FHE Over the Integers: Decomposed and Batched in the Post-Quantum Regime

Daniel Benarroch⋆,[1], Zvika Brakerski⋆⋆,[1], and Tancrède Lepoint⋆⋆⋆,[2]

[1] Weizmann Institute of Science, Israel
[2] SRI International, USA

Abstract. Fully homomorphic encryption over the integers (FHE-OI) is currently the only alternative to lattice-based FHE. FHE-OI includes a family of schemes whose security is based on the hardness of different variants of the approximate greatest common divisor (AGCD) problem. A lot of effort was made to port techniques from second generation lattice-based FHE (using tensoring) to FHE-OI. Gentry, Sahai and Waters (Crypto 13) showed that third generation techniques (which were later formalized using the "gadget matrix") can also be ported. However, the majority of these works was based on the noise-free variant of AGCD which is potentially weaker than the general one. In particular, the noise-free variant relies on the hardness of factoring and is thus vulnerable to quantum attacks.

In this work, we propose a comprehensive study of applying third generation FHE techniques to the regime of FHE-OI. We present and analyze a third generation FHE-OI based on decisional AGCD without the noise-free assumption. We proceed to showing a batch version of our scheme where each ciphertext can encode a vector of messages and operations are performed coordinate-wise. We use a similar AGCD variant to Cheon et al. (Eurocrypt 13) who suggested the batch approach for second generation FHE, but we do not require the noise-free component or a subset sum assumption. However, like Cheon et al., we do require circular security for our scheme, even for bounded homomorphism. Lastly, we discuss some of the obstacles towards efficient implementation of our schemes and discuss a number of possible optimizations.

1 Introduction

In homomorphic encryption (HE), it is possible to transform a ciphertext $\mathsf{Enc}(x)$ into $\mathsf{Enc}(f(x))$ for some class of functions in a public manner, i.e. without any secret information and without compromising the security of the encrypted message. Rivest, Adleman and Dertouzos [28] proposed the notion of fully homomorphic encryption (FHE) where the scheme is homomorphic w.r.t any efficiently computable f. This will allow to outsource computation to third parties without compromising security. In an exciting breakthrough, Gentry [20] presented the first candidate FHE scheme that was based on ideal lattices. Very shortly afterwards, van Dijk, Gentry, Halevi and Vaikuntanathan [31] proposed a scheme with a similar structure to Gentry's but one that was based on a different hardness assumption, namely the hardness of the approximate greatest common divisor problem (AGCD) [23]. In AGCD, the attacker's goal is to find a hidden prime p given arbitrarily (polynomially) many samples of random

⋆ Supported by the Israel Science Foundation (Grant No. 468/14).

⋆⋆ Supported by the Israel Science Foundation (Grant No. 468/14), the Alon Young Faculty Fellowship and Binational Science Foundation (Grant No. 712307).

⋆⋆⋆ This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) and Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under Contract No. N66001-15-C-4071. The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

"near multiples" of p, i.e. samples of the form $qp + r$ where q is chosen randomly from an appropriate domain, and $r \ll p$ is random noise. In the AGCD schemes, the basic elements are integers rather than lattice vectors, and the scheme was referred to as FHE over the integers (henceforth FHE-OI).

Lattice based FHE developed and new schemes with better security and efficiency guarantees emerged [2, 4–6] in the lattice domain; this is sometimes called second generation FHE. A sequence of works on FHE-OI showed that many of these techniques can be applied in that regime as well, resulting in schemes with comparable efficiency to the lattice setting [9, 15–17]. In particular, Cheon et al. [9] showed how to encrypt a vector of messages in a single ciphertext, similarly to the [3, 4, 30] SIMD approach. In the batch setting, when performing a homomorphic operation, the operation is performed in parallel on all coordinates of the respective message vectors. This allows to increase the throughput of the scheme while preventing the ciphertext size from growing too much. This was later improved by Coron, Lepoint and Tibouchi [15] who presented a "scale invariant" version of the batch scheme, again showing that the scale invariance notion from [2] can be applied in the FHE-OI setting.

These last two works [9, 15], however, deviate from the original formulation of the AGCD assumption in a number of aspects: (i) In order to allow batched ciphertexts, they change p from being prime to being a product of primes, so as to allow encoding of multiple messages using the Chinese Remainder Theorem. Furthermore, they required a circular secure variant of the assumption, where indistinguishability holds even for elements of the form $qp + r + m$ where m depends on the factors of p. We discuss circular security further when we talk about our batch scheme below. (ii) Rather than just assuming that finding p is hard, they assume that the samples $qp + r$ are indistinguishable from uniform samples over some domain. This is now known as *decisional* AGCD. (iii) Only in [9]: They require that the problem remains hard even if a single multiple of p without noise is known, i.e. some $x_0 = q_0 p$. This is known as the noise free variant. However, they show that if this element is given, then there is a reduction from the original AGCD problem to the new decisional batch AGCD. (iv) They make an additional subset sum assumption.

Gentry, Sahai and Waters [21] proposed a new family of techniques for lattice based FHE, sometimes referred to as third generation FHE. They showed how to decrease the size of the public parameters and the asymptotic complexity of performing a single homomorphic multiplication. It was later shown that this approach can be used to weaken the hardness assumption of the scheme and achieve better parameters [1, 7] and even for implementations [11, 18, 24]. It was observed in [21] that their techniques can be applied also to FHE-OI in the noise free variant. Subset sum was no longer required, but a batched scheme was not introduced. It is important to mention that third generation FHE is lacking in terms of information rate. Whereas in second generation FHE, a (post-evaluation) ciphertext of length ℓ can encrypt $\Omega(\ell)$ bits of information, in third generation scheme only $o(\ell)$ bits are possible while maintaining full homomorphism.

Although having a noise free element simplifies the schemes, its impact on security is unclear. On one hand, it allows to reduce from the hardness of the search AGCD all the way to batch decision. On the other hand, it gives the adversary additional information that might be harmful for security. One setting where this additional information is *known* to be harmful is for *post-quantum* security. In post quantum security, we are interested in schemes that can be run on standard classical computers, but which are secure even against adversaries that have a quantum computer. This corresponds to a situation where the majority of the population cannot afford a quantum computer, but big organizations or governments can, and the simple user would still like to maintain security even against these quantum capabilities. Since factoring can be solved using a quantum computer [29], noise free AGCD is *not* post quantum secure. On the other hand, Cheon and Stehlé [10] showed that (non batched) decisional AGCD without the noise free element is at least as hard as the learning with errors (LWE) problem, which is widely regarded as post quantumly secure. We note that even though a similar statement for the batch variant is not known, the [10] result increases our confidence in its post quantum security.

1.1 Our Results

We construct third generation FHE-OI schemes, with and without batching capabilities, based on decisional AGCD (without a noise free element). More accurately, we construct *leveled* FHE: a parameterized homomorphic encryption scheme, such that for any polynomial depth bound $d = \text{poly}(\lambda)$ there is a proper setting of parameters such that the scheme can evaluate all depth d circuits. Leveled FHE schemes can be converted to plain FHE using Gentry's bootstrapping principle [20], albeit at the cost of making an additional hardness assumption (the circular security of the scheme). Since the use of bootstrapping is identical to previous schemes in the literature, we leave it out of this paper and focus on constructing the leveled schemes.

Our first scheme is non-batched, i.e. each ciphertext encrypts a single bit message, and is based on decisional AGCD. This scheme is similar to the [21] proposed construction, but we do away with the noise free element. This scheme is presented in Section 3. Our second scheme is batched and is based on batched decisional AGCD in addition to a circular security assumption (similarly to [9, 15] as mentioned above). This scheme is presented in Section 4. Known attacks against the batched AGCD problem work less well than for the classical AGCD problem: this problem can potentially offer a higher degree of security (cf. [19]).

Both schemes enjoy the same *noise propagation* features as the LWE scheme of [1, 7, 21]: In all known FHE candidates, the limitation on the homomorphism depth stems from noise accumulation. Upon encryption, ciphertexts contain a certain amount of noise (appropriately defined), and performing homomorphic operations increases the noise. For correct decryption, the noise has to be below a predefined threshold, hence the limitation on the number of operations. In our scheme, the noise grows by a $\text{poly}(\lambda)$ factor with every multiplication or logical NAND operation (the exact polynomial depends on the choice of parameters). Furthermore, the noise growth is asymmetric as in [21], i.e. only the noise of one of the two operands grows by a polynomial factor, and the noise of the other does not grow at all. This means that we can apply similar optimizations such as the ones in [1, 7].

Finally, in Section 5, we discuss a number of possible optimizations. We show that using the subset sum assumption we can reduce the public key size, using techniques inspired by

Coron et al. [16]. We show that at the last steps of the homomorphic evaluation, we do not have to pay the full cost of multiplication. We then show that instead of binary decomposition of the ciphertext, which is a major ingredient in [21] (and even earlier in [5]), one can use decomposition over a larger index, specifically roughly polynomial in the security parameter. This will reduce the ciphertext size and the evaluation complexity, but will increase the noise growth. Lastly we discuss the limitations in choosing the actual parameters for future implementation.

1.2 Our Techniques

Let p be a secret prime sampled from the appropriate distribution, and consider a sequence of samples of the form $x_i = q_i p + r_i$, where q_i is again sampled uniformly across some properly defined domain, and r_i is chosen uniformly across a "small" domain so that $|r_i| \ll p$. The decisional AGCD assumption is that the x_i's are computationally indistinguishable from a uniform element modulo a known parameter $N \approx q_{\max} p$ (in fact, the parameters are chosen so that first N is selected and then the distribution q_i's is chosen so that the maximum is approximately equal to N/p). Consider adding a few of the x_i's together, for example taking two such samples, x_1, x_2 and letting $y = x_1 + x_2$. Then $y = (q_1 + q_2)p + (r_1 + r_2)$, and it has a similar structure to the original x's. However, not exactly: First of all, the noise $r_1 + r_2$ is bigger than the original r. Secondly, and as it turns out more importantly, now y might be bigger than the modulus N. Looking ahead, this will translate to ciphertext size growth during homomorphic evaluation, a side effect that we need to prevent. Note that the first intuition of taking the result modulo N does not help here. While the ciphertext size will be reduced, the structure might be lost as well since $y \pmod{N} = y - kN$ for some integer k. The solution, as presented already in [31], is to consider the first sample $x_0 = q_0 p + r_0$ as the modulus. Since we know that y cannot be much larger than N (since it is just the sum of a small number, say 2, of x_i's), this means that k is not so large, and therefore $y \pmod{x_0} = y - kx_0 = (q_1 + q_2 - kq_0)p + (r_1 + r_2 - kr_0)$ both lies in the right domain, which we will now define to be integers modulo x_0, and has the right structure. We note that the aforementioned error free variant is simply taking $r_0 = 0$ which simplifies many of the computations ahead since there will not be an r_0 contribution to the noise term.

A Modified Distribution. The situation here is more challenging than [31], though. For security purposes we would like to argue that $x_i \pmod{x_0}$ is indistinguishable from a uniform element modulo x_0. However, this is actually not true since x_i and x_0 are of similar magnitudes and therefore small elements have a higher probability of appearing in $x_i \pmod{x_0}$. We therefore consider the *conditional* probability distribution of x_i conditioned on $x_i < x_0$. This distribution is indeed indistinguishable from uniform modulo x_0. We will therefore use *rejection sampling* to sample the x_i's: we will sample according to the AGCD distribution, and if we are above x_0, we will discard the sample and repeat. This will result in a distribution of x_i that both has the structure that we desire and is indistinguishable from uniform modulo x_0. The only remaining problem is that perhaps we get x_0 that is so small that we will reject too often thus increasing the computational complexity of generating the x_i's.

There are a number of ways to avoid this problem, we chose to just apply rejection sampling to the choice of x_0 itself and condition on it being larger than $N/2$, which implies that x_i will be rejected with probability at most $1/2$. We note that if x_0 was error free, this process would not be needed since $x_i \pmod{x_0}$ would have been distributed exactly like a fresh AGCD sample with q uniformly modulo $q_{\max}$. The formal analysis of this process appears in Lemma 2.3.

Our Basic Scheme. Now that we have our building blocks, we can construct a GSW-style homomorphic encryption scheme (as formulated by [1]). Consider the operation of taking a number c modulo x_0 and decomposing it into its binary representation, which is a binary column vector of dimension $\lceil \log x_0 \rceil$. We denote this operation by $\mathbf{G}^{-1}(c)$ (for reasons that will become clear later), if $\mathbf{c}$ is a row vector, then $\mathbf{G}^{-1}(\mathbf{c})$ is a binary matrix. The complement of this operation is linear, i.e. there is a vector $\mathbf{g}$ s.t. $\mathbf{g} \cdot \mathbf{G}^{-1}(\mathbf{c}) = \mathbf{c} \pmod{x_0}$ for all $\mathbf{c}$. Therefore, we can devise a scheme where in order to encrypt a bit m, we produce a ciphertext which is a row vector of the form $\mathbf{c} = m\mathbf{g} + \mathbf{q}p + \mathbf{r}$, but which is still computationally indistinguishable from a uniform vector over $\mathbb{Z}_{x_0}$. Such a ciphertext can be generated using standard methods from a public key containing a sequence of x_i's: in a nutshell, a random subset sum of the x_i's will preserve the structure and produce a "fresh" x_i, although with somewhat larger noise, which is indistinguishable from uniform even given the public key due to the leftover hash lemma. One has to take special care when reducing modulo x_0 that the additional kr_0 term does not become too large. Such a ciphertext can be decrypted by first multiplying by $\mathbf{G}^{-1}(p/2)$, and then reducing modulo p, which results in $mp/2 + \mathbf{r}\mathbf{G}^{-1}(p/2)$. Since $\mathbf{G}^{-1}(\cdot)$ always output a binary vector, then if $\mathbf{r}$ is short enough then $\mathbf{r}\mathbf{G}^{-1}(p/2)$ is only slightly longer. So long as the norm of $\mathbf{r}\mathbf{G}^{-1}(p/2)$ is smaller than $p/4$, we will decrypt the correct bit. Homomorphic evaluation is performed as in [1] by computing $\mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2) \pmod{x_0}$ which results in a ciphertext of the form $\mathbf{c} = m_1 m_2 \mathbf{g} + \mathbf{q}' p + (\mathbf{r}_1 \mathbf{G}^{-1}(\mathbf{c}_2) + m_1 \mathbf{r}_2 + \mathbf{k} r_0)$. Thus we have a noise growth very similar to GSW style encryption, but with an additional term that depends on r_0 are comes from taking the result modulo x_0. We therefore need to take into account in our analysis a bound on the modulus $\mathbf{k}$ so that we can bound the noise growth. Note that if x_0 had been noise free, this complication does not arise. See Section 3 for the full details, parameters and analysis.

A Batched Variant. The previous scheme only allows to encode a single bit in a ciphertext vector. This is of course a significant efficiency constraint. We show how to encode multiple bits in a single ciphertext vector using the Chinese Remainder Theorem (CRT), inspired by previous works such as [9]. We will now consider a batched AGCD distribution with samples of the form $x_i = q_i \pi + r_i$, where π is now a composite $\pi = \prod_{i \in [\ell]} p_i$, and the noise r_i is defined as the number modulo π s.t. $|r_i \pmod{p_j}| \ll p_j$. Namely, r_i is not a short element by itself, but rather its CRT coefficients with respect to the p_j's are short. The batched AGCD hardness assumption again asserts that these samples are indistinguishable from random modulo some known N. We will use our modified distribution defined above to again generate x_0 and a distribution over x_i s.t. $x_i \pmod{x_0}$ has the right form and is indistinguishable from uniform. Now consider a ciphertext of the form $\mathbf{c} = m\mathbf{g} + \mathbf{q}\pi + \mathbf{r}$, except now m itself is not a bit, but rather its CRT representation is a sequence of bits. Namely, $m \pmod{p_j} = m_j$ for some bit m_j. Decryption can be performed in analogous way to the basic scheme, multiplying by $\mathbf{G}^{-1}(p_j)$ and reducing modulo p_j will allow to recover m_j. Similarly homomorphic operations are performed in the exact same manner as before, since e.g. multiplying two "CRT containers" m, m' will result in element-wise multiplication in each slot corresponding to a factor p_j.

This still leaves open the question of how to generate ciphertexts with the aforementioned structure. It is not a problem to encrypt zero by generating $\mathbf{c}' = \mathbf{q}\pi + \mathbf{r}$ as above. In the basic scheme this was enough since we could just take $m\mathbf{g} + \mathbf{c}' \pmod{x_0}$ and get our ciphertext. Here, we cannot even generate, given the sequence of m_j, the CRT representation m without using secret information (that is, the factorization of π). We solve this problem by considering a set of messages that span the message space, and placing their encryptions as a part of the public key. Specifically, consider m^*_j s.t. $m^*_j = 0 \pmod{p_{j'}}$ if $j \neq j'$ and $m^*_j = 1 \pmod{p_j}$ and let $\mathbf{c}^*_j = m^*_j\mathbf{g} + \mathbf{q}^*_j\pi + \mathbf{r}^*_j$. The $\mathbf{c}^*_\ell$ vectors can be generated during the key generation process using the factorization of π. Now, in order to encrypt a sequence of bits $\{m_j\}_j$, we take $\sum_j m_j\mathbf{c}^*_j + \mathbf{c}' \pmod{x_0}$, where $\mathbf{c}'$ is a freshly generated zero encryption as defined above. One can verify that this will indeed encrypt the right message.

We were able to solve the functionality problem, but we can no longer base security on batch AGCD, since in order to generate the public key we can no longer use the x_i alone without additional private information. Unfortunately, we do not know how to resolve this problem and we make the explicit assumption that the batch AGCD remains hard even when the $\mathbf{c}^*_j$ vectors are published as a part of the public key. In order to increase our confidence in the validity of this additional assumption, we show that it can be stated as assuming *circular security* for a different scheme which is CPA secure. A circular secure encryption scheme is one that can securely encrypt functions of its own secret key. This notion has been entangled in the FHE literature due to the batching technique which requires circular security in order to transform from leveled FHE to unbounded FHE. It is commonly believed that "normal" encryption schemes should be circular secure unless they are intentionally weakened. Clearly this vague definition does not provide a strong guarantee and there are ongoing attempts to come up with more natural schemes that are not circular secure. Yet, by showing that the security of our scheme relates to the circular security of a CPA secure scheme (which is in turn secure under batch AGCD), we can at least deduce that breaking our scheme will imply a surprising result in the study of encryption. We note that a similar assumption was made in previous works [9, 15] but without an explicit proof of the relation to circular security. Formally, we consider the very encryption scheme which takes $m \in \mathbb{Z}$ (or some restriction thereof) and encrypts it as $\mathbf{c} = m\mathbf{g} + \mathbf{c}' \pmod{x_0}$, where $\mathbf{c}'$ is as above. We do not provide a decryption algorithm since we do not require functionality for this scheme, only security, but we consider the secret key to be the factorization of π. This scheme can be shown to be CPA secure under batch AGCD on one hand, and on the other our m^*_j can be written as a function of the secret key of this scheme. Therefore if this auxiliary scheme is circular secure for any function of the secret key, then our scheme is CPA secure.

For the formal statement and analysis of this scheme, see Section 4.

Optimizations Towards Practicality. In Section 5 we analyze parameters and suggested optimizations towards implementation of our schemes. Let us only mention here that one of the serious constraints appears to be the length of the ciphertext which now becomes a $\log(x_0)$ dimensional vector. We find that this can be mitigated by considering a $\mathbf{G}^{-1}(\cdot)$ function that does not perform binary decomposition but rather decomposes relative to a larger radix B. This will have a negative effect on the noise growth, but will decrease the ciphertext size to $\log(x_0)/\log(B)$ which might enhance performance substantially. This optimization was considered in the lattice setting as well, but we believe that it will be much more effective in our setting.

2 Preliminaries

We denote the parameters of our encryption schemes by Greek letters ($\eta, \rho, \gamma, \tau, \lambda$, etc.), where λ is always the security parameter. Scalars are denoted by lowercase Latin characters (p, q, x, y, r, etc.), whereas vectors are denoted by lowercase bold English letters ($\mathbf{x}, \mathbf{y}, \mathbf{r}, \mathbf{q}$, etc.). Finally matrices are denoted by uppercase English letters. For any integer, z, or any vector of integers $\mathbf{z}$, we denote by $[z]_p$ or $[\mathbf{z}]_p$ the value in $(-p/2, p/2]$ which is the remainder of z or of each coordinate of $\mathbf{z}$ when divided by p. For a vector $\mathbf{z}$, we denote the norm $\|\mathbf{z}\|$ to be the infinity norm of the vector, or the size of the maximum entry, $\|\mathbf{z}\| := \|\mathbf{z}\|_\infty = \max_{z_i \text{ in } \mathbf{z}} |z_i|$. All logarithms mentioned in this paper are base two, unless stated otherwise. We note that for all $\mathbf{a}, \mathbf{b} \in \mathbb{Z}^n$ it holds that

$$\|[\mathbf{a} \pm \mathbf{b}]_p\| \leq \|[\mathbf{a}]_p\| + \|[\mathbf{b}]_p\| \,. \tag{1}$$

Computational Indistinguishability. Distribution ensembles $\{D_{0,\lambda}\}_\lambda$, $\{D_{1,\lambda}\}_\lambda$ are *computationally indistinguishable* if for every polynomial time algorithm $\mathcal{A}$ it holds that

$$\left|\Pr[\mathcal{A}^{D_{0,\lambda}}(1^\lambda) = 1] - \Pr[\mathcal{A}^{D_{1,\lambda}}(1^\lambda)] = 1\right| \leq negl(\lambda),$$

where the oracle $D_{b,\lambda}$ is one that returns a fresh sample from $D_{b,\lambda}$ on every call.

Bit Decomposition. For some $n \in \mathbb{Z}$, define the gadget vector, $\mathbf{g} = (1, 2, 4, ..., 2^n)$ and the gadget function $\mathbf{g}^{-1} \colon \mathbb{Z} \cap [0, 2^{n+1}) \to \{0,1\}^{n+1}$ to be the function that computes the $(n+1)$-th bit decomposition of any integer. For some integer, z, the function is defined as $\mathbf{g}^{-1}(z)^T = \mathbf{v}^T = (v_0, v_1, ..., v_n)$ where $v_i \in \{0, 1\}$ such that $z = \langle \mathbf{g}, \mathbf{v} \rangle$. By extension we define the augmented gadget function $\mathbf{G}^{-1} \colon (\mathbb{Z} \cap [0, 2^{n+1}))^k \to \{0,1\}^{(n+1)\times k}$ to be the function that computes the $(n+1)$-th bit decomposition of every integer in a vector, $\mathbf{z}$, of dimension k, and arranges them as vector columns of an $(n+1)\times k$ binary matrix which we denote $\mathbf{G}^{-1}(\mathbf{z})$. Hence, $\mathbf{g} \cdot \mathbf{G}^{-1}(\mathbf{z}) = \mathbf{z}$.

CRT Representation. We recall the Chinese Remainder Theorem over the integers, which we use during the construction of the batch version of the scheme.

Definition 2.1. *Given k pair-wise co-prime integers $\mathbf{p} = (p_1, ..., p_k)$, let $\pi = \prod_{i=1}^{k} p_i$. For k integers $m_i \in \mathbb{Z}_{p_i}$, we define the CRT representation of $\mathbf{m} = (m_1, ..., m_k)$ with respect to $\mathbf{p}$, to be the unique field element $m \in \mathbb{Z}_\pi$ such that $[m]_{p_i} = m_i$; we write $m = CRT_{p_1,...,p_k}(m_1, ..., m_k)$ and recall that given $\mathbf{p}$ and $\mathbf{m}$, $CRT_{\mathbf{p}}(\mathbf{m})$ is an efficiently computable ring isomorphism from $\prod \mathbb{Z}_{p_i}$ to $\mathbb{Z}_\pi$.*

Leftover Hash Lemma. A family $\mathcal{H}$ of hash functions from X to Y, both finite sets, is said to be 2-universal if for all distinct $x, x' \in X, Pr_{h \leftarrow \mathcal{H}}[h(x) = h(x')] = 1/|Y|$. A distribution D is ε-uniform if its statistical distance from the uniform distribution is at most ε, where the statistical distance between two distributions D_1, D_2 over a finite domain X is $\frac{1}{2}\sum_{x \in X} |D_1(x) - D_2(x)|$.

Lemma 2.1 (Simplified Leftover Hash Lemma (LHL) [22]). *Let $\mathcal{H}$ be a family of 2-universal hash functions from X to Y. Suppose that $h \leftarrow \mathcal{H}$ and $x \leftarrow X$ are chosen uniformly and independently. Then, $(h, h(x))$ is $\frac{1}{2}\sqrt{|Y|/|X|}$-uniform over $\mathcal{H} \times Y$.*

We present the following version of the LHL, specifically adapted to our scheme.

Lemma 2.2. *Set $\mathbf{x} = (x_1, ..., x_m) \leftarrow \mathbb{Z}_M^m$ uniformly and independently, set $\mathbf{S} \leftarrow \{0,1\}^{m \times n}$ for some n; and let $\mathbf{y} = \mathbf{x} \cdot \mathbf{S} \pmod M$. Then $(\mathbf{x}, \mathbf{y})$ is $1/2\sqrt{M/2^m}$-uniform over $\mathbb{Z}_M^{m+n}$.*

Proof. Consider the following hash function family $\mathcal{H} \subseteq \{0,1\}^{m \times n} \to \mathbb{Z}_M^n$. Each function $h \in \mathcal{H}$ is parametrized by the coordinates of $\mathbf{x} \in \mathbb{Z}_M^m$. Now, given any $\mathbf{S} \in \{0,1\}^{m \times n}$, we define $h(\mathbf{S}) = \mathbf{x} \cdot \mathbf{S} \pmod M \in \mathbb{Z}_M^n$. We have that the function family is 2-universal. Therefore by Lemma 2.1, $(h, h(\mathbf{S}))$ is $1/2\sqrt{M/2^m}$-uniform over $\mathbb{Z}_M^{m+n}$. □

2.1 Homomorphic Encryption

A homomorphic (public-key) encryption scheme $\mathsf{HE} = (\mathsf{HE.Setup}, \mathsf{HE.Keygen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Eval})$ with message space $\mathcal{M}$ is a 4-tuple of PPT algorithms as follows (λ is the security parameter):

- **Key generation** $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$**:** Outputs a public encryption key pk and a secret decryption key sk.
- **Encryption** $c \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, \mu)$**:** Using the public key pk, encrypts a message $\mu \in \mathcal{M}$ into a ciphertext c.
- **Decryption** $\mu \leftarrow \mathsf{HE.Dec}(\mathsf{sk}, c)$**:** Using the secret key sk, decrypts a ciphertext c to recover the message $\mu \in \mathcal{M}$.
- **Homomorphic evaluation** $\widehat{c} \leftarrow \mathsf{HE.Eval}(C, (c_1, \ldots, c_\ell), \mathsf{pk})$**:** Using the public key pk, applies a circuit $C \colon \mathcal{M}^\ell \to \mathcal{M}$ to $c_1, \ldots, c_\ell$, and outputs a ciphertext $\widehat{c}$.

A homomorphic encryption scheme is said to be secure if it is semantically secure. It is (perfectly) correct w.r.t a class of circuits $\mathcal{C}$, if for any efficiently computable circuit $C \in \mathcal{C}$ and any set of inputs $\mu_1, \ldots, \mu_\ell$, letting $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$ and $c_i \leftarrow \mathsf{HE.Enc}(\mathsf{pk}, \mu_i)$, it holds that $\mathsf{HE.Dec}(\mathsf{sk}, \mathsf{HE.Eval}(C, (c_1, \ldots, c_\ell), \mathsf{pk})) = C(\mu_1, \ldots, \mu_\ell)$. The scheme is compact if the decryption circuit's size only depends on λ. The scheme is leveled fully homomorphic if for every $L = \text{poly}(\lambda)$ it can be instantiated so that it can evaluate all depth L circuits.

2.2 Approximate GCD (AGCD)

Variants of the Approximate GCD problem have been used for homomorphic encryption in a number of previous works [9, 10, 15–17, 21, 31]. In this work, we consider the decisional noisy variant, both in the standalone and batch regimes (definitions follow). We also show that the decisional noisy variant implies the hardness of "size-bounded" decisional AGCD which had been defined and used implicitly in the noise-free setting but to our knowledge not in the noisy setting.

We start by defining the distribution that underlies the AGCD problem. Essentially, this is a distribution over "near multiples" of a hidden parameter p, followed by a definition of the (standalone) AGCD problem.

Definition 2.2. *The distribution $\mathcal{D}_{\gamma,\rho}(p)$, parameterized by integers γ, ρ and a η-bit prime p, is supported over γ-bit integers and defined as follows.*

$$\mathcal{D}_{\gamma,\rho}(p) = \{\textit{sample } q \leftarrow \mathbb{Z} \cap [0, 2^{\gamma}/p), r \leftarrow \mathbb{Z} \cap (-2^{\rho}, 2^{\rho}) : \textit{Output } x = p \cdot q + r\} \tag{2}$$

Definition 2.3 ((ρ, η, γ)-AGCD [9, 31]). *The (ρ, η, γ)-AGCD problem is to find p given oracle access to $\mathcal{D}_{\gamma,\rho}(p)$, where p is a random η-bit prime. The decisional AGCD problem is to distinguish between $\mathcal{D}_{\gamma,\rho}(p)$ and the uniform distribution on $[0, 2^{\gamma}) \cap \mathbb{Z}$, given oracle access to both distributions.*

We note that these definitions are valid even if p is a non-prime odd integer, and our results carry over to this case as well.

The batched version is defined similarly, except with multiple p's. We start by defining a noise distribution via CRT representation, followed by the batch AGCD problem definition.

Definition 2.4. *Let $p_1, ..., p_l$ be η-bit primes. We define the following distribution:*

$$\Phi_{\rho}(p_1, ..., p_l) = \{r = CRT_{p_1,...,p_l}(r_1, ..., r_l) | r_i \leftarrow \mathbb{Z} \cap (-2^{\rho}, 2^{\rho})\} . \tag{3}$$

Definition 2.5 ((ρ, η, Γ)-l-AGCD). *Let ρ, η, Γ, l be parameters instantiated as a function of the security parameter. Let $p_1, ..., p_l$ be random η-bit sized primes and define $\pi = \prod_{i=1}^{l} p_i$. Given oracle access to the distribution*

$$\mathcal{X}_{\rho,\Gamma}(p_1, ..., p_l) = \{q\pi + r | r \leftarrow \Phi_{\rho}(p_1, ..., p_l), q \leftarrow \mathbb{Z} \cap [0, 2^{\Gamma}/\pi)\} ,$$

output at least one of $p_1, ..., p_l$. The decisional version is to distinguish between $\mathcal{X}_{\rho,\Gamma}(p_1, ..., p_l)$ and the uniform distribution on $\mathbb{Z} \cap [0, 2^{\Gamma})$.

We note that for $l = 1$ we get exactly the non-batched version, so this is a strict generalization.

Size Bounded AGCD. The distribution $\mathcal{D}_{\gamma,\rho}(p)$ defined above (and respectively $\mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)$ in the batch variant) produces elements of varying length. For functionality purposes, we would like to perform arithmetics with a single modulus x_0 in our scheme, and this modulus must itself be $\approx q_0 p$ (and respectively for the batch version). In some previous works [9, 13, 15–17, 25], this was done by defining the modulus as a special noise free element $x_0 = q_0 p$. However, this could potentially weaken security and in particular makes the scheme vulnerable to quantum attacks (since factoring x_0 reveals p). Sampling x_0 from the distribution $\mathcal{D}_{\gamma,\rho}(p)$ itself was proposed already in [15, 31], the former work only in the context of search AGCD, and the latter with an ad-hoc (unmentioned) circular security assumption. Recall that the standalone AGCD is a special case of the batched version, and therefore it is sufficient to take care of the latter.

We start by defining truncated versions of distributions, i.e. a distribution conditioned on some external condition.

Definition 2.6. *Let X be a distribution supported over $\mathbb{Z}$ and let $k \in \mathbb{Z}$. The distribution $X^{(\leq k)}$ is the distribution X conditioned on $X \leq k$. If $\Pr[X \leq k] = 0$ then $X^{(\leq k)}$ is undefined. Analogously we can define $X^{(\geq k)}$.*

Via rejection sampling it is easy to see that if X is efficiently sampleable and $\Pr[X \leq k]$ is noticeable then $X^{(\leq k)}$ is efficiently sampleable up to negligible statistical distance.

Lemma 2.3. *Let $(\rho, \eta, \gamma), l$ be as in the l-AGCD problem. For any polynomial t, we define the following distributions.*

- *The distribution $\vec{\mathcal{X}_t} = (x_0, x_1, \ldots, x_t)$ where $x_0 \leftarrow \mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)^{(\geq 2^{\Gamma}/2)}$ and for $i > 0$, $x_i \leftarrow \mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)^{(\leq x_0)}$.*
- *The distribution $\vec{\mathcal{U}_t} = (u_0, u_1, \ldots, u_t)$ where u_0 is uniform over $[2^{\Gamma}/2, 2^{\Gamma}) \cap \mathbb{Z}$ and for $i > 0$, u_i is uniform over $[0, u_0) \cap \mathbb{Z}$.*

It holds that under the (ρ, η, Γ)-l-AGCD assumption, both distributions are efficiently sampleable, up to negligible statistical distance, and computationally indistinguishable for any polynomial t.

Proof. Let U be the uniform distribution over $[0, 2^{\Gamma}) \cap \mathbb{Z}$. Then an equivalent formulation for $\vec{\mathcal{U}_t}$ is to set $u_0 \leftarrow U^{(\geq 2^{\Gamma}/2)}$ and for $i > 0$, $u_i \leftarrow U^{(\leq u_0)}$. In this formulation, $\vec{\mathcal{U}_t}$ is efficiently sampleable given oracle access to U and using rejection sampling, since for u_0 the rejection probability is at most $1/2$ and since $u_0 \geq 2^{\Gamma}/2$, the same holds for the rest of the u_i's. Note that in expectation only a constant number of samples of U is required to sample each u_i.

Replacing U with $\mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)$ implies the distribution $\vec{\mathcal{X}_t}$. Since U and $\mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)$ are computationally indistinguishable under (ρ, η, Γ)-l-AGCD, it implies that applying the same rejection sampler but with $\mathcal{X}_{\rho,\Gamma}(p_1,...,p_l)$ samples instead of U samples will efficiently sample from $\vec{\mathcal{X}_t}$ and furthermore that the resulting distribution is indistinguishable from $\vec{\mathcal{U}_t}$. □

3 Our Basic Scheme

In this section we will describe the full construction of our decomposed homomorphic encryption scheme, we will analyze it for correctness and efficiency and finally prove its underlying security.

3.1 Construction

We recall that for a specific η-bit odd integer p, we use the distribution from Definition 2.2 over γ-bit integers.

$\mathsf{HE.Keygen}(1^\lambda)$*:* We generate the public parameters $params = \{\gamma, \rho, \eta, \tau\}$ according to the security parameter λ and the parameter constraints in Section 3.3. Sample uniformly an η-bit integer p. Using Eq. (2) and via rejection sampling first sample an integer $x_0 \leftarrow (\mathcal{D}_{\gamma,\rho}(p))^{(\geq 2^{\gamma-1})}$ and then τ integers $\{x_i\}_{1\leq i \leq \tau} \leftarrow (\mathcal{D}_{\gamma,\rho}(p))^{(\leq x_0)}$, such that $(x_0, x_1, \ldots, x_\tau) \leftarrow \vec{\mathcal{X}}_\tau$, as in Definition 2.6. We write $\mathbf{x} = (x_1, ..., x_\tau)$, we let $pk = (params, \mathbf{x}, x_0)$ and $sk = p$.

$\mathsf{HE.Enc}(pk, m)$*:* We randomly sample a matrix $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \gamma}$ and we compute

$$\mathbf{c} = [m \cdot \mathbf{g} + \mathbf{x}\mathbf{S}]_{x_0}$$

which is a vector of dimension $\gamma = \lceil \log x_0 \rceil$.

$\mathsf{HE.Eval}(pk, \mathcal{C}, \mathbf{c}_1, ...\mathbf{c}_t)$*:* Given t ciphertexts and a binary circuit $\mathcal{C}$ with t input bits, compute all the operations in the circuit over the integers and output the resulting integer modulo x_0. For pairwise ciphertexts, we compute the operations $\mathsf{HE.Mult}(\mathbf{c}_1, \mathbf{c}_2)$ and $\mathsf{HE.Nand}(\mathbf{c}_1, \mathbf{c}_2)$ in the following manner.

$$\begin{aligned} \mathbf{c}_{mult} &= \mathsf{HE.Mult}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{c}_1 \mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0}\,. \end{aligned} \tag{4}$$

We similarly define the homomorphic NAND operation.

$$\begin{aligned} \mathbf{c}_{nand} &= \mathsf{HE.Nand}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{g} - \mathbf{c}_1 \mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0}\,. \end{aligned}$$

Furthermore, our scheme allows for addition gates, $\mathsf{HE.Add}$, only in the case when it is known that *at most one* of the plaintext messages is 1. In this case we perform the addition, between two ciphertexts, in the following way

$$\begin{aligned} \mathbf{c}_{add} &= \mathsf{HE.Add}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{c}_1 + \mathbf{c}_2]_{x_0}\,. \end{aligned}$$

$\mathsf{HE.Dec}(sk, \mathbf{c})$: We have that $sk = p$. Hence, as mentioned earlier, this procedure simply computes $f = \mathbf{c} \cdot \mathbf{g}^{-1}(p/2) \pmod p$ and outputs the following

$$m \leftarrow \begin{cases} 1 & \text{if } |f| \geq p/4 \\ 0 & \text{if } |f| < p/4 \end{cases} .$$

3.2 Correctness and Noise Analysis

In this section we prove the correctness of our scheme and at the same time make a parallel analysis of the size of the noise component in the different relevant algorithms. Finally we present the optimal parameters as a function of the circuit depth.

Theorem 3.1. *For a boolean circuit, $\mathcal{C}$, of depth d, for $(sk, pk) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$ and $\mathbf{c} \leftarrow \mathsf{HE.Eval}(\mathcal{C}, \mathbf{c}_1, ..., \mathbf{c}_t)$ such that $\mathbf{c}_i = \mathsf{HE.Enc}(pk, m_i)$, where $m_i \in \{0, 1\}$. We have that*

$$\mathsf{HE.Dec}(sk, \mathbf{c}) = \mathcal{C}(m_1, ..., m_t) .$$

Remark 3.1. We note that we did not present the most general description of the boolean circuit to be evaluated. We mention a circuit of depth d, without considering the different effect of NAND and ADD gates. Our description assumes only NAND gates in the circuit. If we had a circuit with ADD gates, then its depth could be larger since the ADD gates contribute less to the growth of the noise of the ciphertext, as we will see below.

After the encryption procedure, $\mathsf{HE.Enc}$, the resulting ciphertext is a vector of dimension $\lceil \log x_0 \rceil = \gamma$ with each entry being an integer in $[-x_0/2, x_0/2)$. We now formally define the noise component of a ciphertext and analyze the size.

Definition 3.1 (Noise Component HE). *For any ciphertext $\mathbf{c}$, we define its noise component to be $\mathbf{r}_{m,p}(\mathbf{c}) = [\mathbf{c} - m\mathbf{g}]_p$ and define its size to be the norm $r_{m,p}(\mathbf{c}) = \|\mathbf{r}_{m,p}(\mathbf{c})\|$. Note that we consider $\mathbf{r}_{m,p}(\mathbf{c})$ over $\mathbb{Z}$ and not over $\mathbb{Z}_p$.*

In the following lemma we first give an upper bound on the size of the noise of a fresh ciphertext and then an upper bound on the noise growth during the evaluation function for the multiplication, nand and addition functions.

Lemma 3.1 (Noise Size). *Let $(pk, sk) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$. Let $\mathbf{c}_1, \mathbf{c}_2$ be two ciphertexts, respectively encrypting messages $m_1, m_2 \in \{0, 1\}$, we define*

$$B = \max\{r_{m_1,p}(\mathbf{c}_1), r_{m_2,p}(\mathbf{c}_2)\} .$$

The following holds

1. *Given a fresh encryption $\mathbf{c} = \mathsf{HE.Enc}(pk, m)$, the noise component, $\mathbf{r}_{m,p}(\mathbf{c})$ has norm $r_{m,p}(\mathbf{c}) \leq \tau 2^{\rho+1}$.*
2. *For $\mathbf{c}_{mult} = \mathsf{HE.Mult}(\mathbf{c}_1, \mathbf{c}_2)$ and $\mathbf{c}_{nand} = \mathsf{HE.Nand}(\mathbf{c}_1, \mathbf{c}_2)$, we have that $r_{m_{mult},p}(\mathbf{c}_{mult}) = r_{m_{nand},p}(\mathbf{c}_{nand}) \leq (2\gamma + 1)B$, where $m_{mult} = m_1 m_2$ and $m_{nand} = 1 - m_1 m_2$.*

3. For $\mathbf{c}_{add} = \mathsf{HE.Add}(\mathbf{c}_1, \mathbf{c}_2)$*, we have that* $r_{m_{add},p}(\mathbf{c}_{add}) \leq 2B + 2^{\rho+1}$*, where* $m_{add} = m_1 + m_2$*.*

Proof. 1. Let $pk = (x_0, \mathbf{x}) = (r_0 + q_0 \cdot p, \mathbf{r} + \mathbf{q} \cdot p)$. Given

$$\begin{aligned} \mathbf{c} &= [m\mathbf{g} + \mathbf{x}\mathbf{S}]_{x_0} \\ &= m\mathbf{g} + \mathbf{r}\mathbf{S} + \mathbf{q}\mathbf{S} \cdot p + \mathbf{k}x_0 \end{aligned}$$

the noise component is $\mathbf{r}_{m,p}(\mathbf{c}) = \mathbf{r}\mathbf{S} + \mathbf{k}r_0$, where $\mathbf{k}$ is the multiple of $x_0 = r_0 + q_0 \cdot p$ that is added after the $\bmod x_0$ operation on the ciphertext. The first term comes from the matrix operation with the public key vector, $\mathbf{x}\mathbf{S} = \mathbf{q}\mathbf{S}p + \mathbf{r}\mathbf{S}$, and has size $\|\mathbf{r}\mathbf{S}\| \leq \tau 2^{\rho}$. The second term derives from the modx_0 operation on the ciphertext. Since the ciphertext cannot grow by more than $\tau \cdot x_0$ in each coordinate, then we have that $\|\mathbf{k}\| \leq \tau$, such that $\|\mathbf{k}r_0\| \leq \tau 2^{\rho}$. Thus the claim follows and $r_{m,p}(\mathbf{c}) \leq \tau 2^{\rho+1}$.

2. Let $\mathbf{c}_1, \mathbf{c}_2, B$ and $\mathbf{c}_{mult}$ be as in the statement of the lemma. Then we have

$$\begin{aligned} \|\mathbf{r}_{m_{mult},p}(\mathbf{c}_{mult})\| &= \|[\mathbf{c}_{mult} - m_{mult}\mathbf{g}]_p\| \\ &= \|[[\mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0} - m_{mult}\mathbf{g}]_p\| \\ &= \|[\mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2) + \mathbf{k}_{mult}x_0 - m_{mult}\mathbf{g}]_p\| \\ &\leq \|[\mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2) - m_{mult}\mathbf{g}]_p\| + \|[\mathbf{k}_{mult}x_0]_p\| \,, \end{aligned}$$

where the last inequality comes from Eq. (1). Now $\|\mathbf{k}_{mult}\| \leq \gamma$ and $[x_0]_p = r_0$,[3] hence

$$\begin{aligned} &\|\mathbf{r}_{m_{mult},p}(\mathbf{c}_{mult})\| \leq \|[\mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2) - m_{mult}\mathbf{g}]_p\| + \gamma \cdot 2^{\rho} \\ &\leq \|[m_1\mathbf{c}_2 + (\mathbf{c}_1 - m_1\mathbf{g})\mathbf{G}^{-1}(\mathbf{c}_2) - m_{mult}\mathbf{g}]_p\| + \gamma \cdot 2^{\rho} \\ &\leq \|[m_1 m_2\mathbf{g} + m_1(\mathbf{c}_2 - m_2\mathbf{g}) + (\mathbf{c}_1 - m_1\mathbf{g})\mathbf{G}^{-1}(\mathbf{c}_2) - m_{mult}\mathbf{g}]_p\| + \gamma \cdot 2^{\rho} \\ &\leq \|m_1[\mathbf{c}_2 - m_2\mathbf{g}]_p\| + \|[\mathbf{c}_1 - m_1\mathbf{g}]_p\| \cdot \|\mathbf{G}^{-1}(\mathbf{c}_2)\| + \gamma \cdot 2^{\rho} \\ &\leq r_{m_2,p}(\mathbf{c}_2) + r_{m_1,p}(\mathbf{c}_1)\gamma + \gamma \cdot 2^{\rho} \\ &\leq B + B\gamma + \gamma \cdot 2^{\rho} \end{aligned}$$

since $m_{mult} = m_1 m_2$ by definition. Since $r_0 \leq 2^{\rho} \leq B$, we get

$$\begin{aligned} r_{m_{mult},p}(\mathbf{c}_{mult}) &\leq B + \gamma B + \gamma 2^{\rho} \\ &\leq (2\gamma + 1)B \,, \end{aligned}$$

which is exactly what we need. Analogously, the NAND operation causes the same increase in the size of the noise. This proves the statement.

[3] In order to upper bound $\mathbf{k}_{mult}$ we know that $\|\mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2)\| \leq \|\mathbf{c}_1\| \cdot \gamma \leq x_0/2 \cdot \gamma$ since $\mathbf{c}_1 \in [-x_0/2, x_0/2)$. Thus for each coordinate $1 \leq i \leq l$ of $\mathbf{c}_{mult}$ we have that

$$\begin{aligned} -x_0/2 &\leq [(c_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2))[i]]_{x_0} \\ &= (c_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2))[i] - k_{mult}[i]x_0 \leq x_0/2 \,. \end{aligned} \tag{5}$$

Hence, on the one hand we have $-x_0/2 \leq (c_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2))[i] - k_{mult}[i]x_0$, which gives $k_{mult}[i] \leq (\gamma + 1)/2$. On the other hand we have that $(c_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2))[i] - k_{mult}[i]x_0 \leq x_0/2$, which gives $(\gamma - 1)/2 < k_{mult}[i]$. Thus we conclude that $\|\mathbf{k}_{mult}\| \leq \gamma$.

3. Let $\mathbf{c}_1, \mathbf{c}_2, B$ and $\mathbf{c}_{add}$ be as in the statement of the lemma. Then we have

$$\begin{aligned}
\|\mathbf{r}_{m_{add},p}(\mathbf{c}_{add})\| &= \|[\mathbf{c}_{add} - m_{add}\mathbf{g}]_p\| \\
&= \|[[\mathbf{c}_1 + \mathbf{c}_2]_{x_0} - m_1\mathbf{g} - m_2\mathbf{g}]_p\| \\
&= \|[\mathbf{c}_1 - m_1\mathbf{g} + \mathbf{c}_2 - m_2\mathbf{g} - \mathbf{k}_{add}x_0]_p\| \\
&\leq \|[\mathbf{c}_1 - m_1\mathbf{g}]_p\| + \|[\mathbf{c}_2 - m_2\mathbf{g}]_p\| + \|[\mathbf{k}_{add}x_0]_p\| \\
&\leq \|\mathbf{r}_{m_1,p}(\mathbf{c}_1)\| + \|\mathbf{r}_{m_2,p}(\mathbf{c}_2)\| + \|\mathbf{k}_{add}r_0\| \\
&\leq r_{m_1,p}(\mathbf{c}_1) + r_{m_2,p}(\mathbf{c}_2) + 2^{\rho+1}
\end{aligned}$$

since in this case $\|\mathbf{c}_1 + \mathbf{c}_2\| \leq 2x_0$, thus $\|\mathbf{k}_{add}\| \leq 2$. This proves the statement.

□

Generalizing the statement in 2, let us assume that we want to compute a circuit of depth d on ciphertexts whose noise components are bounded by B. The output ciphertext, $\mathbf{c}_d$, an encryption of m_d, will have a noise component with norm $r_{m_d,p}(\mathbf{c}_d) \leq (2\gamma + 1)^d B$. We now prove decryption correctness.

Lemma 3.2 (Correctness Homomorphic Decryption). *For any vector $\mathbf{c} \in \mathbb{Z}_{x_0}^{\gamma}$ and any $m \in \mathbb{Z}_2$, if $r_{m,p}(\mathbf{c}) < p/(4\gamma)$ then $\mathsf{HE.Dec}(sk, \mathbf{c}) = m$.*

Proof. Assume that for a ciphertext $\mathbf{c} \in \mathbb{Z}_{x_0}^{\gamma}$ we have $r_{m,p}(\mathbf{c}) < p/(4\gamma)$, then during decryption, we have that

$$\begin{aligned}
f &= \mathbf{c} \cdot \mathbf{g}^{-1}(p/2) \pmod p \\
&= m\mathbf{g}\mathbf{g}^{-1}(p/2) + \mathbf{r}_{m,p}(\mathbf{c})\mathbf{g}^{-1}(p/2) \\
&\leq m(p/2) + \gamma\mathbf{r}_{m,p}(\mathbf{c}) \,,
\end{aligned}$$

where by assumption $\gamma\|\mathbf{r}_{m,p}(\mathbf{c})\| < p/4$. So by the decryption algorithm, if $|f| \leq m(p/2) + \gamma r_{m,p}(\mathbf{c}) < p/4$ then it necessarily means that $m = 0$. Whereas if instead $|f| \leq m(p/2) + \gamma r_{m,p}(\mathbf{c})$ and $|f| > p/4$ then it must be the case that $m = 1$, or else $\gamma r_{m,p}(\mathbf{c}) > p/4$. Hence in any case we get $\mathsf{HE.Dec}(sk, \mathbf{c}) = m$, which implies decryption correctness. □

As mentioned earlier, one of the most important features of homomorphic operations is the size of the noise component, which must be somewhat controlled and, in our scheme, must be at all times less than $p/(4\gamma)$ in order for the ciphertext to decrypt correctly. This can give us mathematically an upper bound on the number of operations that we can perform, relative to some given parameters. Indeed, we can show the relationship between $\eta - \rho$ and the depth of the evaluation circuit d. Given that our scheme is not initially fully homomorphic, but only leveled-homomorphic, we must be able to compute $\eta - \rho$ given the circuit depth, d. This is proved in the next lemma.

Lemma 3.3. *Given a circuit of depth d and the parameters (τ, γ), correctness of the* HE *scheme implies that the following inequality is satisfied.*

$$\eta - \rho > (d + 1)\log\gamma + \log\tau + \mathcal{O}(1)\,. \tag{6}$$

Proof. Let us assume that the given circuit is of depth d, then in the worst-case scenario, a single ciphertext will undergo at most d multiplications with co-factor of the same noise level. Hence we know that if $\mathbf{c}_d$ is the output of the circuit, then $\|\mathbf{r}_{m_d,p}(\mathbf{c}_d)\| \leq (2\gamma+1)^d B$ by Lemma 3.1. We also know by Lemma 3.2 that the following inequality must hold for decryption correctness

$$\|\mathbf{r}_{m_d,p}(\mathbf{c}_d)\| < p/(4\gamma) < 2^\eta/(4\gamma)$$

and we can get a bound for $\eta - \rho$. More specifically we have that $\log B < (\eta-2) - d\log(2\gamma+1) - \log\gamma$. If we assume that B is the size of the noise of the input ciphertexts, which are fresh out of the encryption procedure, then $B = \tau 2^{\rho+1}$ by Lemma 3.1 and

$$\begin{aligned} & (\eta-2) - d\log(2\gamma+1) - \log\gamma > \log B \\ \Rightarrow \quad & (\eta-2) - d\log(2\gamma+1) - \log\gamma - \log\tau > \rho+1 \\ \Rightarrow \quad & \eta - \rho > (d+1)\log\gamma + \log\tau + \mathcal{O}(1) \end{aligned}$$

as required. □

Remark 3.2. Note that if instead, we are given the parameters, $\{\rho, \eta, \tau, \gamma\}$, one can derive d, an upper bound for the depth of the circuit, by rearranging Eq. (6) as follows

$$d < \frac{(\eta-2) - (\rho+1)\log\tau}{\log(2\gamma+1)} .$$

3.3 Parameters

We present the constraints on the parameters needed in order for the HE scheme to be correct and secure against known attacks. Let d be the depth of the circuit used to evaluate the data and let λ be the security parameter.

- ρ is the bit-length of the noise components r_i's of the public key elements x_i's; $\rho = \omega(\lambda)$, to protect against brute-force attacks on the noise [8, 13, 17];
- η is the bit-length of the secret key p; $\eta = \Omega(\rho + (d+1)\log\gamma + \log\tau)$ in order to have correctness of the evaluation circuit (Lemma 3.3);
- γ is the bit-length of the elements of the public key, the x_i's; $\gamma \geq \Omega(\frac{\lambda}{\log\lambda}(\eta-\rho)^2)$ and $\gamma > \eta^2$, to thwart different lattice reduction attacks on the AGCD as studied in [10, 19] such as the orthogonal lattice attacks [27, 31], the simultaneous Diophantine approximation attack [26, 31] and the multivariate polynomial approach in [12, 23].
- τ is the number of x_i's in the public key; $\tau = \gamma + \Omega(\lambda)$, which is derived from the constraints given by the Leftover Hash Lemma in Section 2, needed in the security proof below. We note that the constraint requires that $1/2\sqrt{x_0/2^\tau}$ be negligible, where $\gamma = \lceil \log x_0 \rceil$.

3.4 Semantic Security

To prove the security of this scheme, we cannot use the same techniques as in DGHV because the use of the gadget vector causes the message to be encrypted in a higher bit, and thus the LSB predictor procedure used in [31] fails to output the correct bit. For this reason, we reduce the security of this scheme to the *decisional* AGCD problem instead. Hence, simply stated, we prove that our scheme is CPA secure under the *decisional* AGCD assumption.

Theorem 3.2. *The above* HE *scheme is CPA secure under the* (ρ, η, γ)-decisional *AGCD assumption.*

Proof. For $pk = (params, \mathbf{x}, x_0) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$ and $\mathbf{c}_b = \mathsf{HE.Enc}(pk, b)$, where $b \leftarrow \{0,1\}$, $pk' = (params, \mathbf{u}, u_0)$ where $u_0 \leftarrow ([0, 2^\gamma) \cap \mathbb{Z})^{(\geq 2^{\gamma-1})}$ and $\mathbf{u} \leftarrow \mathcal{U}([0, u_0) \cap \mathbb{Z})^\tau$, we prove that $(pk, \mathbf{c}_b)$ and $(pk', \mathbf{v})$, where $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^\gamma$, are computationally indistinguishable. In other words, for every polynomial time algorithm $\mathcal{A}$, $\left|\Pr[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1] - \Pr[\mathcal{A}(1^\lambda, pk', \mathbf{v}) = 1]\right| = negl(\lambda)$. In order to do this we will use a three step hybrid argument and use Lemmas 2.2 and 2.3, for $l = 1$, to show indistinguishability between each hybrid. For simplicity we use the notation in Lemma 2.3, where we sample from the distributions $\vec{\mathcal{X}}_\tau = (x_0, x_1, \ldots, x_\tau)$ and $\vec{\mathcal{U}}_\tau = (u_0, u_1, \ldots, u_\tau)$.

Hybrid 0: We define the distribution $(pk, \mathbf{c}_b)$ in the following way, let $pk = (params, \mathbf{x}, x_0) \leftarrow \mathsf{HE.Keygen}(1^\lambda)$, such that $(x_0, x_1, \ldots, x_\tau) \leftarrow \vec{\mathcal{X}}_\tau$, and let $\mathbf{c}_b = \mathsf{HE.Enc}(pk, b) = [b\mathbf{g} + \mathbf{xS}]_{x_0}$ for $b \leftarrow \{0,1\}$. In this case, $(pk, \mathbf{c}_b)$ is distributed exactly as in our HE scheme.

Hybrid 1: We define the distribution $(pk, \mathbf{c}_b)$ as follows, let $pk = (params, \mathbf{u}, u_0)$, such that $(u_0, u_1, \ldots, u_\tau) \leftarrow \vec{\mathcal{U}}_\tau$, and let $\mathbf{c}_b = \mathsf{HE.Enc}(pk, b) = [b\mathbf{g} + \mathbf{uS}]_{u_0}$. Now, by Lemma 2.3 we know that $(pk, \mathbf{c}_b)_{H_0}$ and $(pk, \mathbf{c}_b)_{H_1}$ are computationally indistinguishable because $\vec{\mathcal{X}}_\tau$ and $\vec{\mathcal{U}}_\tau$ are indistinguishable by the lemma. Hence we have

$$\left|\Pr_{H_0}[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1] - \Pr_{H_1}[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1]\right| \leq negl(\lambda)\,.$$

Hybrid 2: In this hybrid we define the distribution $(pk, \mathbf{c}_b)$ as follows. Again, let $pk = (params, \mathbf{u}, u_0)$, where $(u_0, u_1, \ldots, u_\tau) \leftarrow \vec{\mathcal{U}}_\tau$, and let $\mathbf{c}_b = [b\mathbf{g} + \mathbf{v}]_{u_0}$, where $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^\gamma$ is completely random. By the LHL in Lemma 2.2, the statistical distance between $(u_0, \mathbf{u}, \mathbf{uS})$ and $(u_0, \mathbf{u}, \mathbf{v})$ is upper bounded by $\frac{1}{2}\sqrt{u_0/2^\tau} = negl(\lambda)$. Finally, we know that $\mathbf{c}_b \equiv \mathbf{v}$, hence the probability of success for $\mathcal{A}$ in this hybrid is exactly $1/2$ since $\mathbf{v}$ is completely random. So we have that

$$\left|\Pr_{H_1}[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1] - \Pr_{H_2}[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1]\right| \leq \frac{1}{2}\sqrt{u_0/2^\tau}\,.$$

Thus we conclude that

$$\begin{aligned}&\left|\Pr_{H_0}[\mathcal{A}(1^\lambda, pk, \mathbf{c}_b) = 1] - \Pr_{H_2}[\mathcal{A}(1^\lambda, pk, \mathbf{v}) = 1]\right| \\ &\leq negl(\lambda) + \frac{1}{2}\sqrt{u_0/2^\tau} \\ &= negl(\lambda)\end{aligned}$$

as desired. □

4 Batch Generalization Construction

In this section we present a batched version of our scheme, called BHE, which uses the CRT representation to encrypt several messages at a time.

4.1 Overview

In this section we generalize our construction to allow for encryption of several messages at the same time. Given the messages $\mathbf{m} = (m_1, ..., m_l)$ where $m_i \in \{0, 1\}$ we want to pack these $l \in \mathbb{Z}$ messages into a single ciphertext. For this we will use the Chinese Remainder Theorem (CRT). It follows from the CRT and from modular arithmetic that homomorphic multiplication, i.e. MULT , and hence the NAND operation, will apply in parallel and component-wise. Let us look at the general idea first. We sample l primes of η-bit length $p_1, ..., p_l$ and define $\pi = \prod_{i=1}^{l} p_i$. Given $x_0 \in \mathbb{Z} \cap [0, 2^\Gamma)$ from the public key, the ciphertext has the following structure:

$$\mathbf{c} = [m\mathbf{g} + \mathbf{r} + \mathbf{q}\pi]_{x_0}$$

where we have that $m = \mathrm{CRT}_{p_1,...,p_l}(m_1, ..., m_l)$ for $\mathbf{r} = (r_0, ..., r_n)$ such that $r_i = \mathrm{CRT}_{p_1,...,p_l}(r_{i,1}, ..., r_{i,l})$ for $r_{i,j} \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)$ and finally $\mathbf{q} = (q_1, ..., q_n)$ such that $q_i \leftarrow \mathbb{Z} \cap [0, 2^\Gamma/\pi)$.

This packing method still allows for homomorphic multiplication, in the same way as for the single message construction,

$$\begin{aligned}\mathbf{c}_3 &= \mathsf{BHE.Mult}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0}.\end{aligned}$$

Finally, decryption happens in a very similar way, we compute $f_i = \mathbf{c}\mathbf{g}^{-1}(p_i/2) \mod p_i$ for all $1 \leq i \leq l$. Then for each f_i if $|f_i| \geq p_i/4$ output $m_i = 1$, otherwise $m_i = 0$.

4.2 The Batch Construction

$\mathsf{BHE.Keygen}(1^\lambda)$: We first generate the parameters $params = \{\Gamma, \rho, \eta, \tau, l\}$ according to the security parameter λ and correctness as explained in Section 4.4 below. Then we sample l η-bit primes $p_1, ..., p_l$ and let $\pi = \prod_{i=1}^{l} p_i$. We define $\pi_i = \frac{\pi}{p_i}$ and we let $y_i =$

$CRT_{p_1,...,p_l}(0,...,1,...,0) = \pi_i(\pi_i^{-1} \mod p_i) \mod \pi$ where only the i-th coordinate is nonzero. We will post encryptions of the y_i as a part of the public key so as to allow public encryption.

We sample $(x_0, x_1, \ldots, x_\tau) \leftarrow \vec{\mathcal{X}}_\tau$, where $\vec{\mathcal{X}}_\tau$ is as defined in Lemma 2.3.

We denote $\mathbf{x} = (x_1, ..., x_\tau)$, $\mathbf{r} = (r_1, ..., r_\tau)$ and $\mathbf{q} = (q_1, ..., q_\tau)$. Thus by sampling l matrices $\mathbf{W}_i \leftarrow \{0,1\}^{\tau\times(\mathit{\Gamma}+1)}$, the ciphertexts of the y_i's look like

$$\begin{aligned}\mathbf{y}_i &= [y_i\mathbf{g} + \mathbf{x}\mathbf{W}_i]_{x_0} \\ &= [y_i\mathbf{g} + \mathbf{r}_i + \mathbf{q}_i\pi]_{x_0} \end{aligned} \tag{7}$$

where $\mathbf{r}_i = \mathbf{r}\mathbf{W}_i$ and $\mathbf{q}_i = \mathbf{q}\mathbf{W}_i$. We set $\mathbf{Y} = (\mathbf{y}_1 \ ... \ \mathbf{y}_l)$. The public key is defined as $pk = (params, x_0, \mathbf{x}, \mathbf{Y})$ and the secret key as $sk = (p_1, ..., p_l)$.

$\mathsf{BHE.Enc}(pk, m_1, ..., m_l)$*:* For $m_i \in \{0,1\}$, sample a matrix $\mathbf{S} \leftarrow \{0,1\}^{\tau\times\mathit{\Gamma}}$. Then we encrypt the messages $\mathbf{m} = (m_1, ..., m_l)$ as follows:

$$\mathbf{c} = [\mathbf{m}\mathbf{Y} + \mathbf{x}\mathbf{S}]_{x_0}\,.$$

$\mathsf{BHE.Eval}(pk, \mathcal{C}, \mathbf{c}_1, ...\mathbf{c}_t)$*:* For any boolean circuit $\mathcal{C}$ we can homomorphically compute the operations $\mathsf{BHE.Mult}$ and $\mathsf{BHE.Nand}$ in an almost identical manner as in the non batch version. The former is computed as follows

$$\begin{aligned}\mathbf{c}_{mult} &= \mathsf{BHE.Mult}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0}\end{aligned}$$

and the latter as

$$\begin{aligned}\mathbf{c}_{nand} &= \mathsf{BHE.Nand}(\mathbf{c}_1, \mathbf{c}_2) \\ &= [\mathbf{g} - \mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0}\,.\end{aligned}$$

$\mathsf{BHE.Dec}(sk, \mathbf{c})$*:* Given $\mathbf{c}$ we simply compute for each $i = 1, ..., l,\ f_i = \mathbf{c}\cdot\mathbf{g}^{-1}(p_i/2) \mod p_i$. So for each f_i if $|f_i| \geq p/4$ then $m_i = 1$, otherwise $m_i = 0$.

4.3 Correctness and Noise Analysis

The goal of this section is again to prove the correctness of our scheme.

Theorem 4.1. *For a boolean circuit of depth d, $\mathcal{C}$, for $(sk, pk) \leftarrow \mathsf{BHE.Keygen}(1^\lambda)$ and $\mathbf{c} \leftarrow \mathsf{BHE.Eval}(\mathcal{C}, \mathbf{c}_1, ..., \mathbf{c}_t)$ such that $\mathbf{c}_i = \mathsf{BHE.Enc}(pk, \mathbf{m}_i)$, where $\mathbf{m}_i \in \{0,1\}^l$. We have that*

$$\mathsf{BHE.Dec}(sk, \mathbf{c}) = \mathcal{C}(\mathbf{m}_1, ..., \mathbf{m}_t)\,.$$

Remark 3.1 also applies in this case.

We want to make an analysis of the noise components similar to the one in Section 3.2. We will start by defining the noise component of the ciphertext for the batched version of the scheme and then we will prove decryption correctness and show how the size behaves in this batch version of the scheme. We first note that after encryption, $\mathsf{BHE.Enc}(pk, \mathbf{m})$, the resulting ciphertext $\mathbf{c}$ is a vector of dimension $\lceil \log x_0 \rceil = \mathit{\Gamma}$ with each entry being an integer in $[-x_0/2, x_0/2)$.

Definition 4.1 (Batch Noise Component). *For any ciphertext* $\mathbf{c}$*, we define its noise components to be* $\mathbf{r}_{m,p_i}(\mathbf{c}) = [\mathbf{c} - m\mathbf{g}]_{p_i}$ *for any* $i = 1, ..., l$*. Therefore its size is the norm* $r_{m,p_i}(\mathbf{c}) = \|\mathbf{r}_{m,p_i}(\mathbf{c})\|$*. Here* $m = CRT_{p_1,...,p_l}(m_1, ..., m_l)$*. We will further define the overall noise component of a ciphertext* $\mathbf{c}$ *to be* $r_{m,\pi}(\mathbf{c}) = \max_{0 \le i \le l} r_{m,p_i}(\mathbf{c})$*. We consider* $\mathbf{r}_{m,p_i}(\mathbf{c})$ *over* $\mathbb{Z}$ *and not over* $\mathbb{Z}_{p_i}$*.*

Lemma 4.1. *Given a vector* $\mathbf{m} \in \{0,1\}^l$ *and a public key* $pk = (params, \mathbf{x}, \mathbf{Y}, x_0)$*, the ciphertext is of the form* $\mathbf{c} = \mathsf{BHE.Enc}(pk, \mathbf{m}) = [CRT_{p_1,...,p_l}(m_1\mathbf{g} + \mathbf{r}_1, ..., m_l\mathbf{g} + \mathbf{r}_l) + q \cdot \pi]_{x_0}$*.*

Proof. The encryption procedure computes

$$\begin{aligned}
\mathsf{BHE.Enc}(pk,\mathbf{m}) &= [\mathbf{mY} + \mathbf{xS}]_{x_0} \\
&= [\sum_{i=1}^{l}(m_i y_i \mathbf{g} + m_i \mathbf{r}\mathbf{W}_i + m_i \mathbf{q}\mathbf{W}_i \pi) + \mathbf{xS}]_{x_0} \\
&= [CRT_{p_1,...,p_l}(m_1\mathbf{g}, ..., m_l\mathbf{g}) + \sum_{i=1}^{l}(m_i \mathbf{r}\mathbf{W}_i + m_i \mathbf{q}\mathbf{W}_i \cdot \pi) + \mathbf{xS}]_{x_0} ,
\end{aligned}$$

where the last steps is obtained by the linear nature of the CRT representation, which allows for scalar multiplications and point-wise additions. Since $\mathbf{x} = \mathbf{r} + \mathbf{q} \cdot \pi$, we have that for each p_i

$$\begin{aligned}
&\sum_{i=1}^{l}(m_i \mathbf{r}\mathbf{W}_i) + \sum_{i=1}^{l}(m_i \mathbf{q}\mathbf{W}_i \cdot \pi) + \mathbf{rS} + \mathbf{qS} \cdot \pi \\
&= \sum_{i=1}^{l} m_i \sum_{j=1}^{\tau} \mathbf{r}_j \cdot \mathbf{w}_{j,i} + \sum_{j=1}^{\tau} \mathbf{r}_j \cdot \mathbf{s}_j \pmod{p_i} \\
&= \hat{\mathbf{r}}_i
\end{aligned}$$

where $\mathbf{x} = \mathbf{r}_j \pmod{p_i}$ are vectors whose coordinates are in $\mathit{\Phi}_\rho(p_1, ..., p_l)$, as in Eq. (3). So by combining the two we get that $\mathbf{c} = m_i\mathbf{g} + \hat{\mathbf{r}}_i \pmod{p_i}$ and hence

$$\mathbf{c} = \mathsf{BHE.Enc}(pk, \mathbf{m}) = [CRT_{p_1,...,p_l}(m_1\mathbf{g} + \mathbf{r}_1, ..., m_l\mathbf{g} + \mathbf{r}_l) + q \cdot \pi]_{x_0} \tag{8}$$

as required. □

In the following lemma we first give an upper bound on the size of the noise of a fresh ciphertext in the batch scheme and then an upper bound on the noise growth during the evaluation function for the multiplication, nand and addition functions.

Lemma 4.2 (Batch Noise Size). *Let* $(pk, sk) \leftarrow \mathsf{BHE.Keygen}(1^\lambda)$*. For any two ciphertexts* $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{Z}_{x_0}^{\Gamma}$*, encrypting messages* $\mathbf{m}_1, \mathbf{m}_2 \in \{0,1\}^l$*, we define* $B = \max\{r_{m_1,\pi}(\mathbf{c}_1), r_{m_2,\pi}(\mathbf{c}_2)\}$*. The following holds*

1. *Given a fresh encryption* $\mathbf{c} = \mathsf{BHE.Enc}(pk, \mathbf{m})$*, the noise component,* $\mathbf{r}_{m,\pi}(\mathbf{c})$ *has norm* $r_{m,\pi}(\mathbf{c}) \leq (l+2)\tau 2^\rho$*.*
2. *For* $\mathbf{c}_{mult} = \mathsf{BHE.Mult}(\mathbf{c}_1, \mathbf{c}_2)$ *and* $\mathbf{c}_{nand} = \mathsf{BHE.Nand}(\mathbf{c}_1, \mathbf{c}_2)$*, we have that* $r_{m_{mult},\pi}(\mathbf{c}_{mult}) = r_{m_{nand},\pi}(\mathbf{c}_{nand}) \leq (2\Gamma + 1)B$*, where* $m_{mult} = m_1 m_2$ *and* $m_{nand} = 1 - m_1 m_2$*.*
3. *For* $\mathbf{c}_{add} = \mathsf{BHE.Add}(\mathbf{c}_1, \mathbf{c}_2)$*, we have that* $r_{m_{add},\pi}(\mathbf{c}_{add}) \leq 2B + 2^{\rho+1}$*, where* $m_{add} = m_1 + m_2$*.*

Proof. 1. Let $pk = (x_0, \mathbf{x}) = (r_0 + q_0 \cdot \pi, \mathbf{r} + \mathbf{q} \cdot \pi)$. From Eq. (8) we know that

$$\begin{aligned}\mathbf{c} &= [m\mathbf{g} + \hat{\mathbf{r}} + \hat{\mathbf{q}} \cdot \pi]_{x_0} \\ &= m\mathbf{g} + \hat{\mathbf{r}} + \hat{\mathbf{q}} \cdot \pi + \mathbf{k}x_0\end{aligned}$$

where $\hat{\mathbf{r}} = \sum_{i=1}^{l}(m_i \mathbf{r}\mathbf{W}_i) + \mathbf{r}\mathbf{S}$ and $\hat{\mathbf{q}} = \sum_{i=1}^{l}(m_i \mathbf{q}\mathbf{W}_i) + \mathbf{q}\mathbf{S}$. Thus the noise component is $\mathbf{r}_{m,\pi}(\mathbf{c}) = (\sum_{i=1}^{l} m_i \mathbf{r}\mathbf{W}_i) + \mathbf{r}\mathbf{S} + r_0\mathbf{k}$, where $\mathbf{k}$ is the multiple of x_0 that is added after the $\mod x_0$ operation on the ciphertext. As we have seen in Lemma 4.1 the first term, $(\sum_{i=1}^{l} m_i \mathbf{r}\mathbf{W}_i)$, comes from the matrix operation $\mathbf{m}\mathbf{Y}$ such that $\|\sum_{i=1}^{l} m_i \mathbf{r}\mathbf{W}_i\| \leq l\tau 2^\rho$; the second term of the noise above comes from the matrix operation with the public key vector, $\mathbf{x}\mathbf{S} = \mathbf{q}\mathbf{S}p + \mathbf{r}\mathbf{S}$, with size $\|\mathbf{r}\mathbf{S}\| \leq \tau 2^\rho$. Finally, the last term derives from the mod x_0 operation on the ciphertext. We have that $\|\mathbf{k}\| \leq \tau$, such that $\|r_0\mathbf{k}\| \leq \tau 2^\rho$. The claim follows since

$$r_{m,\pi}(\mathbf{c}) \leq l\tau 2^\rho + \tau 2^\rho + \tau 2^\rho = (l+2)\tau 2^\rho \tag{9}$$

as required.

2. Let $\mathbf{c}_1, \mathbf{c}_2, B$ and $\mathbf{c}_{mult}$ be as in the statement of the lemma. Then we have

$$\begin{aligned}
\|\mathbf{r}_{m_{mult},\pi}(\mathbf{c}_{mult})\| &= \max_{i=1}^{l}(\|[\mathbf{c}_{mult} - m_{mult}\mathbf{g}]_{p_i}\|) \\
&= \max_{i=1}^{l}(\|[[\mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2)]_{x_0} - m_{mult}\mathbf{g}]_{p_i}\|) \\
&= \max_{i=1}^{l}(\|[\mathbf{c}_1\mathbf{G}^{-1}(\mathbf{c}_2) + \mathbf{k}_{mult}x_0 - m_{mult}\mathbf{g}]_{p_i}\|) \\
&= \max_{i=1}^{l}(\|[m_1\mathbf{c}_2 + (\mathbf{c}_1 - m_1\mathbf{g})\mathbf{G}^{-1}(\mathbf{c}_2) + \mathbf{k}_{mult}x_0 - m_{mult}\mathbf{g}]_{p_i}\|) \\
&= \max_{i=1}^{l}(\|[m_1 m_2\mathbf{g} + m_1(\mathbf{c}_2 - m_2\mathbf{g}) + (\mathbf{c}_1 - m_1\mathbf{g})\mathbf{G}^{-1}(\mathbf{c}_2) \\
&\qquad\qquad + \mathbf{k}_{mult}x_0 - m_{mult}\mathbf{g}]_{p_i}\|) \\
&\leq \max_{i=1}^{l}(\|m_{1,i}[\mathbf{c}_2 - m_2\mathbf{g}]_{p_i}\| + \|[\mathbf{c}_1 - m_1\mathbf{g}]_{p_i}\| \cdot \mathit{\Gamma} + \|[\mathbf{k}_{mult}x_0]_{p_i}\|) \\
&\leq \max_{i=1}^{l}(r_{m_2,p_i}(\mathbf{c}_2) + r_{m_1,p_i}(\mathbf{c}_1) \cdot \mathit{\Gamma} + \|\mathbf{k}_{mult}r_{0,i}\|) \\
&\leq r_{m_2,\pi}(\mathbf{c}_2) + r_{m_1,\pi}(\mathbf{c}_1) \cdot \mathit{\Gamma} + \|\mathbf{k}_{mult}2^{\rho}\| \\
&\leq B + B\mathit{\Gamma} + \|\mathbf{k}_{mult}\|2^{\rho} \,,
\end{aligned}$$

where $m_{j,i}$ is the i-th coordinate of $\mathbf{m}_j$ and $r_{0,i} = [r_0]_p \leq 2^{\rho}$ such that $x_0 = r_0 + q_0 \cdot \pi$ and $r_0 \leftarrow \mathit{\Phi}_{\rho}(p_1, ..., p_l)$. In order to upper bound $\|\mathbf{k}_{mult}\|$ we use a method analogous to the one in Lemma 3.1 as described with Eq.(5) to conclude that $\|\mathbf{k}_{mult}\| \leq \mathit{\Gamma}$. Hence we get that

$$\begin{aligned}
r_{m_{mult},\pi}(\mathbf{c}_{mult}) &\leq B + \mathit{\Gamma}B + \mathit{\Gamma}2^{\rho} \\
&\leq (2\mathit{\Gamma} + 1)B \,,
\end{aligned}$$

which is exactly what we need. Analogously, the NAND operation causes the same increase in the size of the noise. This proves the statement.

3. Let $\mathbf{c}_1, \mathbf{c}_2, B$ and $\mathbf{c}_{add}$ be as in the statement of the lemma. Then we have

$$\begin{aligned}
\|\mathbf{r}_{m_{add},\pi}(\mathbf{c}_{add})\| &= \max_{i=1}^{l}(\|[\mathbf{c}_{add} - m_{add}\mathbf{g}]_{p_i}\| \\
&= \max_{i=1}^{l}(\|[\mathbf{c}_1 - m_1\mathbf{g} + \mathbf{c}_2 - m_2\mathbf{g} + \mathbf{k}_{add}x_0]_{p_i}\|) \\
&\leq \max_{i=1}^{l}(\|[\mathbf{c}_1 - m_1\mathbf{g}]_{p_i}\| + \|[\mathbf{c}_2 - m_2\mathbf{g}]_{p_i}\| + \|[\mathbf{k}_{add}x_0]_{p_i}\|) \\
&\leq \max_{i=1}^{l}(\|\mathbf{r}_{m_1,p_i}(\mathbf{c}_1)\| + \|\mathbf{r}_{m_2,p_i}(\mathbf{c}_2)\| + \|\mathbf{k}_{mult}r_{0,i}\|) \\
&\leq r_{m_1,\pi}(\mathbf{c}_1) + r_{m_2,\pi}(\mathbf{c}_2) + 2^{\rho+1}
\end{aligned}$$

since we have that $\|\mathbf{c}_1 + \mathbf{c}_2\| \leq 2x_0$, thus $\|\mathbf{k}_{add}\| \leq 2$. As before we have that $r_{0,i} = [r_0]_p \leq 2^{\rho}$ such that $x_0 = r_0 + q_0 \cdot \pi$ and $r_0 \leftarrow \mathit{\Phi}_{\rho}(p_1, ..., p_l)$. This proves the statement. □

Generalizing the statement in 2, let us assume that we want to compute a circuit of depth d on ciphertexts whose noise components are bounded by B. The output ciphertext, $\mathbf{c}_d$, an encryption of m_d, will have a noise component with norm $r_{m_d,\pi}(\mathbf{c}_d) \leq (2\varGamma + 1)^d B$. We now prove decryption correctness.

Lemma 4.3 (Correctness Homomorphic Decryption BHE). *For any vector* $\mathbf{c} \in \mathbb{Z}_{x_0}^{\varGamma}$ *encrypting some messages* $m_1, ..., m_l \in \mathbb{Z}_2$ *under our scheme, such that* $m = CRT_{p_1,...,p_l}(m_1, ..., m_l)$*, we have that if* $r_{m,p_i}(\mathbf{c}) < p_i/(4\varGamma)$ *for all* $i = 1, ..., l$ *then* $\mathsf{HE.Dec}(\mathbf{c}) = (m_1, ..., m_l)$*.*

Proof. Assume that for a ciphertext $\mathbf{c} \in \mathbb{Z}_{x_0}^{\varGamma}$ we have $r_{m,p_i}(\mathbf{c}) < p/(4\varGamma)$, then during decryption, we have that

$$\begin{aligned} f_i &= \mathbf{c} \cdot \mathbf{g}^{-1}(p_i/2) \pmod{p_i} \\ &= m\mathbf{g}\mathbf{g}^{-1}(p_i/2) + \mathbf{r}_{m,\pi}(\mathbf{c})\mathbf{g}^{-1}(p_i/2) \pmod{p_i} \\ &\leq m_i(p_i/2) + \varGamma r_{m,p_i}(\mathbf{c}) \end{aligned}$$

where by assumption $\varGamma\|\mathbf{r}_{m,p_i}(\mathbf{c})\| < p_i/4$ for all $1 \leq i \leq l$. So by the decryption algorithm, if $|f_i| \leq m_i(p_i/2) + \varGamma r_{m,p_i}(\mathbf{c}) < p_i/4$ then it necessarily means that $m_i = 0$. Whereas if instead $|f_i| \leq m_i(p_i/2) + \varGamma r_{m,p_i}(\mathbf{c})$ and $|f| > p_i/4$ then it must be the case that $m_i = 1$, or else $\varGamma r_{m,p_i}(\mathbf{c}) > p_i/4$. Hence in any case we get that $\mathsf{BHE.Dec}(sk, \mathbf{c}) = (m_1, ..., m_l)$, which implies decryption correctness. □

4.4 Parameters

Our BHE scheme uses the same parameters as the HE scheme except for

$\varGamma$ is the bit-length of the elements of the public key, the x_i's, we change the symbol for convenience;

l is both the number of messages in the batch and the number of primes in the secret key;

In the batched version of the decomposed scheme, some of the parameters differ since we are now including several primes in the secret key. The following are the constraints on the parameters needed in order for the BHE scheme to be correct and secure against known attacks. Let d be the depth of the circuit used to evaluate the data.

- $\rho = \omega(\lambda)$, to protect against brute-force attacks on the noise [8, 13, 17];
- $\eta = \varOmega(\rho + (d+1)\log\varGamma + \log\tau + \log l)$ in order to have correctness of the evaluation circuit (Lemma 4.2);
- $\varGamma \geq \varOmega(\frac{\lambda}{\log\lambda}(\eta - \rho)^2)$ and $\varGamma > \eta^2$, to thwart different lattice reduction attacks on the AGCD as studied in [10, 12, 19, 23, 26, 27, 31].
- $\tau = \varGamma + \varOmega(\lambda)$, which is derived from the constraints given by the Leftover Hash Lemma in section 2, needed in the security proof in section 4.5.

Remark 4.1. The previous constraints are similar to those of Section 3.3 (with η depending additionally on $\log l$) and comes from the fact that there is no known attack on the $(\rho, \eta, \varGamma)$-l-AGCD (Def. 2.5) that exploits the CRT structure [19, Sec. 2.1]: the best known attack is to attack the AGCD on a single prime $p \in \{p_1, \dots, p_l\}$. Thus, Γ has to be set larger than η^2. Informally, this shows that for the same parameters (Γ, η) as in Sec. 3, one can encrypt close to $l = \eta$ bits without increasing the ciphertext size while still maintening correctness. (Note that the public key contains l additional ciphertexts compared to the scheme of Sec. 3.)

Remark 4.2. Note that we necessarily have that $\Gamma \geq l \cdot \eta$. When $\Gamma = l \cdot \eta$ (and thus $l > \eta$ by the previous remark), the l-AGCD distribution becomes

$$\mathcal{X}_{\rho,\Gamma} = \Phi_\rho = \{r = CRT_{p_1,\dots,p_l}(r_1, ..., r_l) | r_i \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\},$$

i.e., a multiple of π is no longer added to the samples of Φ_ρ. This distribution is at the core of the multilinear maps of [14]. As far as we known, there is no known attack against this distribution that exploits the additional structure (attacks on [14] comes from the zeroizing parameter therein).

4.5 Security

In this section we would like to prove, similar to the non-batched version of the scheme, the semantic security. Unfortunately, the assumption in Definition 2.5 is not enough to ensure the security of the batched version of the scheme as it does not assure security when an encryption of key dependent messages is published, needed to compute the CRT representation of a batch of messages during the encryption procedure.

One way to go is to assume that even in spite of the new elements in the public key, the vector $\mathbf{x}$ is still indistinguishable from uniform. This would allow us to apply the same proof strategy as in the previous section. However, to increase our confidence in the validity of this assumption, we show that one can view it as assuming *circular security* for a different auxiliary scheme, one that we can actually prove secure under Definition 2.5. Furthermore, since our auxiliary scheme is only used in the proof of security, we do not even require that it is properly decryptable, only that it is CPA secure (we proved correctness for our actual scheme).

In what follows we introduce our auxiliary encoding scheme, AHE, which encrypts large messages, instead of the CRT representation of a batch of bits. We show that this scheme can be extended to the BHE. We prove that AHE is secure under the decisional batch AGCD assumption (Definition 2.5) and finally show that by adding the circular security assumption to AHE, we can make the BHE scheme secure.

Auxiliary HE Scheme As explained above, we only require key generation and encryption for this scheme.

AHE.Keygen(1^λ)*:* We first generate the parameters $params = \{\Gamma, \rho, \eta, \tau, l, k\}$ according to the security parameter λ and correctness. Then we sample l η-bit primes $p_1, ..., p_l$ and we define $\pi = \prod_{i=1}^{l} p_i$, as above. After this, using the distribution in Definition 2.4 and rejection sampling, we first sample an integer $x_0 \leftarrow (\mathcal{X}_{\rho,\Gamma}(p_1, ..., p_l))^{(\geq 2^{\Gamma-1})}$ and then τ integers $\{x_i\}_{1 \leq i \leq \tau} \leftarrow (\mathcal{X}_{\rho,\Gamma}(p_1, ..., p_l))^{(\leq x_0)}$, such that $(x_0, x_1, \dots, x_\tau) \leftarrow \vec{\mathcal{X}}_\tau$, as in Lemma 2.3. We write $\mathbf{x} = (x_1, ..., x_\tau)$, $\mathbf{r} = (r_1, ..., r_\tau)$ and $\mathbf{q} = (q_1, ..., q_\tau)$. We let the message space be $\mathbb{Z}_k$ for some $k \leq \pi$, the public key $pk = (params, x_0, \mathbf{x})$ and the secret key $sk = (p_1, ..., p_l)$.

$\mathsf{AHE.Enc}(pk, m)$*:* For a message $m \in \mathbb{Z}_k$, we sample a matrix $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \Gamma}$ and we compute

$$\mathbf{c} = [m\mathbf{g} + \mathbf{xS}]_{x_0} \tag{10}$$

where again, $\mathbf{c}$ is a vector of dimension $\Gamma = \lceil \log x_0 \rceil$.

Remark 4.3. Given a message $m \in \mathbb{Z}_\pi$, the ciphertext generated by the AHE.Enc is decryptable in a similar way to BHE, where the secret key is the prime factorization of π. This only works in the case that $m = CRT_{p_1,\ldots,p_l}(m_1, ..., m_l)$.

Security of AHE We start by proving a lemma that will be useful in proving both of the next two theorems. Simply said we prove that the structure $\mathbf{xS} \mod x_0$, where $\mathbf{x}$ are AGCD samples, is computationally indistinguishable from uniform. This will help us avoid redundancy in the proofs of security of AHE and BHE since this structure is present in both of the encryption algorithms.

Lemma 4.4. *Let* $(x_0, x_1, \ldots, x_\tau) \leftarrow \vec{\mathcal{X}}_\tau$ *and* $(u_0, u_1, \ldots, u_\tau) \leftarrow \vec{\mathcal{U}}_\tau$ *as in Lemma 2.3, where* $\mathbf{x} = (x_1, ..., x_\tau)$ *and* $\mathbf{u} = (u_1, ..., u_\tau)$*. Then for* $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^{\Gamma}$ *and* $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \Gamma}$*, the distribution* $(\mathbf{x}, x_0, [\mathbf{xS}]_{x_0})$ *is computationally indistinguishable from the distribution* $(\mathbf{u}, u_0, \mathbf{v})$*.*

Proof. For convenience we write $pk = (\mathbf{x}, x_0)$ and $pk' = (\mathbf{u}, u_0)$. On the one hand, we know by Lemma 2.3 that pk and pk' are indistinguishable, so it follows that $(pk, [\mathbf{xS}]_{x_0})$ is indistinguishable from $(pk', [\mathbf{uS}]_{u_0})$ where $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \Gamma}$. On the other hand, we have by the LHL from Lemma 2.2 that the statistical distance between $(pk', [\mathbf{uS}]_{u_0})$ and $(pk', \mathbf{v})$ for $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^{\Gamma}$ is upper bounded by $\frac{1}{2}\sqrt{u_0/2^\tau} = negl(\lambda)$. Hence by transitivity, $(pk, [\mathbf{xS}]_{x_0})$ is indistinguishable from $(pk', \mathbf{v})$ for $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^{\Gamma}$.

In other words, for every polynomial time algorithm $\mathcal{A}$, we have that

$$\left|\Pr[\mathcal{A}(1^\lambda, \mathbf{x}, x_0, [\mathbf{xS}]_{x_0}) = 1] - \Pr[\mathcal{A}(1^\lambda, \mathbf{u}, u_0, \mathbf{v}) = 1]\right| \leq negl(\lambda)\,.$$

□

Theorem 4.2. *The above* AHE *scheme is CPA secure under the* $(\rho, \eta, \Gamma) - l$-decisional *AGCD assumption.*

Proof. For $pk = (params, \mathbf{x}, x_0) \leftarrow \mathsf{AHE.Keygen}(1^\lambda)$ and $\mathbf{c} = \mathsf{AHE.Enc}(pk, b)$, where $b \in \mathbb{Z}_k$ is chosen by the adversary, $pk' = (params, \mathbf{u}, u_0)$, as in Lemma 4.4, we prove that $(pk, \mathbf{c})$ and $(pk', \mathbf{v})$, where $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^{\Gamma}$, are computationally indistinguishable.

In order to do this we will use a two step hybrid argument and use Lemma 4.4 to show indistinguishability between the hybrids. Each hybrid is an interactive exchange between a challenger, $\mathcal{C}$, and a polynomial time adversary, $\mathcal{A}$.

Hybrid 0: The adversary, $\mathcal{A}$, gets $pk_{H_0} = pk$ from the challenger and then chooses a message $b \in \mathbb{Z}_k$, which he sends back to the challenger. In turn, $\mathcal{C}$ computes $\mathbf{c} = \mathsf{AHE.Enc}(pk, b) = [b\mathbf{g} + \mathbf{xS}]_{x_0}$, which is then sent to the adversary. In this case, $(pk, \mathbf{c})$ is distributed exactly as in the AHE scheme.

Hybrid 1: The adversary, $\mathcal{A}$, gets $pk_{H_1} = pk'$ from the challenger and then chooses a message $b \in \mathbb{Z}_k$, which he sends back to the challenger. In turn, $\mathcal{C}$ computes $\mathbf{c} = [b\mathbf{g} + \mathbf{v}]_{u_0}$, which is then sent to the adversary. Now, by Lemma 4.4 we know that $(pk, \mathbf{c})_{H_0}$ and $(pk, \mathbf{c})_{H_1}$ are computationally indistinguishable because $(pk, [\mathbf{xS}]_{x_0})$ is indistinguishable from $(pk', [\mathbf{uS}]_{u_0})$. Finally, we know that $\mathbf{c} \equiv \mathbf{v}$, hence the probability of success for $\mathcal{A}$ in this hybrid is exactly $1/2$ since $\mathbf{v}$ is completely random.

In other words, for every polynomial time algorithm $\mathcal{A}$,

$$\left|\Pr[\mathcal{A}(1^\lambda, pk, \mathbf{c}) = 1] - \Pr[\mathcal{A}(1^\lambda, pk', \mathbf{v}) = 1]\right| = negl(\lambda)\,.$$

□

Security of BHE As mentioned earlier, in order to prove security of BHE we must prove first that it is an extension of the AHE scheme under specific conditions, mainly we have the following definition.

Definition 4.2. *Let $\mathcal{E}(\mathcal{M}_\mathcal{E}, \mathcal{C}_\mathcal{E})$ be a public encoding scheme with message space $\mathcal{M}_\mathcal{E}$ and ciphertext space $\mathcal{C}_\mathcal{E}$. We say that $\mathcal{E}'(\mathcal{M}_{\mathcal{E}'}, \mathcal{C}_{\mathcal{E}'})$ is an* extension by ciphertext *of $\mathcal{E}$ if for some integer n there exists $\mathbf{c}_1, ..., \mathbf{c}_n$ and function $f : \mathcal{C}_\mathcal{E}^n \times \mathcal{M}_{\mathcal{E}'} \leftarrow \mathcal{C}_{\mathcal{E}'}$ such that for all $m \in \mathcal{M}_{\mathcal{E}'}$, $\mathcal{E}'.\mathsf{Enc}(pk_{\mathcal{E}'}, m) = f(c_1, ..., c_n, m)$. Furthermore, $\mathcal{E}'$ is a public encryption scheme if decryption is correct.*

Lemma 4.5. *The* BHE *public encryption scheme is an extension by ciphertext of the* AHE *scheme.*

Proof. Let $(pk, sk) \leftarrow \mathsf{AHE}$ and let $f : (\mathbb{Z}_{x_0}^\Gamma)^l \times \{0,1\}^l \leftarrow \mathbb{Z}_{x_0}^\Gamma$ be the function

$$f(\mathbf{c}_1, ..., \mathbf{c}_l, \mathbf{m}) = [\mathbf{mC} + \mathbf{xS}]_{x_0}$$

where $\mathbf{S} \leftarrow \{0,1\}^{\tau \times \Gamma}$ and $\mathbf{C} = (\mathbf{c}_1 ... \mathbf{c}_l)$. Now in order to obtain exactly the BHE scheme we must specify the ciphertexts used, which in this case are $\mathbf{c}_i = \mathsf{AHE.Enc}(pk, y_i)$ where $y_i = CRT_{p_1,...,p_l}(0, ..., 0, 1, 0, ..., 0) = \pi_i(\pi_i^{-1} \mod p_i)$. This can be done as long as the AHE encryptions are done privately in the key generation process and we let $pk_{\mathsf{BHE}} = (pk, \mathbf{C})$ for $\mathbf{C} = (\mathbf{c}_1 ... \mathbf{c}_l)$ and we let $sk_{\mathsf{BHE}} = sk$. Correctness of BHE follows by the previous section. □

We now connect the circular security of AHE with the CPA security of BHE secure. We start by defining the flavor of circular security we require. Note that this definition applies to encoding schemes (that do not have decryption) and not just to encryption schemes.

Definition 4.3. *A public key encoding scheme* $(\mathsf{Keygen}, \mathsf{Enc})$ *is weakly circular secure, if for any polynomial sequence of functions f_i from the secret key space to the message space, it holds that* $(\mathsf{pk}, \{\mathsf{Enc}_\mathsf{pk}(f_i(\mathsf{sk}))\}_i)$ *is computationally indistinguishable from* $(\mathsf{pk}, \{\mathsf{Enc}_\mathsf{pk}(0)\}_i)$.

The security of BHE follows by applying the circularity of AHE to account for the key dependent information in $\mathbf{Y}$ and then applying the standard security argument. Hence we show that

Theorem 4.3. *If* AHE *is circular secure and the* $(\rho, \eta, \Gamma)-l-AGCD$ *assumption holds, then* BHE *is CPA secure.*

Proof. Let us assume that AHE is circular secure, where we write $\mathcal{S}$ for the secret key space and $\mathcal{M}$ for the message space. We then let the sequence of functions, $f_i : \mathcal{S} \to \mathcal{M}$, be the following

$$f_i(p_1, ..., p_l) = CRT_{p_1,...,p_l}(0, ..., 1, ..., 0)\,.$$

We prove that for all polynomial time adversaries $\mathcal{B}$, the probability of distinguishing in the BHE scheme between an encryption of any message in $\{0,1\}^l$, $\mathbf{c}_1$, and a uniform vector is $negl(\lambda)$.

We proceed to prove this by using a hybrid argument. In each of the hybrids, some form of a CPA game is played between a challenger $\mathcal{A}$ and an adversary $\mathcal{B}$. Let $(pk, sk) \leftarrow \mathsf{AHE.Keygen}(1^\lambda)$ and let $\mathbf{c}_b$ be the BHE ciphertext that $\mathcal{B}$ receives during the game after choosing the message $\mathbf{m}_b \in \{0,1\}^l$. We have that $pk = (params, \mathbf{x}, x_0)$ where $(\mathbf{x}, x_0) = (x_0, ..., x_\tau) \leftarrow \vec{\mathcal{X}_\tau}$ and we let $(\mathbf{u}, u_0) = (u_0, u_1, \ldots, u_\tau) \leftarrow \vec{\mathcal{U}_\tau}$ as in Lemma 2.3.

Hybrid 0: In this Hybrid, the challenger $\mathcal{A}$ generates AHE encryptions of functions of the secret key, $\mathbf{c}_i = \mathsf{AHE.Enc}(pk, m_i)$, where $m_i = f_i(p_1, ..., p_l)$. Then $\mathcal{A}$ generates the BHE public key $pk' = (pk, \mathbf{Y})$, where $\mathbf{Y} = (\mathbf{c}_1 ... \mathbf{c}_l)$ and sends pk' to $\mathcal{B}$. By Lemma 4.5, we have that pk' is distributed exactly as the public key in $\mathsf{BHE.Keygen}$. The BHE-CPA game is then played and $\mathcal{B}$ has some probability of success.

Hybrid 1: In this Hybrid, the challenger $\mathcal{A}$ generates AHE encryptions of zero, $\mathbf{c}_i = \mathsf{AHE.Enc}(pk, m_i)$, where $m_i = 0$. Then $\mathcal{A}$ generates the public key $pk' = (pk, \mathbf{Y})$, where $\mathbf{Y} = (\mathbf{c}_1 ... \mathbf{c}_l)$ and sends pk' to $\mathcal{B}$. The BHE-CPA game is then played and we claim that the probability of success for $\mathcal{B}$ is negligibly close to the probability of success for $\mathcal{B}$ in Hybrid 0. Otherwise it would imply that $\mathcal{B}$ can distinguish the AHE encryptions of key dependent messages from encryptions of zero, contradicting the circular security assumption. Thus we have that

$$\left| \Pr_{H_0}[\mathcal{B}(1^\lambda, pk', \mathbf{c}_b) = 1] - \Pr_{H_1}[\mathcal{B}(1^\lambda, pk', \mathbf{c}_b) = 1] \right| = negl(\lambda)\,.$$

Hybrid 2: In this Hybrid, the challenger $\mathcal{A}$ generates AHE encryptions of zero, $\mathbf{c}_i = \mathsf{AHE.Enc}(pk, m_i)$, where $m_i = 0$. Then $\mathcal{A}$ generates the public key $pk' = (params, \mathbf{u}, u_0, \mathbf{Y})$, where $\mathbf{Y} = (\mathbf{c}_1 ... \mathbf{c}_l)$ and sends pk' to $\mathcal{B}$. Here $\mathcal{A}$ does not send $\mathcal{B}$ some encryption of a message or of zero, like in the CPA game, instead it sends the vector $\mathbf{c}_b = [\mathbf{m}_b \mathbf{Y} + \mathbf{v}]_{u_0}$ where $\mathbf{v} \leftarrow \mathbb{Z}_{u_0}^{\Gamma}$. By Lemma 4.4 we know that $(pk', \mathbf{c}_b)_{H_1}$ is indistinguishable from $(pk', \mathbf{c}_b)_{H_2}$, which implies that $\mathcal{B}$ has probability of success that is negligibly close to that of Hybrid 1, hence

$$\left| \Pr_{H_1}[\mathcal{B}(1^\lambda, pk', \mathbf{c}_b) = 1] - \Pr_{H_2}[\mathcal{B}(1^\lambda, pk', \mathbf{c}_b) = 1] \right| = negl(\lambda)\,.$$

Furthermore, we know that $\mathbf{c}_b \equiv \mathbf{v}$ and since $\mathbf{v}$ is completely random, the probability of success of $\mathcal{B}$ is exactly 1/2. Thus, by transitivity, we have that

$$\left| \Pr_{H_0}[\mathcal{B}(1^\lambda, pk', \mathbf{c}_b) = 1] - \Pr_{H_2}[\mathcal{B}(1^\lambda, pk', \mathbf{v}) = 1] \right| = negl(\lambda)\,.$$

Hence we have showed that if the AHE scheme is circular secure, then there does not exist an adversary that has non-negligible advantage in a CPA game in the BHE scheme. □

5 Towards Practicality

In this section, we suggest some optimizations to improve the asymptotic and concrete parameters of our schemes, and discuss the obstacles towards efficient implementation thereof. Overcoming these limitations and obtaining a scheme as efficient as the lattice-based variants [11, 18, 24] remains a challenging open problem. In this section, for simplicity we focus on the scheme of Sec. 3 (our optimizations are easily generalizable to the batch variant of Sec. 4).

5.1 Reducing the Public-Key Size

To satisfy the constraints on the parameters of Sec. 3 for a depth-d circuit, we can take

$$\rho = 2\lambda, \eta = \tilde{\mathcal{O}}(\lambda + d), \gamma = \tilde{\mathcal{O}}(\lambda^2 + d^2) \text{ and } \tau = \tilde{\mathcal{O}}(\lambda^2 + d^2).$$

This gives a public key of size $\tilde{\mathcal{O}}(\lambda^6 + d^6)$. To reduce the size of the public key, we can use the technique suggested in [16] to use a subset-sum with words rather than bits. In particular, to encrypt a message $m \in \{0, 1\}$ as in Eq. (3.1), we sample a random matrix $\mathbf{S} \in [0, \beta)^{\tau\times\gamma}$ instead of a binary matrix. We have the following corollary of Lemma 2.1.

Corollary 5.1. *Let $\beta \geq 2$. Set $\mathbf{x} = (x_1, \ldots, x_m) \leftarrow \mathbb{Z}_M^m$ uniformly and independently, set $\mathbf{S} \leftarrow [0, \beta)^{m\times n}$ for some n; and let $\mathbf{y} = \mathbf{x} \cdot \mathbf{S} \pmod M$. Then $(\mathbf{x}, \mathbf{y})$ is $1/2\sqrt{M/2^{\log_2 \beta \cdot m}}$-uniform over $\mathbb{Z}_M^{m+n}$.*

Proof. Let us consider the hash function family $\mathcal{H}$ from $[0, 2^\beta)^m$ to $\mathbb{Z}_M^n$. Each member $h \in \mathcal{H}$ is parametrized by the element $(x_1, \ldots, x_m) \in \mathbb{Z}_M^m$. Given $\mathbf{S} \in [0, \beta)^{m\times n}$, we define $h(\mathbf{S}) = \mathbf{x} \cdot \mathbf{S}$. The family $\mathcal{H}$ is a 2-universal family of hash functions, and by Lemma 2.1 we get the desired result. □

In the proof of security (**case 2**) instead of concluding that the statistical distance between $(\mathbf{x}, \mathbf{xS})$ and $(\mathbf{x}, \mathbf{u})$ where $\mathbf{u} \leftarrow \mathcal{U}([0, 2^\gamma)) \cap \mathbb{Z}$ is bounded by $\frac{1}{2}\sqrt{x_0/2^\tau}$, we get that it is bounded by $\frac{1}{2}\sqrt{x_0/2^{\log_2 \beta \cdot \tau}}$. Also in Lemma 3.1, the noise of a fresh encryption $\mathbf{c}$ of m now has norm

$$r_{m,p}(\mathbf{c}) \leq \tau\beta \cdot 2^{\rho+1} \, .$$

Eventually, this gives the following new parameter constraints:

$$\log \beta \cdot \tau \geq \gamma + \mathcal{O}(\lambda), \qquad \eta - \rho > (d + 1) \log \gamma + \log \tau + \log \beta + \mathcal{O}(1) \, ,$$

and by taking $\log \beta = \tilde{\mathcal{O}}(\lambda + d)$, we reduce the public key size to $\tilde{\mathcal{O}}(\lambda^5 + d^5)$.

5.2 Evaluating Partial Gates

Let us recall that the decryption procedure first computes

$$f = \mathbf{c} \cdot \mathbf{g}^{-1}(p/2) \pmod{p}\,,$$

and outputs $m = 1$ if $|f| \geq p/4$ and $m = 0$ otherwise. Now, since $p/2 \leq 2^{\eta-1}$, we have that

$$\mathbf{g}^{-1}(p/2) = (P_0, P_1, \ldots, P_{\eta-1}, 0, \ldots, 0),$$

where $P_0, \ldots, P_{\eta-1} \in \{0, 1\}$. In particular, this shows that only the first η coefficients of $\mathbf{c}$ are useful during the decryption procedure (the other coefficients are required for correctness when homomorphically processing the MULT and NAND gates; see Section 3.2). Therefore, when evaluating a circuit, one can only compute the η first coefficients of the outputs of the last MULT and NAND gates, and all the subsequent homomorphic additions; this reduces the computation cost by a multiplicative factor $\approx \gamma/\eta$.

5.3 Trade-off on the Multiplication Complexity and the Ciphertext Size

Recall that the homomorphic multiplication (Eq. (4)) of two ciphertexts $\mathbf{c}_1$ and $\mathbf{c}_2$ is given by:

$$\mathbf{c}_{mult} = \mathbf{c}_1 \cdot \mathbf{G}^{-1}(\mathbf{c}_2) \bmod x_0\,.$$

In particular, computing $\mathbf{c}_{mult}$ requires to compute γ times (for each coefficient) about $\gamma/2$ modular additions of γ-bit numbers, i.e. a computational complexity of $\mathcal{O}(\gamma^2 \log(\gamma))$.

Now, assume that instead of using the gadget $\mathbf{g} = (1, 2, \ldots, 2^\gamma)$, we use the gadget

$$\mathbf{g}_\omega = (1, \omega, \ldots, \omega^{\gamma'_\omega})\,,$$

where $\omega \geq 2$ and $\gamma'_\omega = \lceil \gamma' / \log_2 \omega \rceil$ assuming that we now work with γ'-bit integers; i.e. we perform a word decomposition instead of a bit decomposition (taking $\gamma' \geq \gamma$ to get the same homomorphic functionality). Then, computing $\mathbf{c}_{mult}$ requires to compute γ'_ω times (for each coefficient) γ'_ω modular multiplications between an element of $\mathbb{Z}_{\gamma'}$ and of $\mathbb{Z}_\omega$; i.e. an approximate complexity of $\mathcal{O}(\gamma'^2_\omega \log(\gamma') \log(\omega))$ (via a schoolbook multiplication).

Now, if one works with $\omega \geq 2$, the noise bounds have to be revisited. In particular, for any two ciphertexts $\mathbf{c}_1, \mathbf{c}_2$, where $B = \max\{r_{m_1,p}(\mathbf{c}_1), r_{m_2,p}(\mathbf{c}_2)\}$, then

$$r_{m_{mult},p}(\mathbf{c}_{mult}) = r_{m_{nand},p}(\mathbf{c}_{nand}) \leq (2\gamma'_\omega \omega + 1)B\,,$$

and the decryption condition of a ciphertext $\mathbf{c}$ of a message m becomes

$$r_{m,p}(\mathbf{c}) < p/(4\gamma'_\omega \omega)\,.$$

We have the following new parameter constraint:

$$\eta - \rho > (d+1)(\log \gamma'_\omega + \log \omega) + \log \tau + \log \beta + \mathcal{O}(1).$$

By taking $\log \omega = \tilde{\mathcal{O}}(\lambda)$, we obtain the same asymptotic complexities a before, but concrete complexities will differ. We will see in Sec. 5.4 how this trade-off (increasing ω also increases γ') impacts concrete parameters.

5.4 Limitations

In order to choose concrete parameters, we use the analyses of the concrete attacks against the AGCD problem from [8, 10, 16, 17, 19, 31]. In particular, we deduce from [8, 17] that ρ should be conservatively set to 2λ, and from [10, 19] that the best lattice attacks require to work in dimension $t \geq (\gamma - \rho)/(\eta - \rho)$.

In [10], the AGCD based scheme is also a leveled homomorphic encryption scheme and its parameters can be set significantly smaller than previous works [9, 15–17]: indeed, $\eta - \rho$ can be selected to be small, and so can γ as long as (say) $800 \geq (\gamma - \rho)/(\eta - \rho)$. In the regular scheme of Sec. 3, we also have that $\eta - \rho$ is small, and hence that γ can be small. Unfortunately, our ciphertexts are γ time larger than the ciphertexts in [9], and the complexity of the homomorphic multiplication is at least quadratic in γ. In practice, for (say) $\lambda = 80, \rho = 160$ and $\eta = 172$ (which would only allow for one homomorphic multiplication), then $\gamma \approx 12000$, and the homomorphic multiplication would consist of about 10^9 modular additions, which takes several seconds on a modern CPU. Using the optimization of Section 5.3, one can reduce the homomorphic multiplication complexity at first; e.g. when $\omega = 2^{32}$ and $\gamma \approx 75000$, performing an homomorphic multiplication costs now $0.43 \cdot 10^9$ schoolbook modular multiplications between 32-bit words and 75000-bit integers (which still takes several seconds on a modern CPU). Unfortunately, as the gap between η and ρ widens, γ has to be significantly increased so that the AGCD problem remains hard. As ω increases, the number of unit operations increases again, and the unit operations becomes more and more costly (namely, modular multiplication between $\log \omega$-bit integers and γ-bit integers). It follows from our experiments that the size of the ciphertext is a bottleneck to make our schemes practical; a back of the hand computation shows that they are about two order of magnitude slower than their competitors [11, 24]. We leave as a challenging open problem to improve the efficiency of this scheme. Note however that our batch variant can encrypt up to γ/η plaintexts in parallel for roughly the same computational cost (cf. Remark 4.1), which decreases the computational cost per bit of plaintext.

References

1. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In *CRYPTO (1)*, volume 8616 of *LNCS*, pages 297–314. Springer, 2014.
2. Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *CRYPTO*, volume 7417 of *LNCS*, pages 868–886. Springer, 2012.
3. Zvika Brakerski, Craig Gentry, and Shai Halevi. Packed ciphertexts in lwe-based homomorphic encryption. In *Public Key Cryptography*, volume 7778 of *LNCS*, pages 1–13. Springer, 2013.
4. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, pages 309–325. ACM, 2012.
5. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS*, pages 97–106. IEEE Computer Society, 2011. Full version in `https://eprint.iacr.org/2011/344.pdf`.
6. Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, volume 6841 of *LNCS*, pages 505–524. Springer, 2011.
7. Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based FHE as secure as PKE. In *ITCS*, pages 1–12. ACM, 2014.
8. Yuanmi Chen and Phong Q. Nguyen. Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers. In *EUROCRYPT*, volume 7237 of *LNCS*, pages 502–519. Springer, 2012.

9. Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In *EUROCRYPT*, volume 7881 of *LNCS*, pages 315–335. Springer, 2013.
10. Jung Hee Cheon and Damien Stehlé. Fully homomophic encryption over the integers revisited. In *EUROCRYPT (1)*, volume 9056 of *LNCS*, pages 513–536. Springer, 2015.
11. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *ASIACRYPT (1)*, volume 10031 of *Lecture Notes in Computer Science*, pages 3–33, 2016.
12. Henry Cohn and Nadia Heninger. Approximate common divisors via lattices. *The Open Book Series*, 1(1):271–293, 2013.
13. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Batch fully homomorphic encryption over the integers. *IACR Cryptology ePrint Archive*, 2013:36, 2013.
14. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *CRYPTO (1)*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493. Springer, 2013.
15. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Scale-invariant fully homomorphic encryption over the integers. In *Public Key Cryptography*, volume 8383 of *LNCS*, pages 311–328. Springer, 2014.
16. Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Fully homomorphic encryption over the integers with shorter public keys. In *CRYPTO*, volume 6841 of *LNCS*, pages 487–504. Springer, 2011.
17. Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In *EUROCRYPT*, volume 7237 of *LNCS*, pages 446–464. Springer, 2012.
18. Léo Ducas and Daniele Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In *EUROCRYPT (1)*, volume 9056 of *LNCS*, pages 617–640. Springer, 2015.
19. Steven D. Galbraith, Shishay W. Gebregiyorgis, and Sean Murphy. Algorithms for the approximate common divisor problem. *IACR Cryptology ePrint Archive*, 2016:215, 2016.
20. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178. ACM, 2009.
21. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO (1)*, volume 8042 of *LNCS*, pages 75–92. Springer, 2013.
22. Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
23. Nick Howgrave-Graham. Approximate integer common divisors. In *CaLC*, volume 2146 of *LNCS*, pages 51–66. Springer, 2001.
24. Alhassan Khedr, P. Glenn Gulak, and Vinod Vaikuntanathan. SHIELD: scalable homomorphic implementation of encrypted data-classifiers. *IEEE Trans. Computers*, 65(9):2848–2858, 2016.
25. Jinsu Kim, Moon Sung Lee, Aaram Yun, and Jung Hee Cheon. CRT-based fully homomorphic encryption over the integers. *IACR Cryptology ePrint Archive*, 2013:57, 2013.
26. Jeffrey C Lagarias. The computational complexity of simultaneous diophantine approximation problems. *SIAM Journal on Computing*, 14(1):196–209, 1985.
27. Phong Q. Nguyen and Jacques Stern. The two faces of lattices in cryptology. In *CaLC*, volume 2146 of *LNCS*, pages 146–180. Springer, 2001.
28. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–177. Academic Press, 1978.
29. Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *FOCS*, pages 124–134. IEEE Computer Society, 1994.
30. Nigel P. Smart and Frederik Vercauteren. Fully homomorphic SIMD operations. *Des. Codes Cryptography*, 71(1):57–81, 2014.
31. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, volume 6110 of *LNCS*, pages 24–43. Springer, 2010.