On Zero-Testable Homomorphic Encryption and Publicly Verifiable Non-Interactive Arguments [*]

Omer Paneth[†] Guy N. Rothblum[‡]

April 3, 2018

Abstract

We define and study *zero-testable homomorphic encryption* (ZTHE) – a *semantically secure*, somewhat homomorphic encryption scheme equipped with a *weak zero test* that can identify *trivial zeros*. These are ciphertexts that result from homomorphically evaluating an arithmetic circuit computing the zero polynomial over the integers. This is a relaxation of the (strong) zero test provided by the notion of graded encodings, which identifies all encodings of zero.

We show that ZTHE can suffice for powerful applications. Based on any ZTHE scheme that satisfies the additional properties of correctness on adversarial ciphertexts and multi-key homomorphism, we construct publicly verifiable non-interactive arguments for delegating computation. Such arguments were previously constructed from indistinguishability obfuscation or based on so-called knowledge assumptions. The arguments we construct are adaptively sound, based on an efficiently falsifiable assumption, and only make black-box use of the underlying cryptographic primitives.

We also show that a ZTHE scheme that is sufficient for our application can be constructed based on an efficiently-falsifiable assumption over so-called "clean" graded encodings.

[*]This work subsumes an earlier report posted on the Cryptology ePrint Archive [PR14]. The current version features new results and addresses correctness issues with the earlier report.

[†]MIT. Research supported in part by NSF Grants CNS-1350619, CNS-1414119 and CNS-1413920, by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226 and W911NF-15-C-0236 and by Simons Investigator Award Agreement Dated 6-5-12.

[‡]Weizmann Institute of Science.

Contents

1 Introduction **1**
1.1 Non-Interactive Arguments . . . 2
1.2 Our Results in More Details . . . 4
1.2.1 Zero-testable homomorphic encryption. . . . 4
1.2.2 Graded encoding. . . . 5
1.3 Non-Interactive Arguments from Zero-Testable Homomorphic Encryption . . . 6
1.4 Zero-Testable Homomorphic Encryption from Graded Encodings . . . 8
1.5 Organization . . . 9

2 Preliminaries **10**
2.1 Arithmetic Circuits. . . . 10
2.2 Multi-linear Extension. . . . 11
2.3 Publicly-Verifiable Non-Interactive Arguments . . . 11

3 Zero-Testable Homomorphic Encryption **12**
3.1 Homomorphic Encryption . . . 12
3.2 Correctness for Adversarial Ciphertexts . . . 13
3.3 Zero Test . . . 14
3.4 Weak Decryption . . . 14
3.5 Multi-Key Zero-Testable Homomorphic Encryption . . . 15

4 A Non-Interactive Argument **17**
4.1 Overview . . . 17
4.2 Adaptive Local-Assignment Generator . . . 19
4.3 The Core Protocol . . . 20
4.3.1 Construction. . . . 21
4.3.2 Completeness. . . . 23
4.3.3 Adaptive local soundness. . . . 25
4.4 The Augmented Circuit . . . 27
4.4.1 Transformation outline. . . . 28
4.5 The Final Protocol . . . 31
4.5.1 Construction. . . . 31
4.5.2 Analysis. . . . 31

5 Zero-Testable Homomorphic Encryption from Graded Encodings **32**
5.1 Graded Encodings . . . 32
5.2 Construction . . . 36
5.3 Analysis . . . 40
5.4 On Noisy Graded Encoding . . . 45

6 Acknowledgements **46**

1 Introduction

Recent breakthroughs in the study of fully homomorphic encryption [Gen09] and program obfuscation [GGH$^+$13b] have revolutionized the foundations of cryptography. Fully homomorphic encryption (FHE) allows arbitrary polynomial-time computations to be performed "homomorphically" on encrypted data, while ensuring that semantic security is maintained and *nothing* about the data can be learned. While this powerful security guarantee enables important applications, other scenarios require more fine-grained control: allowing some information about the data to be exposed, while other information remains hidden. Multilinear maps [BS02] and graded encodings [GGH13a] are basic building blocks that have proven to be incredibly useful in such scenarios. Intuitively, a graded encoding scheme is a *somewhat homomorphic encryption*, supporting homomorphic evaluation of low-degree algebraic computations, with an additional capability: an efficient *zero test* procedure that publicly identifies encodings of zero. Graded encodings cannot be semantically secure: the zero test procedure leaks *partial information* on the encoded elements. Nevertheless, other information can remain hidden (in particular, inverting the encoding might still be hard). This balance between functionality and security makes the notion of graded encoding incredibly useful for computing on encrypted data, with applications such as indistinguishability obfuscation and functional encryption [GGH$^+$13b, GGHZ16].

While homomorphic encryption can by based on the Learning with Errors assumption [BV11, GSW13], the situation for graded encodings is less clear. Analyzing the security of existing candidates and designing new ones are central challenges [GGH13a, CLT15, GGH15, CHL$^+$15, HJ16, MSZ16, GMM$^+$16].

Zero-testable homomorphic encryption. In this work we define and study a new relaxation of graded encodings that we call zero-testable (somewhat) homomorphic encryption (ZTHE). A ZTHE is a semantically secure somewhat homomorphic encryption scheme equipped with a *weak zero test* that can only identify *trivial zeros*. These are ciphertexts that result from homomorphically evaluating an arithmetic circuit computing the zero polynomial over $\mathbb{Z}$. The weak zero test should accept such trivial zeros, but reject ciphertexts that encrypt non-zero values.

Importantly, an efficient weak zero test poses no contradiction to semantic security, since it does not allow to distinguish between encryptions of two different values. Given a ciphertext c it is possible to homomorphically evaluate a circuit P on c and test if the result is a trivial zero. However, this does not give any information on the value encrypted in c, since the zero test only required to pass if P vanishes on *all* values. Intuitively, the zero test is giving information on the evaluated computation P rather then on the ciphertext c. Indeed, semantic security implies that if P only vanishes on some values, then even if the evaluated ciphertext encrypts zero it will not pass the weak zero test (except with negligible probability). Otherwise, the zero test would have revealed information on the original encrypted evaluation point.

From ZTHE to delegation. The main technical result in this work demonstrates that ZTHE can suffice for powerful applications. Based on any ZTHE scheme that satisfies the additional properties of correctness on adversarial ciphertexts and multi-key homomorphism (we elaborate on these additional properties below), we construct *publicly verifiable non-interactive arguments for delegating computation*. Such arguments were previously constructed from indistinguishability obfuscation or based on so-called knowledge assumptions. Our construction follows a new approach and has important properties, such as adaptive soundness, reduction to an efficiently falsifiable assumption, and black-box use of the underlying cryptographic primitives. We note that the additional properties we assume (adversarial correctness and multi-key homomorphism) make ZTHE incomparable to "vanilla" graded encodings: the weak zero test assumption is more relaxed than the strong zero test of graded encodings schemes, but we require a stronger correctness property (namely correctness on adversarially generated ciphertexts).

ZTHE Candidate. We study the feasibility of constructing ZTHE. First, we observe that several existing somewhat homomorphic encryption schemes [Gen09, vDGHV10] admit a simple weak zero test. These schemes, however, do not satisfy the additional properties required for our non-interactive arguments. We construct ZTHE that is sufficient for our application based on an efficiently-falsifiable assumption over graded encodings with strong properties such as adversarial correctness. Our construction cannot be instantiated based on the existing graded encoding candidates (so-called "clean" graded encodings [Zim15, LV16] do guarantee these stronger properties). We leave the question of ZTHE instantiations as an important open problem and hope it will lead to new and improved deletion protocols based on weaker assumptions, as well as other applications.

Organization. In the rest of this introduction we elaborate on our results and techniques. Section 1.1 gives background on non-interactive arguments and discusses our main technical result, a construction of non-interactive arguments from ZTHE. In Section 1.2 we present our results in more detail. The construction of non-interactive arguments from ZTHE is described in Section 1.3. The construction of ZTHE from graded encodings is described in Section 1.4. The full details are given in the body.

1.1 Non-Interactive Arguments

Background. The power of efficiently verifiable proof systems is a foundational issue in the study of computation. A central goal is constructing proof systems that can be used by a powerful prover to convince a weak verifier of the correctness of a complex computational statement, usually framed as proving membership of an input x in a language $\mathcal{L}$. Beyond its foundational importance in the theory of computation, this question has real-world applications, such as *delegating computation*. In this setting, a powerful server (playing the role of the prover) can run a complex computation for a much weaker client (playing the role of the verifier), and provide a proof of the output's correctness.

A similar question was raised by Babai, Lund, Fortnow and Szegedy [BFLS91] in the PCP setting. Kilian [Kil92] and Micali [Mic94] gave the first candidate scheme for delegating computation. The question re-emerged in the theoretical literature in the work of Goldwasser, Kalai and Rothblum [GKR08], and became the focus of a rich body of research spanning theory and systems. See the recent survey by Walfish and Blumberg [WB13].

A "holy grail" for delegating computations is *fully non-interactive proofs*, comprised of a single message sent from the prover to the verifier with unconditional soundness, as in classic NP or Merlin-Arthur proofs. Unfortunately, there are serious barriers to constructing such proofs for delegating general deterministic computations (in particular, they imply Merlin-Arthur speedups for deterministic computations). Thus, a body of research has focused on computationally sound proofs in the common reference string model, where:

1. Soundness is only required to hold against *efficient* cheating provers. Computationally sound proof systems are commonly called *argument systems*.

2. There is a (public) *common reference string* (CRS), generated in advance by a trusted authority (or the verifier herself). This CRS can be used (repeatedly) by different parties to verify proofs. The prover and the verifier both have access to the CRS, but neither has access to the secret coins used to generate the CRS.

We focus on non-interactive argument systems for polynomial-time computations, where the verifier should be super-efficient (nearly-linear in the input length), and the honest prover should run in polynomial time. Non-interactive arguments are especially attractive for delegating computation, as any untrusted server can simply use the CRS to generate proofs and send them off (non-interactively and asynchronously), to be verified at the clients' convenience. We refer to such a system as a *publicly verifiable non-interactive argument for delegating computation*. For the remainder of this work, we use the term *non-interactive argument* as shorthand.

Prior works on non-interactive arguments. In his seminal work, Micali [Mic94] gave the first construction of non-interactive arguments in the random oracle model. However, instantiating random oracle model constructions in a provably secure way is notoriously difficult, and often impossible [CGH04, GW11]. A rich body of research has aimed to construct non-interactive arguments in the plain model led to a variety of beautiful constructions based on strong cryptographic assumptions.

One line of works based non-interactive arguments on non-falsifiable[1] *knowledge assumptions* such as the *knowledge of exponent assumption* in bilinear groups [Gro10, Lip12, DFH12, GGPR13, BCI$^+$13, BCCT13]. A recent sequence of works [SW14, BGL$^+$15, CHJV14, KLW14] show how to base non-interactive arguments on *indistinguishability obfuscation* (IO). Based on standard assumptions such as somewhat-homomorphic encryption or private information retrieval schemes, the works of [KRR13, KRR14, BHK16] achieve the weaker notion of *designated-verifier arguments*. These are two-message arguments where, in the first message, the verifier samples the CRS and sends it to the prover. The secret coins used to sample the CRS are required to verify the proof sent in the second message.

This work. Our main technical result is a construction of non-interactive arguments from any ZTHE with the additional properties mentioned above (see Section 1.2). Our construction follows a different approach from previous works and leverages ideas and techniques that were previously used only in the context of designated-verifier arguments [KRR14, BHK16], such as efficient probabilistically checkable proofs and no-signaling soundness. As a result, our non-interactive arguments have some notable advantages compared to previous works:

- **Efficiently falsifiable assumptions.** Our arguments are based on the semantic security of the underlying ZTHE - an efficiently falsifiable assumption. Moreover, in our candidate construction of ZTHE from graded encodings, we further base semantic security of the ZTHE on a simple and efficiently falsifiable assumption on the graded encodings. Taken together, we can base soundness of the argument system on a falsifiable assumption on graded encodings.

  In contrast, the constructions of publicly verifiable non-interactive argument are based on assumptions that are not efficiently falsifiable. IO was recently constructed from simpler primitives such as multi-linear maps or functional encryption. However, these construction involve a sub-exponential security loss. While many applications of IO can be based directly on polynomially secure functional encryption, currently non-interactive arguments still require the full power of IO. For more information on this line of work, see [GPSZ17] and references therein.

  We note that for any particular non-interactive argument candidate, the assumption that the candidate is secure is efficiently falsifiable. Therefore, our focus will be on falsifiable assumptions that are elementary and natural compared to the tautological assumption that the candidate is secure.

- **Adaptive soundness.** The soundness of our non-interactive arguments is adaptive: it holds even when the statement proven is chosen as a function of the CRS. Adaptive soundness is required in many applications, and it is especially important in settings where the CRS is set "once and for all".

  We note that any sound argument can be turned into an adaptively sound one via "complexity leveraging". However, this reduction incurs an exponential loss in security, and therefore cannot be based on efficiently falsifiable assumptions.

- **Black-box construction.** In contrast to all previous construction of non-interactive arguments, our construction makes only black-box use of the underlying cryptographic primitives.[2] Under-

[1] A "falsifiable" assumption [Nao03] is one that can be efficiently refuted. Falsifiability is a basic "litmus test" for cryptographic assumptions.

[2] One exception is instantiating Micali's random oracle construction with a cryptographic hash function. However, beyond assuming this construction is secure, we do not know how to reduce its security to a simpler assumption.

standing the feasibility and limitation of black-box constructions in cryptography is the subject of a rich body of work motivated both by theoretical interest as well as efficiency considerations.

1.2 Our Results in More Details

In this section we present our results in more details. We start by describing the basic notion of zero testable homomorphic encryption and the additional properties we consider.

1.2.1 Zero-testable homomorphic encryption.

A homomorphic encryption is a semantically secure public key encryption equipped with a public evaluation algorithm that adds, subtracts and multiplies values homomorphically "under the encryption". We focus on somewhat homomorphic encryption that only supports homomorphic evaluation of polynomial-size arithmetic circuits of logarithmic degree. That is, of degree $c \cdot \log \lambda$ for any constant c, where λ is the security parameter. We require that ciphertexts are succinct: their size is bounded by some fixed polynomial in λ that is independent of c.

A zero-testable somewhat homomorphic encryption (ZTHE) has an additional zero test procedure that takes a ciphertext and tests if it is a trivial zero. In more detail, we consider the homomorphic evaluation of a circuit P over freshly encrypted ciphertexts $c_1, \ldots, c_n$, resulting in the evaluated ciphertext c. If the polynomial computed by P is identically zero over $\mathbb{Z}$, then we require that c passes the zero test. We also require that a ciphertext c' that decrypts to a non-zero value does not pass the zero-test. If c decrypts to zero, but it is not a trivial zero, we make no requirement on the outcome of the zero test. However, as discussed above, it follows from the semantic security of the encryption that such a ciphertext should not pass the zero test. Moreover, we note that even if P vanishes on all boolean inputs, but it is not identically zero as a polynomial over $\mathbb{Z}$, we still expect the zero test to fail. Otherwise, the zero test can be used to efficiently decide the satisfiability of P.

We further study the following additional properties of ZTHE, which we use in our construction of non-interactive arguments:

Multi-key evaluation. In multi-key homomorphic encryption, introduced by López-Alt et al. [LTV12], homomorphic computation can be executed over ciphertexts encrypted under different keys. To ensure semantic security, decrypting the result requires all secret keys. We use ZTHE for three keys. That is, it is possible to homomorphically compute over ciphertexts encrypted under at most three different keys, and to run a weak zero test on the result. Importantly, a system can generate ciphertext under an unbounded number of keys and any three of them can be combined in a homomorphic computation. The encryption may also use shared public parameters to generate all keys.

Correctness for adversarially generated ciphertexts. We require that an efficient adversary, given the public key, cannot generate a pair of ciphertexts that result in an evaluation error. A pair of ciphertexts c_1, c_2 cause an evaluation error if computing a homomorphic operation $\star$ over c_1, c_2 and decrypting the evaluated ciphertext c give a different result than decrypting c_1 and c_2 and computing $\star$ on the decrypted values. If c_1 and c_2 are generated honestly, this follows from the standard correctness guarantee of the encryption. However, we require correctness even when the ciphertext are not generated honestly. Note that the zero test is only required to accept honest ciphertexts that are trivially zero. However, even a malformed ciphertext that decrypts to a non-zero value should make the zero test reject.

In known constructions of somewhat homomorphic encryption, there exist invalid ciphertexts that do not represent an encryption of any value. To account for such candidates, we allow the decryption algorithm to fail. If c_1 or c_2 are invalid (fail to decrypt) we require that the evaluated ciphertext c is invalid as well. If both c_1 and c_2 are valid, we require that c is either invalid or it decrypts to the correct value.

Theorem 1.1 (Informal)**.** *Assuming a 3-key zero-testable somewhat homomorphic encryption scheme with correctness for adversarially-generated ciphertexts, there exists an adaptively-secure publicly-verifiable non-interactive argument for delegating all polynomial time computations. The non-interactive argument uses the encryption scheme as a black box.*

Instantiations: discussion. We observe that existing constructions of somewhat homomorphic encryption, such as the ones in [Gen09, vDGHV10], already support zero testing: simply test if the ciphertext is zero in the ring of ciphertexts. More generally, in any encryption scheme where ciphertexts are elements of some ring, and the homomorphic operations on ciphertext identify with the ciphertext-ring operations, every trivial zero is represented by the zero of the ciphertext ring. While these construction satisfy the weak zero test requirement, they do not seem to support the additional properties stated above.

Following the observations in [LTV12, GHV10, HRSV11], any homomorphic encryption scheme that supports homomorphic computations of sufficiently large degree can be generically modified to satisfy both multi-key evaluation for a constant number of keys and correctness for adversarially generated ciphertexts. This transformation, however, may not preserve the weak zero test property. Roughly speaking, the generic transformation is based on the idea of bootstrapping [Gen09], where the evaluated circuit is modified to include the decryption circuit of the scheme itself. Now, even if we evaluate a circuit computing the zero polynomial, the modified circuit, which now runs the scheme's decryption circuit, will not be identically zero.

We show that ZTHE satisfying both additional properties can be constructed from graded encodings with additional properties described below.

1.2.2 Graded encoding.

A graded encoding is an encoding scheme for elements of a ring. We consider a symmetric graded encoding that supports homomorphic computations of bounded degree Δ. The encoding scheme also features a (strong) zero test that identifies encodings of zero (even non-trivial ones). In Section 1.4 we describe the interface of a graded encoding scheme in more detail.

We consider graded encodings that satisfy a simple and natural decisional assumption.

Assumption 1.2 (Informal)**.** *Given encoded coefficients $\alpha_0, \dots, \alpha_\Delta$ of a random degree Δ polynomial, it is hard to distinguish an encoding of a root from an encoding of a random element.*

Intuitively, this problem should be hard since testing if the given encoding is a root requires a homomorphic computation of degree $\Delta + 1$.

To reduce the semantic security of the ZTHE to the above assumption on the graded encoding, we need the graded encodings to support a re-randomization operation. Intuitively, re-randomizing an encoding results in a new encoding of the same value that is otherwise independent of the original encodings. As in many other applications of graded encoding (for example [GLSW15]), the re-randomization operation is only needed in the reduction and not in the construction. We note that it is possible to avoid the use of randomization, but this requires making a more complicated and less natural (though still efficiently falsifiable) hardness assumption.

Correctness for adversarially generated encodings. In order to construct a ZTHE scheme with correctness for adversarially generated ciphertexts we need to require that the graded encoding themselves have correctness for adversarially generated ciphertexts. This is a non-standard requirement for graded encoding schemes, and it is not required in other applications such as obfuscation (where all encodings are generated by an honest party).

The correctness requirement for adversarially generated encodings is somewhat stronger than in the context of encryption. We require that it is hard to find a pair of valid encodings such that a homomorphic operation on them results in an invalid encoding. In order to support "noisy" candidates, where such an evaluation error always occurs after a large enough number of homomorphic evaluations, we also consider a relaxed requirement. Intuitively, it should be possible to publicly test that the level of noise in an adversarially generated encoding is low. If we determine that an encoding has low noise, it should support a large number of homomorphic operation without an error.

Theorem 1.3 (Informal)**.** *Assuming a graded encoding scheme satisfying Assumption 1.2, there exists a $O(1)$-key zero-testable somewhat homomorphic encryption scheme. Moreover, if the graded encoding scheme is correct for adversarially generated encodings, then the encryption scheme is correct for adversarially generated ciphertexts.*

Instantiations: discussion. The existing constructions of graded encodings [GGH13a, CLT15, GGH15] that support re-randomization do not satisfy our hardness assumption [GGH13a, CHL$^+$15, HJ16]. We don't know if in existing constructions of graded encodings it is possible to publicly test for low noise level. One potential strategy to implement such a test would be to combine the re-randomization and zero test operations. We note that so-called "clean" graded encoding schemes (see for example [Zim15, LV16]), where every element has a unique encoding, trivially satisfy correctness for adversarially generated encodings, and support re-randomization.

1.3 Non-Interactive Arguments from Zero-Testable Homomorphic Encryption

Our construction is based on ideas developed in the context of designated-verifier arguments.

Designated-verifier arguments. Aiello *et al* [ABOR00] suggested the following approach to constructing designated verifier arguments: The prover computes a probabilistically checkable proof (PCP) for the statement. The verifier's message contains PCP queries, encrypted using an FHE scheme, where each query is encrypted under a different key. The prover computes the PCP answers homomorphically, and the verifier decrypts and verifies the answers. The hope was that since a cheating prover couldn't tailor its answer to one query depending on other queries's values, the argument would inherit the PCP's soundness. Dwork *et al.* [DLN$^+$04, DNR16] showed obstacles to proving this construction's soundness. Nonetheless, Kalai, Raz and Rothblum [KRR14] proved that when the underlying PCP satisfies a strong notion of soundness called *no-signaling* soundness, the suggested arguments are in fact sound.

Leaking information on queries: a failed attempt. A naive attempt to turn the above designated-verifier protocol into a publicly verifiable non-interactive argument would be to place the verifier's encrypted queries in the CRS, and provide some leakage on encrypted queries that allows verifying the evaluated answers, but (somehow) does not compromise the soundness of the protocol. We argue, however, that any such leakage must (inherently) compromise soundness. A cheating prover can begin with an accepting PCP proof, changing it into a rejecting proof one symbol at a time. By observing which of the intermediate proofs makes the verifier reject, the prover can recover the encrypted queries and break soundness.

Our approach: intuition. Our protocol follows the blueprint described above: the CRS contains encrypted queries, and the prover homomorphically evaluates the PCP and sends the evaluated queries as the proof. However, to make the proof publicly verifiable we *do not leak any information about the encrypted queries or their answers*. The main idea is to encrypt the queries with a ZTHE. By executing a sequence of homomorphic evaluations and zero tests on the evaluated ciphertexts in the proof, the verifier *learns information about the PCP proof computed by the prover*, which is sufficient to verify its validity.

Next we elaborate on this idea. We start by giving some background on the PCP system we use.

The BFLS PCP. The PCP of Babai et al. [BFLS91] proves that a given computation accepts its input. The tableau of the computation is translated into a multi-variate low-degree polynomial P_0 and the PCP proof contains all the evaluations of P_0 over some finite field. Testing the validity of the tableau is reduced to testing that P_0 is indeed a low-degree polynomial and that it vanishes on all *boolean* inputs. The proof that P_0 vanishes on all boolean inputs is based on the well-known sum-check protocol. The sum-check proof contains auxiliary polynomials $P_1, \dots, P_m$ and the verifier tests that these polynomials satisfy some local low-degree relations of the form $R(P_i, P_{i+1}) \equiv 0$. These tests are carried out by probing the polynomials on a small number of random inputs and testing that the relations are satisfied.

A sketch of our protocol. As described above, the CRS contains encryptions $c_1, \dots, c_m$ that specify queries to the PCP. Each triplet c_j, c_k, c_ℓ specifies an evaluation point for the polynomials $P_1, \dots, P_m$. For every such triplet, and for every polynomial P_i, the proof contains the homomorphically evaluated answer $d_i = P_i(c_j, c_k, c_\ell)$. To verify the relation $R(P_i, P_{i+1}) \equiv 0$, the verifier homomorphically evaluates $R(d_i, d_{i+1})$ and tests that the evaluation results in a trivial zero . Since the different queries are encrypted under different keys, we use a *multi-key* homomorphic encryption scheme. While the CRS contains encryptions under m different keys, the verifier only computes homomorphically on three keys at a time, therefore we only need 3-key homomorphism.

The proof strategy. Intuitively, if the prover is cheating and $R(P_i, P_{i+1}) \not\equiv 0$ it follows from sematic security that the verifier's zero test fails. Alas, this intuition is fundamentally flawed. A cheating prover may not derive its answers by homomorphically evaluating the low degree polynomials $P_1, \dots, P_m$, or any other polynomial for that matter. Our actual proof strategy is inspired by that of Kalai, Raz and Rothblum [KRR14] and consists of the following steps.

1. Since the encryption is semantically secure, the prover's answers are *no-signaling*, meaning that the decrypted answer to one query gives no information on the other queries values.

2. In the BFLS PCP, it is possible to reconstruct any small subset of entries L of the computation's tableau based on PCP values in some small set of locations $q(L)$. We show that our proof satisfies the following local soundness guarantee: if the verifier's encrypted queries include the locations $q(L)$ and if the verifier accepts the prover's encrypted answers then the reconstructed subset of the tableau is *locally consistent*. That is, it obeys the computation's local constrains. To show that this is the case even when the prover sends malformed answers we use the fact that the encryption scheme is correct for *adversarially generated ciphertext*.

3. By the semantic security of encrypted queries, and by the fact that the protocol is publicly verifiable, we deduce that if the verifier accepts the answers to *any* queries encrypted in the CRS (say the all-0 queries), it would also accept the answers to the to queries $q(L)$, for *every* subset L.

4. It follows that we can turn any convincing prover in our protocol into an algorithm that samples local assignments for any subset L of the computation's tableau that are guaranteed to be *both no-signaling and locally consistent*.

5. Based on the *augmented circuit technique* of [KRR14], we show how to use such a *local-assignment generator* to reconstruct a complete and valid tableau.

We note that our soundness proof is significantly simpler than that of [KRR14]. In particular we only use a striped down version of the BFLS PCP without any low-degree tests, and we do not argue that this PCP has no-signaling soundness. Intuitively, what enables this simplification is that in the publicly-verifiable setting we can move from local consistency for one subset to local consistency on all subsets using semantic security (see Step 3 above) and without using global properties of the PCP.

Proving *adaptive* soundness presents additional challenges. To argue adaptive soundness, we use ideas inspired by the recent work of Brakerski et al. [BHK16], who constructed an adaptively sound arguments in the designated-verifier setting. Roughly, they show how to reconstruct a tableau from any local-assignment generator that can chose the statement adaptively as a function of the subset L.

On the notion of local-assignment generator. The augmented circuit technique as well as the technique of reconstructing the computation's tableau by reading subsets that are no-signaling and locally-consistent originates from the analysis of [KRR14]. The notion of local-assignment generator and the generic transformation from a local-assignment generator to global soundness first appeared in an earlier version of this work [PR14]. Since then the local-assignment generator abstraction played a key role in achieving stronger designated-verifier arguments for RAM computations [KP16] and Batch-NP computations [BHK16], as well as in achieving adaptive soundness [BHK16]. In the current version of this work we use the adaptive local-assignment generator of [BHK16].

1.4 Zero-Testable Homomorphic Encryption from Graded Encodings

We start by describing the interface of a graded encoding scheme in more details. The scheme has public parameters that define a ring R and a maximal degree Δ. The scheme encodes elements in R and supports homomorphic computations up to degree Δ. Every encoding has a level. Freshly generated encodings are of level 1 and level-δ encodings are the result of a degree-δ homomorphic computation. We also refer to the elements of R as level-0 encoding. Following the standard formulation of graded encodings, we do not assume that the ring R is public. Instead, there is a public interface for sampling random level-0 encodings and evaluating the ring operations. We also assume that the public parameters include encodings of the constants 0 and 1 in every level.

The graded encoding supports a (strong) zero test that can publicly identify encodings of zero in any level. It also supports a re-randomization operation that, given an encoding, generates a new random encoding of the same element. For example, re-randomizing an encoding can be used to hide the homomorphic computation that generated it.

The ZTHE scheme. We construct multi-key ZTHE from graded encoding as follows. The scheme's public parameters are the parameters of a graded encoding scheme with degree bound Δ. The secret key is a random ring element $t \in R$ and the corresponding public key is a level-1 encoding of t.

An encryption c of a message $m \in \{0, 1\}$ is given by a random degree-Δ univariate polynomial P such that $P(t) = m$. The ciphertext c consists of level-1 encodings of the $\Delta + 1$ coefficients $\alpha_0, \ldots, \alpha_\Delta$ of P. The semantic security of this encryption follows from Assumption 1.2 that states that even given the public key encoding of t, the encodings in c are indistinguishable from encodings of random elements, independent of m.

Encryption. We need to sample such an encryption using only the public parameters and the public key encoding of t. A naive approach would be to sample all the coefficients of P except for the free coefficient α_0 randomly and then homomorphically compute an encoding of α_0. However, this would result in an encoding in level Δ instead of level 1. Instead we can sample all the coefficients of P as linear functions of t. We sample random ring elements $r_1, \ldots, r_\Delta$ and homomorphically compute encodings of the coefficients

$$\alpha_0 = m - r_1 \cdot t, \quad \ldots \quad , \alpha_i = r_i - r_{i+1} \cdot t, \quad \ldots \quad , \alpha_\Delta = r_\Delta \ .$$

Note that $\alpha_0, \ldots, \alpha_\Delta$ are indeed random subject to $\sum \alpha_i \cdot t^i = m$. Finally, we re-randomize the encoded coefficient to hide the process in which they where sampled (which depends on m).

We note that the re-randomization operation is only used during encryption. In our non-interactive argument the ZTHE encryption procedure is only used to generate the CRS and in the security proof. As noted above, we could avoid the use of re-randomization at the cost of making a more complicated assumption on the graded encoding that implies the CPA security of our encryption scheme in the secret key setting.

Same-Key homomorphic evaluation. Let c_1 and c_2 be ciphertexts encrypting messages m_1 and m_2

respectively under the same secret key t. Let P_1 and P_2 be the polynomials encoded by c_1 and c_2, where

$$P_1(t) = m_1 \quad , \quad P_2(t) = m_2 \ .$$

To evaluate a homomorphic operation $\star \in \{+, -, \times\}$ we homomorphically compute the encoded coefficients of the polynomial $P_1 \star P_2$. Correctness follows since

$$(P_1 \star P_2)(t) = P_1(t) \star P_1(t) = m_1 \star m_2 \ .$$

For addition and subtraction, the homomorphic computation of the new coefficients is a linear operation (over the input coefficients), and the degree of the resulting polynomial is the maximal degree of the two input polynomials. For multiplication, we homomorphically compute a convolution of the input coefficients, and the degree of the resulting polynomial is the sum of the degrees of the input polynomials. Thus, the evaluation of a degree-δ homomorphic computation yields coefficients that are encoded in level-δ of the graded encoding scheme, and the resulting (univariate) polynomial has degree $(\delta \cdot \Delta)$. It follows that the encryption supports degree-Δ homomorphic computations, before the level of encoded coefficient exceeds the degree bound.

Multi-Key homomorphic evaluation. To compute a homomorphic operation $\star$ over ciphertexts c_1, c_2 encrypted under different secret keys t_1, t_2, we homomorphically compute the coefficients of the *bivariate* polynomial $P(x, y) \equiv P_1(x) \star P_2(y)$, where P_1 and P_2 are the polynomials encoded by c_1 and c_2 respectively. In general, a homomorphic computation involving ciphertexts under d different keys will result in a ciphertext encoding a d-variate polynomial. Since the number of coefficients grows exponentially with d, we only support homomorphic computation involving a constant number of keys.

Decryption. To decrypt a ciphertext c, we homomorphically evaluate the polynomial P it encodes on the secret key t. Since the secret key is a level-0 encoding, this homomorphic evaluation does not exceed the degree bound Δ. We then use the graded encoding zero test to compare the evaluated encoding to an encoding of 0 or of 1. If none of the tests succeed decryption fails.

Note that in homomorphic evaluation, the algebraic operation on the plaintexts are evaluated over the ring R. However, since our decryption only obtains *an encoding of* the plaintext, we can only decrypt messages in $\{0, 1\}$ (or more generally, messages taken from a small plaintext space). This is analogous to the behaviour of the additively-homomorphic ElGamal encryption and other schemes [BGN05]. Such decryption is sufficient for our application, where we evaluate arithmetic circuits (over $\mathbb{Z}$) whose outputs are expected to be boolean.

Zero Test. A ciphertext c that results from a homomorphic evaluation of a polynomial that is identically zero always encodes a polynomial $P \equiv 0$. We can test this by using the zero test procedure of the graded encoding, testing that all the encoded coefficient of P are zero. It is also the case that a ciphertext that passes the zero test must encode a polynomial $P \equiv 0$ and therefore it must decrypt to zero.

Correctness for adversarially generated ciphertexts. If the graded encoding scheme is correct even on adversarially generated encodings, we inherit this strong correctness guarantee also for the ciphertext. Note, however, that even a ciphertext that consists of valid encodings may encode a polynomial P such that $P(t) \notin \{0, 1\}$, and therefore fail to decrypt. To deal with this case, we consider an alternative decryption algorithm that is inefficient and can decrypt any value in R. The correctness requirement for adversarially generated ciphertexts is therefore defined with respect to this inefficient decryption procedure. The weaker correctness requirement suffices for proving the computational soundness of the non-interactive argument, even though it considers an inefficient decryption algorithm: once the correctness requirement is guaranteed, the remainder of the soundness proof is information theoretic. .

1.5 Organization

The definition of non-interactive arguments and other preliminaries are given in Section 2. In Section 3 we define the notion of ZTHE and the additional properties we use. Section 4 describes the construction of non-interactive argument from ZTHE. Section 5 describes the construction of ZTHE from graded encodings.

2 Preliminaries

For a sequence $\mathbf{x} = (x_1, \ldots, x_n)$, we denote by $\mathbf{x}_{-i}$ the sequence with the i-th elements removed

$$\mathbf{x}_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, x_n) \ .$$

For a pair of sequences $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_{n'})$ we denote by $\mathbf{x} \,|\, \mathbf{y}$ the concatenated sequence

$$\mathbf{x} \,|\, \mathbf{y} = (x_1, \ldots, x_n, y_1, \ldots, y_{n'}) \ .$$

2.1 Arithmetic Circuits.

We consider arithmetic circuits with binary addition, subtraction and multiplication gates. We only allow use of the constants $\{0, 1\}$.

Degree. For an arithmetic circuit C, the degree (resp. total degree) of C is the individual (resp. total) degree of the formal polynomial computed by C. A degree-1 circuit is said to be multi-linear.

Equivalence. An arithmetic circuit C is said to be identically zero (denoted by $C \equiv 0$) if the formal polynomial computed by C is identically zero over $\mathbb{Z}$. Two arithmetic circuits C_1, C_2 are said to be equivalent (denoted by $C_1 \equiv C_2$) if $C_1 - C_2 \equiv 0$.

Computing boolean functions. An arithmetic circuit C is said to compute a boolean function f if C agrees with f when evaluated over $\mathbb{Z}$. That is, if f takes n inputs, then for every $x \in \{0, 1\}^n$ we have that $f(x) = C(x)$ when C is evaluated over $\mathbb{Z}$.

Fact 2.1. *Let C_1 and C_2 be arithmetic circuits with n inputs wires computing boolean functions f_1 and f_2 respectively.*

1. *The circuit $1 - C_1$ computes the boolean function $1 - f_1$.*
2. *The circuit $C_1 \cdot C_2$ computes the boolean function $f_1 \cdot f_2$.*
3. *If for every $x \in \{0, 1\}^n$, at most one of the values $C_1(x)$ and $C_2(x)$ is non-zero, then the circuit $C_1 + C_2$ computes the boolean function $f_1 + f_2$.*

Circuit restrictions. Let C be an arithmetic circuit with n inputs wires and individual degree δ. For $i \in [n]$ let $C|_{i,0}, \ldots, C|_{i,\delta}$ be the arithmetic circuits with $n - 1$ inputs wires and individual degree δ such that

$$C(x_1, \ldots, x_n) \equiv \sum_{j \in [0,\delta]} C|_{i,j}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \cdot x_i^j \ . \quad (1)$$

For $j > \delta$ let $C|_{i,j}$ denote the identically 0 circuit.

Fact 2.2. *There is an procedure that given an arithmetic circuit C with n inputs wires and individual degree δ and given an index $i \in [n]$ computes $C|_{i,0}, \ldots, C|_{i,\delta}$ in time* $\mathrm{poly}(|C|, \delta)$.

2.2 Multi-linear Extension.

A multi-linear extension of a boolean function f is a multi-linear arithmetic circuit C computing f. Next we describe a multi-linear extension circuit of an arbitrary boolean function f.

Let β_n be the multi-linear arithmetic circuit with $2n$ inputs computing the boolean identity function. That is, for every $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$, $\beta_n(\mathbf{x}, \mathbf{y}) = 1$ if and only if $\mathbf{x} = \mathbf{y}$. The arithmetic circuit β_n is given by the expression

$$\beta_n(x_1, \dots, x_n, y_1, \dots, y_n) = \prod_{i \in [n]} x_i y_i + (1 - x_i)(1 - y_i) \ . \tag{2}$$

We sometimes omit the subscript n when it is clear from the context.

The multi-linear extension of a boolean function f with n inputs is given by the arithmetic circuit

$$C(\mathbf{x}) = \sum_{\mathbf{y} \in \{0,1\}^n} \beta_n(\mathbf{x}, \mathbf{y}) \cdot f(\mathbf{y}) \ . \tag{3}$$

Since for every $\mathbf{x} \in \{0,1\}^n$ there exist only one value of $\mathbf{y} \in \{0,1\}^n$ such that $\beta_n(\mathbf{x}, \mathbf{y}) \neq 0$, it follows by Fact 2.1 that C computes the boolean function f.

2.3 Publicly-Verifiable Non-Interactive Arguments

In this section we define publicly verifiable non-interactive arguments.

Let $\mathcal{U}$ be the universal language such that $(x, \mathsf{T}) \in \mathcal{U}$ for $x = (M, y)$ if and only if the Turing machine M accepts the input y within at most T steps.

Syntax. A a publicly verifiable non-interactive argument scheme for the universal language $\mathcal{U}$ consists of PPT algorithms $(\mathsf{Del.Gen}, \mathsf{Del.P}, \mathsf{Del.V})$ with the following syntax.

$\mathsf{Del.Gen}$**:** Given the security parameter 1^λ, outputs a common reference string CRS.

$\mathsf{Del.P}$**:** Given the common reference string, a time bound 1^T in unary representation and an instance $x \in \{0,1\}^*$, outputs a proof Π.

$\mathsf{Del.V}$**:** Given the common reference string, a time bound T in binary representation, an instance $x \in \{0,1\}^*$ and a proof Π, outputs a bit.

Definition 2.3. *A publicly verifiable non-interactive argument scheme* $(\mathsf{Del.Gen}, \mathsf{Del.P}, \mathsf{Del.V})$ *for the universal language $\mathcal{U}$ satisfies the following requirements*

Completeness: *For every $\lambda \in \mathbb{N}$ and every $(x, \mathsf{T}) \in \mathcal{U}$*

$$\Pr\left[\mathsf{Del.V}(\mathsf{CRS}, \mathsf{T}, x, \Pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Del.Gen}(1^\lambda) \\ \Pi \leftarrow \mathsf{Del.P}(\mathsf{CRS}, 1^\mathsf{T}, x) \end{array}\right] = 1 \ .$$

Efficiency: *In the above (honest) experiment the size of the proof* Π *is* $\mathrm{poly}(\lambda, \log \mathsf{T})$. *The running time of* $\mathsf{Del.V}$ *is* $|x| \cdot \mathrm{poly}(|\mathsf{CRS}|, |\Pi|, \log \mathsf{T})$.

Adaptive Soundness: *For every polynomial* T *and for every poly-size cheating prover* P^* *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[\begin{array}{l} (x^*, \mathsf{T}(\lambda)) \notin \mathcal{U} \\ \mathsf{Del.V}(\mathsf{CRS}, \mathsf{T}, x^*, \Pi^*) = 1 \end{array} \;\middle|\; \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Del.Gen}(1^\lambda) \\ (x^*, \Pi^*) \leftarrow \mathsf{P}^*(\mathsf{CRS}) \end{array}\right] \leq \mu(\lambda) \ ,$$

3 Zero-Testable Homomorphic Encryption

In this section we define the notion of zero-testable homomorphic encryption. We also define a multi-key variant [LTV12].

3.1 Homomorphic Encryption

We start by recalling the notion of homomorphic encryption.

Syntax. A homomorphic encryption scheme consists of PPT algorithms

$$(\mathsf{HE.KeyGen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$$

with the following syntax.

$\mathsf{HE.KeyGen}$**:** Given the security parameter 1^λ, outputs a secret key sk, a public key pk and a description of a ring R.

$\mathsf{HE.Enc}$**:** Given the public key pk and a message $m \in \{0,1\}$, outputs a ciphertext c.

$\mathsf{HE.Dec}$**:** Given the secret key sk and a ciphertext c, outputs a ring element $\alpha \in R$ or a special symbol $\perp$.

$\mathsf{HE.Eval}$**:** Given e public key pk, an operation $\star \in \{+,-,\times\}$, and a pair of ciphertexts c_1, c_2, outputs a ciphertext c or a special symbol $\perp$.

Evaluating circuits. Some formulations of homomorphic encryption only consider an evaluation algorithm for circuits and not individual gates. By explicitly requiring that the evaluation is performed gate by gate, we ensure correctness for a "multi-hop" evaluation [GHV10] where ciphertexts that result from a homomorphic computation support further homomorphic operations.

Homomorphic evaluation of an arithmetic circuit C is implemented by iteratively applying the basic evaluation algorithm $\mathsf{HE.Eval}$ for every gate in C. This process is described formally below.

We only consider arithmetic circuits containing constants from $\{0,1\}$, which can be evaluated over any ring. When evaluating a gate that takes a constant $b \in \{0,1\}$ we do not generate a fresh random encryption of b. Instead, we assume that the public key includes ciphertexts $\hat{0}$ and $\hat{1}$ of 0 and 1 respectively. This evaluation strategy guarantees that all occurrences of a constant in C are replaced with the same ciphertext. This will be crucial later when we introduce the notion of zero-testable homomorphic encryption.

For an arithmetic circuit C, and ciphertexts $(c_1,\ldots,c_n)$ encrypted under public key pk we denote by $\langle C(c_1,\ldots,c_n)\rangle$ the evaluated ciphertext c computed as follows.

- If C is the constant 0 then $c = \hat{0}$.
- If C is the constant 1 then $c = \hat{1}$.
- If C is the i-th input wire then $c = c_i$.
- If C is of the form $C = C_1 \star C_2$ then

$$c = \mathsf{HE.Eval}\left(\mathsf{pk}, \star, \left(\langle C_1(c_1,\ldots,c_n)\rangle, \langle C_2(c_1,\ldots,c_n)\rangle\right)\right) \enspace .$$

Definition 3.1 (Homomorphic Encryption)**.** *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be an ensemble of circuits. A homomorphic encryption scheme* $(\mathsf{HE.KeyGen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$ *for $\mathcal{C}$ satisfies the following requirements.*

Correctness: *For every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$ with n inputs wires, and every $m_1, \dots, m_n \in \{0,1\}$*

$$\Pr\left[C(m_1,\dots,m_n) = \alpha \;\middle|\; \begin{array}{l} (\mathsf{sk},\mathsf{pk},R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ \forall i \in [n] : c_i \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, m_i) \\ c \leftarrow \langle C(c_1,\dots,c_n)\rangle \\ \alpha \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c) \end{array} \right] = 1 \enspace ,$$

where C is evaluated over R.

Compactness: *There exists a polynomial L such that in the above honest experiment $|c| \leq L(\lambda)$ (independently of $|C|$).*

Semantic Security: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[m = m' \;\middle|\; \begin{array}{l} m \leftarrow \{0,1\} \\ (\mathsf{sk},\mathsf{pk},R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ c \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, m) \\ m' \leftarrow \mathsf{Adv}(\mathsf{pk}, c) \end{array} \right] \leq \frac{1}{2} + \mu(\lambda) \enspace .$$

Definition 3.2 (Somewhat Homomorphic Encryption). *For $B, \Delta \in \mathbb{N}$ let $\mathcal{C}_{B,\Delta}$ be the set of arithmetic circuits of size at most B and total degree at most Δ. Let $B = B(\lambda), \Delta = \Delta(\lambda)$ be polynomially bounded functions. A homomorphic encryption scheme is (B,Δ)-somewhat homomorphic if it satisfies Definition 3.1 for the circuit ensemble $\left\{\mathcal{C}_{B(\lambda),\Delta(\lambda)}\right\}_{\lambda\in\mathbb{N}}$. A scheme is Δ-somewhat homomorphic if it is (B,Δ)-somewhat homomorphic for every polynomial B.*

3.2 Correctness for Adversarial Ciphertexts

We formulate an additional correctness requirement that considers evaluation of adversatively generated ciphertexts. Informally, we require that an efficient adversary cannot generate a pair of ciphertexts that cause en evaluation error. A homomorphic evaluation $\langle c_1 \star c_2 \rangle$ is erroneous if the following two experiments have different outputs

1. Homomorphically evaluate $\langle c_1 \star c_2 \rangle$ and output the decryption of the evaluated ciphertext.

2. Decrypt c_1, c_s. If one of the ciphertexts fails to decrypt (decryption output $\perp$), then output $\perp$. Otherwise output the evaluation of $\star$ on the decrypted elements.

Many existing homomorphic encryption candidates only support a polynomially bounded number of homomorphic operations before the noise in the ciphertexts becomes too large and causes an evaluation error. Therefore, in such candidates, ciphertexts that cause en evaluation error are easy to generate. To support candidate of this nature we allow the output of the first experiment above to be $\perp$ even if the output of the second experiment is different than $\perp$.

Correctness for Adversarial Ciphertexts: For every poly-size adversary Adv there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and for every operation $\star \in \{+,-,\times\}$

$$\Pr\left[\alpha \notin \{\alpha_1 \star \alpha_2, \perp\} \;\middle|\; \begin{array}{l} (\mathsf{sk},\mathsf{pk},R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ c_1, c_2 \leftarrow \mathsf{Adv}(\mathsf{pk}) \\ c \leftarrow \mathsf{HE.Eval}(\mathsf{pk}, \star, (c_1, c_2)) \\ \forall i \in \{1,2\} : \alpha_i \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c_i) \\ \alpha \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c) \end{array} \right] \leq \mu(\lambda) \enspace ,$$

where in the probability above, if $\alpha_1, \alpha_2 \in R$, the expression $\alpha_1 \star \alpha_2$ is evaluated over R. If either $\alpha_1 = \perp$ or $\alpha_1 = \perp$ then $\alpha_1 \star \alpha_2 = \perp$.

3.3 Zero Test

A zero test for a homomorphic encryption scheme is a PPT algorithm $\mathsf{HE.ZT}$ that can identify trivial encryptions of 0. These are ciphertexts that result from homomorphically evaluating an arithmetic circuit that is identically zero. We additionally require that the zero test never incorrectly identifies encryptions of non-zero values. This holds even for adversatively generated ciphertexts.

Given the public key pk and a ciphertext c, the zero test $\mathsf{HE.ZT}$ outputs a bit. The zero test satisfies the following requirements.

Zero-Test Completeness: For every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$ with n inputs wires such that C is identically zero, and every $m_1, \dots, m_n \in \{0,1\}$

$$\Pr\left[\mathsf{HE.ZT}(\mathsf{pk}, c) = 1 \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{pk}, R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ \forall i \in [n] : c_i \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, m_i) \\ c \leftarrow \langle C\left(c_1, \dots, c_n\right)\rangle \end{array}\right] = 1 \ .$$

Zero-Test Soundness: For every poly-size adversary Adv there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$

$$\Pr\left[\begin{array}{l} \mathsf{HE.ZT}(\mathsf{pk}, c) = 1 \\ \alpha \neq 0 \end{array} \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{pk}, R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ c \leftarrow \mathsf{Adv}(\mathsf{pk}) \\ \alpha \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c) \end{array}\right] \leq \mu(\lambda) \ .$$

3.4 Weak Decryption

We define a relaxation of homomorphic encryption where

- The decryption procedure $\mathsf{HE.Dec}$ is not required to be PPT.
- Instead we require that there exists a weak decryption procedure $\mathsf{HE.WeakDec}$ which is PPT but does not decrypt messages outside $\{0,1\}$.
- The weak decryption result should be consistent with the inefficient decryption result even for adversarially generated ciphertexts.

The encryption scheme constructed in Section 5 will only satisfy this relaxation which is sufficient for our application.

Given the secret key sk and a ciphertext c, the weak decryption procedure $\mathsf{HE.WeakDec}$ outputs a message $m \in \{0,1\}$ or a special symbol $\perp$. The weak decryption procedure satisfies the following requirement.

Weak Decryption: For every poly-size adversary Adv there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$

$$\Pr\left[m \neq \alpha' \;\middle|\; \begin{array}{l} (\mathsf{sk}, \mathsf{pk}, R) \leftarrow \mathsf{HE.KeyGen}(1^\lambda) \\ c \leftarrow \mathsf{Adv}(\mathsf{pk}) \\ \alpha \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c) \\ m \leftarrow \mathsf{HE.WeakDec}(\mathsf{sk}, c) \end{array}\right] \leq \mu(\lambda) \ ,$$

where in the above probability, $\alpha' = \alpha$ if $\alpha \in \{0,1\}$ and $\alpha' = \perp$ otherwise.

3.5 Multi-Key Zero-Testable Homomorphic Encryption

In this section we define a multi-key variant of homomorphic encryption that also satisfies the other requirements defined above. In multi-key homomorphic encryption, introduced by López-Alt et al. [LTV12] homomorphic computation can be executed over ciphertexts encrypted under d different keys. To ensure semantic security, decrypting the result requires all secret keys. Importantly, a system can generate ciphertext under an unbounded number of keys and any d of them can be combined in a homomorphic computation. We assume that the number of different keys d is constant. We also allow for common public parameters used to generate all keys.

Syntax. A d-key zero-testable homomorphic encryption scheme consists of PPT algorithms

$$(\mathsf{MHE.ParamGen}, \mathsf{MHE.KeyGen}, \mathsf{MHE.Enc}, \mathsf{MHE.WeakDec}, \mathsf{MHE.Eval}, \mathsf{MHE.ZT})$$

and an unbounded algorithm $\mathsf{MHE.Dec}$ with the following syntax.

$\mathsf{MHE.ParamGen}$**:** Given the security parameter 1^λ, outputs public parameters pp and a description of a ring R.

$\mathsf{MHE.KeyGen}$**:** Given the public parameters pp, outputs a secret key sk and a public key pk.

$\mathsf{MHE.Enc}$**:** Given public parameters pp, a public key pk and a message $m \in \{0,1\}$, outputs a ciphertext c.

$\mathsf{MHE.Dec}$**:** Given public parameters pp, d secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_d$ and a ciphertext c, outputs a ring element $\alpha \in R$ or a special symbol $\bot$.

$\mathsf{MHE.WeakDec}$**:** Given public parameters pp, d secret keys $\mathsf{sk}_1, \ldots, \mathsf{sk}_d$ and a ciphertext c, outputs a message $m \in \{0,1\}$ or a special symbol $\bot$.

$\mathsf{MHE.Eval}$**:** Given public parameters pp, a pair of public keys $\mathsf{pk}_1, \mathsf{pk}_2$, an operation $\star \in \{+,-,\times\}$ and a pair c_1, c_2, outputs a ciphertext c or a special symbol $\bot$.

$\mathsf{MHE.ZT}$**:** Given public parameters pp, d public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_d$ and a ciphertext c, outputs a bit.

Remark 3.3 (Superfluous keys). The decryption and zero test algorithms take d keys, even if the input ciphertext results from a computation involving less keys. We assume without loss of generality that adding superfluous keys does not affect the procedures functionality.

Definition 3.4 (Multi-Key Zero-Testable Homomorphic Encryption)**.** *Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be an ensemble of circuits. A d-key zero-testable homomorphic encryption scheme*

$$(\mathsf{MHE.ParamGen}, \mathsf{MHE.KeyGen}, \mathsf{MHE.Enc}, \mathsf{MHE.Dec}, \mathsf{MHE.WeakDec}, \mathsf{MHE.Eval}, \mathsf{MHE.ZT})$$

for $\mathcal{C}$ satisfies the following requirements.

Correctness: *There exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$ with n inputs wires, every $m_1, \ldots, m_n \in \{0,1\}$ and every indices $j_1, \ldots, j_n \in [d]$*

$$\Pr\left[C(m_1,\ldots,m_n) = \alpha \;\middle|\; \begin{array}{l} (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ \forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ \forall i \in [n] : c_i \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}_{j_i}, m_i) \\ c \leftarrow \langle C(c_1,\ldots,c_n)\rangle \\ \alpha \leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1,\ldots,\mathsf{sk}_d), c) \end{array} \right] \geq 1 - \mu(\lambda) \ ,$$

where C is evaluated over R.

Compactness: *There exists a polynomial L (that may depend on d) such that in the above honest experiment $|c| \leq L(\lambda)$ (independently of $|C|$).*

Correctness for Adversarial Ciphertexts: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and for every operation $\star \in \{+, -, \times\}$*

$$\Pr\left[\alpha \notin \{\alpha_1 \star \alpha_2, \bot\} \;\middle|\; \begin{array}{l} (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ \forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ c_1, c_2 \leftarrow \mathsf{Adv}(\mathsf{pp}, \mathsf{pk}_1, \ldots, \mathsf{pk}_d) \\ c \leftarrow \mathsf{MHE.Eval}(\mathsf{pp}, (\mathsf{pk}_1, \ldots, \mathsf{pk}_d), \star, (c_1, c_2)) \\ \forall i \in \{1, 2\} : \alpha_i \leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d), c_i) \\ \alpha \leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d), c) \end{array}\right] \leq \mu(\lambda) \;,$$

where in the probability above, if $\alpha_1, \alpha_2 \in R$, the expression $\alpha_1 \star \alpha_2$ is evaluated over R. If either $\alpha_1 = \bot$ or $\alpha_1 = \bot$ then $\alpha_1 \star \alpha_2 = \bot$.

Zero Test Completeness: *There exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$ with n inputs wires that is identically zero, every $m_1, \ldots, m_n \in \{0, 1\}$, and every indices $j_1, \ldots, j_n \in [d]$*

$$\Pr\left[b = 1 \;\middle|\; \begin{array}{l} (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ \forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ \forall i \in [n] : c_i \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}_{j_i}, m_i) \\ c \leftarrow \langle C\,(c_1, \ldots, c_n)\rangle \\ b \leftarrow \mathsf{MHE.ZT}(\mathsf{pp}, (\mathsf{pk}_1, \ldots, \mathsf{pk}_d), c) \end{array}\right] \geq 1 - \mu(\lambda) \;.$$

Zero-Test Soundness: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[\begin{array}{c} b = 1 \\ \alpha \neq 0 \end{array} \;\middle|\; \begin{array}{l} (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ \forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ c \leftarrow \mathsf{Adv}(\mathsf{pp}, \mathsf{pk}_1, \ldots, \mathsf{pk}_d) \\ \alpha \leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d)\,, c) \\ b \leftarrow \mathsf{MHE.ZT}(\mathsf{pp}, (\mathsf{pk}_1, \ldots, \mathsf{pk}_d), c) \end{array}\right] \leq \mu(\lambda) \;.$$

Weak Decryption: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[m \neq \alpha' \;\middle|\; \begin{array}{l} (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ \forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ c \leftarrow \mathsf{Adv}(\mathsf{pp}, \mathsf{pk}_1, \ldots, \mathsf{pk}_d) \\ \alpha \leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d)\,, c) \\ m \leftarrow \mathsf{MHE.WeakDec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d)\,, c) \end{array}\right] \geq 1 - \mu(\lambda) \;,$$

where in the above probability, $\alpha' = \alpha$ if $\alpha \in \{0, 1\}$ and $\alpha' = \bot$ otherwise.

Semantic Security: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[m = m' \;\middle|\; \begin{array}{l} m \leftarrow \{0, 1\} \\ (\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \\ (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{MHE.KeyGen}(1^\lambda) \\ c \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}, m) \\ m' \leftarrow \mathsf{Adv}(\mathsf{pk}, c) \end{array}\right] \leq \frac{1}{2} + \mu(\lambda) \;.$$

4 A Non-Interactive Argument

This section describes our publicly-verifiable non-interactive arguments. We start with an overview of the construction.

4.1 Overview

We construct a non-interactive argument system for the universal language $\mathcal{U}$. Given an instance $x = (M, y) \in \{0,1\}^n$ and a time bound T the verifier wants to ascertain that $(x, \mathsf{T}) \in \mathcal{U}$, that is, that the Turing machine M accepts the input y within T steps. The protocol should be *adaptively sound*: even an adaptive cheating prover, who first sees the CRS and then picks an instance $(x, \mathsf{T}) \notin \mathcal{U}$ adaptively, should not be able to generate am accepting proof.

In the protocol, the prover and verifier translate the instance (x, T) into a 3CNF formula φ over $\mathrm{poly}(n, \mathsf{T})$ variables, which is satisfiable if and only if $(x, \mathsf{T}) \in \mathcal{U}$. φ has a "short" implicit description via an arithmetic circuit Φ of small size and degree that, given the the labels of three literals, determines whether their disjunction is a clause in φ. Note that given φ, the formula Φ and the original instance (x, T) can be efficiently reconstructed. More over, if $(x, \mathsf{T}) \in \mathcal{U}$, a satisfying assignment for φ can be efficiently computed. With this formula in mind, the argument system has two main ingredients:

Ingredient 1: the core protocol. The first ingredient is a publicly-verifiable non-interactive "core protocol". The prover in the core protocol is presented with a CRS, a circuit Φ describing a 3CNF φ (as above), and a satisfying assignment σ to φ. It generates a proof Π that will convince the verifier that the 3CNF described by Φ is satisfiable.

The core protocol has a relaxed soundness property: it is *not* guaranteed that an adaptive cheating prover P^* cannot generate a circuit Φ describing an unsatisfiable 3CNF together with a proof Π^* that makes the verifier accept. Rather, the soundness guarantee is that any adaptive cheating prover for the core protocol can be used to derive a *no-signalling adaptive local assignment generator* Assign. The adaptive assignment generator Assign is a randomized algorithm that gets as input a small set S of variables, and outputs a pair (Φ, σ), where $\sigma : S \to \{0,1\}$ is a local assignment to the variables in S. The algorithm Assign satisfies the following properties:

1. **No-signalling.** Given a set S of variables, Assign outputs a pair (Φ, σ). Intuitively, the joint distribution of Φ and the values assigned to any subset of the variables in S are independent of the other variables in S. More precisely, for every two sets of variables S_1, S_2 both containing a subset T, the distributions obtained by executing Assign on S_1 and on S_2 to obtain (Φ, σ), and then restricting σ to the variables in T, are computationally indistinguishable.

2. **Adaptive local soundness.** We consider an execution of the cheating prover P^* in the core protocol that generates a pair (Φ, Π^*). Additively, for every small subset S of variables, we consider an execution of Assign on the set S that generates a pair (Φ', σ'). We require that Φ' is indistinguishable from Φ, and moreover, if the proof Π^* is accepting, then the assignment σ' is *locally-consistent* with the 3CNF φ' described by Φ'. We say that the assignment $\sigma' : S \to \{0,1\}$ is locally-consistent with φ' if σ' satisfies all clauses of φ' that are comprised entirely of variables in S.

   In particular, we have that if P^* has a noticeable probability of generating a pair (Φ, Π^*) such that Φ describes an unsatisfiable 3CNF, but the verifier accepts Π^*. Then for every small subset S of variables, running Assign on the set S has a noticeable probability of producing a pair (Φ', σ') where Φ' describes an unsatisfiable 3CNF φ', but σ is a locally-consistent with Φ'.

Some remarks are in order. First, we note that the relaxed soundness property has a flavor of "knowledge extraction": while we do not claim that any cheating prover for the core protocol must "know" a satisfying assignment to the 3CNF (indeed, the 3CNF might not be satisfiable, in which case no such assignment exists), a cheating prover *can* be used to generate "locally consistent" assignments on any set of variables. This extraction property is slightly more involved because it is concerned with *adaptive* cheating provers: the 3CNF is not fixed in advance. Rather, an adaptive cheating prover for the core protocol can be used to adaptively generate, given a set S of variables, an unsatisfiable 3CNF together with a locally-consistent assignment for those variables in S. The distribution of 3CNFs generated by the core protocol cheating prover (together with the bit indicating whether the verifier accepts the jointly-generated proof) is computationally indistinguishable from the distribution of 3CNFs generated by the assignment generator (together with the bit indicating whether the jointly-generated assignment is locally satisfiable). We note further that the no-signalling property implies that for any two sets S and S', the distributions of the circuit Φ generated by Assign are themselves computationally indistinguishable.

While the core protocol's soundness guarantee is robust to adaptive provers, it is weak in the sense that it only guarantees *local* consistency of the assignment generator. Even for a fixed 3CNF (let alone for an adaptively-generated one) the existence of no-signalling locally-consistent assignments does not imply that the 3CNF is satisfiable! As in prior works, we provide a "circuit-augmentation" procedure that encodes a Turing Machine computation as a 3CNF with a particular structure. The existence of a (no-signalling) locally-consistent assignment generator for the augmented 3CNF guarantees that the Turing Machine accepts its input. Here too, we need to take care to handle adaptive adversaries. This is the second main ingredient of our delegation protocol.

Ingredient 2: adaptive augmented circuit. To build an adaptively-sound delegation protocol we need an adaptive variant of the the *augmented circuit construction* from [KRR14]. We describe this as a circuit-augmentation algorithm that transforms an instance (x, T) for $\mathcal{U}$ into an arithmetic circuit Φ of small size and degree, which describes a 3CNF φ. The 3CNF φ should be satisfiable if and only if $(x, \mathsf{T}) \in \mathcal{U}$. This property alone, of course, is not sufficient, since the core protocol does not prove the 3CNF's global satisfiability. Prior work showed a transformation where if $(x, \mathsf{T}) \notin \mathcal{U}$, then it is not possible to generate even locally-consistent assignments in a no-signalling manner. Since we want an adaptively-sound delegation protocol, we need an even stronger property: let Assign be a no-signalling adaptive assignment generator as above. We assume that Assign generates the circuit Φ by applying the adaptive circuit-augmentation procedure to an instance (x, T). Then for some small set S^* of variables the probability that $(x, \mathsf{T}) \notin \mathcal{U}$ but Assign generates a locally-consistent assignment for S^* is negligible. The transformation and its proof are based on [KRR14, PR14, BHK16].

There is a (slight) gap between the soundness we consider in the augmented-circuit transformation and in the core protocol: the core protocol is simply concerned with 3CNFs described by small circuits. The augmented-circuit transformation, on the other hand, considers (and relies on) the procedure used to derive these 3CNFs from a computation described by a Turing Machine. This gap makes the presentation of the core protocol considerably simpler and more modular (in particular, there is no need to consider Turing Machines in the core protocol). We bridge the gap by noting that the augmentation procedure Aug is easy to invert: given a circuit Φ, it is easy to recover the instance (x, T) from which it was derived (or to output $\perp$ if Φ is not an output of Aug). This allows us to argue that for two computationally indistinguishable distributions on Φ, if the first distribution is over outputs of Aug, then the second must be over such outputs too (except with negligible probability). Moreover, given a circuit Φ produced by Aug, we can determine whether it describes a satisfiable 3CNF by recovering the original instance for $\mathcal{U}$ and testing (in polynomial time) whether the Turing Machine accepts or rejects.

Putting it together. To derive a delegation protocol, we use the core protocol's CRS. Given an instance (x, T), the prover and verifier both use the augmented-circuit transformation to derive Φ and execute the core protocol on Φ. A prover P^* that cheats with noticeable probability can be used to derive a no-signalling adaptive local assignment generator Assign^*. By the core protocol's soundness we conclude that for every set S of variables, with noticeable probability Assign^* generates pairs (Φ, σ) where Φ

describes an unsatisfiable 3CNF, but σ is locally consistent. Moreover, Φ is derived by running the augmented circuit construction on an instance $(x, \mathsf{T}) \notin \mathcal{U}$ (this is true for the execution of the core protocol, by computational indistinguishability it holds also for the outputs of Assign^*). However, the the augmented circuit construction guarantees that no such assignment generator exists, leading to a contradiction.

Organization. We define adaptive local assignment generators in Section 4.2. The core protocol is given in Section 4.3. The augmented-circuit transformation is discussed in Section 4.4. Finally, we combine these ingredients in Section 4.5, where we describe the full delegation protocol.

4.2 Adaptive Local-Assignment Generator

Before stating the properties of the core protocol, we introduce some notation and formalize the notion of an adaptive local-assignment generator.

Succinct formula representation $\mathcal{I}_\varphi$. Let φ be a 3CNF boolean formula with variables $\alpha_1, \dots, \alpha_B$. Let $B = 2^m$ and identify the indices in $[B]$ with strings in $\{0,1\}^m$. We define a boolean *indicator function* $\mathcal{I}_\varphi : \{0,1\}^{3m+3} \to \{0,1\}$ of φ as follows. For every indices $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3 \in \{0,1\}^m$ and for every bits $b_1, b_2, b_3 \in \{0,1\}^3$, we have that

$$\mathcal{I}_\varphi(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, b_1, b_2, b_3) = 1 \ ,$$

if and only if φ contains the clause:

$$(\alpha_{\mathbf{u}_1} = b_1) \vee (\alpha_{\mathbf{u}_2} = b_2) \vee (\alpha_{\mathbf{u}_3} = b_3) \ .$$

The locally consistency verifier $\mathsf{V}_{\mathsf{local}}$. We denote by $\mathsf{V}_{\mathsf{local}}$ the verification algorithm for local assignments to φ. The algorithm is given as input

- An arithmetic circuit Φ computing a boolean function with $3m + 3$ inputs (we think of Φ as computing the indicator function $\mathcal{I}_\varphi$ for some formula φ).
- A partial assignments $\sigma : S \to \{0,1\}$ for a set $S \subseteq \{0,1\}^m$.

$\mathsf{V}_{\mathsf{local}}(\Phi, \sigma)$ accepts if an only if the assignment σ is locally consistent with the formula described by Φ. That is, for every $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3 \in S$ and every $b_1, b_2, b_3 \in \{0,1\}$

$$\Phi(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, b_1, b_2, b_3) = 1 \quad \Rightarrow \quad (\sigma(\mathbf{u}_1) = b_1) \vee (\sigma(\mathbf{u}_2) = b_2) \vee (\sigma(\mathbf{u}_3) = b_3) \ .$$

Adaptive local-assignment generator. Let $Q = Q(\lambda), B = B(\lambda)$ be functions and let $B = 2^m$. An adaptive Q-local-assignment generator Assign for B-variate formulas is a probabilistic algorithm with the following syntax: given the the security parameter 1^λ and a set of indices $S \subseteq \{0,1\}^m$ of size at most Q, Assign outputs

- An arithmetic circuit Φ computing a boolean function with $3m + 3$ inputs.
- A partial assignment $\sigma : S \to \{0,1\}$.

We also allow the local-assignment generator to fail and output $\Phi = \perp$. An adaptive local-assignment generator should satisfy the following properties.

Definition 4.1 (Adaptive Local-Assignment Generator). *A Q-local-assignment generator* Assign *for $B = 2^m$-variate formulas satisfies:*

Computational No-Signaling: *For every polynomial-size distinguisher D there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every subsets $S \subseteq S' \subseteq \{0,1\}^m$ of size at most Q*

$$\left| \Pr_{(\Phi,\sigma)\leftarrow \mathsf{Assign}(1^\lambda,S)} [D(\Phi,\sigma(S))=1] - \Pr_{(\Phi,\sigma')\leftarrow \mathsf{Assign}(1^\lambda,S')} [D(\Phi,\sigma'(S))=1] \right| \le \mu(\lambda) \enspace .$$

Everywhere Local Consistency: *There exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every set $S \subseteq \{0,1\}^m$ of size at most Q*

$$\Pr_{(\Phi,\sigma)\leftarrow \mathsf{Assign}(1^\lambda,S)} \left[\begin{array}{l} \Phi \neq \perp \\ \mathsf{V}_{\mathsf{local}}(\Phi,\sigma)=0 \end{array} \right] \le \mu(\lambda) \enspace .$$

Remark 4.2. The everywhere local consistency requirement allows the assignment generator to fail and output $\Phi = \perp$. We only require that if the assignment generator does not fail, it generates an assignment that is locally consistent with the formula described by Φ. Looking ahead, to leverage the existence of some assignment generator we first bound the probability it fails.

4.3 The Core Protocol

In this section we describe the syntax and the properties of the core delegation protocol. The protocol itself is given in Section 4.3.1.

Syntax. Let $\Delta = \Delta(\lambda)$ be a polynomially bounded function. The core protocol with degree bound Δ consists of PPT algorithms $(\mathsf{Core.Gen}, \mathsf{Core.P}, \mathsf{Core.V})$ with the following syntax. Let φ be a B-variate 3CNF boolean formula where $B = 2^m$ and let Φ be an arithmetic circuit of total degree $\delta \le \Delta$ computing the function $\mathcal{I}_\varphi$.

$\mathsf{Core.Gen}$**:** Given the security parameter 1^λ and a locality parameter 1^Q outputs a common reference string CRS.

$\mathsf{Core.P}$**:** Given the common reference string CRS, the circuit Φ and an assignment $\sigma : \{0,1\}^m \to \{0,1\}$, outputs a proof Π.

$\mathsf{Core.V}$**:** Given the common reference string CRS, the circuit Φ and the proof Π outputs a bit.

The protocol satisfies the following requirements.

Completeness. For every security parameter $\lambda \in \mathbb{N}$, every 3CNF boolean formula φ with B variables, every satisfying assignment σ, every arithmetic circuit Φ of individual degree $\delta \le \Delta$ computing the function $\mathcal{I}_\varphi$, and every locality parameter $Q \in [B]$

$$\Pr\left[\mathsf{Core.V}(\mathsf{CRS},\Phi,\Pi)=1 \;\middle|\; \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Core.Gen}(1^\lambda,1^Q) \\ \Pi \leftarrow \mathsf{Core.P}(\mathsf{CRS},\Phi,\sigma) \end{array} \right] = 1 \enspace .$$

Efficiency. There exists a polynomial L such that in the above honest experiment $|\Pi| \le L(\lambda)\cdot Q \cdot \delta$ where δ is the individual degree of the circuit Φ. Additionally the verifier's running time is bounded by $L(|\mathsf{CRS}|)\cdot(|\Phi|+|\Pi|)$.

Adaptive local soundness. For every polynomially bounded functions $Q = Q(\lambda), B = B(\lambda)$ there exists an algorithm Assign such that for every poly-size cheating prover P^* the following holds

- $\mathsf{Assign}^{\mathsf{P}^*}$ is a adaptive Q-local-assignment generator for B-variate formulas.

- For every polynomial-size distinguisher D there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, letting $B = 2^m$, for every set of indices $S \subseteq \{0,1\}^m$ of size at most Q

$$\left| \begin{array}{l} \Pr\left[D(\Phi) = 1 \;\middle|\; \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Core.Gen}(1^\lambda, 1^Q) \\ (\Phi, \Pi^*) \leftarrow \mathsf{P}^*(\mathsf{CRS}) \\ b \leftarrow \mathsf{Core.V}(\mathsf{CRS}, \Phi, \Pi^*) \\ \text{If } b = 0 \text{ Set } \Phi \leftarrow \bot \end{array} \right] \\ \\ - \quad \Pr\left[D(\Phi) = 1 \;\middle|\; (\Phi, \sigma) \leftarrow \mathsf{Assign}^{\mathsf{P}^*}(1^\lambda, S) \right] \end{array} \right| \le \mu(\lambda) \ .$$

4.3.1 Construction.

Let $\Delta = \Delta(\lambda)$ be the function bounding the total degree of the circuit Φ. The core protocol makes use of a 3-key zero-testable 2Δ-somewhat homomorphic encryption scheme

$$(\mathsf{MHE.ParamGen}, \mathsf{MHE.KeyGen}, \mathsf{MHE.Enc}, \mathsf{MHE.Dec}, \mathsf{MHE.WeakDec}, \mathsf{MHE.Eval}, \mathsf{MHE.ZT}) \ .$$

The CRS generator. The CRS generation algorithm Core.Gen is given as input the security parameter 1^λ and a locality parameter 1^Q. It outputs a common reference string CRS as follows.

1. Sample public parameters for the encryption scheme

$$(\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \ .$$

2. For every $q \in [Q]$, generate a key pair

$$(\mathsf{sk}^q, \mathsf{pk}^q) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \ ,$$

   and λ encryptions of 0

$$\{c_i^q \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}^q, 0)\}_{i \in [\lambda]} \ .$$

3. Output a reference string containing the public parameters and all the public keys and ciphers

$$\mathsf{CRS} = \left(\mathsf{pp}, \left\{\mathsf{pk}^q, (c_1^q, \dots, c_\lambda^q)\right\}_{q \in [Q]}\right) \ .$$

The prover. The prover algorithm Core.P is given as input

- The common reference string

$$\mathsf{CRS} = \left(\mathsf{pp}, \left\{\mathsf{pk}^q, (c_1^q, \dots, c_\lambda^q)\right\}_{q \in [Q]}\right) \ .$$

- An (individual) degree δ arithmetic circuit Φ computing a boolean function with $3m + 3$ inputs.
- An assignment $\sigma : \{0,1\}^m \to \{0,1\}$.

We start by introducing some notation.

1. For every query $q \in [Q]$, let $\mathbf{c}^q = (c_1^q, \dots, c_m^q)$. We refer to the ciphertext vector $\mathbf{c}^q$ as an encryption of the the q-th CRS index (in an honestly generated CRS the index value is always 0^m).
2. Let Σ be a multi-linear extension of σ (See Section 2.2).

3. For every triplet of bits $\mathbf{b} = (b_1, b_2, b_3) \in \{0,1\}^3$ let $P_0^{\mathbf{b}}$ be the degree $\delta + 1$ arithmetic circuit taking $3m$ inputs

$$P_0^{\mathbf{b}}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = \Phi(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{b}) \cdot \prod_{k \in [3]} \left(1 - \beta(b_k, \Sigma(\mathbf{x}_k))\right) \quad . \tag{4}$$

(See Section 2.2 for the definition of the circuit β.)

4. For every $i \in [3m]$, let $P_i^{\mathbf{b}}$ be the linearization of the first i variables of the circuit $P_0^{\mathbf{b}}$. That is, $P_i^{\mathbf{b}}$ is the following arithmetic circuit taking $3m$ inputs which is multilinear in its first i variables, and of degree at most $\delta + 1$ in its other variables.

$$P_i^{\mathbf{b}}(x_1, \dots, x_{3m}) = \sum_{y_1, \dots y_i \in \{0,1\}} \beta(y_1, \dots y_i, x_1, \dots x_i) \cdot P_0^{\mathbf{b}}(y_1, \dots, y_i, x_{i+1}, \dots, x_{3m}) \quad . \tag{5}$$

Core.P outputs a proof Π as follows.

1. For every $q \in [Q]$ obtain an encryption of the assignment Σ evaluated on the q-th CRS index. That is, homomorphically obtain the ciphertext $d^q = \langle \Sigma\left(\mathbf{c}^q\right) \rangle$.

2. For every triplet of bits $\mathbf{b} \in \{0,1\}^3$, triplet of queries $\mathbf{q} = (q_1, q_2, q_3) \in [Q]^3$, and $i \in [3m]$ obtain the encrypted coefficients of the circuit $P_{i-1}^{\mathbf{b}}$ evaluated on the CRS indices $\mathbf{q}$ and restricted to its i-th input variable (see Section 2.1). Since the individual degree of $P_{i-1}^{\mathbf{b}}$ is at most $\delta + 1$, the restricted polynomial will have at most $\delta + 2$ coefficients. That is, homomorphically obtain the sequence of $\delta + 2$ ciphertexts $\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}$

$$\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}} = \left(\left\langle P_{i-1}^{\mathbf{b}} \Big|_{i,j} \left((\mathbf{c}^{q_1} \,|\, \mathbf{c}^{q_2} \,|\, \mathbf{c}^{q_3})_{-i} \right) \right\rangle \right)_{j \in [0, \delta+1]} \quad .$$

3. Output a proof Π that contains all the ciphertexts

$$\Pi = \left(\{d^q\}_{q \in [Q]} \quad , \quad \left\{ \mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}} \right\}_{\mathbf{b} \in \{0,1\}^3, \mathbf{q} \in [Q]^3, i \in [3m]} \right) \quad .$$

The verifier. The verifier algorithm Core.V is given as input

- The common reference string

$$\mathsf{CRS} = \left(\mathsf{pp}, \left\{ \mathsf{pk}^q, (c_1^q, \dots, c_\lambda^q) \right\}_{q \in [Q]} \right) \quad .$$

- A degree δ arithmetic circuit Φ computing a boolean function with $3m + 3$ inputs.

- An proof

$$\Pi = \left(\{d^q\}_{q \in [Q]} \quad , \quad \left\{ \mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}} \right\}_{\mathbf{b} \in \{0,1\}^3, \mathbf{q} \in [Q]^3, i \in [3m]} \right) \quad .$$

Core.V performs the following tests for every triplet of bits $\mathbf{b} = (b_1, b_2, b_3) \in \{0,1\}^3$ and triplet of queries $\mathbf{q} = (q_1, q_2, q_3) \in [Q]^3$. Core.V accepts only if all tests pass.

First, Core.V homomorphically evaluates the following ciphertexts

- Let $\tilde{P}^{\mathbf{b}}$ be the following arithmetic circuit taking $3m+3$ inputs

$$\tilde{P}^{\mathbf{b}}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, y_1, y_3, y_3) = \Phi(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{b}) \cdot \prod_{k \in [3]} (1 - \beta(b_k, y_k)) \quad . \tag{6}$$

  Evaluate the ciphertext

$$f'_0 = \left\langle \tilde{P}^{\mathbf{b}} \left(\mathbf{c}^{q_1}, \mathbf{c}^{q_2}, \mathbf{c}^{q_3}, d^{q_1}, d^{q_2}, d^{q_3} \right) \right\rangle \quad .$$

- Let F be the following arithmetic circuit taking $\delta + 3$ inputs

$$F(x, y_0, \ldots, y_{\delta+1}) = \sum_{j \in [0, \delta+1]} y_j \cdot x^j \quad . \tag{7}$$

  For $i \in [3m]$, evaluate the ciphertext f_{i-1} that encrypts the evaluation of the univariate polynomial with encrypted coefficient $\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}$ on the i-th input bit of the concatenated CRS indices $\mathbf{q}$. Recall that $\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}$ are supposedly the encrypted coefficients of the circuit $P_{i-1}^{\mathbf{b}}$ evaluated on the CRS indices $\mathbf{q}$ and restricted to its i-th input variable. Therefore, f_{i-1} is suppose to encrypt the evaluation of $P_{i-1}^{\mathbf{b}}$ on the CRS indices $\mathbf{q}$.

$$f_{i-1} = \left\langle F \left(\left(\mathbf{c}^{q_1} \,|\, \mathbf{c}^{q_2} \,|\, \mathbf{c}^{q_3} \right)_i, \mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}} \right) \right\rangle \quad .$$

- Let F' be the following arithmetic circuit taking $\delta + 3$ inputs

$$F'(x, y_0, \ldots, y_{\delta+1}) = \sum_{z \in \{0,1\}} \beta(z, x) \cdot F(z, y_0, \ldots, y_{\delta+1}) \quad .$$

  For $i \in [3m]$, evaluate the ciphertext f'_i that encrypts the linearization of the univariate polynomial with encrypted coefficient $\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}$ evaluated on the on the i-th input bit of the concatenated CRS indices $\mathbf{q}$. Therefore, f_{i-1} is suppose to encrypt the evaluation of the circuit $P_{i-1}^{\mathbf{b}}$ with its i-th variable linearized on the CRS indices $\mathbf{q}$.

$$f'_i = \left\langle F' \left(\left(\mathbf{c}^{q_1} \,|\, \mathbf{c}^{q_2} \,|\, \mathbf{c}^{q_3} \right)_i, \mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}} \right) \right\rangle \quad .$$

- Let $f_{3m} = \hat{0}$.

For every $i \in [0, 3m]$, $\mathsf{Core.V}$ tests that

$$\mathsf{MHE.ZT}\left(\mathsf{pp}, \left(\mathsf{pk}^{q_1}, \mathsf{pk}^{q_2}, \mathsf{pk}^{q_3}\right), \langle f_i - f'_i \rangle\right) = 1 \quad .$$

4.3.2 Completeness.

Fix a security parameter $\lambda \in \mathbb{N}$, a B-variate 3CNF boolean formula φ, a satisfying assignment σ, a degree δ arithmetic circuit Φ computing the function $\mathcal{I}_\varphi$, and a locality parameter $Q \in [B]$. Fix a triplet of bits $\mathbf{b} \in \{0,1\}^3$ and a triplet of queries $q_1, q_2, q_3 \in [Q]^3$.

Recall that the honest execution of the protocol proceeds as follows

- The CRS ciphertexts are generated including

$$\left(\mathbf{c}^{q_1}, \mathbf{c}^{q_2}, \mathbf{c}^{q_3} \right) \quad .$$

- The prover Core.P homomorphically evaluates arithmetic circuits over the CRS ciphertexts and obtains the proof ciphertexts including

$$(d^{q_1}, d^{q_2}, d^{q_3}) \quad , \quad \mathbf{e}_0^{\mathbf{q},\mathbf{b}}, \ldots, \mathbf{e}_{3m-1}^{\mathbf{q},\mathbf{b}} \ .$$

- The verifier Core.V, in the iteration corresponding to $\mathbf{q}, \mathbf{b}$, homomorphically evaluates arithmetic circuits over the CRS ciphertexts and the proof ciphertexts

$$f_0, f_0', \cdots, f_{3m}, f_{3m}' \ .$$

We consider the arithmetic circuits obtained by composing together the circuits evaluated by the prover and the circuits evaluated by the verifier that map the CRS ciphertexts $\mathbf{c}^{q_1}, \mathbf{c}^{q_2}, \mathbf{c}^{q_3}$ to the verifier's ciphertexts f_i, f_i'. Specifically, for $i \in [0, 3m]$ let G_i and G_i' be arithmetic circuits such that

$$\begin{aligned} f_i &= \langle G_i\left(\mathbf{c}^{q_1}, \mathbf{c}^{q_2}, \mathbf{c}^{q_3}\right)\rangle \ , \\ f_i' &= \left\langle G_i'\left(\mathbf{c}^{q_1}, \mathbf{c}^{q_2}, \mathbf{c}^{q_3}\right)\right\rangle \ . \end{aligned}$$

By inspecting the construction we have that for $i \in [3m]$

$$\begin{aligned} G_0'(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) &= \tilde{P}^{\mathbf{b}}\left(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \Sigma(\mathbf{x}_1), \Sigma(\mathbf{x}_2), \Sigma(\mathbf{x}_3)\right) \ , \\ G_{i-1}(\mathbf{x}) &= F\left(x_i, P_{i-1}^{\mathbf{b}}\Big|_{i,1}(\mathbf{x}_{-i}), \ldots, P_{i-1}^{\mathbf{b}}\Big|_{i,\delta+1}(\mathbf{x}_{-i})\right) \ , \\ G_i'(\mathbf{x}) &= \sum_{z \in \{0,1\}} \beta(z, x_i) \cdot F\left(z, P_{i-1}^{\mathbf{b}}\Big|_{i,1}(\mathbf{x}_{-i}), \ldots, P_{i-1}^{\mathbf{b}}\Big|_{i,\delta+1}(\mathbf{x}_{-i})\right) \ , \\ G_{3m}(\mathbf{x}) &= \hat{0} \ . \end{aligned}$$

By the correctness of the homomorphic encryption zero test we have that Core.V accepts if $G_i \equiv G_i'$ for every $i \in [0, 3m]$. We argue that this is the case with the following sequence of equivalences.

1. By the definition of the circuit $P_0^{\mathbf{b}}$ in (4) and the definition of the circuit $\tilde{P}^{\mathbf{b}}$ in (6) we have that $G_0' \equiv P_0^{\mathbf{b}}$.

2. By the definition of the circuit F in (7), and property (1) of the circuit restrictions, we have that for $i \in [3m]$:

$$\begin{aligned} G_{i-1}(\mathbf{x}) &\equiv P_{i-1}^{\mathbf{b}}(\mathbf{x}) \ , \\ G_i'(\mathbf{x}) &\equiv \sum_{z \in \{0,1\}} \beta(z, x_i) \cdot P_{i-1}^{\mathbf{b}}\left(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_{3m}\right) \ . \end{aligned}$$

3. By the definition of the circuit $P_i^{\mathbf{b}}$ in (5) and by the definition of the function β in (2) we have that $G_i' \equiv P_i^{\mathbf{b}}$.

4. Σ is an arithmetic circuit that computes the boolean function σ, which is a satisfying assignment for φ and the circuit Φ computes the the function $\mathcal{I}_\varphi$. Thus, it follows by the definition of the circuit $P_0^{\mathbf{b}}$ in (4) and by Fact 2.1 that $P_0^{\mathbf{b}}$ computes the boolean zero function. Therefore, by the definition of the circuit $P_{3m}^{\mathbf{b}}$ in (5), we have that $P_{3m}^{\mathbf{b}} \equiv \hat{0}$.

It follows from sequence of equivalences above that $G_i \equiv G_i'$ for every $i \in [0, 3m]$ as required.

4.3.3 Adaptive local soundness.

The assignment generator Assign**.** We start by describing the algorithm Assign. Let $Q = Q(\lambda), B = B(\lambda)$ be polynomially bounded functions and let $B = 2^m$. Let P^* be a poly-size cheating prover. Assign has oracle assess to P^* and is given the security parameter 1^λ and a set of indices $S \subseteq \{0,1\}^m$ of size at most Q. Assign outputs an arithmetic circuit Φ and a partial assignment σ as follows.

1. Sample public parameters for the encryption scheme
$$(\mathsf{pp}, R) \leftarrow \mathsf{MHE.ParamGen}(1^\lambda) \ .$$
2. For every $q \in [Q]$, generate a key pair
$$(\mathsf{sk}^q, \mathsf{pk}^q) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \ .$$
3. For every $q \in [Q]$, let $\mathbf{u}^q \in \{0,1\}^m$ be the q-th index in S in lexicographical order, or 0^m if $q > |S|$. Let $\tilde{\mathbf{u}}^q \in \{0,1\}^\lambda$ be the string $\mathbf{u}^q$ padded with 0's to length λ. Obtain the encryptions
$$\{c_i^q \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}^q, \tilde{\mathbf{u}}_i^q)\}_{i\in[\lambda]} \ .$$
4. Query P^* with the reference string
$$\mathsf{CRS}^* = \left(\mathsf{pp}, \{\mathsf{pk}^q, (c_1^q, \ldots, c_\lambda^q)\}_{q\in[Q]}\right) \ ,$$
and obtain a circuit Φ and a proof
$$\Pi^* = \left(\{d^q\}_{q\in[Q]} \quad , \quad \left\{\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}\right\}_{\mathbf{b}\in\{0,1\}^3, \mathbf{q}\in[Q]^3, i\in[3m]}\right) \ .$$
5. If $\mathsf{Core.V}(\mathsf{CRS}^*, \Phi, \Pi^*) = 0$ output $(\Phi = \bot, \sigma = \bot)$.
6. Otherwise, output (Φ, σ) where $\sigma : S \to \{0,1\}$ is the assignment such that
$$\sigma(\mathbf{u}^q) = \mathsf{MHE.WeakDec}(\mathsf{pp}, \mathsf{sk}^q, d^q) \ .$$

Note that Assign is require to be efficient and therefore it must use the weak decryption algorithm $\mathsf{MHE.WeakDec}$. We will show that if the proof is accepting d^q decrypts to a value in $\{0,1\}$, and therefore weak decryption will not fail.

To prove adaptive local soundness we need to argue that $\mathsf{Assign}^{\mathsf{P}^*}$ satisfies the computational no-signaling and everywhere local consistency properties and that for every polynomial-size distinguisher D, security parameter $\lambda \in \mathbb{N}$, and set of indices $S \subseteq \{0,1\}^m$ of size at most Q

$$\left| \begin{array}{l} \Pr\left[D(\Phi) = 1 \;\middle|\; \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Core.Gen}(1^\lambda, 1^Q) \\ (\Phi, \Pi^*) \leftarrow \mathsf{P}^*(\mathsf{CRS}) \\ b \leftarrow \mathsf{Core.V}(\mathsf{CRS}, \Phi, \Pi^*) \\ \text{If } b = 0 \text{ Set } \Phi \leftarrow \bot \end{array} \right] \\ \\ - \ \Pr\left[D(\Phi) = 1 \;\middle|\; (\Phi, \sigma) \leftarrow \mathsf{Assign}^{\mathsf{P}^*}(1^\lambda, S) \right] \end{array} \right| \leq \mathrm{negl}(\lambda) \ . \tag{8}$$

The computational no-signaling property as well as (8) follows directly from the construction of Assign and the semantic security of the encryption scheme. We omit the proof.

We continue to show that $\mathsf{Assign}^{\mathsf{P}^*}$ is everywhere local consistent. an adaptive Q-local-assignment generator for B-variate formulas. We need to show that there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every set $S \subseteq \{0,1\}^m$ of size at most Q

$$\Pr_{(\Phi,\sigma)\leftarrow\mathsf{Assign}(1^\lambda,S)}\left[\begin{array}{l}\Phi \neq \bot \\ \mathsf{V}_{\mathsf{local}}(\Phi,\sigma) = 0\end{array}\right] \leq \mu(\lambda) \ .$$

Since Assign outputs $\Phi = \bot$ whenever $\mathsf{Core.V}(\mathsf{CRS},\Phi,\Pi^*) = 0$ we need to show

$$\Pr_{\mathsf{Assign}^{\mathsf{P}^*}(1^\lambda,S)}\left[\begin{array}{l}\mathsf{Core.V}(\mathsf{CRS}^*,\Phi,\Pi^*) = 1 \\ \mathsf{V}_{\mathsf{local}}(\Phi,\sigma) = 0\end{array}\right] < \mu(\lambda) \ .$$

Next, we prove that with overwhelming probability, if Core.V accepts, $\mathsf{V}_{\mathsf{local}}$ also accepts. To this end, we consider an intermediate verifier $\mathsf{V}_{\mathsf{clear}}$ that first decrypts the proof ciphertexts and then emulates the tests performed by the verifier Core.V "in the clear". We show that with overwhelming probability:

1. If Core.V accepts $\mathsf{V}_{\mathsf{clear}}$ also accepts. This follows from the correctness of the encryption zero-test.
2. If $\mathsf{V}_{\mathsf{clear}}$ accepts $\mathsf{V}_{\mathsf{local}}$ also accepts. This follows by a simple information theoretic argument.

The verifier $\mathsf{V}_{\mathsf{clear}}$. We define a new verification procedure $\mathsf{V}_{\mathsf{clear}}$ that emulates the verifier Core.V "in the clear". That is:

- Instead of operating over the ciphertexts in the CRS and in the proof, $\mathsf{V}_{\mathsf{clear}}$ operates on the corresponding plaintexts.
- Instead of homomorphically evaluating arithmetic circuits over the ciphertexts, $\mathsf{V}_{\mathsf{clear}}$ evaluates the same arithmetic circuits directly over the plaintexts.
- Instead of using the encryption's zero-test operation on the evaluated ciphertexts, $\mathsf{V}_{\mathsf{clear}}$ simply tests if the output of arithmetic circuit evaluate to zero.

Specifically, in the execution of $\mathsf{Assign}^{\mathsf{P}^*}(1^\lambda, S)$, let $\mathsf{CRS}_{\mathsf{clear}}$ be the elements encrypted in CRS^*

$$\mathsf{CRS}_{\mathsf{clear}} = \{\mathbf{u}^q \in \{0,1\}^m\}_{q\in[Q]} \ .$$

Let Π_{clear} be the elements encrypted in the proof Π^*. That is, for

$$\Pi^* = \left(\{d^q\}_{q\in[Q]} \quad , \quad \left\{\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}\right\}_{\mathbf{b}\in\{0,1\}^3,\mathbf{q}\in[Q]^3,i\in[3m]}\right) \ ,$$

let

$$\begin{aligned}
s^q &= \mathsf{MHE.Dec}(\mathsf{pp},\mathsf{sk}^q,d^q) \ , \\
\mathbf{t}_{i-1}^{\mathbf{q},\mathbf{b}} &= \left(\mathsf{MHE.Dec}\left(\mathsf{pp},(\mathsf{sk}^{q_1},\mathsf{sk}^{q_2},\mathsf{sk}^{q_3}),\left(\mathbf{e}_{i-1}^{\mathbf{q},\mathbf{b}}\right)_j\right)\right)_{j\in[0,\delta+1]} \ , \\
\Pi_{\mathsf{clear}} &= \left(\{s^q\}_{q\in[Q]} \quad , \quad \left\{\mathbf{t}_{i-1}^{\mathbf{q},\mathbf{b}}\right\}_{\mathbf{b}\in\{0,1\}^3,\mathbf{q}\in[Q]^3,i\in[3m]}\right)
\end{aligned}$$

Note that the above values may not be in $\{0,1\}$ and are therefore computed inefficiently via the algorithm MHE.Dec. Accordingly, the rest of the argument is information theoretic.

$\mathsf{V}_{\mathsf{clear}}$ is given as input $\mathsf{CRS}_{\mathsf{clear}}$, the arithmetic circuit Φ and Π_{clear}. If any of the elements in Π_{clear} is $\bot$, $\mathsf{V}_{\mathsf{clear}}$ rejects. Otherwise, $\mathsf{V}_{\mathsf{clear}}$ performs the following tests for every triplet of bits $\mathbf{b} = (b_1,b_2,b_3) \in \{0,1\}^3$ and triplet of queries $\mathbf{q} = (q_1,q_2,q_3) \in [Q]^3$ and accepts only if all tests pass.

For $i \in [3m]$ let

$$\tilde{P}^{\mathbf{b}}_{i-1}(x) = \prod_{j \in [0,\delta+1]} \left(\mathbf{t}^{\mathbf{q},\mathbf{b}}_{i-1}\right)_j \cdot x^j \ .$$

Let $\mathbf{u} = \mathbf{u}^{q_1} \mid \mathbf{u}^{q_2} \mid \mathbf{u}^{q_3}$, and let

$$\begin{aligned} v'_0 &= \Phi(\mathbf{u}, \mathbf{b}) \cdot \prod_{k \in [3]} (1 - \beta(b_k, s^{q_k})) \ , \\ v_{i-1} &= \tilde{P}^{\mathbf{b}}_{i-1}(\mathbf{u}_i) \ , \\ v'_i &= \sum_{z \in \{0,1\}} \beta(z, \mathbf{u}_i) \cdot \tilde{P}^{\mathbf{b}}_{i-1}(z) \ , \\ v_{3m} &= 0 \ . \end{aligned}$$

For every $i \in [0, 3m]$, $\mathsf{V}_{\mathsf{clear}}$ tests that $v_i = v'_i$.

By the zero-test soundness and the correctness for adversarial ciphertexts properties of the encryption there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and every set of indices $S \subseteq \{0,1\}^m$ of size at most Q

$$\Pr_{\mathsf{Assign}^{\mathsf{P}^*}(1^\lambda, S)} \left[\begin{array}{l} \mathsf{Core.V}(\mathsf{CRS}^*, \Phi, \Pi^*) = 1 \\ \mathsf{V}_{\mathsf{clear}}(\mathsf{CRS}_{\mathsf{clear}}, \Phi, \Pi_{\mathsf{clear}}) = 0 \end{array} \right] < \mu(\lambda) \ .$$

Therefore, it remains to prove that for every $\lambda \in \mathbb{N}$ and every set of indices $S \subseteq \{0,1\}^m$ of size at most Q

$$\Pr_{\mathsf{Assign}^{\mathsf{P}^*}(1^\lambda, S)} \left[\begin{array}{l} \mathsf{V}_{\mathsf{clear}}(\mathsf{CRS}_{\mathsf{clear}}, \Phi, \Pi_{\mathsf{clear}}) = 1 \\ \mathsf{V}_{\mathsf{local}}(\Phi, \sigma) = 0 \end{array} \right] = 0 \ .$$

If $\mathsf{V}_{\mathsf{clear}}(\mathsf{CRS}_{\mathsf{clear}}, \Phi, \Pi_{\mathsf{clear}}) = 1$ then none of the elements in Π_{clear} are $\perp$ and for every $\mathbf{b} = (b_1, b_2, b_3) \in \{0,1\}^3$, every $q_1, q_2, q_3 \in [Q]^3$ and every $i \in [0, 3m]$, we have $v_i = v'_i$. Note that since $\mathbf{u}_i \in \{0,1\}$

$$v'_i = \sum_{z \in \{0,1\}} \beta(z, \mathbf{u}_i) \cdot \tilde{P}^{\mathbf{b}}_{i-1}(z) = \tilde{P}^{\mathbf{b}}_{i-1}(\mathbf{u}_i) = v_{i-1} \ ,$$

Therefore

$$\Phi(\mathbf{u}, \mathbf{b}) \cdot \prod_{k \in [3]} (1 - \beta(b_k, s^{q_k})) = v'_0 = v_0 = v'_1 = v_1 = \cdots = v_{3m} = 0 \ .$$

Since $\sigma(\mathbf{u}^q) = s^q$ it follows that

$$\Phi(\mathbf{u}^{q_1}, \mathbf{u}^{q_2}, \mathbf{u}^{q_3}, \mathbf{b}) = 1 \quad \Rightarrow \quad (\sigma(\mathbf{u}^{q_1}) = b_1) \vee (\sigma(\mathbf{u}^{q_2}) = b_2) \vee (\sigma(\mathbf{u}^{q_3}) = b_3) \ ,$$

and therefore $\mathsf{V}_{\mathsf{local}}(\Phi, \sigma)$ accepts.

4.4 The Augmented Circuit

Syntax. Let $\mathcal{U}$ be the universal language (see Section 2.3). The augmented-circuit transformation consists of deterministic polynomial time algorithms $(\mathsf{Aug}, \mathsf{Aug}^{-1}, \mathsf{Trans})$ with the following syntax.

Aug**:** the circuit-augmentation procedure takes as input an instance $x = (M, y)$ and a time bound T for $\mathcal{U}$. It outputs an arithmetic circuit Φ computing the indicator function $\mathcal{I}_\varphi$ of the "augmented formula" φ (see Section 4.2)). We say that Φ *represents* φ.

Aug^{-1}**:** the inversion procedure takes as input an arithmetic circuit Φ. It either outputs (x, T) or fails and outputs $\perp$.

Trans: the assignment generation procedure takes as input an instance x and a time bound T for $\mathcal{U}$. It outputs an assignment σ for φ.

These procedures satisfy the following properties:

Efficiency. For $x \in \{0,1\}^n$

- $\mathsf{Aug}(x, \mathsf{T})$ runs in time $n \cdot \mathrm{polylog}(\mathsf{T})$ and outputs an arithmetic circuit Φ such that
  - Φ is of size $n \cdot \mathrm{polylog}(\mathsf{T})$.
  - Φ is of total degree $\delta = \delta(n, \mathsf{T}) = \mathrm{polylog}(n, \mathsf{T})$
  - Φ represents a formula φ on $B = B(n, \mathsf{T}) = \mathrm{poly}(n, \mathsf{T})$ variables.
- $\mathsf{Aug}(x, \mathsf{T})$ and $\mathsf{Aug}^{-1}(\Phi)$ run in time $n \cdot \mathrm{polylog}(\mathsf{T})$.
- $\mathsf{Trans}(x, \mathsf{T})$ runs in time $\mathrm{poly}(n, \mathsf{T})$.

Inversion. For every $(x, \mathsf{T}) \in \{0,1\}^*$

$$\mathsf{Aug}^{-1}(\mathsf{Aug}(x, \mathsf{T})) = (x, \mathsf{T}) \ .$$

Completeness. For every $(x, \mathsf{T}) \in \mathcal{U}$, $\mathsf{Trans}(x, \mathsf{T})$ outputs a satisfying assignment σ for the formula φ represented by the output of $\mathsf{Aug}(x, \mathsf{T})$.

Soundness. At a high level, the soundness guarantees that there does not exist an adaptive local-assignment generator (see Section 4.2) that for *every* small set of indices S generates a circuit $\Phi = \mathsf{Aug}(x, \mathsf{T})$, such that $(x, \mathsf{T}) \notin \mathcal{U}$, together with partial assignment $\sigma : S \to \{0,1\}$ that is locally consistent with the formula represented by Φ.

Lemma 4.3 (Augmented Circuit Soundness). *There exists a function $Q = \mathrm{polylog}(\lambda)$ such that for every polynomially bounded function $B = B(\lambda)$, and every polynomial-time Q-local-assignment generator* Assign *for B-variate formulas there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, letting $B = 2^m$, there exists a set $S^* \subseteq \{0,1\}^m$ of size at most Q such that*

$$\Pr\left[\begin{array}{l} \Phi \neq \perp \\ \mathsf{Aug}^{-1}(\Phi) \notin \mathcal{U} \cup \{\perp\} \end{array} \;\middle|\; (\Phi, \sigma) \leftarrow \mathsf{Assign}(1^\lambda, S^*) \right] \leq \mu(\lambda) \ .$$

4.4.1 Transformation outline.

We provide a proof sketch, focusing on guaranteeing the *adaptive* soundness property. A full description of the augmented circuit construction in the non-adaptive setting can be found in [KRR14, PR14]. Recall that the augmented-circuit transformation takes an instance (x, T) where $x \in \{0,1\}^n$, and outputs an arithmetic circuit Φ of size $n \cdot \mathrm{polylog}(\mathsf{T})$ and degree $\mathrm{polylog}(n, \mathsf{T})$ that describes a 3CNF. The construction is identical to the one in [PR14], but we prove the stronger *adaptive* soundness property stated above.

Construction. We describe the construction at a high level, we refer the reader to [PR14] for details. We consider a boolean circuit of size $\mathrm{poly}(n, \mathsf{T})$ that runs the computation of the universal Turing machine on input x for T steps. The circuit is *layered*: it is divided into T layers, one for each step of the machine, and every wire is connecting two successive layers. The Cook-Levin reduction gives such a circuit. We "augment" each layer ℓ in this circuit, by computing a low-degree extension of the gate-values in that layer. The low-degree extension is taken over finite field of size $\mathrm{polylog}(n, \mathsf{T})$. In addition, for each layer ℓ, the augmented circuit runs a low-degree test to verify that restricting the low-degree extension to each possible line yields a low-degree univariate polynomial. The low-degree extension, and the low-degree tests (one for each line) are performed as part of each layer ℓ in the augmented circuit. We note that the low-degree extension and the low-degree tests are redundant in the sense that they do not "help" the circuit compute the correct functionality, but they add redundancy that is helpful for obtaining soundness.

This augmented circuit was described as an arithmetic circuit over a small finite field. We transform it into a boolean circuit by replacing each wire carrying an arithmetic value with a collection of boolean wires carrying a representation of that value, and replacing each arithmetic gate with boolean sub-circuit implementing its finite field operations. We obtain a 3CNF from this boolean circuit, where we have a variable for each internal wire in the circuit and for each input and output wire. We add clauses to ensure that: (i) each gate's two input wires are consistent with each of its output wires (as in the Cook-Levin reduction), (ii) the input wires are consistent with the input y, and (iii) the output wires of every low-degree test and of the entire circuit all have the value 1. This fully specifies the CNF φ produced by the augmented-circuit transformation.

We observe that this construction lends itself naturally to producing a circuit Φ of size $n \cdot \mathrm{polylog}(\mathsf{T})$ and degree $\mathrm{polylog}(n, \mathsf{T})$ that describes φ. Given the names of three literals (negated or unnegated variables), Φ determines whether these three literals form a clause in φ. This follows from the fact that the augmented boolean circuit described above is "constructible", see the details in [PR14]. Finally, the construction is invertible because we hardwire the instance (x, T) into the arithmetic circuit Φ.

Setup and Notations. Let Q be the bound on the size of the set S in the soundness condition. For a set W of variables in the CNF φ (corresponding to wires in the augmented circuit) of size at most Q, and for a sample $(\Phi, \sigma) \leftarrow \mathsf{Assign}(1^\lambda, W)$, we say that (Φ, σ) is *correct* on a subset $W' \subseteq W$ of variables, if the assigned values $\sigma|_{W'}$ are consistent with the (unique) satisfying assignment to the CNF described by Φ. That is, the assignment is consistent with the wire values of a correct computation of the boolean augmented circuit on x. We denote this event by $\mathsf{CR}((\Phi, \sigma), W')$. We emphasize that correctness is only measured with respect to the computation described by the instance (x, T) that is embedded in Φ. In particular, *a correct assignment need not be locally consist.* This can happen when $(x, \mathsf{T}) \notin \mathcal{U}$: the *correct* assignment to the variable corresponding to the output wire equals 0, and it does not satisfy the clause checking that the variable corresponding to the output wire equals 1. Intuitively, local consistency is a condition that can be checked easily given access to Φ (That is, in polylogarithmic time). Correctness of a variable, on the other hand, cannot be checked easily: in the above example, ascertaining correctness of the output wire variable requires running the entire computation.

For each layer $\ell \in [\mathsf{T}]$ of the augmented circuit, we keep track of c random variables. These random variables are sampled by picking $c = \log^2(\lambda)$ independent and uniformly random points in the LDE of layer ℓ. We denote these random variables by $W_\ell = (\mathbf{z}_{\ell,1}, \ldots, \mathbf{z}_{\ell,c})$, where each $\mathbf{z}_{\ell,i}$ is an independently random location in the LDE of layer ℓ. In the soundness analysis we consider (and analyze) various events defined over these random variables. We emphasize that the randomness is always taken over the choice of these points in W_ℓ (unless we explicitly note otherwise). We abuse notation by referring to W_ℓ (which describe coordinates in the LDE) as variables in the CNF described by Φ, rather than referring to the *boolean* variables carrying bits encoding the field elements in the chosen coordinates of layer ℓ's LDE.

Let (Φ, σ) be a sample from $\mathsf{Assign}(1^\lambda, W_\ell)$. Let CR_ℓ be the event $\mathsf{CR}((\Phi, \sigma), W_\ell)$, indicating that the assignment in σ is correct for the CNF described by Φ on the variables W_ℓ.

Soundness Overview. Suppose for contradiction that there exists an adaptive local-assignment generator Assign and a polynomial $p(\lambda)$, such that for every variable set S of size at most Q, when Assign runs on S, with probability at least $1/p(\lambda)$ it output (Φ, σ) such that $\Phi \neq \perp$ is the output of Aug on an instance $(x, \mathsf{T}) \notin \mathcal{U}$. We show that this implies a contradiction.

As a first step, we amplify Assign's success probability. Consider Assign' that on input S runs $(\lambda \cdot p(\lambda))$ independent executions of Assign until either Assign's outputs (Φ, σ) that satisfies the above condition, in which case Assign' outputs (Φ, σ), or if all iterations fail then Assign' fails and outputs $\Phi = \bot$. Note that the satisfiability of Φ can be checked in polynomial time. This amplification step preserves the no-signaling property since Assign' filters the output of Assign based only on the value of Φ and independently of the input S. It follows that Assign' is also a polynomial-time adaptive local-assignment generator.

Similarly to the proof of [KRR14] (see the description in [PR14]), we use an inductive argument. For (Φ, σ) produced by Assign', let $\neg\mathsf{SAT}$ denote the event that Φ is Aug's output on an input that is not in $\mathcal{U}$. The main difference from prior works showing non-adaptive soundness, is that here we continually keep track of whether the event $\neg\mathsf{SAT}$ occurs. By the above, for every set S, this event occurs with all but negligible probability. We show that for every layer i, the probability that the event $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$ occurs is almost 1. This is done via an inductive argument:

1. For the input layer (the induction basis), local consistency implies that for any subset $T \subseteq S$ of variables corresponding to input wires of the augmented circuit, when Assign' outputs (Φ, σ) that are locally consistent, the values that σ assigns to variables in T agree with the instance x and are thus correct (in addition to being locally-consistent). This is simply because φ checks that variables corresponding to input wires are assigned the correct values.

   Thus when we run Assign' on any set T of variables corresponding to input wires, with all but negligible probability, the event $\neg\mathsf{SAT}$ occurs and these variables are all assigned correct values. Now when we extend to asking about sets $S \supseteq T$, by the no-signalling property of Assign, the same must hold. Local consistency guarantees that any variables in S corresponding to wires whose values are determined by the input wires in T are also correct (up to some negligible loss).

   We can continue to use local consistency together with the no-signalling guarantee, and conclude that when we run Assign on *any* set S of variables corresponding to (any) wires in the input layer (including wires that compute the low-degree extension), with all but negligible probability the event $\neg\mathsf{SAT}$ occurs and σ is correct on every variable in S. This uses the fact that all wires within in the layer of the augmented circuit can be computed by a circuit of logarithmic depth, so the negligible errors we incur due to signalling are not amplified too much, and remain negligible. We conclude in particular that the probability of $(\mathsf{CR}_1 \wedge \neg\mathsf{SAT})$ is $1 - \mathrm{negl}(\lambda)$.

2. In the inductive step, we show that if the probability of $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$ is at least $p \geq 0.99$, then the probability of $(\mathsf{CR}_{i+1} \wedge \neg\mathsf{SAT})$ is at least $p - \mathrm{negl}(\lambda)$.

   This follows similarly to the inductive step in [KRR14]. Fix any wire w in layer i. If we sample a set $U \leftarrow (\{w\} \cup W_i)$ and query $(\Phi, \sigma) \leftarrow \mathsf{Assign}'(1^\lambda, U)$ and condition on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$, then with all but negligible probability σ is also correct for the variable w. We emphasize that this negligible failure probability (conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$) does not grow with the layer (or with the probability the $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$ doesn't occur). This argument uses the low degree tests and no-signalling of Assign', see [KRR14] (and the presentation in [PR14]).

   To complete the inductive step, we show that conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$, we also have that when we query any variable w in W_{i+1}, we get that with all but negligible probability $\neg\mathsf{SAT}$ occurs and the assignment to w is correct. As above, this uses local consistency, which holds for every gate in the computation of layer $(i+1)$'s LDE from the LDE of layer i. Local consistency guarantees that, for each gate in this computation, the probability of an error in the output wire (conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$), is at most the sum of error probabilities in the input wires (conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$). Since the computation of the LDE has only logarithmic depth, we get that the accumulated error probability for each point in W_{i+1} (conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$) remains negligible. Taking a union bound over all points in W_{i+1}, we get that they are all correct with all but negligible probability (conditioned on $(\mathsf{CR}_i \wedge \neg\mathsf{SAT})$). The inductive step follows.

By induction, we conclude that also for the output layer, the probability of $(\mathsf{CR}_\mathsf{T} \wedge \neg\mathsf{SAT})$ is $1 - \mathrm{negl}(\lambda)$. Similarly to the above, this implies that when we query a variable corresponding to any wire in layer T, with all but negligible probability $\neg\mathsf{SAT}$ occurs and the assignment given to this variable is correct. In particular, this is true for the output wire w^*. We conclude that when we query $(\Phi, \sigma) \leftarrow \mathsf{Assign}'(1^\lambda, \{w^*\})$, with all but negligible probability $\neg\mathsf{SAT}$ occurs and σ gives the correct value to w^*. This correct value is 0 (because $\neg\mathsf{SAT}$ occurred), whereas local consistency requires that the variable corresponding to the output wire gets value 1. Thus, local consistency doesn't hold, and we obtain a contradiction.

4.5 The Final Protocol

In this section we combine the core protocol (Section 4.3) with the augmented-circuit transformation (Section 4.4) to get our full non-interactive argument.

4.5.1 Construction.

Let $(\mathsf{Aug}, \mathsf{Aug}^{-1}, \mathsf{Trans})$ be the the augmented-circuit transformation. Let $\delta = \delta(n, \mathsf{T})$ be the total degree of the circuit Φ generated by the circuit-augmentation procedure. Let $\Delta = \Delta(\lambda)$ be polynomially bounded functions such that for every polynomials $n = n(\lambda), \mathsf{T} = \mathsf{T}(\lambda)$ we have that $\Delta(n) = \omega(\delta(n(\lambda), \mathsf{T}(n)))$. Let $(\mathsf{Core.Gen}, \mathsf{Core.P}, \mathsf{Core.V})$ be the core protocol for degree bound Δ.

The CRS generator. The CRS generation algorithm $\mathsf{Del.Gen}$ is given as input the security parameter 1^λ. Let $Q = Q(\lambda)$ be the locality parameter given by Lemma 4.3. $\mathsf{Del.Gen}$ invokes the core protocol CRS generator

$$\mathsf{CRS} \leftarrow \mathsf{Core.Gen}(1^\lambda, 1^Q) \ .$$

The prover. The prover algorithm $\mathsf{Del.P}$ is given as input the common reference string CRS, a time bound 1^T and an instance $x \in \{0,1\}^\lambda$. It outputs a proof Π. If $(x, \mathsf{T}) \notin \mathcal{U}$, $\mathsf{Del.P}$ aborts. Otherwise $\mathsf{Del.P}$ obtains the circuit $\Phi = \mathsf{Aug}(x, \mathsf{T})$ representing a formula φ with $B = 2^m$ variables and a satisfying assignment $\sigma \leftarrow \mathsf{Trans}(x, \mathsf{T})$ for φ. It invokes the core protocol prover to compute the proof

$$\Pi \leftarrow \mathsf{Core.P}(\mathsf{CRS}, \Phi, \sigma) \ .$$

The verifier. The verifier algorithm $\mathsf{Core.V}$ is given as input the common reference string CRS, a time bound T, an instance $x \in \{0,1\}^\lambda$ and a proof Π. It obtains the circuit $\Phi = \mathsf{Aug}(x, \mathsf{T})$ and invokes the core protocol verifier and output b

$$b \leftarrow \mathsf{Core.V}(\mathsf{CRS}, \Phi, \Pi) \ .$$

4.5.2 Analysis.

The correctness and efficiency properties of the of the final protocol follow directly from the correctness and efficiency properties of the core protocol and the augmented-circuit transformation. We prove the adaptive soundness property.

Assume towards contradiction that there exits a polynomial $\mathsf{T} = \mathsf{T}(\lambda)$, a poly-size cheating prover P^* and a polynomial p such that for infinity many $\lambda \in \mathbb{N}$

$$\Pr\left[\begin{array}{l} (x^*, \mathsf{T}) \notin \mathcal{U} \\ \mathsf{Del.V}(\mathsf{CRS}, \mathsf{T}, x^*, \Pi^*) = 1 \end{array} \middle| \begin{array}{l} \mathsf{CRS} \leftarrow \mathsf{Del.Gen}(1^\lambda) \\ (x^*, \Pi^*) \leftarrow \mathsf{P}^*(\mathsf{CRS}) \end{array}\right] \geq \frac{1}{p(\lambda)} \ .$$

By the definition of the protocol, for every such λ

$$\Pr\left[\begin{array}{l}(x^*,\mathsf{T})\notin\mathcal{U}\\ \mathsf{Core.V}(\mathsf{CRS},\Phi,\Pi^*)=1\end{array}\;\middle|\;\begin{array}{l}\mathsf{CRS}\leftarrow\mathsf{Core.Gen}(1^\lambda,1^Q)\\ (x^*,\Pi^*)\leftarrow\mathsf{P}^*(\mathsf{CRS})\\ \Phi\leftarrow\mathsf{Aug}(x^*,\mathsf{T})\end{array}\right]\geq\frac{1}{p(\lambda)}\ .$$

Let P' be a cheating prover that given CRS emulates P^*, obtains (x^*,Π^*), obtains the augmented circuit $\Phi\leftarrow\mathsf{Aug}(x^*,\mathsf{T})$ and outputs (Φ,Π^*). By the properties of the inversion procedure Aug^{-1} we have that

$$\Pr\left[\begin{array}{l}\mathsf{Aug}^{-1}(\Phi)\notin\mathcal{U}\cup\{\perp\}\\ \mathsf{Core.V}(\mathsf{CRS},\Phi,\Pi^*)=1\end{array}\;\middle|\;\begin{array}{l}\mathsf{CRS}\leftarrow\mathsf{Core.Gen}(1^\lambda,1^Q)\\ (\Phi,\Pi^*)\leftarrow\mathsf{P}'(\mathsf{CRS})\end{array}\right]\geq\frac{1}{p(\lambda)}\ .$$

By the adaptive local soundness of the protocol there exists a polynomial $B=B(\lambda)$, an adaptive Q-local-assignment generator for B-variate formulas $\mathsf{Assign}^{\mathsf{P}^*}$ and a polynomial p' such that for infinity many $\lambda\in\mathbb{N}$, letting $B=2^m$, for every set of indices $S\subseteq\{0,1\}^m$ of size at most Q

$$\Pr\left[\begin{array}{l}\mathsf{Aug}^{-1}(\Phi)\notin\mathcal{U}\cup\{\perp\}\\ \mathsf{V}_{\mathsf{local}}(\Phi,\sigma)=1\end{array}\;\middle|\;(\Phi,\sigma)\leftarrow\mathsf{Assign}^{\mathsf{P}'}(1^\lambda,S)\right]\geq\frac{1}{p'(\lambda)}\ .$$

In contradiction to the augmented circuit soundness property.

5 Zero-Testable Homomorphic Encryption from Graded Encodings

In this section we construct multi-key zero-testable somewhat homomorphic encryption from graded encodings.

5.1 Graded Encodings

We start by defining the notion of graded encoding. A graded encoding scheme encodes elements of some ring R. Every element is encoded with respect to a level $\delta\in[0,\Delta]$. It is possible to homomorphically evaluate algebraic operation on the encodings, subject to some simple constraints on their levels, and it is also possible to identify encodings of zero. While the encoding and decoding operations require the secret parameters, the homomorphic operation and zero test only require the public parameters. It is also possible to sample a level-0 encoding of a random element and to re-randomize an arbitrary encoding given only the public parameters. We note that not all applications of graded encodings require a re-randomization operation.

Syntax. A graded encoding scheme consists of PPT algorithms

$$(\mathsf{GE.ParamGen},\mathsf{GE.Enc},\mathsf{GE.Dec},\mathsf{GE.Samp},\mathsf{GE.Eval},\mathsf{GE.Rand},\mathsf{GE.ZT})\ ,$$

with the following syntax.

$\mathsf{GE.ParamGen}$**:** Given the security parameter 1^λ and a maximal level 1^Δ, outputs secret parameters sp, public parameters pp, and the description of a ring R.

$\mathsf{GE.Enc}$**:** Given the secret parameters sp, a ring element $\alpha\in R$ and a level $\delta\in[0,\Delta]$ outputs an encoding e.

$\mathsf{GE.Dec}$**:** Given the secret parameters sp and an encoding e outputs an element $\alpha\in R\cup\{\perp\}$ and a level $\delta\in[0,\Delta]$.

$\mathsf{GE.Samp}$**:** Given the public parameters pp outputs a level-0 encoding e of a random element.

$\mathsf{GE.Eval}$**:** Given the public parameters pp, an operation $\star \in \{+, -, \times\}$, and a pair of encodings e_1, e_2, outputs an encoding e or a special symbol $\perp$.

$\mathsf{GE.Rand}$**:** Given the public parameters pp and an encoding e, outputs a new random encoding e'.

$\mathsf{GE.ZT}$**:** Given the public parameters pp and an encoding e, outputs a bit.

Evaluating circuits. Homomorphic evaluation of an arithmetic circuit C is implemented by iteratively applying the basic evaluation algorithm $\mathsf{GE.Eval}$ for every gate in C. For simplicity, we only consider arithmetic circuits without constants. We assume that the public parameters include level-0 encodings $[0]_0$ and $[1]_0$ of 0 and 1 respectively, and a level-1 encoding $[1]_1$ of 1.

For an arithmetic circuit C, and encodings $(e_1, \ldots, e_n)$ under public parameters pp we denote by $\langle C(e_1, \ldots, e_n) \rangle$ the evaluated encoding e computed as follows.

- If C is the i-th input wire then $e = e_i$.
- If C is of the form $C = C_1 \star C_2$ then

$$e = \mathsf{GE.Eval}\left(\mathsf{pp}, \star, \left(\langle C_1(e_1, \ldots, e_n)\rangle, \langle C_2(e_1, \ldots, e_n)\rangle\right)\right) \ .$$

We also consider a randomized evaluation of an arithmetic circuit C where after every application of the the basic evaluation algorithm $\mathsf{GE.Eval}$ the resulting encoding is randomized with the operation $\mathsf{GE.Rand}$. We denote by $\langle C(e_1, \ldots, e_n) \rangle_{\mathsf{rand}}$ the evaluated encoding e computed as follows.

- If C is the i-th input wire then $e = \mathsf{GE.Rand}(\mathsf{pp}, e_i)$.
- If C is of the form $C = C_1 \star C_2$ then

$$e = \mathsf{GE.Rand}\left(\mathsf{pp}, \mathsf{GE.Eval}\left(\mathsf{pp}, \star, \left(\langle C_1(e_1, \ldots, e_n)\rangle_{\mathsf{rand}}, \langle C_2(e_1, \ldots, e_n)\rangle_{\mathsf{rand}}\right)\right)\right) \ .$$

Valid arithmetic circuits. Next we define a set of valid arithmetic circuits that can be correctly evaluated over a sequence of encodings. Encodings in the same level can be added and subtracted correctly and the result is an encoding in the same level. Encoding can be multiplied correctly if the sum of their levels does not exceed Δ and the result is an encoding in the sum of the levels. An arithmetic circuit C taking n inputs is valid with respect to a maximal level Δ and a sequence of levels $(\delta_1, \ldots, \delta_n)$ if there exists a function Lvl from the wires of C to $\mathbb{N}$ such that

- The i-th input wire of C satisfies $\mathsf{Lvl}(w_i) = \delta_i$.
- For every addition and subtraction gate in C connecting wires w_i and w_j to a wire w_k,

$$\mathsf{Lvl}(w_i) = \mathsf{Lvl}(w_j) = \mathsf{Lvl}(w_k) \ .$$

- For every multiplication gate in C connecting wires w_i and w_j to a wire w_k,

$$\mathsf{Lvl}(w_i) + \mathsf{Lvl}(w_j) = \mathsf{Lvl}(w_k) \ .$$

- The output wire w of C satisfies $\mathsf{Lvl}(w) \leq \Delta$.

For a valid C we denote by $\mathsf{Lvl}(C)$ the value $\mathsf{Lvl}(w)$ for the output wire w.

Adversarially generated encodings. The notion of graded encoding formulated below differs from previous formulation in the literature and requires correctness of homomorphic evaluation even for encodings that are adversarially generated. Informally, we require that an efficient adversary cannot generate a pair of encodings that cause en evaluation error. A homomorphic evaluation of an operation $\star$ on encodings e_1, e_2 is erroneous if the following two experiments have different outputs

- Homomorphically evaluate $e_1 \star e_2$ and output the decryption of the evaluated encoding.
- Decrypt e_1 and e_2. If one of the encoding fails to decrypt (decryption output $\perp$) then output $\perp$. Otherwise output the evaluation of $\star$ on the decrypted element.

We note that graded encodings that are so-called "clean" (see, for example [Zim15, LV16]) where every element has a unique encoding, trivially satisfy correctness for adversarially generated encodings, and support re-randomization. However in existing graded encodings candidates, encodings have noise that grows with the number of homomorphic operations. When evaluating circuits beyond a certain size, the noise growth results in an evaluation error. Therefore, in such candidates, encodings that cause en evaluation error are easy to generate. We discuss the possibility of supporting such "noisy" candidates in Section 5.4.

Definition 5.1 (Graded Encoding)**.** *Let $\Delta = \Delta(\lambda)$ be a polynomially bounded function. A graded encoding scheme*

$$(\mathsf{GE.ParamGen}, \mathsf{GE.Enc}, \mathsf{GE.Dec}, \mathsf{GE.Samp}, \mathsf{GE.Eval}, \mathsf{GE.Rand}, \mathsf{GE.ZT}) \ ,$$

satisfies the following requirements.

Correctness for Adversarial Encodings: *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, every pair of levels $\delta_1, \delta_2 \in [0, \Delta]$ and every arithmetic circuit C of the form $C(x_1, x_2) = x_1 \star x_2$ such that $\star \in \{+, -, \times\}$ and such that C is valid with respect to Δ and (δ_1, δ_2)*

$$\Pr\left[\begin{array}{l|l} & (\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\ (\delta'_1, \delta'_2) = (\delta_1, \delta_2) \Rightarrow \delta' = \mathsf{Lvl}(C) & e_1, e_2 \leftarrow \mathsf{Adv}(\mathsf{pp}) \\ \alpha = C(\alpha_1, \alpha_2) & e \leftarrow \langle C(e_1, e_2) \rangle \\ & \forall i \in \{1, 2\} : (\alpha_i, \delta'_i) \leftarrow \mathsf{GE.Dec}(\mathsf{sp}, e_i) \\ & (\alpha, \delta') \leftarrow \mathsf{GE.Dec}(\mathsf{sp}, e) \end{array}\right] \geq 1 - \mu(\lambda) \ ,$$

where in the probability above, if $\alpha_1, \alpha_2 \in R$, then $C(\alpha_1, \alpha_2)$ is evaluated over R. If either $\alpha_1 = \perp$ or $\alpha_2 = \perp$ then $C(\alpha_1, \alpha_2) = \perp$.

Compactness: *There exists a polynomial L such that for every secret parameters* sp *in the support of* $\mathsf{GE.ParamGen}(1^\lambda, 1^\Delta)$ *and for every encoding e*

$$|e| \geq L(\lambda, \Delta) \Rightarrow \mathsf{GE.Dec}(\mathsf{sp}, e) = \perp \ .$$

Sampling: *For every polynomial-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{l|l} & b \leftarrow \{0, 1\} \\ & (\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\ b = b' & \alpha \leftarrow R \\ & e_0 \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha, 0) \\ & e_1 \leftarrow \mathsf{GE.Samp}(\mathsf{pp}) \\ & b' \leftarrow \mathsf{Adv}^{O_R, \mathsf{GE.Enc}(\mathsf{sp}, \cdots)}(\mathsf{pp}, e_b) \end{array}\right] \leq \frac{1}{2} + \mu(\lambda) \ ,$$

where O_R is an oracle that samples random elements from R and implements the operations $\{+, -, \times\}$ in R.

Re-Randomization: *For every polynomial-size adversaries* $\mathsf{Adv}_1, \mathsf{Adv}_2$ *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$*, every pair of levels* $\delta_1, \delta_2 \in [0, \Delta]$ *and every arithmetic circuit* C *of the form* $C(x_1, x_2) = x_1 \star x_2$ *such that* $\star \in \{+, -, \times\}$ *and such that* C *that is valid with respect to* Δ *and* (δ_1, δ_2)

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} b \leftarrow \{0,1\} \\ (\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\ \alpha_1, \alpha_2 \leftarrow \mathsf{Adv}_1^{O_R}(\mathsf{pp}) \\ e_i \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha_i, \delta_i) \\ f_0 \leftarrow \langle C\,(e_1, e_2) \rangle_{\mathsf{rand}} \\ f_1 \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, C(\alpha_1, \alpha_2), \mathsf{Lvl}(C)) \\ b' \leftarrow \mathsf{Adv}_2^{O_R, \mathsf{GE.Enc}(\mathsf{sp}, \cdots)}(\mathsf{pp}, e_1, e_2, f_b) \end{array} \right] \leq \frac{1}{2} + \mu(\lambda) \ .$$

where C *is evaluated over* R *and where* O_R *is an oracle that samples random elements from* R *and implements the operations* $\{+, -, \times\}$ *in* R.

Zero Test: *For every polynomial-size adversary* Adv *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$,

$$\Pr\left[\alpha = 0 \Leftrightarrow b = 1 \;\middle|\; \begin{array}{l} (\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\ e \leftarrow \mathsf{Adv}(\mathsf{pp}) \\ (\alpha, \delta) \leftarrow \mathsf{GE.Dec}(\mathsf{sp}, e) \\ b \leftarrow \mathsf{GE.ZT}(\mathsf{pp}, e) \end{array} \right] \geq 1 - \mu(\lambda) \ ,$$

where C *is evaluated over* R.

Remark 5.2. Applications of the graded encoding scheme may rely on the sampling and re-randomization properties for multiple encodings. Therefore in Definition 5.1 we require that the sampling and re-randomization properties hold even when the adversary can sample random elements from the ring R, evaluate the operations $\{+, -, \times\}$, and encode ring elements. Instead of giving the adversary the secret parameters sp and the description of the ring R, we formulate a weaker requirement where the adversary is only given oracle access to the procedure $\mathsf{GE.Enc}(\mathsf{sp}, \cdots)$ and access to an oracle O_R that samples a random element from R and implements the operations $\{+, -, \times\}$ in R.

We consider the following computational assumption. Informally, we assume that given level-1 encodings of random coefficients $\alpha_0, \ldots, \alpha_\Delta$ it is hard to distinguish a level-1 encoding of a root of the polynomial with coefficients $\{\alpha_i\}$ from an encoding of a random element.

Assumption 5.3. *Let* $\Delta = \Delta(\lambda)$ *be a polynomially bounded function. For every poly-size adversary* Adv *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[b = b' \;\middle|\; \begin{array}{l} b \leftarrow \{0,1\} \\ (\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\ \alpha_0^0, \alpha_1, \ldots \alpha_\Delta, t \leftarrow R \\ \alpha_0^1 = -\sum_{i \in [\Delta]} \alpha_i \cdot t^i \\ f \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, t, 1) \\ e_0 \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha_0^b, 1) \\ \forall i \in [\Delta] : e_i \leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha_i, 1) \\ b' \leftarrow \mathsf{Adv}(\mathsf{pp}, f, e_0, \ldots e_\Delta) \end{array} \right] \leq \frac{1}{2} + \mu(\lambda) \ .$$

5.2 Construction

Let $B = B(\lambda), \Delta = \Delta(\lambda)$ be polynomially bounded functions. Let $d \in \mathbb{N}$ be a constant. Let

$$(\mathsf{GE.ParamGen}, \mathsf{GE.Enc}, \mathsf{GE.Dec}, \mathsf{GE.Samp}, \mathsf{GE.Eval}, \mathsf{GE.Rand}, \mathsf{GE.ZT}) \ ,$$

be a graded encoding scheme satisfying Assumption 5.3. We construct d-key zero-testable (B, Δ)-somewhat homomorphic encryption scheme

$$(\mathsf{MHE.ParamGen}, \mathsf{MHE.KeyGen}, \mathsf{MHE.Enc}, \mathsf{MHE.Dec}, \mathsf{MHE.WeakDec}, \mathsf{MHE.Eval}, \mathsf{MHE.ZT}) \ .$$

Parameter generation. The parameter-generation procedure $\mathsf{MHE.ParamGen}$ is given the security parameter 1^λ. It generates parameters for the graded encoding scheme

$$(\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \ .$$

It outputs the public parameters pp and the ring R (the secret parameters are not used).

Key generation. The parameter generation procedure $\mathsf{MHE.ParamGen}$ is given the public parameters pp. It outputs a secret key sk and a public key pk. The secret key is a random level-0 encoding, and the public key is a random level level-1 encoding of the same element. That is, the public key is homomorphically computed by multiplying the secret key encoding with a level-1 encoding of 1 and re-randomizing.

$$\mathsf{sk} \leftarrow \mathsf{GE.Samp}(\mathsf{pp}) \ ,$$
$$\mathsf{pk} \leftarrow \langle \mathsf{sk} \times [1]_1 \rangle_{\mathsf{rand}} \ .$$

Ciphertext structure. Before describing the encryption and decryption procedure we describe the structure of a ciphertext. A ciphertext consist of the following components

Public keys. Any ciphertext is encrypted under a set $\mathbf{pk}$ of at most d public keys. A "fresh" ciphertext which is generated by $\mathsf{MHE.Enc}$ is always encrypted under a singleton set. When evaluating a homomorphic operation on a pair of ciphertexts encrypted under sets $\mathbf{pk}_1$ and $\mathbf{pk}_2$, the resulting ciphertext is encrypted under $\mathbf{pk}_1 \cup \mathbf{pk}_2$.

Degree. Any ciphertext has a degree $\delta \in [\Delta]$. A "fresh" ciphertext has degree 0. When adding or subtracting a pair of ciphertexts of degrees δ_1 and δ_2, the resulting ciphertext is of degree $\max(\delta_1, \delta_2)$. When multiplying the ciphertexts, the resulting ciphertext is of degree $\delta_1 + \delta_2$. Note that the degree of a ciphertext never exceeds Δ, since the encryption only supports homomorphic evaluation of circuit of total degree Δ.

Encoded coefficients. Any ciphertext contains an (encoded) description of a polynomial P over $|\mathbf{pk}|$ variables, and of total degree $(\delta \cdot \Delta)$. Each variable is associated with one of the public keys in $\mathbf{pk}$. A ciphertext encrypting a message $m \in \{0, 1\}$ satisfies the following property. When the variable associated with the public key $\mathsf{pk} \in \mathbf{pk}$ is assigned the value encoded by pk, P evaluates to m .

The polynomial P is represented by random level-δ encoding of its coefficients. Every coefficient is indexed by a function g describing its monomial. The function g maps every public key in $\mathbf{pk}$ to the power its associated variable appears in the monomial. The sum powers in a monomial is bounded by P's total degree. Formally, let $F_{\mathbf{pk},\delta}$ be the set of functions

$$F_{\mathbf{pk},\delta} = \left\{ g : \mathbf{pk} \to [0, \Delta] \;\middle|\; \sum_{\mathsf{pk} \in \mathbf{pk}} g(\mathsf{pk}) \leq \Delta \cdot \delta \right\} \ .$$

We index the monomials of P by functions in $F_{\mathbf{pk},\delta}$.

Overall, a ciphertext c has the following structure

$$c = \left(\mathbf{pk}, \delta, \{e_g\}_{g \in F_{\mathbf{pk},\delta}}\right) \quad ,$$

where e_g is a level-δ encoding.

Encryption. The encryption procedure $\mathsf{MHE.Enc}$ is given a public parameters pp, the public key pk and a message $m \in \{0, 1\}$. It outputs a ciphertext

$$c = \left(\{\mathsf{pk}\}, 1, \{e_g\}_{g \in F_{\{\mathsf{pk}\},1}}\right) \quad .$$

The ciphertext c encodes a random univariate polynomial P of degree Δ that evaluates to m on the element α encoded by pk. e_g is the encoded level-1 coefficient of the monomial indexed by g. We simplify notation and identify each function $g \in F_{\{\mathsf{pk}\},1}$ with an integer $i \in [0, \Delta]$ such that $g(\mathsf{pk}) = i$. That is e_i is the degree-i coefficient of P. Next we describe how the encodings e_i are sampled.

Since e_i and pk are level-1 encodings, the coefficient encoded in e_i must be expressed as a linear function of α. To this end, we sample P as follows

- Sample a random linear combination of the the following Δ polynomials

$$\left\{x^i - \alpha x^{i-1}\right\}_{i \in [\Delta]} \quad .$$

  this is a random degree Δ polynomial that vanishes on 0.

- Add m to the free coefficient.

Specifically, the encodings $e_0, \ldots, e_\Delta$ are sampled as follows.

1. Sample encodings $r_0, \ldots, r_{\Delta+1}$ as follows
   - $r_0 = [m]_0$.
   - For $i \in [\Delta]$ sample a random level-0 encoding $r_i \leftarrow \mathsf{GE.Samp}(\mathsf{pp})$.
   - $r_{\Delta+1} = [0]_0$.
2. For $i \in [0, \Delta]$ let
$$e_i \leftarrow \langle r_i \times [1]_1 - r_{i+1} \times \mathsf{pk} \rangle_{\mathsf{rand}} \quad .$$

Decryption. The inefficient decryption procedure $\mathsf{MHE.Dec}$ is given

- The public parameters pp
- A set $\mathbf{sk} = \{\mathsf{sk}_j\}_{j \in [d]}$ of d secret keys.
- A ciphertext
$$c = \left(\mathbf{pk}, \delta, \{e_g\}_{g \in F_{\mathbf{pk},\delta}}\right) \quad ,$$
  where e_g is a level-δ encoding.

It outputs a message $m \in R$ or a special symbol $\perp$.

For $j \in [d]$, let pk_i be the public key associated with sk_i (we assume without loss of generality that sk_i contains also pk_i). If $\mathbf{pk} \not\subseteq \left\{\mathsf{pk}_j\right\}_{j \in [d]}$ output $\perp$.

The ciphertext c encodes a polynomial P with $|\mathbf{pk}|$ variables of total degree $(\delta \cdot \Delta)$. $\mathsf{MHE.Dec}$ homomorphically evaluates P on the secret keys encodings. Note the despite the high degree of P, this evaluation is valid since the secret keys contain a level-0 encoding. Finally, $\mathsf{MHE.Dec}$ inefficiently decrypts the evaluated encoding.

Specifically, the message is decrypted as follows. Homomorphically evaluate the level-δ encoding

$$f = \left\langle \sum_{g \in F_{\mathbf{pk},\delta}} e_g \times \prod_{\mathsf{pk}_i \in \mathbf{pk}} (\mathsf{sk}_i)^{g(\mathsf{pk}_i)} \right\rangle \ .$$

Inefficiently recover the secret parameters sp from the public parameters pp and obtain the decryption

$$(\alpha, \delta) \leftarrow \mathsf{GE.Dec}(\mathsf{sp}, f) \ ,$$

and output α.

Weak Decryption. The weak decryption procedure $\mathsf{MHE.WeakDec}$ is defined exactly as the inefficient decryption procedure $\mathsf{MHE.Dec}$ except that the final inefficient decryption step is replaced with an efficient zero test.

Specifically, $\mathsf{MHE.WeakDec}$ performs the same tests as $\mathsf{MHE.Dec}$ and obtains the encoding f in the same manner as $\mathsf{MHE.WeakDec}$. If there exists $m \in \{0,1\}$ such that

$$\mathsf{GE.ZT}\left(\mathsf{pp}, \langle f - [m]_1 \rangle\right) = 1 \ ,$$

it outputs m, otherwise it outputs $\perp$.

Homomorphic evaluation. The evaluation procedure $\mathsf{MHE.Eval}$ is given the public parameters pp, an operation $\star \in \{+, -, \times\}$, and a pair of ciphertexts

$$c^1 = \left(\mathbf{pk}^1, \delta^1, \left\{e^1_g\right\}_{g \in F_{\mathbf{pk}^1,\delta^1}}\right) \ ,$$
$$c^2 = \left(\mathbf{pk}^2, \delta^2, \left\{e^2_g\right\}_{g \in F_{\mathbf{pk}^2,\delta^2}}\right) \ .$$

It outputs en evaluated ciphertext c. Intuitively, c^1 and c^2 encode the coefficients of polynomials P^1 and P^2 respectively, then c will encoded the coefficients of the polynomial $P^1 \star P^2$.

$\mathsf{MHE.Eval}$ first "extends" each ciphertext such that it is encrypted under the public keys $\mathbf{pk} = \mathbf{pk}^1 \cup \mathbf{pk}^2$. This is done by mapping the encoded coefficients of the monomials in $F_{\mathbf{pk}^i,\delta^i}$ to the equivalent monomials in $F_{\mathbf{pk},\delta^i}$, and initializing the coefficient of all remanding monomials in $F_{\mathbf{pk},\delta^i}$ to 0.

Formally, we describe how to extend an arbitrary ciphertext

$$c' = \left(\mathbf{pk}', \delta, \left\{e'_g\right\}_{g \in F_{\mathbf{pk}',\delta}}\right) \ .$$

to a new ciphertext c encrypted under a set $\mathbf{pk} \supseteq \mathbf{pk}'$.

$$c = \left(\mathbf{pk}, \delta, \left\{e_g\right\}_{g \in F_{\mathbf{pk},\delta}}\right) \ .$$

All the non-zero coefficients encoded in c are extensions of the coefficients $\left\{e'_{g'}\right\}_{g' \in F_{\mathbf{pk}',\delta}}$ where we extend every index $g' \in F_{\mathbf{pk}',\delta}$ to an index in $g \in F_{\mathbf{pk},\delta}$ by setting $g(\mathsf{pk}) = 0$ for every $\mathsf{pk} \in \mathbf{pk} \setminus \mathbf{pk}'$.

Specifically, for every $g \in F_{\mathbf{pk},\delta}$, the encoding e_g is as follows

- If there exists $\mathsf{pk} \in \mathbf{pk} \setminus \mathbf{pk}'$ such that $g(\mathsf{pk}) > 0$, then e_g is a level-δ encoding of 0

$$e_g = \left\langle [0]_0 \times ([1]_1)^\delta \right\rangle \ .$$

- Otherwise, $e_g = e_{g'}$ where $g' \in F_{\mathbf{pk}',\delta}$ is such that for every $\mathsf{pk} \in \mathbf{pk}'$, $g'(\mathsf{pk}) = g(\mathsf{pk})$.

We denote the extended ciphertexts by

$$d^1 = \left(\mathbf{pk}, \delta^1, \{e_g^1\}_{g \in F_{\mathbf{pk},\delta^1}}\right) \ ,$$
$$d^2 = \left(\mathbf{pk}, \delta^2, \{e_g^2\}_{g \in F_{\mathbf{pk},\delta^2}}\right) \ .$$

Next $\mathsf{MHE.Eval}$ evaluates the operation $\star$ on the encoded polynomial. We sperate between the case where $\star \in \{+,-\}$ and the case where $\star = \times$.

- If $\star \in \{+,-\}$, $\mathsf{MHE.Eval}$ first brings both ciphertexts d^1, d^2 to be of degree $\delta = \max(\delta^1, \delta^2)$ by multiplying their coefficient by a power of the encoding $[1]_1$. Specifically, for every $i \in \{1,2\}$ and for every $g \in F_{\mathbf{pk},\delta^i}$ let

$$f_g^i = \left\langle e_g^i \times ([1]_1)^{\delta-\delta^i} \right\rangle \ ,$$

  and for every $g \in F_{\mathbf{pk},\delta} \setminus F_{\mathbf{pk},\delta^i}$ let

$$f_g^i = \left\langle [0]_0 \times ([1]_1)^\delta \right\rangle \ .$$

  For $g \in F_{\mathbf{pk},\delta}$, let $e_g = \langle f_g^1 \star f_g^2 \rangle$.

- If $\star = \times$, let $\delta = \delta^1 + \delta^2$. If $\delta \geq \Delta$, output $\perp$. Otherwise, for every $g \in F_{\mathbf{pk},\delta}$ evaluate the encoding e_g

$$e_g = \left\langle \sum_{g^1 \in F_{\mathbf{pk},\delta^1}} e_{g^1} \times e_{g-g^1} \right\rangle \ .$$

If there exists $g \in F_{\mathbf{pk},\delta}$ such that $e_g = \perp$ (resulting from an evaluation error) output $c = \perp$. Otherwise output

$$c = \left(\mathbf{pk}, \delta, \{e_g\}_{g \in F_{\mathbf{pk},\delta}}\right) \ .$$

Zero Test. The zero test procedure $\mathsf{MHE.ZT}$ is given the public parameters pp, d public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_d$ and a ciphertext

$$c = \left(\mathbf{pk}, \delta, \{e_g\}_{g \in F_{\mathbf{pk},\delta}}\right) \ .$$

It outputs a bit.

If $\mathbf{pk} \not\subseteq \{\mathsf{pk}_i\}_{i \in [d]}$ output $\perp$. Otherwise, test that the ciphertext c encodes the zero polynomial, that is, all its coefficients are 0. Specifically, if there exists $g \in F_{\mathbf{pk},\delta}$ such that $\mathsf{GE.ZT}(\mathsf{pp}, e_g) = 0$, output 0. Otherwise, output 1.

5.3 Analysis

In this section we prove that under Assumption 5.3, the construction in Section 5.2 is a d-key zero-testable (B, Δ)-somewhat homomorphic encryption scheme according to Definitions 3.4 and 3.2.

Setup. Let Setup be the experiment

$$(\mathsf{sp}, \mathsf{pp}, R) \leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta)$$
$$\forall j \in [d] : (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp})$$

Let $\mathbf{pk} = \{\mathsf{pk}_j\}_{j\in[d]}$. For $j \in [d]$, let t_j be the decryption of the secret and public key $\mathsf{sk}_j, \mathsf{pk}_j$. By the correctness property of the graded encoding both keys decrypt to the same value in R.

$$t_j = \mathsf{GE.Dec}(\mathsf{sp}, \mathsf{sk}_j) = \mathsf{GE.Dec}(\mathsf{sp}, \mathsf{pk}_j) \neq \perp \ .$$

The polynomial P^c. Let c be a ciphertext

$$c = \left(\mathbf{pk}', \delta, \{e_g\}_{g\in F_{\mathbf{pk}',\delta}}\right) \ .$$

The encoded coefficients in c define a polynomial P^c as follows. If $\mathbf{pk}' \not\subseteq \mathbf{pk}$, or if $\delta > \Delta$ let $P^c = \perp$. For every $g \in F_{\mathbf{pk},\delta}$ let α_g be as follows

- If $g \in F_{\mathbf{pk}',\delta}$, let $(\alpha_g, \cdot) \leftarrow \mathsf{GE.Dec}(\mathsf{sp}, e_g)$.
- If $g \notin F_{\mathbf{pk}',\delta}$, $\alpha_g = 0$.

If there exists $g \in F_{\mathbf{pk},\delta}$ such $\alpha_g = \perp$ let $P^c = \perp$. Otherwise

$$P^c(x_1, \dots, x_d) = \sum_{g\in F_{\mathbf{pk},\delta}} \alpha_g \cdot \prod_{i\in[d]} x_i^{g(\mathsf{pk}_i)} \ .$$

If $P^c = \perp$ let $P^c(t_1, \dots, t_d) = \perp$. For a pair of ciphertexts c_1, c_2 and operation $\star \in \{+, -, \times\}$ if either $P^{c_1} = \perp$ or $P^{c_1} = \perp$ let $P^{c_1} \star P^{c_2} = \perp$.

Basic claims. The following claims capture the basic properties of the polynomial P^c.

Claim 5.4 (Encryption). *There exists a negligible function μ such that for every $\lambda \in \mathbb{N}$, every $m \in \{0, 1\}$ and for every index $j \in [d]$*

$$\Pr_{\mathsf{Setup}}\left[\ P^c(t_1, \dots, t_d) = m \ \middle|\ c \leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}_j, m)\ \right] \geq 1 - \mu(\lambda) \ .$$

The proof of the claim follows directly from the correctness property of the graded encoding scheme and by the definition of the procedure $\mathsf{MHE.Enc}$.

Claim 5.5 (Evaluation). *For every poly-size adversary* Adv *there exists a negligible function μ such that for every $\lambda \in \mathbb{N}$ and for every operation $\star \in \{+, -, \times\}$*

$$\Pr_{\mathsf{Setup}}\left[\begin{array}{l} c = \perp \Rightarrow P^c = \perp \\ c \neq \perp \Rightarrow P^c \equiv P^{c_1} \star P^{c_2} \end{array} \middle| \begin{array}{l} c_1, c_2 \leftarrow \mathsf{Adv}(\mathsf{pp}, \mathbf{pk}) \\ c \leftarrow \mathsf{MHE.Eval}(\mathsf{pp}, \mathbf{pk}, \star, (c_1, c_2)) \end{array}\right] \geq 1 - \mu(\lambda) \ .$$

The proof of the claim follows directly from the correctness for adversarial encodings property of the graded encoding scheme and by the definition of the procedure $\mathsf{MHE.Eval}$.

Claim 5.6 (Decryption)**.** *For every poly-size adversary* Adv *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr_{\mathsf{Setup}}\left[\ \alpha = P^c(t_1,\dots,t_d)\ \middle|\ \begin{array}{l} c \leftarrow \mathsf{Adv}(\mathsf{pp},\mathbf{pk}) \\ \alpha \leftarrow \mathsf{MHE.Dec}(\mathsf{pp},(\mathsf{sk}_1,\dots,\mathsf{sk}_d),c) \end{array}\right] \geq 1-\mu(\lambda)\ .$$

The proof of the claim follows directly from the correctness for adversarial encodings property of the graded encoding scheme and by the definition of the procedure MHE.Dec.

Claim 5.7 (Zero Test)**.** *For every poly-size adversary* Adv *there exists a negligible function* μ *such that for every* $\lambda \in \mathbb{N}$ *and for every operation* $\star \in \{+,-,\times\}$

$$\Pr_{\mathsf{Setup}}\left[\ b = 1 \Leftrightarrow P^c \equiv 0\ \middle|\ \begin{array}{l} c \leftarrow \mathsf{Adv}(\mathsf{pp},\mathbf{pk}) \\ b \leftarrow \mathsf{MHE.ZT}(\mathsf{pp},\mathbf{pk},c) \end{array}\right] \geq 1-\mu(\lambda)\ .$$

The proof of the claim follows directly from the correctness for adversarial encodings and zero test properties of the graded encoding scheme and by the definition of the procedure MHE.ZT.

Correctness. Let $\lambda \in \mathbb{N}$. Let C be an arithmetic circuit with n inputs of size at most B and total degree at most Δ. Let $\{m_i \in \{0,1\}, j_j \in [d]\}_{i\in[n]}$ be a sequence of messages and indices. We consider an experiment Exp that starts just as Setup and continues as follows

$$\begin{aligned} \forall i \in [n] : c_i &\leftarrow \mathsf{MHE.Enc}(\mathsf{pp},\mathsf{pk}_{j_i},m_i) \\ c &\leftarrow \langle C\left(c_1,\dots,c_n\right)\rangle \\ \alpha &\leftarrow \mathsf{MHE.Dec}(\mathsf{pp},(\mathsf{sk}_1,\dots,\mathsf{sk}_d),c) \end{aligned}$$

By Claim 5.4 for every $i \in [n]$

$$\Pr_{\mathsf{Exp}}\left[P^{c_i}(t_1,\dots,t_d) = m\right] \geq 1-\mathrm{negl}(\lambda)\ .$$

By Claim 5.5 and by induction on the structure of C

$$\Pr_{\mathsf{Exp}}\left[P^c \equiv C(P^{c_1},\dots,P^{c_n})\right] \geq 1-\mathrm{negl}(\lambda)\ .$$

Here we use the fact that C is of total degree at most Δ to conclude that MHE.Eval never outputs $\perp$.

By Claim 5.6

$$\Pr_{\mathsf{Exp}}\left[\alpha = P^c(t_1,\dots,t_d) = C(m_1,\dots,m_n)\right] \geq 1-\mathrm{negl}(\lambda)\ .$$

Compactness. The compactness property follows directly from the compactness property of the graded encoding scheme. Note that ciphertext size grows exponentially with d, however, we only consider a constant d.

Correctness for adversarial encodings. Let Adv be a poly-size adversary. For every $\lambda \in \mathbb{N}$ and for every operation $\star \in \{+,-,\times\}$ we consider an experiment Exp that starts just as Setup and continues as follows

$$\begin{aligned} (\mathsf{sp},\mathsf{pp},R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda,1^\Delta) \\ \forall j \in [d] : (\mathsf{pk}_j,\mathsf{sk}_j) &\leftarrow \mathsf{MHE.KeyGen}(\mathsf{pp}) \\ c_1,c_2 &\leftarrow \mathsf{Adv}(\mathsf{pp},\mathbf{pk}) \\ c &\leftarrow \mathsf{HE.Eval}(\mathsf{pp},\mathbf{pk},\star,(c_1,c_2)) \\ \forall i \in \{1,2\} : \alpha_i &\leftarrow \mathsf{MHE.Dec}(\mathsf{pp},(\mathsf{sk}_1,\dots,\mathsf{sk}_d),c_i) \\ \alpha &\leftarrow \mathsf{MHE.Dec}(\mathsf{pp},(\mathsf{sk}_1,\dots,\mathsf{sk}_d),c) \end{aligned}$$

In the above experiment, if $\alpha_1, \alpha_2 \in R$, the expression $\alpha_1 \star \alpha_2$ is evaluated over R. If $\perp \in \{\alpha_1, \alpha_2\}$ then let $\alpha_1 \star \alpha_2 = \perp$.

By Claim 5.5

$$\Pr_{\mathsf{Exp}}\left[P^c \in \{P^{c_1} \star P^{c_2}, \perp\}\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

By Claim 5.6

$$\Pr_{\mathsf{Exp}}\left[\begin{array}{l} \alpha_i = P^{c_i}(t_1, \ldots, t_d) \\ \alpha = P^c(t_1, \ldots, t_d) \end{array}\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Together we have that

$$\Pr_{\mathsf{Exp}}\left[\ \alpha \in \{\alpha_1 \star \alpha_2, \perp\}\ \right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Zero test completeness. Let $\lambda \in \mathbb{N}$. Let C be an identically zero arithmetic circuit with n inputs of size at most B and total degree at most Δ. Let $\{m_i \in \{0,1\}, j_j \in [d]\}_{i \in [n]}$ be a sequence of messages and indices. We consider an experiment Exp that starts just as Setup and continues as follows

$$\begin{aligned} \forall i \in [n] : c_i &\leftarrow \mathsf{MHE.Enc}(\mathsf{pp}, \mathsf{pk}_{j_i}, m_i) \\ c &\leftarrow \langle C\left(c_1, \ldots, c_n\right)\rangle \\ b &\leftarrow \mathsf{MHE.ZT}(\mathsf{pp}, \mathbf{pk}, c) \end{aligned}$$

By Claim 5.4 for every $i \in [n]$

$$\Pr_{\mathsf{Exp}}\left[P^{c_i} \neq \perp\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

By Claim 5.5 and by induction on the structure of C

$$\Pr_{\mathsf{Exp}}\left[P^c \equiv C(P^{c_1}, \ldots, P^{c_n}) \equiv 0\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Here we use the fact that C is of total degree at most Δ to conclude that MHE.Eval never outputs $\perp$.

By Claim 5.7

$$\Pr_{\mathsf{Exp}}\left[b = 1\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Zero test soundness. Let Adv be a poly-size adversary. For every $\lambda \in \mathbb{N}$ we consider an experiment Exp that starts just as Setup and continues as follows

$$\begin{aligned} c &\leftarrow \mathsf{Adv}(\mathsf{pp}, \mathbf{pk}) \\ \alpha &\leftarrow \mathsf{MHE.Dec}(\mathsf{pp}, (\mathsf{sk}_1, \ldots, \mathsf{sk}_d), c) \\ b &\leftarrow \mathsf{MHE.ZT}(\mathsf{pp}, \mathbf{pk}, c) \end{aligned}$$

By Claim 5.7

$$\Pr_{\mathsf{Exp}}\left[b = 1 \Rightarrow P^c \equiv 0\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

By Claim 5.6

$$\Pr_{\mathsf{Exp}}\left[\alpha = P^c(t_1, \ldots, t_d)\right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Together we have that

$$\Pr_{\mathsf{Exp}}\left[\ b = 1 \Rightarrow \alpha = 0\ \right] \geq 1 - \mathrm{negl}(\lambda) \ .$$

Weak decryption. The weak decryption property follows directly from de definition of the procedure $\mathsf{MHE.WeakDec}$ and the zero test property of the graded encoding scheme.

Semantic security. Let Adv be poly-size adversary. We consider a sequence of hybrid experiments.

Experiment Exp_1**.** This experiment samples a ciphertext c encrypting a random bit m.

$$\begin{aligned}
m &\leftarrow \{0,1\}\\
(\mathsf{sp},\mathsf{pp},R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda,1^\Delta)\\
\mathsf{sk} &\leftarrow \mathsf{GE.Samp}(\mathsf{pp})\\
\mathsf{pk} &\leftarrow \langle \mathsf{sk}\times[1]_1\rangle_{\mathsf{rand}}\\
r_0 &\leftarrow [m]_0\\
\forall i\in[\Delta]: r_i &\leftarrow \mathsf{GE.Samp}(\mathsf{pp})\\
r_{\Delta+1} &\leftarrow [0]_0\\
\forall i\in[0,\Delta]: e_i &\leftarrow \langle r_i\times[1]_1 - r_{i+1}\times\mathsf{pk}\rangle_{\mathsf{rand}}\\
c &\leftarrow \left(\{\mathsf{pk}\},1,\{e_i\}_{i\in[0,\Delta]}\right)
\end{aligned}$$

We need to show that

$$\Pr_{\mathsf{Exp}_1}[\mathsf{Adv}(\mathsf{pp},\mathsf{pk},c)=m] \le \frac{1}{2}+\mathrm{negl}(\lambda)\ .$$

Experiment Exp_2**.** In this experiment c is sampled as in Exp_1 except that instead of sampling level-0 encodings with the public operation $\mathsf{GE.Samp}$ we use the secret parameters to encode random elements.

$$\begin{aligned}
m &\leftarrow \{0,1\}\\
(\mathsf{sp},\mathsf{pp},R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda,1^\Delta)\\
t,\alpha_1,\dots,\alpha_\Delta &\leftarrow R\\
\mathsf{sk} &\leftarrow \mathsf{GE.Enc}(\mathsf{sp},t,0)\\
\mathsf{pk} &\leftarrow \langle \mathsf{sk}\times[1]_1\rangle_{\mathsf{rand}}\\
r_0 &\leftarrow [m]_0\\
\forall i\in[\Delta]: r_i &\leftarrow \mathsf{GE.Enc}(\mathsf{sp},\alpha_i,0)\\
r_{\Delta+1} &\leftarrow [0]_0\\
\forall i\in[0,\Delta]: e_i &\leftarrow \langle r_i\times[1]_1 - r_{i+1}\times\mathsf{pk}\rangle_{\mathsf{rand}}\\
c &\leftarrow \left(\{\mathsf{pk}\},1,\{e_i\}_{i\in[0,\Delta]}\right)
\end{aligned}$$

It follows from the sampling property of the graded encoding scheme that

$$\left|\Pr_{\mathsf{Exp}_1}[\mathsf{Adv}(\mathsf{pp},\mathsf{pk},c)=1] - \Pr_{\mathsf{Exp}_2}[\mathsf{Adv}(\mathsf{pp},\mathsf{pk},c)=1]\right| \le \mathrm{negl}(\lambda)\ .$$

In more details, we move between Exp_1 and Exp_2 through a sequence of hybrid experiments where in every experiment one invocation of the sampling procedure $\mathsf{GE.Samp}$ is replaced by the encoding procedure $\mathsf{GE.Enc}$. Recall that the adversary braking the sampling property of the graded encoding scheme is not given the scheme's secret parameters or the description of the ring R. However, it is given access to the encoding procedure $\mathsf{GE.Enc}(\mathsf{sp},\cdots)$ and an oracle O_R that samples a random element from R.

Experiment Exp_3**.** In this experiment c is sampled as in Exp_2 except that instead of computing the public key pk homomorphically from the secret key sk and re-randomizing, we encode pk directly with the secret parameters.

$$\begin{aligned}
m &\leftarrow \{0,1\} \\
(\mathsf{sp}, \mathsf{pp}, R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\
t, \alpha_1, \ldots, \alpha_\Delta &\leftarrow R \\
\mathsf{pk} &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, t, 1) \\
r_0 &\leftarrow [m]_0 \\
\forall i \in [\Delta] : r_i &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha_i, 0) \\
r_{\Delta+1} &\leftarrow [0]_0 \\
\forall i \in [0,\Delta] : e_i &\leftarrow \langle r_i \times [1]_1 - r_{i+1} \times \mathsf{pk} \rangle_{\mathsf{rand}} \\
c &\leftarrow \Big(\{\mathsf{pk}\}, 1, \{e_i\}_{i \in [0,\Delta]} \Big)
\end{aligned}$$

It follows from the re-randomization property of the graded encoding scheme that

$$\left| \Pr_{\mathsf{Exp}_2} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] - \Pr_{\mathsf{Exp}_3} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] \right| \leq \mathrm{negl}(\lambda) \ .$$

Recall that the adversary braking the re-randomization property of the graded encoding scheme is not given the scheme's secret parameters or the description of the ring R. However, it is given access to the encoding procedure $\mathsf{GE.Enc}(\mathsf{sp}, \cdots)$ and an oracle O_R that samples a random element from R.

Experiment Exp_4**.** In this experiment c is sampled as in Exp_3 except that instead of computing the encodings e_i homomorphically from the encodings $r_i, r_{i+1}, \mathsf{pk}$ and re-randomizing, we encode e_i directly with the secret parameters.

$$\begin{aligned}
m &\leftarrow \{0,1\} \\
(\mathsf{sp}, \mathsf{pp}, R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\
t, \alpha_1, \ldots, \alpha_\Delta &\leftarrow R \\
\alpha_0 &\leftarrow m \\
\alpha_{\Delta+1} &\leftarrow 0 \\
\mathsf{pk} &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, t, 1) \\
\forall i \in [0,\Delta] : e_i &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha_i - \alpha_{i+1} \cdot t, 1) \\
c &\leftarrow \Big(\{\mathsf{pk}\}, 1, \{e_i\}_{i \in [0,\Delta]} \Big)
\end{aligned}$$

It follows from the re-randomization property of the graded encoding scheme that

$$\left| \Pr_{\mathsf{Exp}_3} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] - \Pr_{\mathsf{Exp}_4} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] \right| \leq \mathrm{negl}(\lambda) \ .$$

In more details, we move between Exp_3 and Exp_4 through a sequence of hybrid experiments where in every experiment one homomorphic evaluation followed by the re-randomization operation $\mathsf{GE.Rand}$ are replaced by the encoding procedure $\mathsf{GE.Enc}$. Recall that the adversary braking the re-randomization property of the graded encoding scheme is not given the scheme's secret parameters or the description of the ring R. However, it is given access to the encoding procedure $\mathsf{GE.Enc}(\mathsf{sp}, \cdots)$ and an oracle O_R that samples a random element from R and implements the operations $\{+, -, \times\}$ in R.

For $i \in [0, \Delta]$, let $\alpha'_i = \alpha_i - \alpha_{i+1} \cdot t$. Observe that

- The elements $\alpha'_1, \dots, \alpha'_\Delta$ are independently uniform in R.
- $\alpha'_0 = \alpha_0 - \sum_{i \in [\Delta]} \alpha'_i \cdot t^i$.

Therefore, we can rewrite Exp_4 as follows.

$$\begin{aligned}
m &\leftarrow \{0,1\} \\
(\mathsf{sp}, \mathsf{pp}, R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\
t, \alpha'_1, \dots, \alpha'_\Delta &\leftarrow R \\
\alpha'_0 &\leftarrow m - \sum_{i \in [\Delta]} \alpha'_i \cdot t^i \\
\mathsf{pk} &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, t, 1) \\
\forall i \in [0, \Delta] : e_i &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha'_i, 1) \\
c &\leftarrow \Big(\{\mathsf{pk}\}, 1, \{e_i\}_{i \in [0,\Delta]} \Big)
\end{aligned}$$

Experiment Exp_5**.** In this experiment c is sampled as in Exp_4 except that α'_0 is uniform in R.

$$\begin{aligned}
m &\leftarrow \{0,1\} \\
(\mathsf{sp}, \mathsf{pp}, R) &\leftarrow \mathsf{GE.ParamGen}(1^\lambda, 1^\Delta) \\
t, \alpha'_0, \dots, \alpha'_\Delta &\leftarrow R \\
\mathsf{pk} &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, t, 1) \\
\forall i \in [0, \Delta] : e_i &\leftarrow \mathsf{GE.Enc}(\mathsf{sp}, \alpha'_i, 1) \\
c &\leftarrow \Big(\{\mathsf{pk}\}, 1, \{e_i\}_{i \in [0,\Delta]} \Big)
\end{aligned}$$

It follows from Assumption 5.3 that

$$\left| \Pr_{\mathsf{Exp}_4} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] - \Pr_{\mathsf{Exp}_5} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = 1] \right| \le \mathrm{negl}(\lambda) \ .$$

Since the view of the adversary is indistinguishable in every pair of experiments, and since in Exp_5, c is independent of m we conclude that

$$\Pr_{\mathsf{Exp}_1} [\mathsf{Adv}(\mathsf{pp}, \mathsf{pk}, c) = m] \le \frac{1}{2} + \mathrm{negl}(\lambda) \ .$$

5.4 On Noisy Graded Encoding

In this section we discuss a possible relaxation of the correctness for adversarially generated encodings property that may hold even for "noisy" candidates. In Definition 5.1 we require that the adversary cannot find a pair of encodings e_1 and e_2 that decrypt to ring elements α_1 and α_2 respectively, but the evaluated encoding $\langle e_1 \star e_2 \rangle$ decrypts to something other than $\alpha_1 \star \alpha_2$. We relax this property and allow $\langle e_1 \star e_2 \rangle$ to decrypt either to $\alpha_1 \star \alpha_2$ or to $\perp$. For example, the evaluated encoding may fail to decrypt if the combined noise in the encoding e_1, e_2 crosses some threshold.

This relaxed correctness property, however, is not sufficient for proving that the encryption scheme in Section 5.2 satisfies correctness for adversarially generated ciphertexts. Specifically we can no longer prove that computing a homomorphic operation on two ciphertexts c_1, c_2 such that at least one of them is invalid (decrypts to $\perp$) necessarily results in an invalid ciphertext c.

Recall that a ciphertext contains encoded coefficients of a polynomial, and decrypting is performed by homomorphically evaluating the polynomial on the element given in the secret key. It may be the case that the ciphertext c_1 contains only valid encodings, however, c_1 is still an invalid ciphertext since one of the homomorphic operations performed during decryption results in an invalid encoding. Still, when homomorphically computing the ciphertext $\langle c_1 \star c_2 \rangle$ and decrypting it, it may be the case that none of the homomorphic operations fail and the evaluated ciphertext decrypts correctly.

To overcome this gap we put another requirement on the graded encoding scheme. Intuitively, we assume that it is possible to publicly test that the level of noise in an adversarially generated encoding is low. In more details, when generating the scheme's public parameters we specify a noise "budget". Fresh encodings should have low noise and homomorphic computation increase the noise in some controlled way. We require that there is a public *noise test* such that

- Encodings with low noise should pass the test.
- It is hard for find encodings that pass the test, but also cause an evaluation error as described above.

We note that the noise test may be randomized and could potentially utilize the public zero test and re-randomization operation. We do not know if existing candidate graded encodings support such public noise test.

Given the noise test we would modify the construction as follows. In the homomorphic evaluation, given a pair of ciphertexts c_1, c_2, the procedure will first check that all the encoding in the ciphertexts pass the noise test and should, therefore, decrypt without errors. If this is not the case, the evaluation fails (outputs $\perp$).

6 Acknowledgements

We thank Zvika Brakerski, Yael Kalai, Ron Rothblum and Nir Bitansky for many helpful and illuminating conversations.

References

[ABOR00] William Aiello, Sandeep N. Bhatt, Rafail Ostrovsky, and Sivaramakrishnan Rajagopalan. Fast verification of any remote procedure call: Short witness-indistinguishable one-round proofs for np. In *ICALP*, pages 463–474, 2000.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for snarks and proof-carrying data. In *STOC*, pages 111–120, 2013.

[BCI$^+$13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In *TCC*, pages 315–333, 2013.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31, 1991.

[BGL$^+$15] Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Sidharth Telang. Succinct randomized encodings and their applications. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 439–448, 2015.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, pages 325–341, 2005.

[BHK16] Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive RAM and batch NP delegation from any PIR. *IACR Cryptology ePrint Archive*, 2016:459, 2016.

[BS02] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *IACR Cryptology ePrint Archive*, 2002:80, 2002.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *IACR Cryptology ePrint Archive*, 2011:344, 2011.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[CHJV14] Ran Canetti, Justin Holmgren, Abhishek Jain, and Vinod Vaikuntanathan. Indistinguishability obfuscation of iterated circuits and ram programs. Cryptology ePrint Archive, Report 2014/769, 2014. `http://eprint.iacr.org/`.

[CHL$^+$15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 3–12, 2015.

[CLT15] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, pages 267–286, 2015.

[DFH12] Ivan Damgård, Sebastian Faust, and Carmit Hazay. Secure two-party computation with low communication. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 54–74, 2012.

[DLN$^+$04] Cynthia Dwork, Michael Langberg, Moni Naor, Kobbi Nissim, and Omer Reingold. Succinct proofs for NP and spooky interactions. Unpublished manuscript, 2004. `http://www.cs.bgu.ac.il/~kobbi/papers/spooky_sub_crypto.pdf`.

[DNR16] Cynthia Dwork, Moni Naor, and Guy N. Rothblum. Spooky interaction and its discontents: Compilers for succinct two-message argument systems. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, pages 123–145, 2016.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178, 2009.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, pages 1–17, 2013.

[GGH$^+$13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 498–527, 2015.

[GGHZ16] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Functional encryption without obfuscation. In *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part II*, pages 480–511, 2016.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 626–645, 2013.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. i-hop homomorphic encryption and rerandomizable yao circuits. In *CRYPTO*, pages 155–172, 2010.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: interactive proofs for muggles. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 113–122, 2008.

[GLSW15] Craig Gentry, Allison Bishop Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 151–170, 2015.

[GMM$^+$16] Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, pages 241–268, 2016.

[GPSZ17] Sanjam Garg, Omkant Pandey, Akshayaram Srinivasan, and Mark Zhandry. Breaking the sub-exponential barrier in obfustopia. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pages 156–181, 2017.

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In *ASIACRYPT*, pages 321–340, 2010.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 75–92, 2013.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, pages 99–108, 2011.

[HJ16] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, pages 537–565, 2016.

[HRSV11] Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. *J. Cryptology*, 24(4):694–719, 2011.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 723–732, 1992.

[KLW14] Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. Cryptology ePrint Archive, Report 2014/925, 2014. `http://eprint.iacr.org/`.

[KP16] Yael Tauman Kalai and Omer Paneth. Delegating RAM computations. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, pages 91–118, 2016.

[KRR13] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. Delegation for bounded space. In *STOC*, pages 565–574, 2013.

[KRR14] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 485–494, 2014.

[Lip12] Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 169–189, 2012.

[LTV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1219–1234, 2012.

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 11–20, 2016.

[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, 1994.

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 629–658, 2016.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *Proceedings of the 23rd Annual International Cryptology Conference*, pages 96–109, 2003.

[PR14] Omer Paneth and Guy N. Rothblum. Publicly verifiable non-interactive arguments for delegating computation. Cryptology ePrint Archive, Report 2014/981, 2014. `http://eprint.iacr.org/2014/981`.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *STOC*, 2014.

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, pages 24–43, 2010.

[WB13] Michael Walfish and Andrew J. Blumberg. Verifying computations without reexecuting them: from theoretical possibility to near-practicality. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:165, 2013.

[Zim15] Joe Zimmerman. How to obfuscate programs directly. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 439–467, 2015.