New and Improved Key-Homomorphic Pseudorandom Functions

Abhishek Banerjee[1⋆] and Chris Peikert[1⋆⋆]

School of Computer Science, Georgia Institute of Technology

Abstract. A *key-homomorphic* pseudorandom function (PRF) family $\{F_s\colon D \to R\}$ allows one to efficiently compute the value $F_{s+t}(x)$ given $F_s(x)$ and $F_t(x)$. Such functions have many applications, such as distributing the operation of a key-distribution center and updatable symmetric encryption. The only known construction of key-homomorphic PRFs without random oracles, due to Boneh *et al.* (CRYPTO 2013), is based on the learning with errors (LWE) problem and hence on worst-case lattice problems. However, the security proof relies on a very strong LWE assumption (i.e., very large approximation factors), and hence has quite inefficient parameter sizes and runtimes.

In this work we give new constructions of key-homomorphic PRFs that are based on much weaker LWE assumptions, are much more efficient in time and space, and are still highly parallel. More specifically, we improve the LWE approximation factor from exponential in the input length to exponential in its *logarithm* (or less). For input length λ and 2^λ security against known lattice algorithms, we improve the key size from λ^3 to λ bits, the public parameters from λ^6 to λ^2 bits, and the runtime from λ^7 to $\lambda^{\omega+1}$ bit operations (ignoring polylogarithmic factors in λ), where $\omega \in [2, 2.373]$ is the exponent of matrix multiplication. In addition, we give even more efficient ring-LWE-based constructions whose key sizes, public parameters, and *incremental* runtimes on consecutive inputs are all *quasi-linear* $\tilde{O}(\lambda)$, which is optimal up to polylogarithmic factors. To our knowledge, these are the first *low-depth* PRFs (whether key homomorphic or not) enjoying any of these efficiency measures together with nontrivial proofs of 2^λ security under any conventional assumption.

1 Introduction

A *pseudorandom function (PRF)* family [GGM84] $\mathcal{F} = \{F_s\colon D \to R\}$ is a finite set of (deterministic) functions with common domain D and range R (both

⋆ Research supported by the second author's grants.

⋆⋆ This material is based upon work supported by the National Science Foundation under CAREER Award CCF-1054495, by the US-Israel Binational Science Foundation Grant 2010296, by the Alfred P. Sloan Foundation, and by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under Contract No. FA8750-11-C-0098. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the National Science Foundation, the BSF, the Sloan Foundation, DARPA or the U.S. Government.

finite), for which a randomly chosen $F_s \leftarrow \mathcal{F}$ cannot be efficiently distinguished from a uniformly random function $U\colon D \to R$, given adaptive oracle access. The index s of function F_s is often called its *(secret) key* or *seed*. The family $\mathcal{F}$ is *key homomorphic* if the set of keys has a group structure and if there is an efficient algorithm that, given $F_s(x)$ and $F_t(x)$ (but not s or t), outputs $F_{s+t}(x)$.

Naor, Pinkas, and Reingold [NPR99] constructed, in the random oracle model, a very simple key-homomorphic PRF family based on the decisional Diffie-Hellman problem, and gave applications like distributing the operation of a Key Distribution Center. Recently, Boneh *et al.* [BLMR13] constructed the first key-homomorphic PRFs *without* random oracles, and described many more applications (all of which are very efficient in their use of the PRF), including symmetric-key proxy re-encryption, updatable encryption, and PRFs secure against related-key attacks (cf. [BC10, LMR14]). The construction of Boneh *et al.* is based on the (appropriately parameterized) *learning with errors* (LWE) problem [Reg05], and builds upon ideas used in the *non*-key-homomorphic LWE-based PRFs of Banerjee, Peikert, and Rosen [BPR12].

One drawback of the construction and proof from [BLMR13] is its rather strong LWE assumption and, by consequence, large parameters and runtimes. For example, to obtain a PRF of input length λ with exponential 2^λ provable security against known lattice attacks, the secret keys and public parameters respectively need to be at least λ^3 and λ^6 bits, and the runtime to evaluate the function is at least λ^7 bit operations (to produce λ^2 output bits), not counting some polylogarithmic $\log^{O(1)} \lambda$ factors. It is worth noting that among the several LWE-based PRFs given in [BPR12], the most highly parallelizable "direct" construction (which can be implemented in $TC^0 \subseteq NC^1$) relies on roughly the same strong assumptions and so has similarly low efficiency as the one from [BLMR13]. However, the synthesizer-based construction (in $TC^1 \subseteq NC^2$) and sequential GGM-based one from [BPR12] can be proved secure under much weaker LWE assumptions, and hence can have much better parameters and runtimes. A natural question, therefore, is whether there exist *key-homomorphic* PRFs with similar security and efficiency characteristics.

Our results. In this work we answer the above question in the affirmative, by giving new constructions of key-homomorphic PRFs that have substantially better efficiency, and still enjoy very high parallelism. As compared with [BLMR13], we improve the key size from λ^3 to λ bits, the public parameters from λ^6 to λ^2 bits, and the runtime from λ^7 to $\lambda^{\omega+1}$ bit operations (always omitting $\log^{O(1)} \lambda$ factors), where $\omega \in [2, 2.373]$ is the exponent of matrix multiplication. Functions having these parameters can be implemented in $TC^1 \subseteq NC^2$, though seemingly not in TC^0 or NC^1.

We also give even more efficient key-homomorphic PRFs based on the *ring*-LWE problem [LPR10, LPR13]. Compared with the ring-based analogue of [BLMR13], and again ignoring $\log^{O(1)} \lambda$ factors, here our keys and public parameters are only λ bits (improving upon λ^3 and λ^4, respectively), and the runtime is only λ^2 bit operations to produce λ output bits (from λ^5 to produce λ^2). In addition, the *incremental* computation of our PRF on successive inputs (e.g.,

in a counter-like mode) has runtime only λ. See Figure 1 for a full comparison with [BPR12, BLMR13].

To our knowledge, ours are the first *low-depth* PRFs (whether key homomorphic or not) having nontrivial proofs of exponential 2^λ security under any conventional assumption along with *quasi-optimal* $\tilde{O}(\lambda)$ key sizes or incremental runtimes, or quasilinear $\tilde{O}(\lambda)$ nonincremental runtime per output bit. For example, the GGM construction [GGM84] can have small keys and quasilinear nonincremental runtime per output bit (using a quasi-optimal PRG), but it is highly sequential. The Naor-Reingold constructions [NR95, NR97], which are highly parallel, have at least quadratic λ^2 key sizes and runtime per output bit, even assuming exponential security of the underlying hard problems. And factoring-based constructions [NRR00] fare much worse due to subexponential-time factoring algorithms.

In their parallelism and underlying LWE assumptions, our functions are qualitatively very similar to the synthesizer- and GGM-based ones from [BPR12] (see Figure 1); however, the constructions and proofs are completely different. Instead, our construction can be seen as a substantial generalization of the one of Boneh *et al.* [BLMR13], in that theirs is an instantiation of ours with a linear-depth "left spine" tree. By contrast, our construction can be securely instantiated with *any* binary tree, thanks to a new proof technique that may be of use elsewhere. The shape of the tree determines the final parameters and parallelism of the resulting function: roughly speaking, its "left depth" determines the strength of the LWE assumption in the proof, while its "right depth" determines its parallelism. Interestingly, a complete binary tree turns out to be *very far from optimal* for the parameters we care about. Optimal trees can be found efficiently using dynamic programming, and provide input lengths that are roughly the *square* of those yielded by complete binary trees. This is all discussed in detail in the next section, where we present and analyze our construction.

Other related work. Our construction is reminiscent of those from several recent works on fully homomorphic encryption, attribute-based encryption, and garbled circuits, e.g., [GSW13, BV14, BGG$^+$14]. In particular, these works obtain relatively good LWE assumptions and parameters by appropriately scheduling "bit decomposition" operations to ensure small noise growth, usually at the expense of increased sequentiality. Our work also falls within this theme, though our proof techniques are completely different.

Organization. In Section 2 we give our construction and a detailed analysis of its security and efficiency. In Section 3 we give the proof of the security theorem, first providing an overview of the key ideas in Section 3.1, and giving the formal proof in Section 3.3 (after recalling some necessary technical background in Section 3.2).

Reference	KH?	Expan	Sequen	Key	Params	Time/Out	Out
this work	Y	1	$\lambda - 1$	λ [λ]	λ^2 [λ]	λ^ω [λ]	λ [λ]
this work	Y	$\log_4 \lambda$	$\log_4 \lambda$	λ [λ]	λ^2 [λ]	λ^ω [λ]	λ [λ]
[BLMR13]	Y	$\lambda - 1$	1	λ^3 [λ^3]	λ^6 [λ^4]	λ^5 [λ^3]	λ^2 [λ^2]
[BPR12, GGM]	N	1	λ	λ [λ]	λ^2 [λ]	λ^2 [λ]	λ [λ]
[BPR12, synth]	N	$\log_2 \lambda$	$\log_2 \lambda$	λ^3 [λ^2]	0 [0]	$\lambda^{\omega-1}$ [λ]	λ^2 [λ]
[BPR12, direct]	N	λ	1	λ^5 [λ^3]	0 [0]	λ^4 [λ^2]	λ^2 [λ^2]

Fig. 1. Example instantiations of our key-homomorphic PRF (for input length λ and provable 2^λ security against the best known lattice algorithms) as compared with prior lattice-based PRFs. "KH" denotes whether the construction is key homomorphic, while "Expan" and "Sequen" are respectively the expansion and sequentiality (as defined in Equations (2.4), (2.7)) of the tree T used in the instantiation (or, for prior constructions, their close analogues). Omitting polylogarithmic $\log^{O(1)} \lambda$ factors, "Key" and "Params" are respectively the bit lengths of the secret key and public parameters; "Time/Out" is the best known runtime (in bit operations) per output bit, where $\omega \in [2, 2.373]$ is the exponent of matrix multiplication; and "Out" is the output length in bits. The quantities in brackets refer to the ring-based construction given in Section 2.4.

2 Construction and Analysis

In this section we define and analyze our key-homomorphic PRF, and compare it with prior LWE-based constructions. The construction involves various parameters (e.g., matrix dimension n, modulus q, tree T) which are all chosen so that the algorithms are polynomial-time in the security parameter λ. As in [BLMR13], we work in a model where the PRF family is defined with respect to some *random public parameters* that are known to all parties, including the adversary. These parameters may be generated by a trusted party, or by the user along with the secret key.

We first recall some standard background. For an integer modulus $q \geq 1$, let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ denote the quotient ring of integers modulo q. For an integer $p \leq q$, define the modular "rounding" function $\lfloor \cdot \rceil_p \colon \mathbb{Z}_q \to \mathbb{Z}_p$ as $\lfloor x \rceil_p = \lfloor \frac{p}{q} \cdot x \rceil$, and extend it coordinate-wise to vectors and matrices over $\mathbb{Z}_q$. Let $\ell = \lceil \log q \rceil$ and define the "gadget" (column) vector

$$\mathbf{g} = (1, 2, 4, \ldots, 2^{\ell-1}) \in \mathbb{Z}_q^\ell,$$

and the (deterministic) "binary decomposition" function $\mathbf{g}^{-1} \colon \mathbb{Z}_q \to \{0,1\}^\ell$ as follows: identifying each $a \in \mathbb{Z}_q$ with its integer residue in $\{0, \ldots, q-1\}$, let $\mathbf{g}^{-1}(a) = (x_0, x_1, \ldots, x_{\ell-1}) \in \{0,1\}^\ell$ where $a = \sum_{i=0}^{\ell-1} x_i 2^i$ is the binary

representation of a. Note that by definition, $\langle \mathbf{g}, \mathbf{g}^{-1}(a) \rangle = a$ for all $a \in \mathbb{Z}_q$, which explains our choice of notation.[1]

Similarly, for vectors and matrices over $\mathbb{Z}_q$ we define the function $\mathbf{G}^{-1} \colon \mathbb{Z}_q^{n \times m} \to \{0,1\}^{n\ell \times m}$ by applying $\mathbf{g}^{-1}$ entry-wise. Notice that for all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ we have

$$\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A}, \quad \text{where} \quad \mathbf{G} = \mathbf{g}^t \otimes \mathbf{I}_n = \mathrm{diag}(\mathbf{g}^t, \ldots, \mathbf{g}^t) \in \mathbb{Z}_q^{n \times n\ell} \tag{2.1}$$

is the block matrix with n copies of $\mathbf{g}^t$ as diagonal blocks, and zeros elsewhere.

For a full (but not necessarily complete) binary tree T—i.e., one in which every non-leaf node has two children—let $|T|$ denote the number of its leaves. If $|T| \geq 1$ (i.e., T is not the empty tree), let $T.l, T.r$ respectively denote the left and right subtrees of T (which may be empty trees).

We now define our function families.

Definition 2.1. *Given matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$ and a full binary tree T of at least one node, define the function $\mathbf{A}_T \colon \{0,1\}^{|T|} \to \mathbb{Z}_q^{n \times n\ell}$ recursively as*

$$\mathbf{A}_T(x) = \begin{cases} \mathbf{A}_x & \text{if } |T| = 1 \\ \mathbf{A}_{T.l}(x_l) \cdot \mathbf{G}^{-1}(\mathbf{A}_{T.r}(x_r)) & \text{otherwise,} \end{cases} \tag{2.2}$$

where in the second case we parse $x = x_l \| x_r$ for $x_l \in \{0,1\}^{|T.l|}, x_r \in \{0,1\}^{|T.r|}$.

Construction 2.1 (Key-Homomorphic PRF). The function family

$$\mathcal{F}_{\mathbf{A}_0, \mathbf{A}_1, T, p} = \left\{ F_{\mathbf{s}} \colon \{0,1\}^{|T|} \to \mathbb{Z}_p^{n\ell} \right\}$$

is parameterized by matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$, a binary tree T, and a modulus $p \leq q$, which may all be considered public parameters. A member of the family is indexed by some $\mathbf{s} \in \mathbb{Z}_q^n$, and is defined as

$$F_{\mathbf{s}}(x) := \left\lfloor \mathbf{s}^t \cdot \mathbf{A}_T(x) \right\rceil_p. \tag{2.3}$$

For security based on LWE, we take $\mathbf{A}_0, \mathbf{A}_1$ and the secret key $\mathbf{s}$ to be uniformly random over $\mathbb{Z}_q$; see Theorem 2.1 below for a formal security statement. Similarly to LWE, it may also be possible to prove security when the entries of $\mathbf{s}$ are drawn from the LWE error distribution (see [ACPS09]). However, most applications of key-homomorphic PRFs need to use uniformly random secret keys anyway, so we do not pursue this question further.

Because rounding is nearly linear, i.e., $\lfloor a + b \rceil_p = \lfloor a \rceil_p + \lfloor b \rceil_p + e$ for some $e \in \{0, \pm 1\}$, it is easy to see that the family $\mathcal{F}_{\mathbf{A}_0, \mathbf{A}_1, T, p}$ defined above is "almost"

[1] These are just particular definitions of $\mathbf{g}, \mathbf{g}^{-1}$ that we fix for convenience. Our constructions and proofs only require that $\mathbf{g}^{-1}$ be deterministic, and that $\mathbf{g}^{-1}(a)$ be a "short" integer vector such that $\langle \mathbf{g}, \mathbf{g}^{-1}(a) \rangle = a$ for all $a \in \mathbb{Z}_q$. Alternatives include using a signed ternary decomposition, or a larger (or mixed-radix) base; the bounds in the security theorem are easily adapted to such choices.

additively key homomorphic, as defined in [BLMR13]. That is, for any keys $F_{\mathbf{s}}, F_{\mathbf{t}}$ in the family, we have

$$F_{\mathbf{s}+\mathbf{t}}(x) = F_{\mathbf{s}}(x) + F_{\mathbf{t}}(x) + \mathbf{e}^t,$$

where $\|\mathbf{e}\|_\infty \leq 1$. As long as the entries of the error term $\mathbf{e}$ are sufficiently smaller than the output modulus p, this near-homomorphism is sufficient for all the applications described in [BLMR13], and for obtaining security against related-key attacks [LMR14].

Notice that the vast majority of the cost of computing $F_{\mathbf{s}}(x)$ is in computing $\mathbf{A}_T(x)$, which can done "publicly" without any knowledge of the secret key $\mathbf{s}$.[2] This property can be very important for the efficiency of certain applications, such as the *homomorphic* evaluation of $F_{\mathbf{s}}$ given an encryption of $\mathbf{s}$. In addition, notice that if $\mathbf{A}_T(x)$ has been computed and all the intermediate matrices saved, then $\mathbf{A}_T(x')$ can be *incrementally* computed much more efficiently for an x' that differs from x in just a single bit. Specifically, one only needs to recompute the matrices for the internal nodes of T on the path from the leaf corresponding to the changed bit to the root. As in [BPR12], this can significantly speed up successive evaluations of $F_{\mathbf{s}}$ on related inputs, e.g., in a counter-like mode using a Gray code.

Relation to [BLMR13]. Our key-homomorphic PRF can be viewed as a substantial generalization of the one of Boneh *et al.* [BLMR13]. Specifically, their construction can be obtained from ours by instantiating it with a tree T that consists of a "left spine" with leaves for all its right children. Because all the right subtrees are just leaves, the only matrices ever decomposed with $\mathbf{G}^{-1}$ are $\mathbf{A}_0$ and $\mathbf{A}_1$. Therefore, we can replace them in the public parameters by the binary matrices $\mathbf{B}_b = \mathbf{G}^{-1}(\mathbf{A}_b)$, yielding the construction $F_{\mathbf{r}}(x) = \lfloor \mathbf{r}^t \cdot \prod_{i=1}^{|x|} \mathbf{B}_{x_i} \rceil_p$ from [BLMR13].[3]

The use of a "left-spine" tree T (as in [BLMR13]) yields an instantiation which is *maximally parallel*—in our language (defined below), it has *sequentiality* $s(T) = 1$. The major drawback is that it also has maximal *expansion* $e(T) = |T| - 1$. In our security theorem (Theorem 2.1 below), the LWE approximation factor and modulus q grow *exponentially* with $e(T)$, so using a tree with large expansion leads to a very strong hardness assumption, and therefore large secret keys and public parameters. By contrast, using trees T with better expansion-sequentiality tradeoffs allows us obtain much better key sizes and efficiency. See the discussion in the following subsections and Figure 1 for further details.

[2] For a few choices of the tree T, it can be faster to compute $\mathbf{s}^t \cdot \mathbf{A}_T(x)$ left-to-right without explicitly computing $\mathbf{A}_T(x)$, but such trees are rare and yield bad parameters.

[3] Here we have ignored the small detail that in our construction, the matrix $\mathbf{A}_{x_1}$ corresponding to the leftmost leaf in the tree is not decomposed, so our instantiation is actually $F_{\mathbf{s}}(x) = \lfloor \mathbf{s}^t \cdot \mathbf{A}_{x_1} \cdot \prod_{i=2}^{|x|} \mathbf{B}_{x_i} \rceil$. However, it is easy to verify that in the construction of [BLMR13], the secret key may be of the form $\mathbf{r}^t = \mathbf{s}^t\mathbf{G}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$. Then $\mathbf{r}^t\mathbf{B}_{x_1} = \mathbf{s}^t\mathbf{A}_{x_1}$, which corresponds to our construction.

2.1 Security

In our security proof, the modulus q and underlying LWE error rate, and hence also the dimension n needed to obtain a desired level of provable security, are largely determined by a certain parameter of the tree T which we call the *expansion* $e(T)$. Essentially, the expansion is the maximum number of terms of the form $\mathbf{G}^{-1}(\cdot)$ that are ever consecutively multiplied together when we unwind the recursive definition of $\mathbf{A}_T$, or $\mathbf{A}_{T'}$ for related trees T' considered in the security proof. Formally, the expansion of T is defined by the recurrence

$$e(T) = \begin{cases} 0 & \text{if } |T| = 1 \\ \max\{e(T.l) + 1\ ,\ e(T.r)\} & \text{otherwise.} \end{cases} \tag{2.4}$$

This is simply the "left depth" of the tree, i.e., the maximum length of a root-to-leaf path, counting only edges from parents to their left children.

We can now state our main security theorem.

Theorem 2.1. *Let T be any full binary tree, χ be some distribution over $\mathbb{Z}$ that is subgaussian with parameter $r > 0$ (e.g., a bounded or discrete Gaussian distribution with expectation zero), and*

$$q \geq p \cdot r\sqrt{|T|} \cdot (n\ell)^{e(T)} \cdot \lambda^{\omega(1)}. \tag{2.5}$$

Then over the uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$, the family $\mathcal{F}_{\mathbf{A}_0,\mathbf{A}_1,T,p}$ with secret key chosen uniformly from $\mathbb{Z}_q^n$ is a secure PRF family, under the decision-$\mathsf{LWE}_{n,q,\chi}$ assumption.

An outline of the proof, which contains all the main and new ideas, is given in Section 3.1; the formal proof appears in in Section 3.3.

Notice that the modulus-to-noise ratio for the underlying LWE problem is $q/r \approx (n \log q)^{e(T)}$, i.e., exponential in the expansion $e(T)$. Known reductions [Reg05, Pei09, BLP+13] (for $r \geq 3\sqrt{n}$) guarantee that such an LWE instantiation is at least as hard as (quantumly) approximating various lattice problems in the worst case to within $\approx q/r$ factors on n-dimensional lattices. Known algorithms for achieving such factors take time exponential in $n/\log(q/r) = \tilde{\Omega}(n/e(T))$, so in order to obtain provable 2^λ security against the best known lattice algorithms, the best parameters we can use are

$$n = e(T) \cdot \tilde{\Theta}(\lambda) \quad \text{and} \quad \log q = e(T) \cdot \tilde{\Theta}(1). \tag{2.6}$$

These parameters determine the runtimes and key sizes of the construction, as analyzed below.

We conclude this discussion of security by remarking that, as in [BPR12, BLMR13], and in contrast with essentially all lattice-based *encryption* schemes, it is possible that our PRF is actually secure for *much smaller* parameters than our proof requires. For example, taking $q = \text{poly}(n)$ even for large $e(T)$, with $p|q$ to ensure that rounding produces "unbiased" output, may actually be secure—but

we do not know how to prove it. (We also do not know of any effective attacks against such parameters.) The reason for this possibility is that the function itself does not actually expose any low-error-rate LWE samples to the attacker; they are used only in the proof as part of a thought experiment. Whether any of the constructions from this work or [BPR12, BLMR13] can be proved secure for smaller parameters under a standard assumption is a fascinating open question. For the remainder of the paper, we deal only with parameters for which we can *prove* security under (ring-)LWE.

2.2 Size, Time, and Depth

Here we briefly analyze the secret key and public parameter sizes, runtime, and circuit depth of our PRFs, always normalizing to 2^λ provable security under standard lattice assumptions. In some cases these quantities are not very practical (or even asymptotically good), especially when the tree T has large expansion. In Section 2.4 we give a much more efficient construction using ring-LWE, which can be quasi-optimal in key size, public parameters, and depth (simultaneously).

The secret key, which is a uniformly random element of $\mathbb{Z}_q^n$, has size $\Theta(n \log q)$, which is $e(T)^2 \cdot \tilde{\Theta}(\lambda)$ by Equation (2.6). The public parameters, being two $n \times n\ell$ matrices over $\mathbb{Z}_q$, are $\Theta(n^2 \log^2 q) = e(T)^4 \cdot \tilde{\Theta}(\lambda^2)$ bits.

For runtime, computing $\mathbf{A}_T(x)$ from scratch takes one decomposition with $\mathbf{G}^{-1}$ and one $(n \times n\ell)$-by-$(n\ell \times n\ell)$ matrix multiplication over $\mathbb{Z}_q$ per internal node of T. (As mentioned above, incremental computation of $\mathbf{A}_T(x)$ on related inputs can be much faster.) Using naïve matrix multiplication, this is a total of $\Theta(|T| \cdot n^3 \log^2 q)$ ring operations in $\mathbb{Z}_q$, which translates to $e(T)^6 \cdot \tilde{\Theta}(\lambda^4)$ bit operations by Equation (2.6) (even using quasi-linear-time multiplication in $\mathbb{Z}_q$, which is needed only when $\log q \neq \tilde{O}(1)$). This can be improved somewhat using asymptotically faster matrix multiplication, but still remains a rather large $\Omega(|T| \cdot n^\omega \log^2 q)$, where $\omega \geq 2$ is the exponent of matrix multiplication.

For certain trees T our construction is highly parallelizable, i.e., it can be computed by a low-depth circuit. First, notice that each $\mathbb{Z}_q$-entry of $\mathbf{s}^t \cdot \mathbf{A}_T(x)$ (and hence each $\mathbb{Z}_p$-entry of the PRF output) can be computed independently. This is because each column of $\mathbf{A}_T(x)$ can be computed independently, by induction and the fact that $\mathbf{G}^{-1}$ works independently on the columns of $\mathbf{A}_{T.r}(x_r)$. Next, since linear operations over $\mathbb{Z}_q$ can be computed by depth-one arithmetic circuits (with unbounded fan-in), the circuit depth of our construction is proportional to the maximum nesting depth of $\mathbf{G}^{-1}(\cdot)$ expressions when we fully unwind the definition of $\mathbf{A}_T$. We call this the *sequentiality* $s(T)$ of the tree T, which is formally defined by the recurrence

$$s(T) = \begin{cases} 0 & \text{if } |T| = 1 \\ \max\{e(T.l)\ ,\ e(T.r) + 1\} & \text{otherwise.} \end{cases} \tag{2.7}$$

This is simply the "right depth" of the tree, i.e., the maximum length of a root-to-leaf path, counting only edges from parents to their right children.

2.3 Instantiations

Here we discuss some interesting instantiations of the tree T and the efficiency properties of the resulting functions; see Figure 1 for a summary. Generally speaking, for a given tree size $|T|$ (the PRF input length) there is a tradeoff between expansion $e(T)$ and sequentiality $s(T)$. Flipping this around, given bounds e, s we are interested in obtaining a largest possible tree T such that $e(T) \leq e$ and $s(T) \leq s$; let $t(e, s)$ denote the size of such a tree. At first blush, it may be surprising that under the simplifying restriction $e = s$, a complete binary tree of depth s is *very far* from optimal! To see this, notice that

$$t(e,s) = \begin{cases} 1 & \text{if } e = 0 \text{ or } s = 0 \\ t(e-1,s) + t(e,s-1) & \text{otherwise.} \end{cases} \tag{2.8}$$

The base cases follow from the fact that only a single leaf satisfies the bounds, and in the recursive case, the first and second terms respectively denote the sizes of the optimal left and right subtrees. It is easy to verify that this recurrence is simply the one that defines the *binomial coefficients*:

$$t(e,s) = \binom{e+s}{e} = \binom{e+s}{s}.$$

One can also efficiently construct an optimal tree for given e, s using dynamic programming.

For example, if we restrict to $e = s$, then by Stirling's approximation we get that $t(e,s) = \binom{2s}{s} \approx 4^s/\sqrt{s\pi}$. Said another way, we can get a PRF with input length $|T|$ where the expansion and sequentiality are both $\approx \log_4(|T|)$. By contrast, a complete binary tree with these parameters has size only $2^s \approx \sqrt{|T|}$. By Theorem 2.1 and Equation (2.6), this means we can get a PRF with input length λ and security 2^λ having sequentiality $O(\log \lambda)$ and secret keys of quasi-optimal bit length $\tilde{O}(\lambda)$.

By ignoring parallelism, one can reduce the expansion even further by letting T be a "right spine" with leaves for all its left children. Then $e(T) = 1$ and $s(T) = |T| - 1$, yielding even better parameters: the underlying LWE assumption has a nearly polynomial $n^{\omega(1)}$ approximation factor, and for security level 2^λ we still obtain secret keys of quasi-optimal bit length $\tilde{O}(\lambda)$; moreover, here the hidden factors are at least a $\log \lambda$ factor smaller than in the case above.

2.4 Ring Variant

Due to the several matrix multiplications (of dimension at least n) involved in computing $\mathbf{A}_T(x)$, our LWE-based construction is not very practically efficient. Fortunately, we can obtain a much more efficient analogue based on the ring-LWE problem [LPR10]. Here we just describe the construction and analyze its efficiency. The proof of security based on ring-LWE proceeds in essentially the same way as the one for our main construction, and is therefore omitted.

For concreteness, let $R \cong \mathbb{Z}[X]/(X^n+1)$ where n is a power of two, which is known as the $2n$th cyclotomic ring. (The construction and analysis may be generalized to arbitrary cyclotomic rings using the tools developed in [LPR13].) For a modulus q, let $R_q = R/qR \cong \mathbb{Z}_q[X]/(X^n+1)$, and define a suitable "gadget" vector $\mathbf{g} \in R_q^\ell$ and deterministic function $\mathbf{g}^{-1}\colon R_q \to R^\ell$, so that $\mathbf{g}^{-1}(a)$ is "short" and $\langle \mathbf{g}, \mathbf{g}^{-1}(a)\rangle = a$ for all $a \in R_q$. (E.g., we may let $\mathbf{g} = (1, 2, 4, \ldots, 2^{\ell-1}) \in R_q^\ell$ and define $\mathbf{g}^{-1}(a)$ so that each of its R-entries has $\{0,1\}$-coefficients with respect to an appropriate "short" $\mathbb{Z}$-basis of R.) Extend $\mathbf{g}^{-1}$ to row vectors over R_q by applying $\mathbf{g}^{-1}$ entry-wise.

Construction 2.2. Fix some *row* vectors $\mathbf{a}_0, \mathbf{a}_1 \in R_q^\ell$, and for a binary tree T, define $\mathbf{a}_T\colon \{0,1\}^{|T|} \to R_q^\ell$ recursively as

$$\mathbf{a}_T(x) = \begin{cases} \mathbf{a}_x & \text{if } |T| = 1 \\ \mathbf{a}_{T.l}(x_l) \cdot \mathbf{g}^{-1}(\mathbf{a}_{T.r}(x_r)) & \text{otherwise,} \end{cases} \tag{2.9}$$

where in the second case we parse $x = x_l \| x_r$ for $x_l \in \{0,1\}^{|T.l|}, x_r \in \{0,1\}^{|T.r|}$.

We define the function family

$$\mathcal{F}_{\mathbf{a}_0,\mathbf{a}_1,T,p} = \left\{ F_s\colon \{0,1\}^{|T|} \to R_p^\ell \right\},$$

which is parameterized by row vectors $\mathbf{a}_0, \mathbf{a}_1 \in R_q^\ell$, a binary tree T, and a modulus $p \le q$. A member of the family is indexed by some $s \in R$ (or R_q), and is defined as

$$F_s(x) := \lfloor s \cdot \mathbf{a}_T(x) \rceil_p. \tag{2.10}$$

Analysis. Evaluating $\mathbf{a}_T(x)$ from scratch takes one decomposition with $\mathbf{g}^{-1}$ and one vector-matrix multiplication of dimension $\ell = \log q$ over R_q per internal node of T, for a total of $O(|T| \cdot \ell^2)$ ring operations in R_q. Ring operations in R_q can be performed in $O(n \log n)$ scalar operations over $\mathbb{Z}_q$, and $\mathbf{g}^{-1}$ can be computed in $O(n \log q)$ time. Using a tree T with expansion $e(T) = \tilde{O}(1)$, by Equation (2.6) we can get a PRF with input length λ and 2^λ security (under conventional assumptions) running in $\tilde{O}(\lambda^2)$ bit operations to output at least λ bits. When T has polylogarithmic depth, the incremental cost per invocation is reduced to $\tilde{O}(\lambda)$ bit operations, which is quasi-optimal.

As an optimization, and analogously to the LWE-based construction, each R_q-entry of $\mathbf{a}_T(x) \in R_q^\ell$ can be computed *independently* in $O(|T| \cdot \ell)$ ring operations each. Therefore, we can compute each R_p-entry of the output (yielding at least n output bits) in just $O(|T| \cdot \ell)$ ring operations. This may be useful in applications that do not need the entire large output length.

3 Security Proof

In this section we prove the security theorem, Theorem 2.1, which says that $F_{\mathbf{s}}(x) = \lfloor \mathbf{s}^t \cdot \mathbf{A}_T(x) \rceil_p$ from Construction 2.1 is a PRF under the LWE assumption, for appropriate parameters.

3.1 Proof Outline

We start with an overview of the proof, which highlights the central (new) ideas. (For technical reasons, the formal proof proceeds a bit differently than this outline, but the main ideas are the same.) The basic strategy, first used in [BPR12], is to define a sequence of hybrid games where the function inside the rounding operation $\lfloor \cdot \rceil_p$ changes in ways that are indistinguishable to the adversary, either statistically or computationally. As in prior works [BPR12, BLMR13], these changes include introducing small additive terms that are "rounded away" and hence preserve the input-output behavior (with high probability), and replacing LWE instances with uniformly random ones. In addition, we introduce a new proof technique described within.

Let T be any full binary tree, and suppose its leftmost leaf v is at depth $d > 1$. (If $d = 1$, then $|T| = 1$ and the function is trivially a PRF based on the "learning with rounding" problem, which is as hard as LWE for our choice of parameters, or even slightly better ones [BPR12, AKPW13].) In the real attack game, the adversary has oracle access to $F_{\mathbf{s}}(\cdot)$, which, by unwinding the definition of $\mathbf{A}_T$, is of the form

$$F_{\mathbf{s}}(x) = \left\lfloor \mathbf{s}^t \cdot \mathbf{A}_T(x) \right\rceil_p = \left\lfloor \mathbf{s}^t \cdot \mathbf{A}_{x_0} \cdot \underbrace{\prod_{i=1}^{d} \mathbf{G}^{-1}(\mathbf{A}_{T_i}(x'_i))}_{\mathbf{S}_T(x')} \right\rceil_p ,$$

where subtree T_i is the right child of v's ith ancestor, and $x = x_0 \| x' = x_0 \| x'_1 \| \cdots \| x'_d$ where $|x_0| = 1$ and $|x'_i| = |T_i|$ for all i.

We next consider a hybrid game in which $\mathbf{s}^t \cdot \mathbf{A}_b$ for $b \in \{0, 1\}$ is replaced by an LWE vector $\mathbf{s}^t \cdot \mathbf{A}_b + \mathbf{e}_b^t$, for some short error vectors $\mathbf{e}_0, \mathbf{e}_1$. That is, the adversary instead has oracle access to the function

$$F'_{\mathbf{s},\mathbf{e}_0,\mathbf{e}_1}(x) := \left\lfloor (\mathbf{s}^t \cdot \mathbf{A}_{x_0} + \mathbf{e}_{x_0}^t) \cdot \mathbf{S}_T(x') \right\rceil_p = \left\lfloor \mathbf{s}^t \cdot \mathbf{A}_T(x) + \mathbf{e}_{x_0}^t \cdot \mathbf{S}_T(x') \right\rceil_p .$$

Because $\mathbf{e}_{x_0}$ and any matrix of the form $\mathbf{G}^{-1}(\cdot)$ are short, $\mathbf{e}_{x_0}^t \cdot \mathbf{S}_T(x')$ is short. More precisely, its entries are of magnitude bounded by $\approx (n \log q)^d$, which is much less than q/p because $d \leq e(T)$ and by assumption on q. Therefore, the additive term $\mathbf{e}_{x_0}^t \cdot \mathbf{S}_T(x')$ is very unlikely the change the final rounded value, i.e., with high probability $F'_{\mathbf{s},\mathbf{e}_0,\mathbf{e}_1}(x) = F_{\mathbf{s}}(x)$ for all the adversary's queries x. Therefore, this hybrid game is statistically indistinguishable from the real attack.

In the next hybrid game, we replace each $\mathbf{s}^t \cdot \mathbf{A}_b + \mathbf{e}_b^t$ for $b \in \{0, 1\}$ by uniformly random and independent $\mathbf{u}_b^t$, i.e., the adversary has access to the function

$$F''_{\mathbf{u}_0,\mathbf{u}_1}(x) := \left\lfloor \mathbf{u}_{x_0}^t \cdot \mathbf{S}_T(x') \right\rceil_p = \left\lfloor \mathbf{u}_{x_0}^t \cdot \mathbf{G}^{-1}(\mathbf{A}_{T_1}(x'_1)) \cdot \underbrace{\prod_{i=2}^{d} \mathbf{G}^{-1}(\mathbf{A}_{T_i}(x'_i))}_{\mathbf{S}'_T(x')} \right\rceil_p . \quad (3.1)$$

Since $\mathbf{S}_T(x')$ can be efficiently computed from the public parameters $\mathbf{A}_b$ and the adversary's queries x, this game is computationally indistinguishable from the previous one, under the LWE assumption.

At this point, we would like to be able to proceed by replacing the terms $\mathbf{u}_{x_0}^t \cdot \mathbf{G}^{-1}(\mathbf{A}_{T_1}(x_1'))$ with some "noisy" variants, then replace those with uniform and independent vectors for all values of $x_0 \| x_1'$, etc. Indeed, this is possible if x_1' consists of a *single bit* (i.e., if $|T_1| = 1$ and hence $\mathbf{A}_{T_1}(x_1') = \mathbf{A}_{x_1'}$), using "non-uniform LWE" exactly as is done in [BLMR13]. Unfortunately, non-uniform LWE does not appear to be sufficient when x_1' is more than one bit (i.e., when $|T_1| > 1$), because the matrices $\mathbf{A}_{T_1}(x_1')$ are not independent for different values of x_1'. And requiring $|T_i| = 1$ for all i makes T have maximal expansion $e(T) = |T| - 1$.

Our main new proof technique is a way to deal with the above issue. Going back to Equation (3.1), as "wishful thinking" suppose that each $\mathbf{u}_b$ was of the form $\mathbf{u}_b^t = \mathbf{s}_b^t \cdot \mathbf{G}$ for some (uniform, say) $\mathbf{s}_b \in \mathbb{Z}_q^n$. Then the $\mathbf{G}$ factor would undo the decomposition $\mathbf{G}^{-1}(\cdot)$, and the adversary would have access to the function

$$F'''_{\mathbf{s}_0,\mathbf{s}_1}(x) := \left\lfloor \mathbf{s}_{x_0}^t \cdot \mathbf{A}_{T_1}(x_1') \cdot \mathbf{S}_T'(x') \right\rceil_p = \left\lfloor \mathbf{s}_{x_0}^t \cdot \mathbf{A}_{T'}(x') \right\rceil_p,$$

where T' is the full binary tree obtained from T by removing its leftmost leaf v and promoting v's sibling subtree T_1 to replace their parent. Notice that the above function is just two independent members of our function family instantiated with tree T'. Moreover, T' has expansion $e(T') \leq e(T)$, because expansion is just "left depth." Therefore, the above function would be a PRF simply by induction on $|T|$.

Unfortunately, our "wishful thinking" fails in a very strong sense: a uniformly random $\mathbf{u}^t$ is highly likely to be *very far* from any vector of the form $\mathbf{s}^t \cdot \mathbf{G}$. However, because $\mathbf{G}^t \cdot \mathbb{Z}_q^n$ is a *subgroup* of $\mathbb{Z}_q^{n\ell}$, a uniformly random vector $\mathbf{u} \in \mathbb{Z}_q^{n\ell}$ *can* be decomposed as $\mathbf{u}^t = \mathbf{s}^t \cdot \mathbf{G} + \mathbf{v}^t$ where $\mathbf{s} \in \mathbb{Z}_q^n$ is uniform, and $\mathbf{v}$ is uniform in (some canonical set of representatives of) the quotient group $\mathbb{Z}_q^{n\ell}/(\mathbf{G}^t \cdot \mathbb{Z}_q^n)$ and *independent* of $\mathbf{s}$. Therefore, the function in Equation (3.1) is equivalent to the function

$$F'''_{\mathbf{s}_0,\mathbf{s}_1,\mathbf{v}_0,\mathbf{v}_1}(x) := \left\lfloor \mathbf{s}_{x_0}^t \cdot \mathbf{A}_{T'}(x') \quad + \quad \mathbf{v}_{x_0}^t \cdot \mathbf{S}_T(x') \right\rceil_p,$$

where T' and x' are exactly as in the previous paragraph. Note that $\mathbf{v}_b$ is *not* short, so the extra additive term above does not simply "round away"—but we do not need it to. The main point is that $\mathbf{v}_b$ may be chosen *independently* of (and hence without knowledge of) $\mathbf{s}_b$ by the simulator, and then the additive term may be efficiently computed from it and other public information. Essentially, this allows us to complete the proof by induction on $|T|$. (Again, the actual proof is structured a bit differently, to allow us to simulate the independent additive terms *inside* the rounding operation.)

3.2 Additional Background

Games and indistinguishability. In our security proof, we model interaction with the adversary through a series of probabilistic experiments called *games*. For an adversary $\mathcal{A}$ interacting with two games H_0 and H_1, the *distinguishing advantage* of $\mathcal{A}$, which is implicitly a funtion of the security parameter λ, is

defined as $\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) = |\Pr[\mathcal{A} \text{ accepts in } H_0] - \Pr[\mathcal{A} \text{ accepts in } H_1]|$. Two games H_0 and H_1 are *computationally distinguishable*, denoted $H_0 \overset{c}{\approx} H_1$, if $\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) = \mathrm{negl}(\lambda)$ for any efficient adversary $\mathcal{A}$.

Learning with errors. We use the following form of the learning with errors (LWE) problem, due to Regev [Reg05]. For a positive integer dimension n, a modulus $q \geq 2$, and a probability distribution χ over $\mathbb{Z}$, the decision-$\mathsf{LWE}_{n,q,\chi}$ assumption is that for for any polynomially bounded m, w,

$$(\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{B}^t = \mathbf{S}^t \cdot \mathbf{A} + \mathbf{E}^t \in \mathbb{Z}_q^{w\times m}) \overset{c}{\approx} (\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{B}^t \leftarrow \mathbb{Z}_q^{w\times m}),$$

where on the left $\mathbf{S}^t \leftarrow \mathbb{Z}_q^{w\times n}$ and $\mathbf{E}^t \leftarrow \chi^{w\times m}$. (The assumption for $w = 1$ implies the assumption for larger w, by a routine hybrid argument.)

For $\chi = D_{\mathbb{Z},r}$ where $r \geq 3\sqrt{n}$, and under mild conditions on the form of the modulus q, the decision-$\mathsf{LWE}_{n,q,\chi}$ assumption holds true assuming that various problems on n-dimensional lattices are hard for quantum algorithms to approximate to within $\tilde{O}(n \cdot q/r)$ factors in the worst case [Reg05]; see also [Pei09, BLP+13] and references therein for similar statements assuming only classical (non-quantum) hardness.

3.3 Proof of Security Theorem

In this section we give the formal proof of Theorem 2.1.

To aid the proof we first define a couple of auxiliary function families. The first family simply consists of the "pre-rounded" counterparts of the functions $F_{\mathbf{s}} \in \mathcal{F} = \mathcal{F}_{\mathbf{A}_0,\mathbf{A}_1,T,p}$.

Definition 3.1. *For $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n\times n\ell}$ and a full binary tree T, the family $\mathcal{G} = \mathcal{G}_{\mathbf{A}_0,\mathbf{A}_1,T}$ is the set of functions $G_{\mathbf{s}} \colon \{0,1\}^{|T|} \to \mathbb{Z}_q^{n\ell}$ indexed by some $\mathbf{s} \in \mathbb{Z}_q^n$, and defined as $G_{\mathbf{s}}(x) := \mathbf{s}^t \cdot \mathbf{A}_T(x)$ (where we define $\mathbf{A}_T(\varepsilon) := \mathbf{G}$ for the empty tree T). We endow $\mathcal{G}$ with the distribution where $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ is chosen uniformly at random.*

Note that $F_{\mathbf{s}}(x) = \lfloor G_{\mathbf{s}}(x) \rceil_p$.

The next family $\tilde{\mathcal{G}}$ consists of functions that are certain "noisy" versions of the functions in $\mathcal{G}$. The family $\mathcal{E}$ of "error functions" used in the definition is a family of functions from $\{0,1\}^{|T|}$ to $\mathbb{Z}^{n\ell}$, and is formally defined in Definition 3.5 below. An important point is that the functions in $E \in \mathcal{E}$ have *exponentially* large keys, but they may be efficiently sampled "lazily," as values $E(x)$ are needed. See the discussion following Definition 3.5 for details.

Definition 3.2. *For $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n\times n\ell}$ and a full binary tree T, the family $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}_{\mathbf{A}_0,\mathbf{A}_1,T}$ is the set of functions $\tilde{G}_{\mathbf{s},E} \colon \{0,1\}^{|T|} \to \mathbb{Z}_q^{n\ell}$ indexed by some $G_{\mathbf{s}} \in \mathcal{G}$ and $E \in \mathcal{E} = \mathcal{E}_{\mathbf{A}_0,\mathbf{A}_1,T}$, and defined as $\tilde{G}_{\mathbf{s},E}(x) := G_{\mathbf{s}}(x) + E(x)$. We endow $\tilde{\mathcal{G}}$ with the distribution where $G_{\mathbf{s}} \leftarrow \mathcal{G}$ and $E \leftarrow \mathcal{E}$ are chosen independently.*

The proof of Theorem 2.1 consists of showing that with overwhelming probability, the rounding of $G_{\mathbf{s}} \in \mathcal{G}$ agrees with the rounding of essentially any corresponding $\tilde{G}_{\mathbf{s},E} \in \tilde{\mathcal{G}}$ on all the attacker's queries, because the outputs of the error functions $E \in \mathcal{E}$ are small. This proof follows very similary to the style of the proof of the "degree-k" PRF of [BPR12], and thus we relegate the details to the full version. The main crux of the theorem, which we show in Theorem 3.1 below, is in proving that $\tilde{\mathcal{G}}$ is a PRF family *without* any rounding, and hence with rounding as well. It follows that the rounding of $G_{\mathbf{s}} \leftarrow \mathcal{G}$ (i.e., $F_{\mathbf{s}} \leftarrow \mathcal{F}$) cannot be distinguished from a uniformly random function, as desired.

We now formally define the "error function" family $\mathcal{E} = \mathcal{E}_{\mathbf{A}_0,\mathbf{A}_1,T}$. To define the error functions we first need a couple of simple definitions.

Definition 3.3 (Pruning). *For a full binary tree T of at least one node, define its* pruning $T' = pr(T)$ *inductively as follows: if $|T.l| \leq 1$ then $T' := T.r$; otherwise, $T'.l := pr(T.l)$ and $T'.r := T.r$. We let $T^{(i)}$ denote the ith successive pruning of T, i.e., $T^{(0)} = T$ and $T^{(i)} = pr(T^{(i-1)})$.*

In other words, pruning a tree node removes its leftmost leaf v and replaces the subtree rooted at v's parent (if it exists) with the subtree rooted at v's sibling. Notice that pruning cannot increase the tree's expansion (i.e., left depth; see Equation (2.4)): $e(T') \leq e(T)$.

Definition 3.4. *Given $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$ and a full binary tree T of at least one node, define the function $\mathbf{S}_T \colon \{0,1\}^{|T|-1} \to \mathbb{Z}^{n\ell \times n\ell}$ recursively as follows:*

$$\mathbf{S}_T(x) = \begin{cases} \mathbf{I} \text{ (the identity matrix)} & \text{if } |T| = 1 \\ \mathbf{S}_{T.l}(x_l) \cdot \mathbf{G}^{-1}(\mathbf{A}_{T.r}(x_r)) & \text{otherwise,} \end{cases} \tag{3.2}$$

where $x = x_l \| x_r$ for $|x_l| = |T.l| - 1, |x_r| = |T.r|$.

Notice that if $T' = pr(T)$ and $x = x_1 \| x' \in \{0,1\}^{|T|}$ for $|x_1| = 1$, then it follows directly from the definitions (recalling that $\mathbf{A}_\varepsilon(\varepsilon) = \mathbf{G}$) and by induction that

$$\mathbf{A}_T(x) = \mathbf{A}_{x_1} \cdot \mathbf{S}_T(x'), \tag{3.3}$$

$$\mathbf{G} \cdot \mathbf{S}_T(x') = \mathbf{A}_{T'}(x'). \tag{3.4}$$

Definition 3.5 (Error Functions). *For public matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$ and a full binary tree T, the family $\mathcal{E} = \mathcal{E}_{\mathbf{A}_0,\mathbf{A}_1,T}$ consists of functions from $\{0,1\}^{|T|}$ to $\mathbb{Z}^{n\ell}$, defined inductively as follows.*

- *For $|T| = 0$, the sole function in $\mathcal{E}$ is defined simply as $E(\varepsilon) := \mathbf{0}$.*
- *For $|T| \geq 1$, a function in $\mathcal{E}$ is indexed by some $\mathbf{e}_0, \mathbf{e}_1 \in \mathbb{Z}^{n\ell}$ and $E'_0, E'_1 \in \mathcal{E}' = \mathcal{E}_{\mathbf{A}_0,\mathbf{A}_1,T'}$, where T' is the pruning of T. For $x = x_1 \| x' \in \{0,1\}^{|T|}$, the function is defined as*

$$E_{\mathbf{e}_0,\mathbf{e}_1,E'_0,E'_1}(x) := \mathbf{e}_{x_1}^t \cdot \mathbf{S}_T(x') + E'_{x_1}(x').$$

For a given error function distribution χ over $\mathbb{Z}$, we endow $\mathcal{E}$ with the distribution where $\mathbf{e}_0, \mathbf{e}_1 \leftarrow \chi^{n\ell}$ and $E'_0, E'_1 \leftarrow \mathcal{E}'$ are all chosen independently.

Note that a function $E \in \mathcal{E}$ is fully specified by *exponentially* (in $|T|$) many error vectors (namely, one $\mathbf{e}_w$ for each $w \in \{0,1\}^{\leq |T|}$), and the value $E(x)$ is fully determined by those $\mathbf{e}_w$ where w is a prefix of x (and $\mathbf{A}_0, \mathbf{A}_1$). This large number of error vectors is what prevents $\tilde{\mathcal{G}}$ itself from being usable as a PRF family. However, as needed in the proof of Theorem 2.1, a function $E \leftarrow \mathcal{E}$ can be sampled "lazily" as values $E(x)$ are needed, since each value of $E(x)$ depends on only a small number of the error vectors. The fact that the output of the error function is "small" with very high probability is also used in the proof of the theorem. Proving this fact is a standard technical exercise, and it is furnished in the full version.

We now prove that the function family $\tilde{\mathcal{G}}$ from Definition 3.2 is pseudorandom.

Theorem 3.1. *For any $n, q \geq 1$ and error distribution χ over $\mathbb{Z}$, any full binary tree T, and over the uniformly random and independent choice of $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$, the family $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}_{\mathbf{A}_0, \mathbf{A}_1, T}$ is pseudorandom, assuming the hardness of decision-$\mathsf{LWE}_{n,q,\chi}$.*

Proof. We proceed through a series of games, one for each bit of the input. In each successive game, we modify the function family $\tilde{\mathcal{G}}$ a little, until we are left with the family of all functions from $\{0,1\}^{|T|}$ to $\mathbb{Z}_q^{n\ell}$ (with uniform distribution), and we show that each successive game is computationally indistinguishable under the LWE assumption from the theorem statement.

To define the games formally, we first need some notation. For a bit string x of length at least i, let $x_{(i)} = x_1 x_2 \cdots x_i$ denote the string of its first i bits, and let $x^{(i)}$ denote the remainder of the string. Where $\mathbf{A}_0, \mathbf{A}_1$ and T are clear from context, let $\mathcal{G}^{(i)} = \mathcal{G}_{\mathbf{A}_0, \mathbf{A}_1, T^{(i)}}$ and similarly for $\mathcal{E}^{(i)}$. Let $\mathcal{P} \subset \mathbb{Z}^{n\ell}$ denote an arbitrary set of representatives of the quotient group $\mathbb{Z}_q^{n\ell} / \mathbf{G}^t \cdot \mathbb{Z}_q^n$, and define a family of auxiliary functions $\mathcal{V}^{(i)} = \mathcal{V}^{(i)}_{\mathbf{A}_0, \mathbf{A}_1, T}$ as follows.

Definition 3.6. *For public matrices $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$, a full binary tree T, and $0 \leq i \leq |T|$, the family $\mathcal{V}^{(i)} = \mathcal{V}^{(i)}_{\mathbf{A}_0, \mathbf{A}_1, T}$ consists of functions from $\{0,1\}^{|T|}$ to $\mathbb{Z}^{n\ell}$, and is defined inductively as follows.*

- *The sole function in $\mathcal{V}^{(0)}$ is defined simply as $V(x) := \mathbf{0}$.*
- *For $i \geq 1$, a function in $\mathcal{V}^{(i)}$ is indexed by some $\mathbf{v}_w \in \mathbb{Z}^{n\ell}$ for every $w \in \{0,1\}^i$, and some $V' \in \mathcal{V}^{(i-1)}$. The function is defined as*

$$V_{\{\mathbf{v}_w\}, V'}(x) := \mathbf{v}_{x_{(i)}}^t \cdot \mathbf{S}_{T^{(i-1)}}(x^{(i)}) + V'(x).$$

  We endow $\mathcal{V}^{(i)}$ with the distribution where the $\mathbf{v}_w \leftarrow \mathcal{P}$ and $V' \leftarrow \mathcal{V}^{(i-1)}$ are all chosen independently and uniformly.

Similarly to the family $\mathcal{E}$ of error functions, the description of a function in $\mathcal{V}^{(i)}$ consists of an exponential (in i) number of $\mathbf{v}_w$ vectors, and can be sampled lazily.

We now define game H_i for $0 \leq i \leq |T|$.

Game H_i. Choose $\mathbf{A}_0, \mathbf{A}_1 \leftarrow \mathbb{Z}_q^{n \times n\ell}$ independently, and lazily sample $G_{\mathbf{s}_w} \leftarrow \mathcal{G}^{(i)}$ and $E_w \leftarrow \mathcal{E}^{(i)}$ for each $w \in \{0,1\}^i$, and $V \leftarrow \mathcal{V}^{(i)}$. Give the adversary $\mathbf{A}_0, \mathbf{A}_1$ and oracle access to the function

$$H(x) := G_{\mathbf{s}_{x_{(i)}}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + V(x). \tag{3.5}$$

Claim. Game H_0 corresponds to the real attack game against the family $\tilde{\mathcal{G}}$, and game $H_{|T|}$ corresponds to oracle access to a uniformly random function.

The first claim follows by definition of $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}_{\mathbf{A}_0,\mathbf{A}_1,T}$, and because $\mathcal{V}^{(0)}$ consists solely of the zero function. For the second claim, for $i = |T|$ we have $x_{(i)} = x$, $x^{(i)} = \varepsilon$, and $T^{(i)} = \varepsilon$ (the empty tree), so by Definitions 3.1, 3.5, and 3.6,

$$H(x) = G_{\mathbf{s}_x}(\varepsilon) + E_x(\varepsilon) + V(x) = \mathbf{s}_x^t \cdot \mathbf{G} + \mathbf{v}_x^t + V'(x).$$

Since $\mathbf{s}_x \in \mathbb{Z}_q^n, \mathbf{v}_x \in \mathcal{P}$ are uniformly random and independent for each x, and $\mathcal{P}$ is a set of representatives of the quotient group $\mathbb{Z}_q^{n\ell}/\mathbf{G}^t \cdot \mathbb{Z}_q^n$, the values $\mathbf{s}_x^t \cdot \mathbf{G} + \mathbf{v}_x^t \in \mathbb{Z}_q^{n\ell}$ are uniformly random and independent. Since V' is independent of these as well, H is a uniformly random function.

It remains to prove that successive games are computationally indistinguishable. To do so we define the following games H'_i for $1 \le i \le |T|$.

Game H'_i. Choose $\mathbf{A}_0, \mathbf{A}_1 \leftarrow \mathbb{Z}_q^{n \times n\ell}$ independently, and lazily sample $\mathbf{u}_w \leftarrow \mathbb{Z}_q^{n\ell}$ and $E_w \leftarrow \mathcal{E}^{(i)}$ for each $w \in \{0,1\}^i$, and $V' \leftarrow \mathcal{V}^{(i-1)}$. Give the adversary $\mathbf{A}_0, \mathbf{A}_1$ and oracle access to the function

$$H'(x) = \mathbf{u}_{x_{(i)}}^t \cdot \mathbf{S}_{T^{(i-1)}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + V(x). \tag{3.6}$$

Claim. For $1 \le i \le |T|$, games H_i and H'_i are equivalent.

We can write each uniformly random $\mathbf{u}_w \in \mathbb{Z}_q^{n\ell}$ for $w \in \{0,1\}^i$ as $\mathbf{u}_w^t = \mathbf{s}_w^t \cdot \mathbf{G}^t + \mathbf{v}_w^t$, where $\mathbf{s}_w \in \mathbb{Z}_q^n$ and $\mathbf{v}_w \in \mathcal{P}$ are uniformly random and independent. Therefore, we can rewrite the function $H'(\cdot)$ from Equation (3.6) as

$$\begin{aligned} H'(x) &= \left(\mathbf{s}_{x_{(i)}}^t \cdot \mathbf{G} + \mathbf{v}_{x_{(i)}}^t\right) \cdot \mathbf{S}_{T^{(i-1)}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + V'(x) \\ &= \mathbf{s}_{x_{(i)}}^t \cdot \mathbf{G} \cdot \mathbf{S}_{T^{(i-1)}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + \left(\mathbf{v}_{x_{(i)}}^t \cdot \mathbf{S}_{T^{(i-1)}}(x^{(i)}) + V'(x)\right) \\ &= G_{\mathbf{s}_{x_{(i)}}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + V(x), \end{aligned}$$

where in the final equality we have used Equation (3.4), and we have defined $V(x)$ to be the second parenthesized component of the previous expression. Notice that all the functions $G_{\mathbf{s}_{x_{(i)}}}$, $E_{x_{(i)}}$, and V are drawn independently from $\mathcal{G}^{(i)}$, $\mathcal{E}^{(i)}$, and $\mathcal{V}^{(i)}$ (respectively), and this proves the claim.

Claim. For $0 \le i \le |T| - 1$, games H_i and H'_{i+1} are computationally indistinguishable under the LWE assumption from the theorem statement.

To prove the claim, we design an efficient simulator $\mathcal{S}$ which receives as input a pair of matrices $(\mathbf{A}, \mathbf{B}^t) \in \mathbb{Z}_q^{n \times 2n\ell} \times \mathbb{Z}_q^{Q \times 2n\ell}$, where $Q = \mathrm{poly}(\lambda)$ is the minimum of 2^i and the number of queries that the adversary makes. The simulator parses $\mathbf{A} = [\mathbf{A}_0 \mid \mathbf{A}_1]$ where $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times n\ell}$ and gives them to the adversary. It lazily samples a $V \leftarrow \mathcal{V}^{(i)}$ and an $E_w \leftarrow \mathcal{E}^{(i+1)}$ for every $w \in \{0,1\}^{i+1}$. Then for each query x from the adversary, if a vector $\mathbf{b}^t_{x_{(i)}}$ has not already been defined, it lets $\mathbf{b}^t_{x_{(i)}}$ be a previously unused row of $\mathbf{B}^t$. It parses $\mathbf{b}^t_{x_{(i)}} = (\mathbf{b}^t_{x_{(i)}\|0} \mid \mathbf{b}^t_{x_{(i)}\|1})$, where $\mathbf{b}_{x_{(i)}\|b} \in \mathbb{Z}_q^{n\ell}$ for each $b \in \{0,1\}$. It then answers the query with the value

$$J(x) := \mathbf{b}^t_{x_{(i+1)}} \cdot \mathbf{S}_{T^{(i)}}(x^{(i+1)}) + E_{x_{(i+1)}}(x^{(i+1)}) + V(x).$$

We now analyze the behavior of $\mathcal{S}$ for the two distributions of $(\mathbf{A}, \mathbf{B}^t)$ from the decision-LWE problem. In both cases, $\mathbf{A}$ is uniformly random and so the public parameters are properly distributed. When $\mathbf{B}$ is uniformly random, it can be seen by inspection that the function J is drawn from the same distribution as the function H' in game H'_{i+1} described in Equation (3.6), so the simulator exactly emulates game H'_{i+1}.

We now analyze the other case, namely, $\mathbf{B}^t = \mathbf{S}^t \cdot \mathbf{A} + \mathbf{E}^t$ for independent $\mathbf{S}^t \leftarrow \mathbb{Z}_q^{Q \times n}$ and $\mathbf{E}^t \leftarrow \chi^{Q \times 2n\ell}$. Then letting $\mathbf{s}^t_{x_{(i)}}, (\mathbf{e}^t_{x_{(i)}\|0} \mid \mathbf{e}^t_{x_{(i)}\|1})$ respectively be the rows of $\mathbf{S}^t, \mathbf{E}^t$ corresponding to the row of $\mathbf{B}^t$ used as $\mathbf{b}^t_{x_{(i)}}$, we have

$$\begin{aligned}
J(x) &= \left(\mathbf{s}^t_{x_{(i)}} \cdot \mathbf{A}_{x_{i+1}} + \mathbf{e}^t_{x_{(i)}\|x_{i+1}}\right) \cdot \mathbf{S}_{T^{(i)}}(x^{(i+1)}) + E_{x_{(i+1)}}(x^{(i+1)}) + V(x) \\
&= \mathbf{s}^t_{x_{(i)}} \cdot \mathbf{A}_{T^{(i)}}(x^{(i)}) + \left(\mathbf{e}^t_{x_{(i)}\|x_{i+1}} \cdot \mathbf{S}_{T^{(i)}}(x^{(i+1)}) + E_{x_{(i)}\|x_{i+1}}(x^{(i+1)})\right) + V(x) \\
&= G_{\mathbf{s}_{x_{(i)}}}(x^{(i)}) + E_{x_{(i)}}(x^{(i)}) + V(x),
\end{aligned}$$

where in the second equality we have used Equation (3.3), and in the last expression we have defined $E_{x_{(i)}}(x^{(i)})$ to be the parenthesized component from the previous expression. Notice that by the distributions of all the variables, the functions $G_{\mathbf{s}_w}$, E_w (for each queried prefix $w \in \{0,1\}^i$) and V are all drawn independently from $\mathcal{G}^{(i)}$, $\mathcal{E}^{(i)}$, and $\mathcal{V}^{(i)}$, so in this case the simulator exactly emulates game H_i.

Because the two LWE input distributions are computationally indistinguishable by assumption and $\mathcal{S}$ is efficient, it follows that H_i and H'_{i+1} are computationally indistinguishable, and the claim is proved.

By repeated application of the claims above, we have that $H_0 \overset{c}{\approx} H'_1 \equiv H_1 \overset{c}{\approx} H'_2 \equiv \cdots \equiv H_{|T|-1} \overset{c}{\approx} H'_{|T|} \equiv H_{|T|}$, and so $H_0 \overset{c}{\approx} H_{|T|}$ by the triangle inequality. This completes the proof of Theorem 3.1.

References

[ACPS09] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618. 2009.

[AKPW13] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs. Learning with rounding, revisited - new reduction, properties and applications. In *CRYPTO*, pages 57–74. 2013.

[BC10] M. Bellare and D. Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In *CRYPTO*, pages 666–684. 2010.

[BGG+14] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. In *EUROCRYPT*. 2014.

[BLMR13] D. Boneh, K. Lewi, H. W. Montgomery, and A. Raghunathan. Key homomorphic PRFs and their applications. In *CRYPTO*, pages 410–428. 2013.

[BLP+13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584. 2013.

[BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737. 2012.

[BV14] Z. Brakerski and V. Vaikuntanathan. Lattice-based FHE as secure as PKE. In *ITCS*, pages 1–?? 2014.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. Preliminary version in FOCS 1984.

[GSW13] C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, pages 75–92. 2013.

[LMR14] K. Lewi, H. W. Montgomery, and A. Raghunathan. Improved constructions of prfs secure against related-key attacks. In *ACNS*, pages 44–61. 2014.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM*, 60(6):43:1–43:35, November 2013. Preliminary version in EUROCRYPT '10.

[LPR13] V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-LWE cryptography. In *EUROCRYPT*, pages 35–54. 2013.

[NPR99] M. Naor, B. Pinkas, and O. Reingold. Distributed pseudo-random functions and KDCs. In *EUROCRYPT*, pages 327–346. 1999.

[NR95] M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999. Preliminary version in FOCS 1995.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. Preliminary version in FOCS 1997.

[NRR00] M. Naor, O. Reingold, and A. Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002. Preliminary version in STOC 2000.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. 2009.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.