal{L}^k_{f_n}$ with adaptive unambiguous soundness.

3.1 Instantiations of Outline-and-Batch

Appropriate instantiations of our outline-and-batch protocol (Theorem 7, Corollary 1) can recover the interactive proof systems or non-interactive argument systems constructed in the following works:

1. *The [44] interactive proof system for #SAT and its Fiat-Shamir instantiations [15, 39].* The sumcheck protocol can be viewed as a composition of
   (a) a $(d+1)$-query downward self-reduction that reduces a statement about the sum $\sum_{x_1,\ldots,x_n\in\{0,1\}} p(x_1, \ldots, x_n)$ of a d-degree, n-variate polynomial p (over some finite field) to $d+1$ statements of the form $\sum_{x_2,\ldots,x_n} p(\alpha, x_2, \ldots, x_n)$ (for hard-coded values of α); and
   (b) a $(d+1)$-to-1 batching reduction reducing these $(d+1)$ statements to a single statement about $\sum_{x_2,\ldots,x_n} p(r, x_2, \ldots, x_n)$ for a *uniformly random* r.
2. *The [16, 51] interactive proof system for IS over the* signed quadratic residue group $\mathbb{QR}^+_N$ *and its Fiat-Shamir instantiation in the standard model [43].* Let $x \xrightarrow{T} y$ (now) denote the statement "x^{2^T} equals y over $\mathbb{QR}^+_N$". These protocols consist of a 2-query downward self-reduction from a statement $x \xrightarrow{T} y$ to two statements of the form $x_i \xrightarrow{T/2} y_i$ and a 2-to-1 batching reduction that combines these two statements to a single such statement using a random linear combination.
3. *The [23] continuous VDF adapted to $\mathbb{QR}^+_N$.* This protocol consists of a d-query downward self-reduction from a statement $x \xrightarrow{T} y$ to d statements of the form $x_i \xrightarrow{T/d} y_i$ and a d-to-1 batching reduction from these d statements to a single such statement using, again, a random linear combination. The parameter d is set in their construction to $O(\lambda)$, for the security parameter λ.
4. *The [5] interactive proof system for IS.* In Section 4.3, we describe how this protocol fits the "outline-and-batch" framework and then show how to instantiate Fiat-Shamir for this protocol in the standard model.

4 Non-Interactive Argument for Iterated Squaring in a Trapdoor Group of Unknown Order

We first recall the iterated squaring (IS) problem modulo an integer N and discuss the hardness of IS. This includes a new hardness reduction showing that certain *public-coin* variants of IS are as hard as the "traditional" IS problem in the RSA group. Next, we consider a general IS problem over an arbitrary group of unknown order and construct our unambiguous "outline-and-batch" argument system in this setting.

4.1 Iterated Squaring modulo N

We first define IS and then recall the assumption of [53] on its sequential hardness that our VDF is based on. Our hardness assumption on IS required for **PPAD** hardness is a relaxation of this assumption.

Definition 20 ([9, 53]). IteratedSquaring *(*IS*)*

- Instance.
  1. *Integers* $N, T \in \mathbb{N}$
  2. *Group element* $g \in \mathbb{Z}_N^*$
- Solution. $f(N, g, T) := g^{2^T} \bmod N$

Assumption 8 (Sequential hardness of IS [53])**.** For a security parameter $\lambda \in \mathbb{N}$, let $_{RSA} \in \lambda^{O(1)}$ denote the size of RSA modulus that corresponds to λ bits of security. Sample $N = pq$ as the product of two random $\lambda_{RSA}/2$-bit primes and $g \leftarrow \mathbb{Z}_N^*$. Consider any time parameter $T = 2^{o(\lambda)}$. Any A that uses $2^{o(\lambda)}$ amount of parallelism and computes $f(N, g, T)$ with a probability that is non-negligible in λ requires *sequential time* $T(1 - o(1))$ group operations.

Assumption 9 (Standard hardness of IS [16])**.** For a security parameter $\lambda \in \mathbb{N}$, let N and g be sampled as in Assumption 8. There exists an efficiently computable function $T(1^\lambda)$, such that no $\lambda^{O(1)}$-time algorithm can compute $f(N, g, T)$ with a non-negligible probability.

Remark 6 (Assumption 9 vs. assumption in [16, 51]). The hardness assumption in [16, 51] is slightly different from Assumption 9. Firstly, the modulus N in [16, 51] is sampled a product of two random $\lambda_{RSA}/2$-bit *safe primes* – the statistical soundness of Pietrzak's proof-of-exponentiation (PoE) is guaranteed only in such moduli. Secondly, to attain unambiguity, [16, 51] switch to the algebraic setting of *signed quadratic residues* [26, 31]. In comparison, our assumption is made on the conventional RSA modulus and this suffices since we rely the PoE from [5] which achieves statistical soundness and (as we show) unambiguity for arbitrary groups.

4.2 Trapdoor Groups with Unknown Order

Definition 21 (Group of unknown order). *A* group sampler *for a* group of unknown order *consists of the following two functionalities:*

- *A setup algorithm* $\mathsf{Setup}(1^\lambda)$ *that samples the description of a group* $\mathbb{G}_\lambda$ *of order* at most 2^λ*. For our purposes, a group description consists of a distinguished identity element* $\mathsf{id}_{\mathbb{G}}$ *and efficient membership testing algorithm that takes as input an arbitrary string and decides whether the string is a valid element of* $\mathbb{G}_\lambda$*.*
- *Efficient* $\mathsf{poly}(\lambda)$*-time algorithms, given a description of* $\mathbb{G}_\lambda$*, for:*
  - *Sampling a uniformly random group element,*
  - *Computing the group law* $(g, h) \mapsto gh \in \mathbb{G}_\lambda$*, and*
  - *Computing the inverse map* $g \mapsto g^{-1} \in \mathbb{G}_\lambda$*.*

These efficient group operations generically imply that one can compute exponentiations $g \mapsto g^x$ in time $\mathsf{poly}(\lambda) \cdot \log(x)$ by repeated squaring. For example, this implies that $g \mapsto g^{2^T}$ can be computed in time $T \cdot \mathsf{poly}(\lambda)$ (or T group operations).

Note that if the order of $\mathbb{G}_\lambda$ is known, then the map g^{2^T} can actually be computed in time $\mathsf{poly}(\lambda, \log T)$ by first reducing 2^T modulo the order of the group. However, when the order of the group is unknown, it is plausible that this map requires time roughly T group operations, as originally proposed by [53]. We formulate two flavors of this assumption, matching Assumptions 8 and 9 in the case of RSA groups.

Assumption 10 ((T, p)-Sequential Hardness)**.** Given the description of $\mathbb{G}_\lambda$ and a random group element g, any algorithm running in sequential time $T(1 - o(1))$ with $p(\lambda)$ parallelism outputs g^{2^T} with only negligible probability.

Assumption 11 (Polynomial Hardness of Iterated Squaring)**.** There exists an efficiently computable function $T(1^\lambda)$ such that, given the description of $\mathbb{G}_\lambda$ and a random group element g, no algorithm running in time $\lambda^{O(1)}$ can output g^{2^T} with non-negligible probability.

In order to prove the unambiguous soundness of our non-interactive argument system for IS, we will make use of groups satisfying Assumption 11 that have *trapdoors* allowing for efficient iterated squaring. We formalize this by requiring that the group distribution $\mathbb{G}_\lambda$ could be sampled along with its order (using secret coins).

Definition 22 (Trapdoor group with unknown order). *A* trapdoor group with unknown order *is a group with unknown order (Definition 21) equipped with an additional setup algorithm* $\mathsf{TrapSetup}(1^\lambda)$ *that outputs the description of a group* $\mathbb{G}_\lambda$ along with *its order* M*. We require that the distribution of groups output by* $\mathsf{Setup}(1^\lambda)$ *is statistically indistinguishable from the distribution of groups output by* $\mathsf{TrapSetup}(1^\lambda)$ *(where the order information is dropped).*

RSA groups $\mathbb{Z}^\times_{pq}$ are naturally equipped with the required trapdoor structure, because if $N = pq$ is sampled as the product of two known primes, then the order of $\mathbb{Z}^\times_N$ is equal to $\phi(pq) = (p-1)(q-1)$.

4.3 Interactive Iterated Squaring Protocol

In this section, we recall the interactive proof system Π of [5] for IS and analyze the Fiat-Shamir heuristic applied to Π using an appropriate correlation-intractable hash family. Since the groups output by $\mathsf{Setup}(1^\lambda)$ and $\mathsf{TrapSetup}(1^\lambda)$ are statistically indistinguishable, we assume that $\mathsf{TrapSetup}(1^\lambda)$ is used for the purposes of both the construction and its analysis.

Let $\mathbb{G}_\lambda \leftarrow \mathsf{Setup}(1^\lambda)$ denote a group (distribution) with unknown order and associated generator g. For simplicity, we only consider T of the form $T = 2^t$.[14] For T of this form, we construct an interactive proof system for IS by having the prover invoke the "outline and batch" protocol (Theorem 7) on λ *identical computations of* g^{2^T}, i.e., $((g, \cdots, g), (g^{2^T}, \cdots, g^{2^T}))$. By Theorem 7, it suffices to show that the function $f : g \mapsto g^{2^T}$ has a 2-query downwards self reduction (Definition 18) and a 2λ-to-λ batching reduction (Definition 19).

- **2-Downwards Self-Reduction**: Given an instance of the T-IS problem $f(g, T)$, we can query $f(g, T/2)$ to obtain an intermediate group element μ, and then call $f(\mu, T/2)$ to obtain $\mu^{2^{T/2}} = g^{2^T}$.
- **2λ-to-λ Batching Reduction**: Given 2λ instances $g_1, \ldots, g_{2\lambda}$ for $f(\cdot, T)$ and 2λ candidate outputs $h_1, \ldots, h_{2\lambda}$, the batching reduction samples λ i.i.d. vectors $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_\lambda \leftarrow \{0,1\}^{2\lambda}$. The reduction then outputs λ statements about $f(\cdot, T)$:

$$\left(\prod_{j=1}^{2\lambda} g_j^{r_{1,j}} \xrightarrow{T} \prod_{j=1}^{2\lambda} h_j^{r_{1,j}}, \ldots, \prod_{j=1}^{2\lambda} g_j^{r_{\lambda,j}} \xrightarrow{T} \prod_{j=1}^{2\lambda} h_j^{r_{\lambda,j}} \right).$$

  Completeness of the batching reduction is immediate by group axioms. For soundness, suppose that $g_j^{2^T} \neq h_j$ for some j. The i-th statement output by the reduction is true if and only if

$$\prod_{j=1}^{2\lambda} (g_j^{2^T})^{r_{i,j}} = \prod_{j=1}^{2\lambda} h_j^{r_{i,j}},$$

  which is equivalent to the equation

$$\prod_{j=1}^{2\lambda} (g_j^{2^T} h_j^{-1})^{r_{i,j}} = \mathsf{id}_{\mathbb{G}_\lambda}.$$

  For $\boldsymbol{r}_i \leftarrow \{0,1\}^{2\lambda}$, $i \in [1, \lambda]$, this event occurs with probability at most $1/2$ (see [5, Fact 8.1]). Thus, at least one of the λ resulting statements is false except with probability $2^{-\lambda}$. In fact, this analysis gives a product set description for the "bad challenges" of the batching reduction. For a fixed $\alpha =$

[14] A protocol for general T can be obtained by dividing T by computing a binary decomposition of the resulting integer, and sequentially composing squaring protocols for integers of the form 2^t.

$((g_1, h_1), \ldots, (g_{2\lambda}, h_{2\lambda}))$, the bad set $\mathcal{R}_\alpha = \mathcal{B}_\alpha^{(1)} \times \ldots \times \mathcal{B}_\alpha^{(\lambda)} \subset (\{0,1\}^{2\lambda})^\lambda$, where

$$\mathcal{B}_\alpha^{(i)} = \left\{ \boldsymbol{r} \in \{0,1\}^{2\lambda} : \prod_{j=1}^{2\lambda} (g_j^{2^T})^{r_j} = \prod_{j=1}^{2\lambda} h_j^{r_j} \right\}$$

(in fact, we have that each $\mathcal{B}_\alpha^{(i)} = \mathcal{B}_\alpha$ for a fixed set $\mathcal{B}_\alpha$). As mentioned above, we have that $|\mathcal{B}_\alpha^{(i)}| \leq 2^{2\lambda}/2$ for every j and every false α. Thus, the bad-challenge relation $\mathcal{R}$ is a product relation with the appropriate sparsity, where $\mathcal{R}$ is defined as the set of pairs $\big(\alpha, \beta = (\boldsymbol{r}_1, \ldots, \boldsymbol{r}_\lambda)\big)$ for which at least one of the 2λ statements defined by α is false but all of the λ statements output by the reduction are true.

Finally, we observe that for $(\mathbb{G}_\lambda, M) \leftarrow \mathsf{TrapSetup}(1^\lambda)$, the relation $\mathcal{R}$ is also *efficiently product verifiable*: to verify that $v \in \mathcal{B}_\alpha^{(i)}$, it suffices to check the equation

$$\prod_{j=1}^{2\lambda} (g_j^{2^T})^{r_j} = \prod_{j=1}^{2\lambda} h_j^{r_j}.$$

This can be checked in time $\mathsf{poly}(\lambda, \log T)$ given the order M of $\mathbb{G}_\lambda$, by first computing 2^T modulo M and then checking the equation above using the group law and repeated squaring.

Thus, by Theorem 7 we conclude that there is a $t = \log(T)$-round unambiguous interactive proof system for the T-IS problem with $\mathsf{poly}(\lambda)$ communication. Moreover, by Corollary 1 this protocol can be round-collapsed to a computationally unambiguous non-interactive argument system using a hash function family that is correlation-intractable for the relation R above (where we consider T as part of the input to the relation). Finally, by Theorem 5, such hash functions can be built under the learning with errors assumption. This is captured by the following corollary.

Corollary 2. *For a security parameter $\lambda \in \mathbb{N}$, let $\mathbb{G}_\lambda$ be a trapdoor group of unknown defined in Definition 22. Assuming polynomial hardness of* LWE *(Assumption 4), $\Pi_{\mathrm{FS},\mathcal{H}}$ is an adaptively unambiguously-sound non-interactive argument for the language*

$$\mathcal{L}^\lambda_{\mathbb{G}_\lambda} := \{((g_1, \cdots, g_\lambda), (h_1, \cdots, h_\lambda), T) \in \mathbb{G}_\lambda^\lambda \times \mathbb{G}_\lambda^\lambda \times \mathbb{N} : h_i = g^{2^T} \text{ for all } i \in [1, \lambda]\}.$$

5 PPAD Hardness

In this section, we construct a hard distribution of RSVL from any hard f that is downward self-reducible and batch-reducible, additionally assuming the unambiguous soundness of $\Pi_{\mathrm{FS},\mathcal{H}}$, the non-interactive "outline-and-batch" argument system for $\mathcal{L}^k_{f_n}$ (Corollary 1). By Theorem 3, this implies hardness of EOML, which is complete for **CLS**; since **CLS** $\subseteq$ **PPAD**, **PPAD**-hardness follows.

Our construction follow the blueprint from [15, 16]. Further, since our construction works with *any* f that is downward self-reducible and batch-reducible, it generalises the constructions of RSVL instance in [15, 16] and the continuous VDF in [23]. Indeed, as we saw in Section 3, both iterated squaring and the sum-check problem satisfy downward self-reducibility and batch-reducibility. Due to a lack of space, we only state below the relevant theorems and refer the reader to the full version of the paper for the details.

Assumption 12 (Hardness of f)**.** Let $f : \mathcal{X} \to \mathcal{Y}$ be a function as defined in Section 3 and let X denote a sampler for $\mathcal{X}$. The function f is $(s(\lambda), \epsilon(\lambda))$-hard with respect to X if for every $s(\lambda)$-sized adversary $\mathsf{A} = \{\mathsf{A}_\lambda\}_{\lambda\in\mathbb{N}}$

$$\Pr_{\substack{x\leftarrow\mathsf{X}(1^\lambda)\\ y\leftarrow\mathsf{A}(x)}}[y = f(x)] = O(\epsilon(\lambda)).$$

Theorem 13 (Hardness of RSVL from f and $\Pi_{\mathrm{FS},\mathcal{H}}$). *Let $k, d \in \mathbb{N}$ be parameters and $\lambda \in \mathbb{N}$ be a security parameter. Let*

- *$f : \mathcal{X} \to \mathcal{Y}$ be a d-query downwards self-reducible and dk-to-k batch-reducible function with sampler X; and*
- *$\Pi_{\mathrm{FS},\mathcal{H}} = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ denote the non-interactive outline-and-batch protocol for $\mathcal{L}^k_{f_n}$ from Corollary 1.*

Furthermore, for $H \leftarrow \Pi_{\mathrm{FS},\mathcal{H}}.\mathsf{Setup}(1^\lambda)$ and $x \leftarrow \mathsf{X}(1^\lambda)$, with $n := |x|$, define

$$m = m(d, k, |x|) \in \mathsf{poly}(d, k, |x|) \text{ and } L = L(d, k) := (d + 1)^n, \tag{8}$$

there exists

$$\mathsf{S} : \{0,1\}^m \to \{0,1\}^m \text{ and } \mathsf{V} : \{0,1\}^m \times [0, 2^m - 1] \to \{\mathsf{accept}, \mathsf{reject}\}, \tag{9}$$

hardwired with $(f, H, \mathsf{D}, \widetilde{\mathsf{B}}, x, \Pi_{\mathrm{FS},\mathcal{H}}.\mathsf{V})$. Such that if f is hard with respect to X and $\Pi_{\mathrm{FS},\mathcal{H}}$ is (adaptively) unambiguously sound argument, then $\mathsf{RSVL} := (\mathsf{S}, \mathsf{V}, L, \boldsymbol{s}_{0^n})$ constitutes a hard distribution of RSVL.

On instantiating f with IS as sampled in Assumption 9 and $\Pi_{\mathrm{FS},\mathcal{H}}$ with non-interactive argument from Corollary 2, we get the following corollary to Theorem 13.

Corollary 3 (Hardness of RSVL from IS and LWE). *For a security parameter $\lambda \in \mathbb{N}$, let $(\mathbb{G}_\lambda, g, T)$ be sampled as in Assumption 11, which defines $f_n(g, T) := g^{2^T}$ for $n := \log(T)$. Also, let $\Pi_{\mathrm{FS},\mathcal{H}} = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ denote the non-interactive protocol for $\mathcal{L}^k_{\mathbb{G}_\lambda}$ from Corollary 2, which implies $k \in \lambda^{O(1)}$ and $d = 2$. Furthermore, for $H \leftarrow \Pi_{\mathrm{FS},\mathcal{H}}.\mathsf{Setup}(1^\lambda)$, define*

$$m = m(n, k, \lambda) := n^2 k \cdot \mathsf{poly}(\lambda) \text{ and } L = L(n) = 3^n, \tag{10}$$

there exists

$$\mathsf{S} : \{0,1\}^m \to \{0,1\}^m \text{ and } \mathsf{V} : \{0,1\}^m \times [0, 2^m - 1] \to \{\mathsf{accept}, \mathsf{reject}\}, \tag{11}$$

hardwired with $((\mathbb{G}_\lambda, g, T), H, \mathsf{D}, \widetilde{\mathsf{B}}, \Pi_{\mathrm{FS},\mathcal{H}}.\mathsf{V})$. Such that if Assumption 11 and Assumption 4 hold then $\mathsf{RSVL} := (\mathsf{S}, \mathsf{V}, L, \boldsymbol{s}_{0^n})$ constitutes a hard distribution of RSVL.

6 Conclusion and Open Problems

In this work, we demonstrated hardness in the class **PPAD** assuming the polynomial hardness of iterated squaring and LWE. Moreover, in the full version of this paper, we (1) strengthened this result to show hardness in **UEOPL** $\subseteq$ **PPAD** (which is first cryptographic hardness shown for that class) and (2) constructed a unique VDF based on similar assumptions.

We briefly mention two interesting open questions that are closely related to this work:

- Can the iterated squaring hardness assumption be replaced by a weaker assumption such as the hardness of factoring? This seems plausible since to achieve **PPAD** hardness, it suffices for iterated squaring to be polynomially hard for *some* efficiently computable iteration parameter. This question was also posed in [16].
- Can we show **PPAD**-hardness solely from polynomial hardness of LWE, and thus establish a (polynomially) tight hardness result for quantum algorithms? Currently, only [39] demonstrates post-quantum hardness of **PPAD** (under sub-exponential LWE).

Acknowledgements. Nir Bitansky is a member of the checkpoint institute of information security and is supported by the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (grant agreement No. 101042417, acronym SPP), and by Len Blavatnik and the Blavatnik Family Foundation.

Arka Rai Choudhuri is supported in part by DARPA under Agreement No. HR00112020026, AFOSR Award FA9550-19-1-0200, NSF CNS Award 1936826, and research grants by the Sloan Foundation, and Visa Inc. 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 United States Government or DARPA.

Chethan Kamath is supported by Azrieli International Postdoctoral Fellowship and ISF grants 484/18 and 1789/19. He thanks Alexandros Hollender and Ninad Rajagopal for discussions on the class **UEOPL** and Krzysztof Pietrzak for clarifications about unique VDFs.

Alex Lombardi is supported in part by DARPA under Agreement No. HR00112020023, a grant from MIT-IBM Watson AI, a grant from Analog Devices, a Microsoft Trustworthy AI grant, the Thornton Family Faculty Research Innovation Fellowship and a Charles M. Vest fellowship. 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 United States Government or DARPA.

Omer Paneth is a member of the checkpoint institute of information security and is supported by an Azrieli Faculty Fellowship, Len Blavatnik and the Blavatnik Foundation, the Blavatnik Interdisciplinary Cyber Research Center at Tel Aviv University, and ISF grant 1789/19.

Ron Rothblum was funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

References

1. Arora, S., Barak, B.: Computational Complexity - A Modern Approach. Cambridge University Press (2009) 12
2. Bach, E.: How to generate factored random numbers. SIAM Journal on Computing **17**(2), 179–193 (1988) 11
3. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_1 2
4. Bitansky, N., Paneth, O., Rosen, A.: On the cryptographic hardness of finding a Nash equilibrium. In: Guruswami, V. (ed.) 56th FOCS. pp. 1480–1498. IEEE Computer Society Press (Oct 2015). https://doi.org/10.1109/FOCS.2015.94 2, 3
5. Block, A.R., Holmgren, J., Rosen, A., Rothblum, R.D., Soni, P.: Time- and space-efficient arguments from groups of unknown order. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 123–152. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_5 6, 7, 8, 9, 10, 20, 21, 23
6. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 757–788. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96884-1_25 4
7. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_16 2
8. Brakerski, Z., Koppula, V., Mour, T.: NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 738–767. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_26 5
9. Cai, J.Y., Lipton, R.J., Sedgewick, R., Yao, A.C.: Towards uncheatable benchmarks. In: [1993] Proceedings of the Eigth Annual Structure in Complexity Theory Conference. pp. 2–11 (May 1993). https://doi.org/10.1109/SCT.1993.336546 21
10. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D.: Fiat-Shamir from simpler assumptions. Cryptology ePrint Archive, Report 2018/1004 (2018), https://eprint.iacr.org/2018/1004 9, 17
11. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-Shamir: from practice to theory. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC. pp. 1082–1090. ACM Press (Jun 2019). https://doi.org/10.1145/3313276.3316380 5, 14, 17
12. Canetti, R., Chen, Y., Reyzin, L., Rothblum, R.D.: Fiat-Shamir and correlation intractability from strong KDM-secure encryption. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 91–122. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9_4 5

13. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: 30th ACM STOC. pp. 209–218. ACM Press (May 1998). https://doi.org/10.1145/276698.276741 8, 14
14. Chen, X., Deng, X., Teng, S.H.: Settling the complexity of computing two-player nash equilibria. Journal of the ACM (JACM) **56**(3), 1–57 (2009) 2
15. Choudhuri, A.R., Hubácek, P., Kamath, C., Pietrzak, K., Rosen, A., Rothblum, G.N.: Finding a nash equilibrium is no easier than breaking Fiat-Shamir. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC. pp. 1103–1114. ACM Press (Jun 2019). https://doi.org/10.1145/3313276.3316400 2, 4, 10, 13, 16, 17, 20, 25
16. Choudhuri, A.R., Hubacek, P., Kamath, C., Pietrzak, K., Rosen, A., Rothblum, G.N.: PPAD-hardness via iterated squaring modulo a composite. Cryptology ePrint Archive, Report 2019/667 (2019), https://eprint.iacr.org/2019/667 2, 4, 5, 20, 21, 25, 26
17. Choudhuri, A.R., Jain, A., Jin, Z.: Non-interactive batch arguments for NP from standard assumptions. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part IV. LNCS, vol. 12828, pp. 394–423. Springer, Heidelberg, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84259-8_14 5
18. Choudhuri, A.R., Jain, A., Jin, Z.: SNARGs for **P** from LWE. In: FOCS. pp. 68–79. IEEE (2021) 5
19. Ciampi, M., Parisella, R., Venturi, D.: On adaptive security of delayed-input sigma protocols and fiat-shamir NIZKs. In: Galdi, C., Kolesnikov, V. (eds.) SCN 20. LNCS, vol. 12238, pp. 670–690. Springer, Heidelberg (Sep 2020). https://doi.org/10.1007/978-3-030-57990-6_33 5
20. Couteau, G., Katsumata, S., Ursu, B.: Non-interactive zero-knowledge in pairing-free groups from weaker assumptions. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 442–471. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45727-3_15 5
21. Daskalakis, C., Goldberg, P.W., Papadimitriou, C.H.: The complexity of computing a nash equilibrium. SIAM Journal on Computing **39**(1), 195–259 (2009) 2
22. Daskalakis, C., Papadimitriou, C.H.: Continuous local search. In: Randall, D. (ed.) 22nd SODA. pp. 790–804. ACM-SIAM (Jan 2011). https://doi.org/10.1137/1.9781611973082.62 3
23. Ephraim, N., Freitag, C., Komargodski, I., Pass, R.: Continuous verifiable delay functions. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 125–154. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45727-3_5 2, 4, 5, 20, 25
24. Fearnley, J., Gordon, S., Mehta, R., Savani, R.: Unique end of potential line. In: Baier, C., Chatzigiannakis, I., Flocchini, P., Leonardi, S. (eds.) ICALP 2019. LIPIcs, vol. 132, pp. 56:1–56:15. Schloss Dagstuhl (Jul 2019). https://doi.org/10.4230/LIPIcs.ICALP.2019.56 3, 10
25. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12 5
26. Fischlin, R., Schnorr, C.P.: Stronger security proofs for RSA and Rabin bits. Journal of Cryptology **13**(2), 221–244 (Mar 2000). https://doi.org/10.1007/s001459910008 7, 21
27. Freitag, C., Pass, R., Sirkin, N.: Parallelizable delegation from LWE. Cryptology ePrint Archive, Report 2022/1025 (2022), https://eprint.iacr.org/2022/1025 4, 6

28. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS. pp. 40–49. IEEE Computer Society Press (Oct 2013). https://doi.org/10.1109/FOCS.2013.13 2
29. Garg, S., Pandey, O., Srinivasan, A.: Revisiting the cryptographic hardness of finding a nash equilibrium. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 579–604. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53008-5_20 2, 3
30. Hoffmann, C., Hubáček, P., Kamath, C., Klein, K., Pietrzak, K.: Practical statistically-sound proofs of exponentiation in any group. Cryptology ePrint Archive, Report 2022/1021 (2022), https://eprint.iacr.org/2022/1021 6
31. Hofheinz, D., Kiltz, E.: The group of signed quadratic residues and applications. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 637–653. Springer, Heidelberg (Aug 2009). https://doi.org/10.1007/978-3-642-03356-8_37 7, 21
32. Holmgren, J., Lombardi, A.: Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In: Thorup, M. (ed.) 59th FOCS. pp. 850–858. IEEE Computer Society Press (Oct 2018). https://doi.org/10.1109/FOCS.2018.00085 5
33. Holmgren, J., Lombardi, A., Rothblum, R.D.: Fiat-shamir via list-recoverable codes (or: parallel repetition of GMW is not zero-knowledge). In: STOC. pp. 750–760. ACM (2021) 5, 6, 7, 8, 10, 14, 15
34. Hubácek, P., Yogev, E.: Hardness of continuous local search: Query complexity and cryptographic lower bounds. In: Klein, P.N. (ed.) 28th SODA. pp. 1352–1371. ACM-SIAM (Jan 2017). https://doi.org/10.1137/1.9781611974782.88 2, 3
35. Hulett, J., Jawale, R., Khurana, D., Srinivasan, A.: SNARGs for P from sub-exponential DDH and QR. In: EUROCRYPT 2022, Part II. pp. 520–549. LNCS, Springer, Heidelberg (Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_18 5
36. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over F_p, DLIN, and PRGs in NC^0. Cryptology ePrint Archive, Report 2021/1334 (2021), https://eprint.iacr.org/2021/1334 3
37. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing. pp. 60–73 (2021) 3
38. Jain, A., Jin, Z.: Non-interactive zero knowledge from sub-exponential DDH. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 3–32. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_1 5
39. Jawale, R., Kalai, Y.T., Khurana, D., Zhang, R.Y.: Snargs for bounded depth computations and PPAD hardness from sub-exponential LWE. In: STOC. pp. 708–721. ACM (2021) 2, 4, 5, 20, 26
40. Kalai, A.: Generating random factored numbers, easily. Journal of Cryptology **16**(4), 287–289 (Sep 2003). https://doi.org/10.1007/s00145-003-0051-5 11
41. Kalai, Y.T., Paneth, O., Yang, L.: Delegation with updatable unambiguous proofs and PPAD-hardness. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 652–673. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_23 2, 4
42. Kalai, Y.T., Rothblum, G.N., Rothblum, R.D.: From obfuscation to the security of Fiat-Shamir for proofs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 224–251. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63715-0_8 5

43. Lombardi, A., Vaikuntanathan, V.: Fiat-shamir for repeated squaring with applications to PPAD-hardness and VDFs. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 632–651. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_22 2, 4, 5, 6, 9, 13, 14, 15, 17, 20
44. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. In: 31st FOCS. pp. 2–10. IEEE Computer Society Press (Oct 1990). https://doi.org/10.1109/FSCS.1990.89518 19, 20
45. Megiddo, N., Papadimitriou, C.H.: On total functions, existence theorems and computational complexity. Theoretical Computer Science **81**(2), 317–324 (1991) 2
46. Nash, J.: Non-cooperative games. Annals of mathematics pp. 286–295 (1951) 1
47. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010), https://eprint.iacr.org/2010/556 2
48. Papadimitriou, C.H.: On the complexity of the parity argument and other inefficient proofs of existence. J. Comput. Syst. Sci. **48**(3), 498–532 (1994) 2
49. Peikert, C.: A decade of lattice cryptography. Foundations and Trends® in Theoretical Computer Science **10**(4), 283–424 (2016) 13
50. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_4 5, 14
51. Pietrzak, K.: Simple verifiable delay functions. In: Blum, A. (ed.) ITCS 2019. vol. 124, pp. 60:1–60:15. LIPIcs (Jan 2019). https://doi.org/10.4230/LIPIcs.ITCS.2019.60 5, 6, 9, 20, 21
52. Reingold, O., Rothblum, G.N., Rothblum, R.D.: Constant-round interactive proofs for delegating computation. In: Wichs, D., Mansour, Y. (eds.) 48th ACM STOC. pp. 49–62. ACM Press (Jun 2016). https://doi.org/10.1145/2897518.2897652 6, 15, 16
53. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Tech. rep., Cambridge, MA, USA (1996) 4, 21, 22