Non-interactive Universal Arguments*

Nir Bitansky Omer Paneth Dana Shamir Tomer Solomon

December 2022

Abstract

In 2002, Barak and Goldreich introduced the notion of a *universal argument* and constructed an interactive universal argument for non-deterministic computations based on polynomially hard collision-resistant hash functions. Since then, and especially in recent years, there have been tremendous developments in the construction of *non-interactive* succinct arguments for deterministic computations under standard hardness assumptions. However, the constructed succinct arguments can be proven universal only under *sub-exponential* assumptions.

Assuming *polynomially hard* fully homomorphic encryption and a widely believed worst-case complexity assumption, we prove a general lifting theorem showing that all existing non-interactive succinct arguments can be made universal. The required complexity assumption is that non-uniformity does not allow arbitrary polynomial speedup. In the setting of uniform adversaries, this extra assumption is not needed.

*Tel Aviv University.Email: {nirbitan, omerpa, danashamir1, tomersolomon}@mail.tau.ac.il .

Nir Bitansky and Omer Paneth are members of the checkpoint institute of information security. Nir Bitansky, Dana Shamir, and Tomer Solomon are supported by the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (grant agreement No. 101042417, acronym SPP).

1 Introduction

A succinct argument for a language $\mathcal{L}$ (in **P** or **NP**) allows proving membership in $\mathcal{L}$ so that verification is only polylogarithmic in the time T needed to decide $\mathcal{L}$ (deterministically or non-deterministically). Since the pioneering works of Kilian [Kil92] and Micali [Mic94], succinct arguments have become a central concept in cryptography, with far reaching applications. As such, different notions of succinct arguments have been (and are still being) studied.

One influential notion in this context is that of *universal arguments*, put forth by Barak and Goldreich [BG08]. A universal argument allows proving membership *in any* language in **NP**. More generally, such an argument system enables proving statements $y = (M, x, T)$ in the *universal language* $\mathcal{L}_{\mathcal{U}}$ attesting that *"M non-deterministically accepts x in time T"*. Unlike a succinct argument for a *specific* language $\mathcal{L} \in$ **NP**, here T is not bounded by any specific polynomial and may be, in fact, as large as 2^{λ}, where λ is a security parameter. The prover complexity is accordingly polynomial in T. Soundness of universal arguments is guaranteed against provers that are polynomially bounded in the security parameter λ. Analogously to universal arguments for non-deterministic computations, one may consider a universal argument for deterministic computations, which will be the focus of this work.

The need for universal arguments arises in different scenarios where we are required to verify the correctness of arbitrary computations and do not have a guarantee on the time it will take for them to terminate. In some scenarios, even the honest prover may not know apriori when the computation will terminate. Such is the scenario of *incrementally verifiable computation* [Val08], where the prover gradually computes and updates its proofs as it progresses along the computation. Another salient use case, expressed in the work of Barak and Goldreich, is *diagonalization*. Following the work of Barak [Bar01], universal arguments have been repeatedly used in non-black-box proofs of security in order to prove assertions pertaining to the code of the adversary, whose running time is not known in advance (c.f. [BG08, PR08, DGS09, BKP18]).

Universal Arguments Based on Polynomial Assumptions. The challenge in constructing universal arguments is in basing them on standard *polynomial* assumptions. Indeed, existing non-universal succinct arguments, deterministic and non-deterministic alike, can be made universal by relying on *sub-exponential* assumptions, or assuming super-polynomial security $\ell(\lambda) = \lambda^{-\omega(1)}$, and restricting the universal language to computations of time $T \leq \ell(\lambda)$. Here the complexity of the security reduction does not scale polynomially in that of the adversary as we usually aim for, but rather scales with the complexity of computations that the adversary asserts about, which may be super-polynomially larger. In the regime of polynomial assumptions, this yields a weaker notion that we call *semi-universal arguments* where soundness is only guaranteed when the computation time T is polynomially bounded.

In the interactive setting, Barak and Goldreich overcome this challenge. Assuming *polynomially secure* collision-resistant hash functions, they prove the existence of a four-message argument for non-deterministic computations that is *fully universal* (namely, sound for super-polynomial computation time $T \leq 2^{\lambda}$).

Non-interactive Universal Arguments. In this work, we aim to base *non-interactive* universal arguments on polynomial assumptions. In the non-interactive setting, the verifier generates a short public key (sometime referred to as a common reference string) once and for all, and this public key can be then used to generate the proofs non-interactively. Faced with known barriers on non-interactive succinct arguments [GW11] and focusing on standard assumptions, we follow the common restriction to *deterministic* computations. Here non-interactive succinct arguments have tremendously developed over recent years and can now be based on standard polynomial assumptions, with strong features like public verification as well and incremental proof updatability (c.f. [KPY19, CJJ21b, PP22, DGKV22]). While these recent constructions can too be made universal under sub-exponential assumptions, or semi universal under polynomial assumptions, non-interactive (fully) universal arguments, based on polynomial assumptions, remain out of reach.

1.1 Results

Our main result is a general lifting theorem from semi universal arguments to universal arguments, based on polynomial security assumptions.

Theorem 1.1 (Informal)**.** *Assuming polynomially-secure fully-homomorphic encryption, any non-interactive semi-universal argument can be lifted to a universal argument. The resulting universal argument is secure against polynomial-time uniform adversaries. Assuming that for any $c \in \mathbb{N}$, $\mathbf{P} \not\subset \mathbf{ioSIZE}(n^c)$, it is secure against polynomial-size non-uniform adversaries.*

The theorem holds both in the setting of public verification (maintaining public verifiability) and of private verification. It also holds both for deterministic and non-deterministic computations, as well as interactive arguments, without increasing round complexity. We shall keep our focus on the setting of non-interactive publicly-verifiable arguments for deterministic computations, where new results under standard assumptions are obtained. For instance, relying on recent constructions of SNARGs for $\mathbf{P}$ from polynomial assumptions [CJJ21b, WW22, KLVW22], we obtain the following corollary.

Corollary 1.2 (of Theorem 1.1)**.** *Assuming fully homomorphic encryption and either LWE or DLIN, there exists a universal argument for deterministic computations. (Non-uniform security additionally requires the same complexity assumption as in the theorem.)*

The assumption that for all $c \in \mathbb{N}$, $\mathbf{P} \not\subset \mathbf{ioSIZE}(n^c)$, required in the non-uniform setting, essentially says that circuits of a fixed polynomial size n^c, cannot decide all of $\mathbf{P}$ in the worst case. (Formally, there is a language $\mathcal{L}_c$ in $\mathbf{P}$, which they fail to decide in the worst-case for all large enough instances.) This can be viewed as a natural generalization of *the time hierarchy theorem,* which holds unconditionally. It further follows from widely believed worst-case assumptions such as *the non-uniform exponential time hypothesis* (**nuETH**). In fact, relying on this assumption has a certain win-win flavour. Morally, if the assumption does not hold, then any language in $\mathbf{P}$ can already be verified in fixed polynomial time, making succinct arguments somewhat obsolete. (This connection is not precise because verification is guaranteed infinitely often and requires non-uniform advice, similarly to other constructions in the literature that use the adversary's circuit [KY18, RV22].)

Universal Incrementally Verifiable Computation. We prove an analogous lifting theorem for the case of incrementally verifiable computation. Here we need an extra complexity assumption in both the uniform and non-uniform settings, for simplicity we focus on the latter.

Theorem 1.3 (Informal)**.** *Assuming polynomially-secure fully-homomorphic encryption and that there exists $d \in \mathbb{N}$ such that for any $c \in \mathbb{N}$, $\mathbf{P} \cap \mathbf{DSPACE}(n^d) \not\subset \mathbf{ioSIZE}(n^c)$, any semi-universal argument can be lifted to an incrementally verifiable universal argument. (The universal argument is secure against polynomial-size non-uniform adversaries.)*

Relying on recent constructions of incrementally verifiable arguments from polynomial assumptions [PP22, DGKV22], we obtain the following corollary.

Corollary 1.4 (of Theorem 1.3)**.** *Assuming fully homomorphic encryption, LWE, and the same complexity assumption as in the theorem, there exists an incrementally verifiable universal argument.*

The complexity assumption required here is stronger (leading to a stronger end result), however, we still view it as rather mild; in particular it is still a worst-case assumption and follows from **nuETH**.

A General Approach Based on Cryptographic Puzzles. Behind our lifting theorems is a general approach based on certain average-case cryptographic puzzles. The statements above reflect standard assumptions from which we manage to construct these puzzles. There are in fact other conceivable ways to construct the corresponding puzzles, as well as potential for future constructions under different assumptions. This is further discussed in the technical overview in the next section. We note that in [BS23], a stronger form of cryptographic puzzles (based on indistinguishability obfuscation) is used to avoid super-polynomial assumptions in the context of fully-homomorphic encryption constructions. We compare the approaches at the end of the technical overview.

1.2 Technical Overview

We start with an overview of our main lifting theorem from semi-universal arguments to universal ones. We shall focus on the case of deterministic computations and public verification.

Recall that our starting point is a semi-universal argument for the language

$$\mathcal{L}_{\mathcal{U}} = \{(M, x, T) : M(x) \text{ deterministically accepts in time } T\} \quad .$$

In such an argument, it is possible to generate public verification and proving keys $(\mathsf{vk}, \mathsf{pk})$, corresponding to a given security parameter λ. The prover can use pk to generate a proof for statement $y = (M, x, T) \in \mathcal{L}_{\mathcal{U}}$ in time $\mathrm{poly}(T)$, which the verifier can verify in time $\mathrm{polylog}(T)$.[1] Soundness is guaranteed against efficient provers that only ever cheat on statements $y = (M, x, T) \notin \mathcal{L}_{\mathcal{U}}$ such that T is bounded by some polynomial in the security parameter λ. In contrast, for statements where T is super-polynomial, soundness is no longer guaranteed, even though the prover itself is efficient.

Put Your Money Where Your Mouth Is. At a high level, our approach toward lifting is natural: if the prover wishes to convince the verifier that $(M, x, T) \in \mathcal{L}_{\mathcal{U}}$, then it should:

a. Provide a semi-universal argument for this fact, and

b. **Prove that it actually performed a T-long computation.**

The honest prover that anyhow runs in time $\mathrm{poly}(T)$ should have no trouble doing so. In contrast, a malicious prover that runs in time $\mathrm{poly}(\lambda)$, should now only be able to prove that it performed a computation of length $T \leq \mathrm{poly}(\lambda)$, in which case the the soundness of the semi-universal argument kicks in.

Cryptographic Puzzles. Realizing the above idea requires an appropriate notion of cryptographic puzzles. For such puzzles, a mildly hard instance can be generated fast, but cannot be solved without the investment of some prescribed amount of resources. Starting from the seminal work of Dwork and Naor [DN92], different notions of cryptographic puzzles have been studied in the literature, with different interpretations of the above requirements. For our purpose, we need a rather weak form of puzzles satisfying the following:

- **Fast Sampling:** Given a difficulty parameter T and security parameter λ, it is possible to sample a puzzle Z in time $\mathrm{poly}(\lambda, \log T)$.
- **Completeness:** The puzzle Z can be solved in time $\approx T$.
- **Fast Verification:** A solution to the puzzle Z, can be verified in time $\mathrm{poly}(\lambda, \log T)$.
- **Soundness:** An adversary running in time t, should fail to solve puzzles Z of difficulty $T \gg t^C$ for some constant $C > 1$. In fact, we rely on a relaxed soundness requirement, where the above holds only for polynomially bounded T (analogously to soundness of semi-universal arguments).

Given such puzzles, we realize the previously described strategy as follows. On top of the verification and proving keys $(\mathsf{vk}, \mathsf{pk})$ for the semi-universal argument, we add puzzles $Z_1, \ldots, Z_\lambda$, where Z_i is generated with difficulty $T_i = 2^i$. To prove a statement of the form $(M, x, T) \in \mathcal{L}_{\mathcal{U}}$, with the sole restriction that $T \leq 2^\lambda$, the prover must provide solutions for all puzzles $Z_1, \ldots, Z_{\log T}$ (in addition to the semi-universal argument).

The completeness of puzzles guarantees that the honest prover can generate a proof in time polynomial in T. As for soundness, a cheating adversary of running time $t = \mathrm{poly}(\lambda)$ that cheats on statements $(M, x, T) \notin \mathcal{L}_{\mathcal{U}}$ must violate the soundness of either the semi-universal argument or the underlying puzzles:

- If $T = O(t^C)$, the soundness of the semi-universal arguments is violated.
- If $T \gg t^C$, the soundness of the puzzle $Z_{\log t^C}$ is broken. (The adversary also solves allegedly more difficult puzzles like $Z_{\log T}$, but if T is super-polynomial, this does not constitute an attack according to our relaxed definition.)

[1] Formally, these polynomials also depend on λ, we suppress this dependence to simplify exposition.

Puzzles from FHE and Semi-Universal Arguments. There are several conceivable ways to construct puzzles satisfying the properties that we need. One simple approach is to require the inversion of a one-way function with domain of size $\approx T$ for the difficulty parameter T. The downside is that this requires assuming exponential hardness of the one-way function. Another approach from [BGJ+16] is to combine worst-case mild hardness, for instance based on appropriate hierarchy theorems, together with succinct randomized encodings [BCG+18], which can be obtained assuming indistinguishability obfuscation or (polynomially secure) functional encryption [AL18, GS18, KNTY19]. In our setting, resorting to succinct randomized encodings seems to be an overkill. Indeed the puzzles in [BCG+18] are also required to enable sampling of solved puzzles, which we do not need.

We provide relatively simple constructions of the required puzzles by combining hierarchy theorems, fully homomorphic encryption, and semi-universal arguments. In a nutshell, our approach is the following: Start from puzzles that are only hard in the worst case, based on appropriate hierarchy theorems. Then, lift their hardness to the average case using fully-homomorphic encryption (alla [CKV10]). Finally, make the puzzles verifiable using semi-universal arguments. We next explain these steps in more detail, addressing the different subtleties that arise in the process.

Hierarchy Theorems and Worst-Case Puzzles. Generally speaking, hierarchy theorems are statements of the form *there exist languages that can be decided in the worst case with a certain amount T of resources, but not with significantly smaller amount $t \ll T$.* Basic examples of such theorems are unconditional hierarchy theorems for (uniform) time and space and (non-uniform) circuit size. The specific type of hierarchy theorem needed to carry out our approach depends on whether we consider uniform or non-uniform adversaries. For the first, we can rely on an unconditional hierarchy theorem for probabilistic time by Barak [Bar02], whereas for the latter we need the complexity assumption, mentioned earlier, that for any $c \in \mathbb{N}$, $\mathbf{P} \not\subset \mathbf{ioSIZE}(n^c)$.

In this overview, we will focus on the non-uniform case, which is simpler, yet conveys the main ideas. Here the previously mentioned complexity assumption directly yields worst case puzzles $Z = (M, x, T)$, where T is the difficulty parameter and the solution is $M_T(x)$, the result of running M on x for T steps. The complexity assumption exactly says that for any polynomial $t = \lambda^c$, there exists a language $\mathcal{L}$ and polynomial $T = t^C$, such that $\mathcal{L}$ can be decided in the worst-case by a T-time Turing machine M, but not by circuits of size t (for all large enough inputs).

Average-Case Hardness from FHE. To turn worst-case hardness to average-case hardness, puzzles will now include encrypted pairs (M, x), namely $Z = (\mathsf{Enc}_{\mathsf{pk}}(M, x), T)$, and the solution is $\widehat{\mathsf{Enc}}_{\mathsf{pk}}(M_T(x)) = \mathsf{Eval}(U_T, \mathsf{Enc}_{\mathsf{pk}}(M, x))$, the result of homomorphically computing the universal circuit U_T that emulates T steps of the underlying computation. The actual puzzle sampler will in fact sample encryptions of some arbitrary pair (M, x), say the all-zero string. The basic idea, inspired by [CKV10], is that if an adversary can solve the puzzles on average, namely compute $\mathsf{Eval}(U_T, \cdot)$, on zero-encryptions, then it also does so on encryptions of *any* pair (M, x). This gives a worst case to average case reduction, namely we can use the average-case adversary to obtain a worst-case adversary of roughly the same size.

A subtlety with the above argument is that the adversary might only solve the average case puzzle with some noticeable probability δ, say $1/\lambda$, rather than with probability ≈ 1. The reduction, however, should solve the underlying worst-case *decision* problem with probability noticeably larger than $1/2$. Accordingly, the reduction has to make $\approx \delta^{-1}$ attempts to solve the average case puzzle to make sure it succeeds. The problem is that the reduction cannot test in which of the attempts the adversary actually succeeds, namely computes $\mathsf{Eval}(U_T, \cdot)$, as testing this requires time $\approx T$ rather than $t \ll T$.[2] For exactly the same reason, currently verification of solutions is not fast, but rather requires time $\approx T$.

Adding Semi-Universal Arguments. We remedy both issues raised in the last paragraph in one-shot. We require as part of the solution a semi-universal argument that the claimed result (ciphertext) is actually the result of applying $\mathsf{Eval}(U_T, \cdot)$ to the ciphertext given by the puzzle Z. Relying on the soundness of the universal arguments, we now get fast verification. At the same time, the worst case to average case reduction can now also rely on fast verification to check which of the repeated attempts succeeds. Since

[2] This difficulty does not arise in [CKV10], who use this technique to construct delegation with preprocessing, where the reduction could run for as long as the preprocessing time T.

we use semi-universal arguments, the puzzles are only sound provided a polynomial bound on T, which as argued before is sufficient for our purpose.

Under the Hood. The reductions should deal with additional details, such as accounting for the overhead of encryption, the repeated solving attempts, and verification of the semi-universal arguments. Overall, the overhead is a fixed polynomial in the adversary size t, the inverse breaking probability δ^{-1}, and the security parameter, so this does not present a problem.

We also note that in the actual paper, we define an intermediate notion of puzzle soundness against non-faulty solvers, which do not err in solving the puzzle, but sometimes identifiably fail. This is meant to capture the minimal notion of puzzles sufficient for universal lifting. We refer the reader to the body for the details.

Incremental Verification. The setting of incrementally verifiable universal arguments generalizes the one of universal arguments. Here we think of the universal relation as

$$\mathcal{L}_{\mathcal{U}} = \left\{(M, \mathsf{cf}, \mathsf{cf}', T) : M \text{ transitions from } \mathsf{cf} \text{ to } \mathsf{cf}' \text{ in time } T\right\} \ .$$

Universal (and semi-universal) arguments are defined exactly the same, but now there is an additional *proof update* algorithm that allows to take a proof for $(M, \mathsf{cf}_0, \mathsf{cf}_T, T) \in \mathcal{L}_{\mathcal{U}}$ and update it to a proof of $(M, \mathsf{cf}_0, \mathsf{cf}_{T+1}, T+1) \in \mathcal{L}_{\mathcal{U}}$ in time independent of T. As already mentioned also here there exists semi-universal constructions [PP22, DGKV22]. Our lifting theorem essentially works a similar way to the non-incremental case, only that solutions for the puzzles $Z_1, \ldots, Z_\lambda$ are now computed incrementally. That is the proof for step T also includes the T-th state in the computation of each of the puzzles $Z_1, \ldots, Z_\lambda$.

To keep the incremental nature of the computation it is important though that the puzzles can be computed with a fixed amount of space (independent of T). For this purpose we need to strengthen our complexity theoretic assumption. We now assume that there exists a constant $d \in \mathbb{N}$ such that for any $c \in \mathbb{N}$, $\mathbf{P} \cap \mathbf{DSPACE}(n^d) \not\subset \mathbf{ioSIZE}(n^c)$. This is arguably still a rather mild worst-case assumption (in particular, it still follows from **nuETH**). Using the fact that homomorphically evaluating a space-efficient computation can be done roughly in the same space, computing the corresponding puzzle solution is space efficient.

Future Direction: Polynomial Hardness of PPAD, Generically. One appealing application of incrementally verifiable proofs has been in the context of proving hardness in the complexity class $\mathbf{CLS} = \mathbf{PPAD} \cap \mathbf{PLS}$ (which implies the hardness of finding Nash equilibria and more) [JPY88, Pap94, FGHS21]. While by now there are hardness results based on subexponentially hard LWE [JKKZ21], hardness results based on polynomial assumptions have been harder to achieve [GPS16, BCH+22]. In particular, results based on polynomial LWE, or in fact any post-quantum assumption, are not known.

Incrementally-verifiable universal arguments give a generic way of proving hardness in **CLS**, assuming that they also satisfy *uniqueness,* namely that ambiguous proofs (even for true statements) are hard to find (c.f. [KPY20]). In the body, we prove a stronger lifting theorem then the one described above that also preserves *uniqueness*. That is, if the underlying semi-universal incrementally-verifiable argument has uniqueness, so will the lifted one. (In a nutshell, this is done by also adding incremental proofs for the computation of the puzzles themselves.)

At this point, semi-universal incrementally-verifiable arguments from polynomial LWE [PP22, DGKV22] do not satisfy uniqueness, and achieving this remains an open problem. Solving it will imply, combined with our results, PPAD hardness from polynomial LWE and the worst-case complexity assumption discussed above. Our lifting theorem does imply under these assumptions average-case hardness in **PLS** following [BG20].

Comparison to [BS23]. The work of [BS23] also relies on a form of cryptographic puzzles to reduce super-polynomial assumptions to polynomial ones in the context of constructing fully-homomorphic encryption. They rely on a stronger form of puzzles that can be sampled together with solutions (similarly to time-lock puzzles [RSW96], but with no depth considerations). Such puzzles are constructed in [BGJ+16] based on succinct randomized encodings, which can in turn be based on inditinguishability obfuscation (with small input space). Indeed, the use of puzzles in [BS23] is quite different from the one in this work: it is meant to reduce the number of hybrids in a certain reduction.

We view finding additional settings where cryptographic puzzles can be used to avoid super-polynomial loss as an appealing research direction for future work.

2 Preliminaries

Languages: Given a language $\mathcal{L} \subseteq \{0,1\}^*$, we define $\mathcal{L}(\cdot)$ to be its characteristic function.

Efficient Adversaries:

- PPT stands for probabilistic polynomial-time.
- For a PPT algorithm M, we denote by $M(x;r)$ the output of M on input x and random coins r, and by $M(x)$ the random variable, given by sampling the coins r uniformly at random.
- A polynomial-size circuit family $\mathcal{C}$ is a sequence of circuits $\mathcal{C} = \{C_\lambda\}_{\lambda\in\mathbb{N}}$, such that each circuit C_λ is of polynomial size $\lambda^{O(1)}$. We also consider probabilistic circuits that may toss random coins.

2.1 Homomorphic Encryption

In this section we define a fully homomorphic encryption scheme.

Definition 2.1 (Fully Homomorphic Encryption)**.** *A (public key) fully homomorphic encryption scheme* FHE *consists of four PPT algorithms (*Gen*,* Enc*,* Dec*,* Eval*) satisfying:*

Correctness. *For any polynomial ℓ, large enough $\lambda \in \mathbb{N}$, circuit C of size at most $\ell(\lambda)$, and message m,*

$$\Pr\left[\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}(C,\ \mathsf{Enc}_{\mathsf{pk}}(m))) = C(m) \ \middle|\ (\mathsf{sk},\ \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)\right] = 1 \ .$$

Semantic Security. *For any pair of equal-length messages $m_0, m_1 \in \{0,1\}^*$,*

$$\mathsf{pk}, \mathsf{Enc}_{\mathsf{pk}}(m_0) \approx_c \mathsf{pk}, \mathsf{Enc}_{\mathsf{pk}}(m_1) \ ,$$

where $(\mathsf{sk},\ \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$.

Compactness. *There exists a polynomial $p(\cdot)$, such that for any $\lambda \in \mathbb{N}$,* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, message $m \in \{0,1\}^*$, and circuit C with input size $|m|$,*

$$|\mathsf{Eval}(C, \mathsf{Enc}_{\mathsf{pk}}(m))| = |C(m)| \cdot p(\lambda) \ .$$

Complexity Preservation. *There exists a polynomial q such that any circuit of size t and width w, can be homomorphically evaluated in time $t \cdot q(\lambda)$ and space $w \cdot (\lambda)$.*

On Complexity Preservation and Homomorphic Evaluation of Turing Machines. The complexity preservation property defined above is typically not required in applications and accordingly not explicitly defined. It is satisfied though by typical FHE schemes, such as any *gate-by-gate* FHE.

We also note that complexity preservation implies, by standard reductions, that we can homomorphically evaluate any time-$t(n)$, space-$s(n)$ Turing machine in time $t \cdot \mathrm{poly}(\log t, n, \lambda)$ and space $s \cdot \mathrm{poly}(\log t, n, \lambda)$ for an appropriate polynomial poly. Accordingly, in our construction of puzzles, it will be convenient to directly address homomorphic evaluation of Turing machines.

2.2 Non-Interactive Arguments for Deterministic Computations

In this section we define non-interactive arguments for deterministic computations. Such arguments allow a prover to convince a verifier of the outcome of a long computation. For a T-time computation, the prover should run in time $\text{poly}(T)$ while the verifier runs in time significantly less than T.

The Universal Language. Let $\mathcal{L}_{\mathcal{U}}$ be the language of all quadruplets $(M, \mathsf{cf}, \mathsf{cf}', T)$ such that M is a deterministic Turing machine that starting from configuration cf transitions to configuration cf' in T steps. A Turing machine configuration includes the machine's state and its entire memory.

A non-interactive argument for $\mathcal{L}_{\mathcal{U}}$ consists of algorithms $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ with the following syntax:

Setup: The probabilistic setup algorithm Gen takes as input a security parameter $\lambda \in \mathbb{N}$. It outputs a prover key pk and a verifier key vk.

Prover: The deterministic prover algorithm Prove takes as input a prover key pk and an instance $y \in \mathcal{L}_{\mathcal{U}}$. It outputs a proof π.

Verifier: The deterministic verifier algorithm Verify takes as input a verifier key vk, an instance y and a proof π. It outputs a bit indicating if it accepts or rejects.

We next define the formal requirements from non-interactive arguments for $\mathcal{L}_{\mathcal{U}}$. The definition distinguishes between plain soundness and universal soundness. The former only guarantees soundness for computations that are polynomially bounded (this corresponds to the semi-universal arguments discussed in the introduction), while the later guarantees soundness also for super-polynomial computations.

Definition 2.2. *A non-interactive argument for $\mathcal{L}_{\mathcal{U}}$ satisfies the following requirements:*

Completeness. *For every $\lambda \in \mathbb{N}$ and $y = (M, \mathsf{cf}, \mathsf{cf}', T) \in \mathcal{L}_{\mathcal{U}}$ such that $|y|, T \leq 2^{\lambda}$:*

$$\Pr\left[\mathsf{Verify}(\mathsf{vk}, y, \pi) = 1 \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda) \\ \pi \leftarrow \mathsf{Prove}(\mathsf{pk}, y) \end{array}\right] = 1 \ .$$

Efficiency. *In the completeness experiment above:*

- *The setup algorithm runs in time* $\text{poly}(\lambda)$.
- *The prover algorithm runs in time* $\text{poly}(\lambda, |y|, T)$ *and outputs a proof π of length* $\text{poly}(\lambda)$.
- *The verifier algorithm runs in time* $\text{poly}(\lambda, |y|)$.

Soundness. *For every polynomial $\bar{T} = \bar{T}(\lambda)$ and $\text{poly}(\lambda)$-size adversary $\mathcal{A}$ there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$:*

$$\Pr\left[\begin{array}{l} \mathsf{Verify}(\mathsf{vk}, y, \pi) = 1 \\ T \leq \bar{T}(\lambda) \\ y \notin \mathcal{L}_M \end{array} \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda) \\ \left(y = \left(M, \mathsf{cf}, \mathsf{cf}', T\right), \pi\right) \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{vk}) \end{array}\right] \leq \mu(\lambda) \ .$$

We say that:

- *The scheme has* universal soundness *if we set $\bar{T}$ to be 2^{λ} instead of polynomial in λ. Schemes satisfying universal soundness are called universal arguments. Schemes satisfying (plain) soundness are called semi-universal arguments.*
- *The scheme is* privately verifiable *if it satisfies a weaker notion of soundness where $\mathcal{A}$ is only given pk but not vk.*
- *The scheme is* sound against uniform adversaries*, if the adversary in the soundness requirement is restricted to be a uniform PPT algorithm.*

Below we state existing results on non-interactive arguments for $\mathcal{L}_{\mathcal{U}}$ based on *polynomial* assumptions. These constructions are semi-universal, meaning that they satisfy soundness but are not known to satisfy universal soundness.

Theorem 2.3 ([CJJ21a, WW22, KLVW22])**.** *Assuming the hardness of either the Learning with Errors (LWE) problem or the Decisional Linear (DLIN) problem in bilinear groups, there exist semi-universal non-interactive arguments for $\mathcal{L}_{\mathcal{U}}$.*

Theorem 2.4 ([BHK17])**.** *Assuming PIR schemes exist, there exist semi-universal privately-verifiable non-interactive arguments for $\mathcal{L}_{\mathcal{U}}$.*

2.3 Incrementally Verifiable Computation

In this section we define incrementally verifiable computation. An incrementally verifiable computation scheme is a non-interactive argument for $\mathcal{L}_{\mathcal{U}}$ that is equipped with an additional update algorithm with the following syntax: The deterministic algorithm Update takes as input the public key pk, a statement $y \in \mathcal{L}_{\mathcal{U}}$ and a proof π. It outputs a proof π'.

Definition 2.5. *A non-interactive argument* $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ *for* $\mathcal{L}_{\mathcal{U}}$ *together with an update algorithm* Update *is called* incremental *if it satisfies the following requirements:*

Incremental Completeness. *For every $\lambda \in \mathbb{N}$ and machine M:*

- *For every configuration* cf*:*

$$\Pr\left[\mathsf{Verify}\left(\mathsf{vk}, (M, \mathsf{cf}, \mathsf{cf}, 0), \mathcal{E}\right) = 1 \mid (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda)\right] = 1 \ .$$

  Where $\mathcal{E}$ is the empty proof.

- *For every $T < 2^{\lambda}$, pair of statements $y, y' \in \mathcal{L}_{\mathcal{U}}$ of the form $y = (M, \mathsf{cf}, \mathsf{cf}', T)$ and $y' = (M, \mathsf{cf}, \mathsf{cf}'', T+1)$ and a proof π:*

$$\Pr\left[\begin{array}{l|l} \mathsf{Verify}(\mathsf{vk}, y, \pi) = 1 & (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda) \\ \mathsf{Verify}(\mathsf{vk}, y', \pi') = 0 & \pi' \leftarrow \mathsf{Update}(\mathsf{pk}, y, \pi) \end{array}\right] = 0 \ .$$

Update Efficiency. *In the incremental completeness experiments above, the update algorithm runs in time $|y| \cdot \mathrm{poly}(\lambda)$, and outputs a proof π of length $\mathrm{poly}(\lambda)$.*

2.4 Average-Case Puzzles

In this section we define hard on average puzzles.

Syntax. A puzzle is given by a deterministic uniform algorithm F that takes as input a difficulty parameter $t \in \mathbb{N}$ and $x \in \{0,1\}^*$ and outputs $y \in \{0,1\}^{m(|x|)}$, for some polynomial $m(\cdot)$. An average-case puzzle is also given by a probabilistic uniform sampler D that takes as input a difficulty parameter $t \in \mathbb{N}$ and a security parameter $\lambda \in \mathbb{N}$ and outputs $x \in \{0,1\}^{n(\lambda)}$, for some polynomial $n(\cdot)$. For a function $t(\cdot)$, we denote by $\mathsf{F}_{t,\lambda}(x)$ the function $\mathsf{F}(t(\lambda), x)$, and by F_t the function ensemble $\{\mathsf{F}_{t,\lambda}\}_{\lambda\in\mathbb{N}}$. Similarly, we denote by $\mathsf{D}_{t,\lambda}$ the distribution $\mathsf{D}(t(\lambda), \lambda)$, and by D_t the distribution ensemble $\{\mathsf{D}(t(\lambda), \lambda)\}_{\lambda\in\mathbb{N}}$.

Before defining puzzle average-case we define the notion of non-faulty solver:

Definition 2.6 ($(\delta, \mathcal{D})$-Faulty Solver)**.** *Let $F = \{F_\lambda\}_{\lambda\in\mathbb{N}}$ be a function ensemble, and let $D = \{D_\lambda\}_{\lambda\in\mathbb{N}}$ be a distribution ensemble, and $\delta : \mathbb{N} \to [0,1]$. An algorithm $\mathcal{A}$ is a (δ, D)-faulty F-solver if for all large enough $\lambda \in \mathbb{N}$, $\Pr\left[\mathcal{A}(x) \notin \{\perp, F_\lambda(x)\} \mid x \leftarrow D_\lambda\right] \leq \delta(\lambda)$, where the probability is also over the random coins tosses of $\mathcal{A}$.*

Definition 2.7 (Average-Case Puzzle Against Non-Faulty Solvers)**.** *An average-case puzzle against non-faulty solvers satisfies the following requirements:*

Efficiency.

- $\mathsf{F}(t,x)$ *runs in time* $t \cdot \mathrm{poly}(\log t, |x|)$.
- $\mathsf{D}(t,\lambda)$ *runs in time* $\mathrm{poly}(\log t, \lambda)$.

We further say that the puzzle is space efficient *if* $\mathsf{F}(t,x)$ *runs in space* $\mathrm{poly}(\log t, |x|)$.

Average-case hardness against non-faulty solvers. *For every polynomial p, there exists a constant c, such that for any polynomially bounded $t \geq p^c$, $\delta \leq \frac{0.9}{p}$, any (δ, D_t)-faulty F_t-solver $\mathcal{A}$ with size at most p, and any large enough $\lambda \in \mathbb{N}$,*

$$\Pr\left[\mathcal{A}(x) = \mathsf{F}_{t,\lambda}(x) \mid x \leftarrow \mathsf{D}_{t,\lambda}\right] \leq \frac{1}{p(\lambda)} \ .$$

Such a puzzle is called Average-Case Puzzle Against Uniform Non-Faulty Solvers (denoted ACPU) if it is secure against uniform PPT adversaries with running time at most p.

3 Universal Lifting

In this section we show how to lift any semi-universal non-interactive argument to a universal one, based on average case puzzles.

Theorem 3.1. *Assume there exists a (semi-universal) non-interactive argument for $\mathcal{L}_{\mathcal{U}}$. Additionally, assume there exists an average-case puzzle against non-faulty solvers. Then there exists a non-interactive universal argument for $\mathcal{L}_{\mathcal{U}}$, namely one with universal soundness.*

If the original argument is only privately verifiable then so is the resulting universal argument.

If the the puzzle is only sound against uniform adversaries then so is the resulting universal argument.

In the rest of this section we prove theorem 3.1.

Construction. Let $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ be the semi-universal argument. Let D be the algorithm sampling puzzle instances and let F be the puzzle solver. We construct a universal argument $(\mathsf{Gen}', \mathsf{Prove}', \mathsf{Verify}')$.

We start with a high level overview of the construction. The setup algorithm generates the keys of the original argument and also generates puzzles $\mathsf{Z}_0, \mathsf{Z}_1, \ldots, \mathsf{Z}_\lambda$ with increasing difficulty levels $1, 2, \ldots, 2^\lambda$ that the prover is challenged to solve. Given a statement $y = (M, \mathsf{cf}, \mathsf{cf}', T)$, the prover creates a proof that $y \in \mathcal{L}_{\mathcal{U}}$ using the semi-universal argument. In addition, it attaches a solution to all puzzles up to difficulty level T together with a proof that it is indeed the solution.

More precisely, let $\bar{i} \leq \lambda$ be the largest such that $2^{\bar{i}} \leq T$. Let $\mathsf{cf}[t, \mathsf{Z}]$ be the starting configuration of F containing (t, Z) as input and $\mathsf{cf}'[\mathsf{S}]$ the ending configuration of F containing S as output. Let T_i be the running time of F on puzzles with security parameter λ and difficulty 2^i. For every $0 \leq i \leq \bar{i}$, the prover attaches the solution S_i of Z_i and the proof that $\left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right) \in \mathcal{L}_{\mathcal{U}}$.

We now fully describe $(\mathsf{Gen}', \mathsf{Prove}', \mathsf{Verify}')$.

Setup: Given a security parameter λ:

- Sample $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(\lambda)$.
- Sample $\mathsf{Z}_i \leftarrow \mathsf{D}(2^i, \lambda)$, for every $0 \leq i \leq \lambda$.
- Output $\mathsf{pk}' = \left(\mathsf{pk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right), \mathsf{vk}' = \left(\mathsf{vk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right)$.

Prover: Given a key $\mathsf{pk}' = \left(\mathsf{pk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right)$ and an instance $y = (M, \mathsf{cf}, \mathsf{cf}', T)$:

- Compute $\pi \leftarrow \mathsf{Prove}(\mathsf{pk}, y)$.
- Solve $\mathsf{S}_i \leftarrow \mathsf{F}(2^i, \mathsf{Z}_i)$, for every $0 \leq i \leq \bar{i}$.

- Compute $\pi_i \leftarrow \mathsf{Prove}\left(\mathsf{pk}, \left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right)\right)$, for every $0 \leq i \leq \bar{i}$.
- Output the proof $\pi' = \left(\pi, (\mathsf{S}_i, \pi_i)_{0 \leq i \leq \bar{i}}\right)$.

Verifier: Given a key $\mathsf{vk}' = \left(\mathsf{vk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right)$, an instance $y = (M, \mathsf{cf}, \mathsf{cf}', T)$ and a proof $\pi' = \left(\pi, (\mathsf{S}_i, \pi_i)_{0 \leq i \leq \bar{i}}\right)$:

- Run $\mathsf{Verify}(\mathsf{vk}, y, \pi)$.
- Run $\mathsf{Verify}\left(\mathsf{vk}, \left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right), \pi_i\right)$, for every $0 \leq i \leq \bar{i}$.
- Accept iff all of the verifiers accept.

Completeness and efficiency follow readily from the completeness and efficiency of the underlying argument and the puzzle. We focus on proving universal soundness.

Universal Soundness. We prove soundness against non-uniform adversaries, assuming the puzzle is sound against non-uniform adversaries. The uniform case is analogous.

Assume by contradiction that there exists a $\mathrm{poly}(\lambda)$-size adversary $\mathcal{A}'$ and a function $\delta(\lambda) = 1/\mathrm{poly}(\lambda)$ such that for infinitely many λ, for

$$\begin{array}{l}(\mathsf{pk}', \mathsf{vk}') \leftarrow \mathsf{Gen}'(\lambda) \\ \left(y = \left(M, \mathsf{cf}, \mathsf{cf}', T\right), \pi'\right) \leftarrow \mathcal{A}'(\mathsf{pk}', \mathsf{vk}') \ ,\end{array}$$

it holds that:

$$\Pr\left[\begin{array}{l}\mathsf{Verify}'(\mathsf{vk}', y, \pi') = 1 \\ T \leq 2^\lambda \\ y \notin \mathcal{L}_\mathcal{U}\end{array}\right] \geq \delta \ . \tag{1}$$

Denote this experiment by Exp and say that $\mathcal{A}'$ succeeds in Exp if the event in equation 1 occurs.

First we describe the general idea of the reduction. We consider different cases, where in some, we show a reduction to the soundness of the underlying semi-universal argument, and in the others, a reduction to the average-case hardness of the puzzle:

- If $\mathcal{A}'$ proves a false statement with a small number of steps T, i.e. number of steps that $\mathcal{A}'$ is capable of computing, we can use it to construct an adversary for the semi-universal argument that proves a false statement corresponding to a polynomial number of steps T, and hence breaks soundness.
- If $\mathcal{A}'$ proves a false statement with a large number of steps T, it has to give a solution and a proof to all puzzles up to difficulty level T. We focus on puzzle number i that is not too difficult but difficult enough. i.e., its difficulty parameter 2^i is bounded by a polynomial, but it is too big for $\mathcal{A}'$ to compute.

  Now, if $\mathcal{A}'$ solves the puzzle correctly, we can use it to construct a solver breaking the average case puzzle. If $\mathcal{A}'$ doesn't solve the puzzle correctly, it convinces the verifier with a false statement. In this case we use $\mathcal{A}'$ to construct an adversary that breaks the semi-universal argument.

We now describe the reduction in detail. Let $p = \mathrm{poly}(\lambda)$ be a polynomial such that $p \geq \frac{3}{\delta}$. The polynomial p will in fact satisfy more constraints that depend polynomially on the size of the adversary. We will specify how to exactly fix it later on (see dedicated paragraph within Case 2.a below).

By our choice of p,

$$\Pr[\mathcal{A}' \text{ succeeds}] \geq \frac{3}{p} \ .$$

Let c be the parameter of the puzzle (D, F) such that for any $t \geq p^c$, $\delta' \leq \frac{0.9}{p}$, and (δ', D_t)-faulty F_t-solver $\mathcal{A}^{\mathsf{puz}}$ with size at most p, and any large enough $\lambda \in \mathbb{N}$,

$$\Pr\left[\mathcal{A}^{\mathsf{puz}}(\mathsf{Z}) = \mathsf{F}_{t,\lambda}(\mathsf{Z}) \mid \mathsf{Z} \leftarrow \mathsf{D}_{t,\lambda}\right] \leq \frac{1}{p} \ .$$

Let T be the number of steps in the statement that $\mathcal{A}'$ outputs. We consider several cases according to the value of $\Pr[\mathcal{A}' \text{ succeeds } \wedge T/2 \leq p^c]$.

Case 1. Reduction to Semi-Universal Arguments. For infinitely many λ,

$$\Pr\left[\mathcal{A}' \text{ succeeds } \wedge T/2 \leq p^c\right] \geq \frac{3}{2p} \ .$$

Let $\bar{T} = 2p^c$. We define a polynomial adversary $\mathcal{A}$ against the semi-universal argument, proving false statements with number of steps $\leq \bar{T}$.

$\mathcal{A}(\mathsf{pk}, \mathsf{vk})$ simulates the experiment Exp, only the inner keys are replaced with $(\mathsf{pk}, \mathsf{vk})$. It outputs the main statement y and its proof π that $\mathcal{A}'$ outputs. When $(\mathsf{pk}, \mathsf{vk})$ are generated by Gen, we get the same probability space as in Exp and therefore

$$\Pr\left[\begin{array}{l} \mathsf{Verify}(\mathsf{vk}, y, \pi) = 1 \\ T \leq \bar{T} \\ y \notin \mathcal{L}_{\mathcal{U}} \end{array}\right] \geq \Pr[\mathcal{A}' \text{ succeeds } \wedge T/2 \leq p^c] \geq \frac{3}{2p} \ ,$$

in contradiction to plain soundness.

Case 2. Assume that for infinitely many λ,

$$\Pr[\mathcal{A}' \text{ succeeds } \wedge T/2 > p^c] \geq \frac{3}{2p} \ .$$

Then for such λ, $\exists i \leq \bar{i}$ such that $2^{i-1} \leq p^c \leq 2^i$. Again we split into cases:

Case 2.a. Reduction to Puzzles. Assume that for infinitely many λ,

$$\Pr[\mathcal{A}' \text{ succeeds } \wedge T/2 > p^c \wedge \mathsf{F}(2^i, \mathsf{Z}_i) \neq \mathsf{S}_i] < \frac{1}{3} \cdot \frac{3}{2p} \ .$$

Let $t = 2^i$ and $\delta' = \frac{1}{2p}$. Note that $t \geq p^c$ and $\delta' \leq \frac{0.9}{p}$. We define a (δ', D_t)-faulty F_t-solver $\mathcal{A}^{\mathsf{puz}}$, with size at most p that solves puzzles of difficulty t w.p. $> \frac{1}{p}$.

$\mathcal{A}^{\mathsf{puz}}(\mathsf{Z})$ simulates the experiment Exp, only the i'th puzzle in the key is replaced with Z. It then passes the output of $\mathcal{A}'$ to Verify'. If Verify' doesn't accept or $p^c \geq T/2$, it outputs $\perp$. Otherwise, it outputs $\mathcal{A}'$'s solution to the i'th puzzle: S_i.

When Z is generated by D_t, we get the same probability space as in Exp and so

$$\begin{aligned} &\Pr[\mathcal{A}^{\mathsf{puz}}(\mathsf{Z}) \notin \{\perp, \mathsf{F}(t, \mathsf{Z})\}] = \\ &= \Pr\left[\mathcal{A}' \text{ succeeds } \wedge T/2 > p^c \wedge \mathsf{F}(2^i, \mathsf{Z}_i) \neq \mathsf{S}_i\right] < \frac{1}{2p} = \delta' \ , \end{aligned}$$

and

$$\begin{aligned} &\Pr[\mathcal{A}^{\mathsf{puz}}(\mathsf{Z}) = \mathsf{F}(t, \mathsf{Z})] = \\ &= \Pr\left[\mathcal{A}' \text{ succeeds } \wedge T/2 > p^c \wedge \mathsf{F}(2^i, \mathsf{Z}_i) = \mathsf{S}_i\right] > \frac{2}{3} \cdot \frac{3}{2p} = \frac{1}{p} \ . \end{aligned}$$

This contradicts the average-case hardness of the puzzle provided that the size of $\mathcal{A}^{\mathsf{puz}}$ is at most p.

$\mathcal{A}^{\mathsf{puz}}$'s Size, Choice of p, and the Uniform Case. Note that $\mathcal{A}^{\mathsf{puz}}$'s size is a polynomial in the size of $\mathcal{A}'$ and running times of Gen' and Verify', and all together it is a fixed polynomial in λ. We take p to be the maximum of this polynomial and $\frac{3}{\delta}$, and so the solver runs in time $\leq p$. Also note that this is the only place in the proof where relying on uniform (as opposed to non-uniform) puzzles makes a difference. Here we note that if the adversary $\mathcal{A}'$ is uniform then so is the solver $\mathcal{A}^{\mathsf{puz}}$.

Case 2.b. Reduction to Semi-Universal Arguments. For infinitely λ,

$$\Pr[\mathcal{A}' \text{ succeeds } \wedge T/2 > p^c \wedge \mathsf{F}(2^i, \mathsf{Z}_i) \neq \mathsf{S}_i] \geq \frac{1}{3} \cdot \frac{3}{2p} \ .$$

Recall that $\mathcal{A}'$ outputs, amongst others, a proof π_i that $\left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right) \in \mathcal{L}_{\mathcal{U}}$. In the case that $\mathcal{A}'$ succeeds and $\mathsf{F}(2^i, \mathsf{Z}_i) \neq \mathsf{S}_i$, we get that $\left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right) \notin \mathcal{L}_{\mathcal{U}}$ and Verify accepts π_i. This means that Verify is convinced by a proof of a wrong statement. In addition, the running time T_i of $\mathsf{F}(2^i, \mathsf{Z}_i)$ satisfies $T_i = 2^i \cdot \mathrm{poly}(\lambda) \leq 2p^c \cdot \mathrm{poly}(\lambda)$ and so there exists a polynomial $\bar{T}(\lambda)$ such that $T_i \leq \bar{T}$. We use this to construct an adversary $\mathcal{A}$ against the semi-universal argument, proving false statements with number of steps $\leq \bar{T}$.

$\mathcal{A}(\mathsf{pk}, \mathsf{vk})$ simulates the experiment Exp, only the inner keys are replaced with $(\mathsf{pk}, \mathsf{vk})$. It outputs the instance $y_i = \left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}'\left[\mathsf{S}_i\right], T_i\right)$ and its proof π_i, both generated by $\mathcal{A}'$. When $(\mathsf{pk}, \mathsf{vk})$ are generated by Gen, we get the same probability space as in Exp and therefore:

$$\Pr\left[\begin{array}{l} \mathsf{Verify}(\mathsf{vk}, y_i, \pi_i) = 1 \\ T_i \leq \bar{T} \\ y_i \notin \mathcal{L}_{\mathcal{U}} \end{array}\right] \geq \Pr[\mathcal{A}' \text{ succeeds } \wedge p^c < T/2 \wedge \mathsf{F}(2^i, \mathsf{Z}_i) \neq \mathsf{S}_i] \geq \frac{1}{2p} \ ,$$

contradicting soundness.

On the Security Loss of the Reduction. Most cryptographic reductions in the literature have a fixed security loss — there is a universal constant c, such that if the adversary breaks a given scheme in time t and probability $1/t$, then the reduction breaks the underlying assumption in time t^c and probability $1/t^c$. We note that as is, our reduction does not have such a fixed loss, due to the fact that we assume a very weak form of puzzles that do not have a fixed gap (namely for any polynomial p there exists a constant c such that there is no $(p, 1/p)$-solver for puzzles of difficulty p^c). If we assume a stronger form of puzzles that switches the quantifiers and has a universal gap c (independent of p), then our reduction has a fixed loss as well. Puzzles with a universal gap indeed follow in the uniform case (with no additional assumptions) or in the non-uniform case assuming **nuETH** (see Section 4 for the corresponding constructions).

3.1 Incrementally Verifiable Computation Lifting

In this section we extend the universal non-interactive argument construction and construct a universal IVC based on a semi-universal IVC.

Theorem 3.2. *Assume that there exists an incrementally verifiable computation for $\mathcal{L}_{\mathcal{U}}$. Additionally, assume that there exists a space-efficient average-case puzzle against non-faulty solvers. Then there exists an incrementally verifiable computation for $\mathcal{L}_{\mathcal{U}}$ with universal soundness.*

If the original IVC is only privately verifiable then the resulting universal IVC is also privately verifiable.

If the puzzle is only sound against uniform adversaries then the resulting universal IVC is also sound against uniform adversaries.

In the rest of this section, we prove theorem 3.2. We use the construction from theorem 3.1, only we add to the proof partial computation data, allowing us to define the Update algorithm.

Construction. Let $(\mathsf{Gen}, \mathsf{Prove}, \mathsf{Update}, \mathsf{Verify})$ be the semi-universal IVC. Let (D, F) be the puzzle. We construct a universal IVC $(\mathsf{Gen}', \mathsf{Prove}', \mathsf{Update}', \mathsf{Verify}')$.

We use the same setup algorithm Gen' as in theorem 3.1. i.e., the keys $(\mathsf{pk}', \mathsf{vk}')$ contain the keys of the underlying IVC $(\mathsf{pk}, \mathsf{vk})$, and puzzle instances $\mathsf{Z}_0, \mathsf{Z}_1, \ldots, \mathsf{Z}_\lambda$ with increasing difficulty levels $2^0, 2^1, \ldots, 2^\lambda$.

The proof π' includes the proof of the main statement using the underlying IVC, and partial computations of all the puzzles including proofs of these partial computations (again, using the underlying IVC). More precisely, let q be the polynomial such that $\mathsf{F}(t, \mathsf{Z})$ runs in time $t \cdot q(\lambda)$ (there is an extra factor of $\mathrm{poly}\log t$ in F's running time, but $\log t \leq \lambda$). The proof π' of a statement with number of steps T, contains for every $0 \leq i \leq \lambda$, a configuration cf_i of the machine F, which is the result of computing $T \cdot q(\lambda)$ steps starting from

the configuration $\mathsf{cf}\left[2^i, \mathsf{Z}_i\right]$. In addition, for every $0 \leq i \leq \lambda$ we attach a proof π_i that cf_i is indeed the result of this partial computation.

Note that for every i such that $2^i \leq T$, cf_i is the final configuration of the computation and so it contains the puzzle solution. It is similar to the non-interactive argument proof in 3.1 where we have a solution (and a proof) to puzzle i for every i such that $2^i \leq T$.

We now describe $\mathsf{Update}', \mathsf{Prove}', \mathsf{Verify}'$ in detail.

Update: Given a key $\mathsf{pk}' = \left(\mathsf{pk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right)$, an instance $y = \left(M, \mathsf{cf}, \mathsf{cf}', T\right)$ and a proof $\pi' = \left(\pi, (\mathsf{cf}_i, \pi_i)_{0 \leq i \leq \lambda}\right)$:

- Update the inner proof $\pi^+ \leftarrow \mathsf{Update}(\mathsf{pk}, y, \pi)$.
- For every $0 \leq i \leq \lambda$, advance the ith puzzle computation and update the proof:
  - Let $\mathsf{cf}_i^0 = \mathsf{cf}_i, \pi_i^0 = \pi_i$. If $T = 1$ (i.e., all the components in π' are empty), take $\mathsf{cf}_i^0 = \mathsf{cf}\left[2^i, \mathsf{Z}_i\right]$.
  - For every $1 \leq j \leq q(\lambda)$, compute one step of F from cf_i^{j-1} to cf_i^j. Then update the proof: $\pi_i^j \leftarrow \mathsf{Update}\left(\mathsf{pk}, \left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}_i^j, (T-1)\, q + j\right), \pi_i^{j-1}\right)$.
  - Set $\mathsf{cf}_i^+ \leftarrow \mathsf{cf}_i^q, \pi_i^+ \leftarrow \pi_i^q$.
- Output the proof $\pi'^+ = \left(\pi^+, \left(\mathsf{cf}_i^+, \pi_i^+\right)_{0 \leq i \leq \lambda}\right)$.

Prover: To prove a statement with number of steps T, we run the update algorithm Update' for T times one after the other, starting from an empty proof.

Verifier: Given a key $\mathsf{vk}' = \left(\mathsf{vk}, (\mathsf{Z}_i)_{0 \leq i \leq \lambda}\right)$, an instance $y = \left(M, \mathsf{cf}, \mathsf{cf}', T\right)$ and a proof $\pi' = \left(\pi, (\mathsf{cf}_i, \pi_i)_{0 \leq i \leq \lambda}\right)$, the verifier simply verifies all the inner proofs:

- Verify: $\mathsf{Verify}\left(\mathsf{vk}, y, \pi\right)$.
- For every $0 \leq i \leq \lambda$, verify: $\mathsf{Verify}\left(\mathsf{vk}, \left(\mathsf{F}, \mathsf{cf}\left[2^i, \mathsf{Z}_i\right], \mathsf{cf}_i, T \cdot q\right), \pi_i\right)$.

Incremental Completeness. Since the verifier verifies all the inner proofs $\pi, (\pi_i)_{0 \leq i \leq \lambda}$ using the underlying IVC, and the update algorithm updates these inner proofs with the underlying IVC, incremental completeness follows from incremental completeness of the underlying IVC.

Efficiency. The update algorithm computes $q(\lambda)$ steps of F for all the puzzles and the proof includes the configuration of F after this computation. Since F is space efficient, the configuration is of size $\mathrm{poly}(\lambda)$. This, together with update efficiency of the underlying IVC implies update efficiency of the universal IVC.

Universal Soundness. As mentioned above, for every i such that the non-interactive argument proof contains the solution to the ith puzzle, also the IVC proof contains the solution, since it contains the final configuration of the puzzle computation. Therefore, universal soundness can be proved analogously to the proof in theorem 3.1.

4 Constructing Average-Case Puzzles

In this section we construct hard-on-average puzzles (definition 2.7). Our construction is based on FHE combined with appropriate hierarchy theorems/assumptions, where different hierarchy theorems/assumptions yield different efficiency and security guarantees. For security against uniform adversaries, we rely on a probabilistic time hierarchy theorem for slightly non-uniform computations (theorem 4.6) by Barak [Bar02]. For puzzles against non-uniform adversaries, and for space efficient puzzles we make mild hierarchy assumptions for space bounded computations (4.1 and 4.2). Both assumptions follow from **nuETH** (see assumption 4.3 and theorem 4.1).

4.1 Worst-Case Hardness Assumptions

In this section we define complexity worst-case assumptions for instantiating average-case puzzles.

Assumption 4.1. *For any polynomial $q(\cdot)$ there exists a polynomial $Q(\cdot)$ and a language $\mathcal{L} \in \boldsymbol{DTIME}(Q)$ such that any family $\mathcal{C} = \{\mathcal{C}_\lambda\}$ of size-$q(\lambda)$ circuits fails to decide $\mathcal{L}_\lambda = \mathcal{L} \cap \{0,1\}^\lambda$ for all large enough λ.*

Assumption 4.2. *There exists a polynomial $s(\cdot)$ such that for any polynomial $q(\cdot)$ there exists a polynomial Q and a language $\mathcal{L} \in \boldsymbol{DTIME}(Q) \cap \boldsymbol{DSPACE}(s)$ such that any family $\mathcal{C} = \{\mathcal{C}_\lambda\}$ of size-$q(\lambda)$ circuits fails to decide $\mathcal{L}_\lambda = \mathcal{L} \cap \{0,1\}^\lambda$ for all large enough λ.*

Non-Uniform ETH. For $a, b \in \mathbb{N}$, let $3SAT[a,b]$ be the language of satisfiable 3-CNF formulas with a variables and b clauses.

Assumption 4.3 (nuETH [IP01, IPZ01]). *There exists constants $c, d > 0$ such that any family $\mathcal{C} = \{\mathcal{C}_\lambda\}$ of size-$2^{\lambda/d}$ circuits fails to decide $3SAT[\lambda, c \cdot \lambda]$ for all large enough λ.*

Theorem 4.1. *Assuming **nuETH** (assumption 4.3), assumptions 4.1 and 4.2 hold.*

Proof Sketch. Let c, d be the constants given by **nuETH**. Let $q(\lambda) = \lambda^k$ be a polynomial. For $\lambda \in \mathbb{N}$, let $n = d \cdot k \cdot \log(\lambda)$ and consider the language $3SAT[n, cn, \lambda]$ that consists of satisfiable 3-CNF formulas with n variables and cn clauses, padded to size λ.

By **nuETH**, for large enough λ any family of circuits of size λ^k fails to decide $3SAT[n, cn, \lambda]$. On the other hand, a SAT solver can enumerate through all $\lambda^{d \cdot k}$ possible assignments. The algorithm runs (deterministically) in time $Q(\lambda) = \lambda^{d \cdot k} \cdot \text{polylog}(\lambda)$, implying assumption 4.1. Moreover, the algorithm runs in $O(\log(\lambda))$ space, which also implies assumption 4.2. □

4.2 Average-Case Puzzles from FHE

In this section we prove the following three theorems.

Theorem 4.2. *Assuming FHE, there exists a Average-Case Puzzle Against Uniform Non-Faulty Solvers.*

Theorem 4.3. *Assuming FHE and assumption 4.1, there exists an Average-Case Puzzle Against Non-Faulty Solvers.*

Theorem 4.4. *Assuming FHE and assumption 4.2, there exists a space-efficient Average-Case Puzzle Against Non-Faulty Solvers.*

The constructions behind all of the above follow a common blueprint. We next describe this blueprint, and then address the proof for each one of the theorems.

At a high level, the construction is as follows. The sampler samples an encryption of garbage. The associated puzzle algorithm evaluates the universal machine for t steps on the ciphertext. To prove security, the reduction switches the ciphertext to an encryption of an instance of an underlying worst-case language. In this case, a correct answer of the non-faulty solver decrypts to a correct decision about the input. Finally, the success probability is amplified.

Construction. For the formal definition, we fix some encoding such that $x \in \{0,1\}^*$ is viewed as $x = (M, x')$ where M is a Turing machine and x' is an input for M. Let $U_{t,s}$ be the universal Turing machine, which on input $x = (M, x') \in \{0,1\}^n$ outputs $M(x')$ provided that M halts after at most t steps, using at most space s, and outputs a single bit. Let $U_t = U_{t,t}$ (where there is no space restriction). By known constructions $U_{t,s}$ runs in time at most $t \cdot \text{poly}(\log t, n)$ and uses space at most $s \cdot \text{poly}(\log t, n)$.

We next define a puzzle sampler D and puzzle solvers F and F^s. In what follows, let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an FHE scheme.

D - Puzzle Sampler: Given a difficulty parameter t and a security parameter λ:

- Samples $\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$
- Returns $ct \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0^{3\lambda})$

F **- Average-Case Puzzle:** Given a difficulty parameter t and input x:

- Homomorphically evaluates U_t on x

F^s **- Space-Efficient Average-Case Puzzle:** Given a difficulty parameter t and input x:

- Homomorphically evaluates $U_{t,s}$ on x

We start by proving theorem 4.3, and then extend the proof to account for theorem 4.2 and theorem 4.4.

Proof of theorem 4.3. We show that (D, F) is an Average-Case Puzzle Against Non-Faulty Solvers under Assumption 4.1.

First note that the efficiency requirements follow from the complexity preservation of homomorphic evaluation. We henceforth focus on proving hardness.

Fix any polynomial $p = p(\lambda)$ and assume toward contradiction that for every constant c, there exists a polynomially bounded function $t \geq p^c$, $\delta \leq \frac{0.9}{p}$, and a (δ, D_t)-faulty F_t-solver $\mathcal{A}$ with size at most p, such that for infinitely many λ,

$$\Pr\left[\mathcal{A}(x) = \mathsf{F}_{t,\lambda}(x) \mid x \leftarrow \mathsf{D}_{t,\lambda}\right] > \frac{1}{p(\lambda)} \quad . \tag{2}$$

We prove that there exists a polynomial q such that for any polynomial Q and language $\mathcal{L} \in \mathbf{DTIME}(Q)$, there exists a size-$q$ circuit family $\mathcal{B}$ that decides $\mathcal{L}$ on infinitely many input lengths, thereby contradicting Assumption 4.1.

Let Q be a polynomial and let $\mathcal{L} \in \mathbf{DTIME}(Q)$. Also, let M be a deterministic Turing machine that decides $\mathcal{L}$ in time Q. Consider a sampler $\widehat{\mathsf{D}}_M(x)$ that given $x \in \{0,1\}^\lambda$, samples $\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{Gen}(1^\lambda)$, and outputs an encryption $\widehat{x} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(x')$, where $x' = (M, x)$, padded to size 3λ.

We construct a probabilistic circuit family $\mathcal{B}'$ that decides $\mathcal{L}$ with error $< 2^{-\lambda}$. In particular, we can non-uniformly fix its randomness to get a deterministic decider that decides $\mathcal{L}$ on inputs $x \in \{0,1\}^\lambda$. (The size of $\mathcal{B}'$ will be a polynomial $q(\lambda)$ independent of Q.)

On input $x \in \{0,1\}^\lambda$, $\mathcal{B}'$ repeats the following process $\mathcal{D}$, $k(\lambda)$ times, for some polynomial $k(\cdot)$ to be determined later. $\mathcal{D}(x)$ samples $\widehat{x}_i \leftarrow \widehat{\mathsf{D}}_M(x)$, runs $\mathcal{A}$ on $\widehat{x}_i$, and obtains its output y_i. If $y_i \neq \perp$, $\mathcal{D}$ outputs $b_i = \mathsf{Dec}_{\mathsf{sk}}(y)$, and otherwise outputs $b_i = \perp$. Finally, $\mathcal{B}'$ outputs the majority among the values $\{b_1, \ldots, b_k\} \setminus \{\perp\}$ generated by $\mathcal{D}$.

Claim 4.5. *There exists a polynomial $t \geq Q$ and constants $0 < \alpha < 1 < \beta$, such that for infinitely many λ it holds that for any $x \in \{0,1\}^\lambda$,*

$$\Pr\left[\mathcal{D}(x) = \mathcal{L}(x)\right] \geq \max\left\{\frac{\alpha}{p(\lambda)}, \beta \cdot \Pr\left[\mathcal{D}(x) \notin \{\perp, \mathcal{L}(x)\}\right]\right\} \quad .$$

Proof. Let c be such that $p^c \geq Q$, and let $t \geq p^c$ be such that Equation 2 holds.

For every large enough λ such that 2 holds,

$$\begin{aligned}
&\Pr\left[\mathcal{D}(x) = \mathcal{L}(x)\right] && = && (M \text{ decides } \mathcal{L}) \\
&\Pr\left[\mathcal{D}(x) = M(x)\right] && = && (\text{FHE correctness}) \\
&\Pr\left[\mathcal{A}(\widehat{x}) = \mathsf{F}_{t,\lambda}(\widehat{x}) \;\middle|\; \widehat{x} \leftarrow \widehat{\mathsf{D}}_M(x)\right] && \geq && (\text{FHE semantic security}) \\
&\Pr\left[\mathcal{A}(\widehat{x}) = \mathsf{F}_{t,\lambda}(\widehat{x}) \mid \widehat{x} \leftarrow \mathsf{D}_{t,\lambda}\right] - \frac{0.01}{p(\lambda)} && \geq && (\text{Equation 2}) \\
&\frac{1}{p(\lambda)} - \frac{0.01}{p(\lambda)} && = && \\
&\frac{0.99}{p(\lambda)} \quad . &&&&
\end{aligned}$$

In addition,

$$\begin{aligned}
&\Pr\left[\mathcal{D}(x) \notin \{\bot, \mathcal{L}(x)\}\right] && = && \text{(FHE correctness)}\\
&\Pr\left[\mathcal{A}(\widehat{x}) \notin \{\bot, \mathsf{F}_{t,\lambda}(\widehat{x})\} \;\middle|\; \widehat{x} \leftarrow \widehat{\mathsf{D}}_M(x)\right] && \leq && \text{(FHE semantic security)}\\
&\Pr\left[\mathcal{A}(\widehat{x}) \notin \{\bot, \mathsf{F}_{t,\lambda}(\widehat{x})\} \mid \widehat{x} \leftarrow \mathsf{D}_{t,\lambda}\right] + \frac{0.01}{p(\lambda)} && \leq && (\mathcal{A} \text{ is } \delta \text{ faulty})\\
&\delta(\lambda) + \frac{0.01}{p(\lambda)} && \leq &&\\
&\frac{0.91}{p(\lambda)} \ . &&&&
\end{aligned}$$

In particular, the claim holds with $\alpha = 0.99$ and $\beta = \frac{0.99}{0.91}$. □

So, by claim 4.5, taking $k(\lambda) = C \cdot p(\lambda)$ for a large enough constant C (and assuming w.l.o.g that $p(\lambda) \geq \lambda$), it follows by a Chernoff bound that for any $x \in \{0,1\}^\lambda$,

$$\Pr\left[\mathcal{B}'(x) \neq \mathcal{L}(x)\right] < 2^{-p(\lambda)} \leq 2^{-\lambda} \ .$$

The Size of $\mathcal{B}$. First, note each invocation of $\mathcal{D}$ can be done in fixed polynomial size $\ell(\lambda)$, which depends only on the size $p(\lambda)$ of $\mathcal{A}$ and fixed polynomial running time of the FHE algorithms. Overall the size of $\mathcal{B}'$, and hence also of the derandomized $\mathcal{B}$, is $q = \ell \cdot k = O(\ell \cdot p)$.

Extending to Space-Efficient Puzzles (Proof of theorem 4.4). Let s be the space bound given by Assumption 4.2. First, by the complexity preservation of homomorphic evaluation F^s, which evaluates $U_{t,s}$, can be computed in space $s \cdot \mathrm{poly}(\log t, \lambda)$ space. The proof of security is the same, with the exception that we consider $\mathcal{L} \in \mathbf{DTIME}(Q) \cap \mathbf{DSPACE}(s)$.

The Uniform Case (Proof of theorem 4.2). Recall that here we focus on puzzles against uniform solvers (that are not necessarily space-efficient). We show how to remove Assumption assumption 4.1, on which we relied in the non-uniform case. Instead, we rely on the following (unconditional) hierarchy theorem for slightly non-uniform probabilistic time.

Theorem 4.6 ([Bar02])**.** *For any polynomial q, there exists a polynomial Q such that*

$$\boldsymbol{BPTIME}(Q)_{/\log n} \not\subseteq \boldsymbol{ioBPTIME}(q)_{/\log n} \ .$$

The differences from theorem 4.3 is only in the security proof. The security proof has a similar outline. Below, the parts that are similar to the proof of theorem 4.3 are in grey, whereas only the new parts are in black.

Fix any polynomial $p = p(\lambda)$ and assume toward contradiction that for every constant c, there exists a polynomially bounded function $t \geq p^c$, $\delta \leq \frac{0.9}{p}$, and a (δ, D_t)-faulty F_t-solver $\mathcal{A}$ of running time at most p, such that for infinitely many λ,

$$\Pr\left[\mathcal{A}(x) = \mathsf{F}_{t,\lambda}(x) \mid x \leftarrow \mathsf{D}_{t,\lambda}\right] > \frac{1}{p(\lambda)} \ . \tag{3}$$

We prove that there exists a polynomial q such that for any polynomial Q and language $\mathcal{L} \in \mathbf{BPTIME}(Q)_{/\log n}$, there exists a q-time probabilistic algorithm $\mathcal{B}$ that decides $\mathcal{L}$ on infinitely many input lengths, thereby contradicting theorem 4.6.

Let Q be a polynomial and let $\mathcal{L} \in \mathbf{BPTIME}(Q)_{/\log n}$. Also, let M be a Q-time probabilistic Turing machine with non-uniform description of size $O(\log \lambda)$, for inputs of size λ, that decides $\mathcal{L}$ with error $< 1/3$.

Claim 4.7. *Assuming one-way functions (and in particular, assuming FHE), there exists a polynomial Q' and a Q'-time probabilistic M', with non-uniform description of size $O(\log \lambda)$, and randomness of size λ, that decides $\mathcal{L}$ with error $\frac{0.01}{p(\lambda)}$, on all inputs of large enough size λ.*

Proof Sketch. First consider an amplified version M'' of M with error $\frac{0.01}{2p(\lambda)}$. M' also has non-uniform description of size $O(\log\lambda)$ and polynomial running time, but may use randomness of polynomial size $\gg\lambda$. Then to get M', derandomize M'' using a cryptographic pseudorandom generator with seed length λ (which follows from one-way functions [HILL99]). Then by pseudorandomness, for all large enough λ, the error of M' is at most $\frac{0.01}{2p(\lambda)}+\frac{0.01}{2p(\lambda)}=\frac{0.01}{p(\lambda)}$, and it has some polynomial running time $Q'(\lambda)$, as required. □

Consider a sampler $\widehat{\mathsf{D}}_{M'}(x)$ that given $x\in\{0,1\}^\lambda$, samples $\mathsf{pk},\mathsf{sk}\leftarrow\mathsf{Gen}(1^\lambda)$, $r\leftarrow\{0,1\}^\lambda$, and outputs an encryption $\widehat{x}\leftarrow\mathsf{Enc}_{\mathsf{pk}}(x')$, where $x'=(M'_r,x)$, padded to size 3λ, and M'_r is M' with randomness r hardwired (note that M' has description of size $\lambda+O(\log\lambda)$).

We construct a probabilistic Turing machine $\mathcal{B}$ with non-uniform description of size $O(\log\lambda)$ that decides $\mathcal{L}$ (with error $<1/3$) for infinitely many λ. (The time of $\mathcal{B}$ will be a polynomial $q(\lambda)$ independent of Q.)

On input $x\in\{0,1\}^\lambda$, $\mathcal{B}$ repeats the following process $\mathcal{D}$, $k(\lambda)$ times, for some polynomial $k(\cdot)$ to be determined later. $\mathcal{D}(x)$ samples $\widehat{x}_i\leftarrow\widehat{\mathsf{D}}_{M'}(x)$, runs $\mathcal{A}$ on $\widehat{x}_i$, and obtains its output y_i. If $y_i\neq\perp$, $\mathcal{D}$ outputs $b_i=\mathsf{Dec}_{\mathsf{sk}}(y)$, and otherwise outputs $b_i=\perp$. Finally, $\mathcal{B}$ outputs the majority among the values $\{b_1,\ldots,b_k\}\setminus\{\perp\}$ generated by $\mathcal{D}$.

Claim 4.8. *There exists a polynomial $t\geq Q'$ and constants $0<\alpha<1<\beta$, such that for infinitely many λ it holds that for any $x\in\{0,1\}^\lambda$,*

$$\Pr\left[\mathcal{D}(x)=\mathcal{L}(x)\right]\geq\max\left\{\frac{\alpha}{p(\lambda)},\beta\cdot\Pr\left[\mathcal{D}(x)\notin\{\perp,\mathcal{L}(x)\}\right]\right\}\ .$$

Proof. Let c be such that $p^c\geq Q'$, and let $t\geq p^c$ be such that Equation 2 holds.

For every large enough λ such that 3 holds,

$$\begin{aligned}
&\Pr\left[\mathcal{D}(x)=\mathcal{L}(x)\right] && = && \text{(FHE correctness)}\\
&\Pr\left[\mathcal{A}(\widehat{x})=\mathsf{F}_{t,\lambda}(\widehat{x})\;\middle|\;\widehat{x}\leftarrow\widehat{\mathsf{D}}_{M'}(x)\right]-\Pr\left[M'(x)\neq\mathcal{L}(x)\right] && \geq && (M'\text{ error probability})\\
&\Pr\left[\mathcal{A}(\widehat{x})=\mathsf{F}_{t,\lambda}(\widehat{x})\;\middle|\;\widehat{x}\leftarrow\widehat{\mathsf{D}}_{M'}(x)\right]-\frac{0.01}{p(\lambda)} && \geq && \text{(FHE semantic security)}\\
&\Pr\left[\mathcal{A}(\widehat{x})=\mathsf{F}_{t,\lambda}(\widehat{x})\mid\widehat{x}\leftarrow\mathsf{D}_{t,\lambda}\right]-\frac{0.01}{p(\lambda)}-\frac{0.01}{p(\lambda)} && \geq && \text{(Equation 3)}\\
&\frac{0.98}{p(\lambda)}\ .
\end{aligned}$$

In addition,

$$\begin{aligned}
&\Pr\left[\mathcal{D}(x)\notin\{\perp,\mathcal{L}(x)\}\right] && = && \text{(FHE correctness)}\\
&\Pr\left[\mathcal{A}(\widehat{x})\notin\{\perp,\mathsf{F}_{t,\lambda}(\widehat{x})\}\;\middle|\;\widehat{x}\leftarrow\widehat{\mathsf{D}}_{M'}(x)\right]+\Pr\left[M'(x)\neq\mathcal{L}(x)\right] && \leq && (M'\text{ error probability})\\
&\Pr\left[\mathcal{A}(\widehat{x})\notin\{\perp,\mathsf{F}_{t,\lambda}(\widehat{x})\}\;\middle|\;\widehat{x}\leftarrow\widehat{\mathsf{D}}_{M'}(x)\right]+\frac{0.01}{p(\lambda)} && \leq && \text{(FHE semantic security)}\\
&\Pr\left[\mathcal{A}(\widehat{x})\notin\{\perp,\mathsf{F}_{t,\lambda}(\widehat{x})\}\mid\widehat{x}\leftarrow\mathsf{D}_{t,\lambda}\right]+\frac{0.01}{p(\lambda)}+\frac{0.01}{p(\lambda)} && \leq && (\mathcal{A}\text{ is }\delta\text{ faulty})\\
&\delta(\lambda)+\frac{0.02}{p(\lambda)} && \leq &&\\
&\frac{0.92}{p(\lambda)}\ .
\end{aligned}$$

In particular, the claim holds with $\alpha=0.99$ and $\beta=\frac{0.98}{0.92}$. □

So, by claim 4.8, taking $k(\lambda) = C \cdot p(\lambda)$ for a large enough constant C (and assuming w.l.o.g that $p(\lambda) \geq \lambda$), it follows by a Chernoff bound that for any $x \in \{0,1\}^\lambda$,

$$\Pr\left[\mathcal{B}(x) \neq \mathcal{L}(x)\right] < 2^{-p(\lambda)} \leq 2/3 \ .$$

The Time and Non-Uniformity of $\mathcal{B}$. First, note that each invocation of $\mathcal{D}$ can be done in fixed polynomial time $\ell(\lambda)$, which depends only on the size $p(\lambda)$ of $\mathcal{A}$ and fixed polynomial running time of the FHE algorithms. Overall the running time of $\mathcal{B}$ is $q = \ell \cdot r = O(\ell \cdot p)$. The non-uniform description of $\mathcal{B}$ is dominated by that of M, and hence is of size $O(\log \lambda)$, as required.

References

[AL18] Prabhanjan Ananth and Alex Lombardi. Succinct garbling schemes from functional encryption through a local simulation paradigm. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 455–472. Springer, 2018.

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 106–115. IEEE Computer Society, 2001.

[Bar02] Boaz Barak. A probabilistic-time hierarchy theorem for "slightly non-uniform" algorithms. In *Randomization and Approximation Techniques, 6th International Workshop, RANDOM 2002, Cambridge, MA, USA, September 13-15, 2002, Proceedings*, pages 194–208, 2002.

[BCG+18] Nir Bitansky, Ran Canetti, Sanjam Garg, Justin Holmgren, Abhishek Jain, Huijia Lin, Rafael Pass, Sidharth Telang, and Vinod Vaikuntanathan. Indistinguishability obfuscation for RAM programs and succinct randomized encodings. *SIAM J. Comput.*, 47(3):1123–1210, 2018.

[BCH+22] Nir Bitansky, Arka Rai Choudhuri, Justin Holmgren, Chethan Kamath, Alex Lombardi, Omer Paneth, and Ron D. Rothblum. PPAD is as hard as LWE and iterated squaring. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part II*, volume 13748 of *Lecture Notes in Computer Science*, pages 593–622. Springer, 2022.

[BG08] Boaz Barak and Oded Goldreich. Universal arguments and their applications. *SIAM J. Comput.*, 38(5):1661–1694, 2008.

[BG20] Nir Bitansky and Idan Gerichter. On the cryptographic hardness of local search. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 6:1–6:29. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[BGJ+16] Nir Bitansky, Shafi Goldwasser, Abhishek Jain, Omer Paneth, Vinod Vaikuntanathan, and Brent Waters. Time-lock puzzles from randomized encodings. In Madhu Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 345–356. ACM, 2016.

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Kalai. Non-interactive delegation and batch np verification from standard computational assumptions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 474–482, 2017.

[BKP18] Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. Multi-collision resistance: a paradigm for keyless hash functions. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 671–684. ACM, 2018.

[BS23] Nir Bitansky and Tomer Solomon. Bootstrapping homomorphic encryption via functional encryption. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 17:1–17:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[CJJ21a] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for np from standard assumptions. In *Annual International Cryptology Conference*, pages 394–423. Springer, 2021.

[CJJ21b] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for $\mathcal{P}$ from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021.

[CKV10] Kai-Min Chung, Yael Tauman Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 483–501. Springer, 2010.

[DGKV22] Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-np and applications. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1057–1068. IEEE, 2022.

[DGS09] Yi Deng, Vipul Goyal, and Amit Sahai. Resolving the simultaneous resettability conjecture and a new non-black-box simulation strategy. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 251–260. IEEE Computer Society, 2009.

[DN92] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 139–147. Springer, 1992.

[FGHS21] John Fearnley, Paul W. Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. In *STOC*, pages 46–59. ACM, 2021.

[GPS16] Sanjam Garg, Omkant Pandey, and Akshayaram Srinivasan. Revisiting the cryptographic hardness of finding a nash equilibrium. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 579–604. Springer, 2016.

[GS18] Sanjam Garg and Akshayaram Srinivasan. A simple construction of io for turing machines. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 425–454. Springer, 2018.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 99–108. ACM, 2011.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[JKKZ21] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. Snargs for bounded depth computations and PPAD hardness from sub-exponential LWE. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 708–721. ACM, 2021.

[JPY88] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79 – 100, 1988.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, editors, *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732. ACM, 1992.

[KLVW22] Yael Tauman Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and ram delegation. *Cryptology ePrint Archive*, 2022.

[KNTY19] Fuyuki Kitagawa, Ryo Nishimaki, Keisuke Tanaka, and Takashi Yamakawa. Adaptively secure and succinct functional encryption: Improving security and efficiency, simultaneously. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 521–551. Springer, 2019.

[KPY19] Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1115–1124. ACM, 2019.

[KPY20] Yael Tauman Kalai, Omer Paneth, and Lisa Yang. Delegation with updatable unambiguous proofs and ppad-hardness. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 652–673. Springer, 2020.

[KY18] Ilan Komargodski and Eylon Yogev. On distributional collision resistant hashing. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 303–327. Springer, 2018.

[Mic94] Silvio Micali. CS proofs (extended abstracts). In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 436–453. IEEE Computer Society, 1994.

[Pap94] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994.

[PP22] Omer Paneth and Rafael Pass. Incrementally verifiable computation via rate-1 batch arguments. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1045–1056. IEEE, 2022.

[PR08] Rafael Pass and Alon Rosen. Concurrent nonmalleable commitments. *SIAM J. Comput.*, 37(6):1891–1925, 2008.

[RSW96] Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, MIT, 1996.

[RV22] Ron D. Rothblum and Prashant Nalini Vasudevan. Collision-resistance from multi-collision-resistance. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 503–529. Springer, 2022.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2008.

[WW22] Brent Waters and David J. Wu. Batch arguments for sfnp and more from standard bilinear group assumptions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 433–463. Springer, 2022.