CS 294. Pseudorandom Functions from Lattices

Pseudorandom functions (PRF) can in principle be constructed from LWE (and even SIS) completely generically following the Goldreich-Goldwasser-Micali paradigm that constructs PRFs from pseudorandom generators and even one-way functions. However, direct constructions often come equipped with other nice properties such as parallelism, key homomorphism, constrained evaluation, and more.

1 Pseudorandom Generator from LWE

The LWE function

$$G_{\mathbf{A}}(\mathbf{s}, \mathbf{e}) = \mathbf{s}^T \mathbf{A} + \mathbf{e}^T$$

is a pseudorandom generator with two caveats:

- It is a family of PRGs indexed by $\mathbf{A}$. A random function chosen from the family is then a PRG. This is not a big issue usually (except when considering questions related to who picks the $\mathbf{A}$).
- As-is, the domain seems to be $\mathbb{Z}_q^n \times \mathbb{Z}_q^m$ and the range is $\mathbb{Z}_q^m$ so the function does not even seem to expand! However, in reality, the function takes as input a smaller number of random bits used to sample $\mathbf{e}$, roughly $m \log(\alpha q)$ to sample from a Gaussian of standard deviation αq. When this is done, for sufficiently large m, the function does expand, and is pseudorandom.

2 GGM Construction

Goldreich, Goldwasser and Micali show how to construct a pseudorandom function family starting from any pseudorandom generator. This can well be applied to the LWE PRG described above, however it results in a rather unwieldy construction. We show below constructions that are much prettier, and as a side-effect, give us several advantages such as key homomorphism and parallel evaluation (as we will see today) and constrained evaluation (as we will see in later lectures).

3 BLMR13 Construction

3.1 The Gadget Matrix

We need the *gadget matrix* which will make its appearance several times in the next few lectures.

In a nutshell, our gadget matrix $\mathbf{G}$ is an $n \times m$ matrix (where $m \geq n \log q$) with the property that $\mathbf{G} \cdot \{0,1\}^m \supseteq \mathbb{Z}_q^n$. That is, for every vector $\mathbf{v} \in \mathbb{Z}_q^n$, there is a 0-1 vector $\mathbf{w}$ such that $\mathbf{G}\mathbf{w} = \mathbf{v} \pmod{q}$. For example, the matrix $\mathbf{G} \in \mathbb{Z}_7^{2\times 6}$ is the following matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 2 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 4 \end{bmatrix}$$

Indeed for every vector $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, let $v_1 = v_{12}v_{11}v_{10}$ denote its bit representation (and similarly for v_2). Then,

$$\mathbf{G}\begin{bmatrix} v_{10} \\ v_{11} \\ v_{12} \\ v_{20} \\ v_{21} \\ v_{22} \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

More generally, let $\mathbf{g}$ denote the gadget vector $[1\ 2\ 4\ \ldots 2^{\lceil \log_2 q \rceil - 1}] \in \mathbb{Z}_q^{1 \times \lceil \log_2 q \rceil}$. Then, $\mathbf{G} = \mathbf{g} \otimes \mathbf{I}_n$ is the tensor product of $\mathbf{g}$ with the $n \times n$ identity matrix $\mathbf{I}_n$. (If $m > n\lceil \log q \rceil$, pad this with a block of the zero matrix.)

We will denote the inverse mapping by $\mathbf{G}^-$. That is, $\mathbf{G}^-(\mathbf{v}) = \mathbf{w}$ if (a) $\mathbf{w}$ has 0 or 1 entries; and (b) $\mathbf{G}\mathbf{w} = \mathbf{v} \pmod q$. Note that there could be many such $\mathbf{w}$ that satify these properties, so $\mathbf{G}^{-1}$ is best thought of as a multi-valued function.

3.2 Flipped LWE: Small A, Random s

We start with the proof that LWE with roles reversed, namely where the entries of $\mathbf{A}$ are random small, and $\mathbf{s}$ is random, is as secure as LWE. Note that (a) we showed that "Normal Form LWE" where $\mathbf{A}$ is random and $\mathbf{s}$ is random small, is as secure as LWE (in Lecture 1 and lecture 4) and (b) if both $\mathbf{A}$ and $\mathbf{s}$ have small entries, the problem is easy, as it is essentially just linear regression, a convex optimization problem.

Assume that $\mathbf{A} \leftarrow \{0,1\}^{N \times m}$ and $\mathbf{s} \leftarrow \mathbb{Z}_q^N$ are uniformly random. The Flipped LWE problem asks to distinguish between $(\mathbf{A}, \mathbf{s}^T\mathbf{A} + \mathbf{e}^T)$ from a truly random pair from the same domains. Note that $\mathbf{s}$ is likely not uniquely determined, rather only determined up to small additive error, so if one wanted to define the search version, it should be done with some care.

Lemma 1. *Flipped LWE*$(N = n\log q, m, q, \chi)$ *is as hard as LWE*(n, m, q, χ).

Proof. We show a reduction from (decisional) LWE to flipped-LWE. Given an LWE sample $(\mathbf{A}, b = \mathbf{s}^T\mathbf{A} + e)$, we rewrite it as

$$(\mathbf{A}, b = (\mathbf{G}^T\mathbf{s})^T\mathbf{G}^-(\mathbf{A}) + e)$$

pick a random $\mathbf{s}' \in \mathbb{Z}_q^N$ and compute $b' = b + \mathbf{s}'^T\mathbf{G}^-(\mathbf{A})$. Pass $(\mathbf{G}^-(\mathbf{A}), b')$ to the flipped LWE adversary.

First, notice that $\mathbf{G}^-(\mathbf{A})$ is a uniformly random 0-1 matrix – this is true either when q is close to a power of two, or by extending the definition of the $\mathbf{G}$ matrix by adding more powers of two.

Secondly, notice that

$$b' = b + \mathbf{s}'^T\mathbf{G}^-(\mathbf{A}) = (\mathbf{G}^T\mathbf{s} + \mathbf{s}')^T\mathbf{G}^-(\mathbf{A}) + e$$

which is exactly a flipped LWE sample when b is an LWE sample and uniformly random otherwise.

This transforms a flipped-LWE distinguisher into an LWE distinguisher. □

3.3 Construction

Both constructions we show will follow the following general template. The PRF family will be indexed by a secret seed $\mathbf{s} \in \mathbb{Z}_q^n$, and a sequence of public matrices $\vec{\mathbf{A}} = (\mathbf{A}_0, \mathbf{A}_1, \ldots)$. On input $\mathbf{x} \in \{0,1\}^\ell$, the function will be defined as

$$\mathsf{PRF}_{\mathbf{s},\vec{\mathbf{A}}}(x) = \mathbf{s}^T \mathbf{A}_x + \mathbf{e}_x^T \pmod q$$

where $\mathbf{A}_x$ is defined as some function (depending on the construction) of $\vec{\mathbf{A}}$ and $\mathbf{x} \in \{0,1\}^\ell$.

The first problem that one encounters with this framework is where does the error $\mathbf{e}_x^T$, which is supposed to be different and "pseudo-fresh" for every x, come from? The first trick we will play is to sidestep this question entirely, and go via the learning with rounding paradigm of Banerjee, Peikert and Rosen [BPR12]. That is, we will define

$$\mathsf{PRF}_{\mathbf{s},\vec{\mathbf{A}}}(x) = \lfloor \mathbf{s}^T \mathbf{A}_x + \mathbf{e}_x^T \rceil_p \pmod p$$

where $\lfloor \cdot \rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$ refers to a function that, on input $x \in \mathbb{Z}_q$ outputs the multiple of p that is closest to it. That is,

$$\lfloor x \rceil_p = \left\lfloor \frac{p}{q} x \right\rceil$$

where $\lfloor \cdot \rceil$ refers to the function that rounds to the nearest integer.

In the BLMR construction, the public parameters are $\vec{\mathbf{A}} := (\mathbf{A}_0, \mathbf{A}_1)$ where both matrices are drawn at random from $\mathbb{Z}_q^{n \times n}$ and $\mathbf{A}_x$ is defined as a subset product. We are now ready to define the BLMR construction. The construction sets

$$\mathbf{A}_x = \mathbf{G}^-(\mathbf{A}_{x_1}) \cdot \mathbf{G}^-(\mathbf{A}_{x_2}) \ldots \mathbf{G}^-(\mathbf{A}_{x_\ell}) = \prod_{i=1}^{\ell} \mathbf{G}^-(\mathbf{A}_i)$$

and therefore,

$$\mathsf{PRF}_{\mathbf{s},\mathbf{A}_0,\mathbf{A}_1}(x) = \lfloor \mathbf{s}^T \mathbf{A}_x \rceil_p \pmod p$$

The only remaining loose end is how to choose p. Intuitively, the larger the p, the less secure the construction is. Indeed, if $p = q$, there is no rounding and the PRF is a linear function! The smaller the p, the less efficient the construction is, in terms of how many pseudorandom bits it produces per invocation.

Parallelism. The pseudorandom function can be computed in $\log \ell$ levels of matrix multiplication, or in the complexity class NC^2.

(Approximate) Key Homomorphism. The PRF has the attractive feature that $\mathsf{PRF}_{\mathbf{s}}(x) + \mathsf{PRF}_{\mathbf{s}'}(x)$ (where both PRFs use the same two public matrices $\mathbf{A}_0$ and $\mathbf{A}_1$) is approximately equal to $\mathsf{PRF}_{\mathbf{s}+\mathbf{s}'}(x)$. This feature has a number of applications such as constructing a distributed PRF and a (additively) related-key secure PRF.

3.4 Proof of Security

We will, for simplicity, prove that the truth table of the PRF is indistinguishable from i.i.d. random strings using a reduction that runs in time exponential in the input length, namely ℓ. More refined approaches, following the GGM proof, are possible, but omitted from our exposition.

The proof proceeds in a number of hybrids. Define "intermediate" pseudorandom functions $\mathsf{PRF}^{(i)}$ for $i = 0, \ldots, \ell$ as follows.

$$\mathsf{PRF}^{(i)}_{\mathbf{s}_0,\ldots,\mathbf{s}_{2^i-1}}(x'||x'') = \lfloor \mathbf{s}_{x'}^T \mathbf{A}_{x''} \rceil_p$$

where x' is the i-bit prefix of $x = x'||x''$.

Note that $\mathsf{PRF}^{(0)}$ is exactly the PRF we defined with $\mathbf{s}_0 = \mathbf{s}$. On the other hand, $\mathsf{PRF}^{(\ell)}$ is a random function. The proof goes via a hybrid argument that switches from $\mathsf{PRF}^{(0)}$ to $\mathsf{PRF}^{(\ell)}$ in ℓ steps. We will now show that each such switch is computationally indistinguishable to the adversary. For simplicity, we show this for $\mathsf{PRF}^{(0)}$ versus $\mathsf{PRF}^{(1)}$.

- First, consider

$$\mathsf{PRF}^{(0)}_{\mathbf{s}}(x_1 \ldots x_\ell) = \lfloor \mathbf{s}^T \mathbf{A}_x \rceil_p = \lfloor \mathbf{s}^T \mathbf{G}^-(\mathbf{A}_{x_1}) \cdot \prod_{i=2}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i}) \rceil_p$$

- We first show that this distribution is statistically close to

$$\lfloor (\mathbf{s}^T \mathbf{G}^-(\mathbf{A}_{x_1}) + \mathbf{e}_{x_1}) \cdot \prod_{i=2}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i}) \rceil_p$$

  Indeed, the intuition is that the difference between the distributions is only noticeable when the addition of $\mathbf{e}_{x_1} \cdot \prod_{i=2}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})$ flips over one of the coordinates of the vector $\mathbf{s}^T \prod_{i=1}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})$ over a multiple of p. First, notice that since $\prod_{i=1}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})$ is full-rank w.h.p. and $\mathbf{s}$ is uniformly random, so is $\mathbf{s}^T \prod_{i=1}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})$. The probability of flipping over is at most $N \cdot ||\mathbf{e}_{x_1} \cdot \prod_{i=2}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})||_\infty/(q/p)$ which is negligible if $||\mathbf{e}_{x_1}||_\infty \ll q/p \cdot 1/N^{\ell+1} \cdot 2^{-\omega(\log \lambda)}$. Assume that $p = \Omega(q)$, this is like assuming LWE with noise-to-modulus ratio that is roughly N^ℓ. In turn, this translates to assuming that gapSVP is hard to approximate to within N^ℓ, a factor exponential in the input length of the PRF.

- Next, observe that this is computationally indistinguishable from

$$\mathbf{s}_{x_1}^T \prod_{i=2}^{\ell} \mathbf{G}^-(\mathbf{A}_{x_i})$$

  by LWE. Finally, this distribution is precisely $\mathsf{PRF}^{(1)}$.

4 BP14 Construction

The only difference between the BLMR13 and BP14 constructions is in the definition of $\mathbf{A}_x$. Let $x = x_1 x_2 \ldots x_\ell$. BP14 defines $\mathbf{A}_x$ recursively as follows. $\mathbf{A}_\varepsilon = \mathbf{I}_{m\times m}$ (where ε is the empty string) and

$$\mathbf{A}_{bx} = \mathbf{G}^-(\mathbf{A}_b \cdot \mathbf{A}_x)$$

Thus,

$$\mathbf{A}_x = \mathbf{G}^{-}(\mathbf{A}_{x_1} \cdot \mathbf{G}^{-}(\mathbf{A}_{x_2} \dots \mathbf{G}^{-}(\mathbf{A}_{x_\ell})))$$

This allows us to base security on LWE with slightly superpolynomial noise-to-modulus ratio.

Roughly speaking, we will switch from

$$\lfloor \mathbf{s}^T \mathbf{G} \mathbf{A}_x \rceil_p = \lfloor \mathbf{s}^T \mathbf{A}_{x_1} \cdot \mathbf{A}_{x_{2\dots\ell}} \rceil_p$$

to

$$\lfloor (\mathbf{s}^T \mathbf{A}_{x_1} + \mathbf{e}_{x_1}) \cdot \mathbf{A}_{x_{2\dots\ell}} \rceil_p$$

by a statistical argument similar to the above. However, now, the norm of $\mathbf{A}_{x_{2\dots\ell}}$ is polynomial in N, independent of ℓ which makes the argument considerably more efficient. We still will need the $2^{-\omega(\log\lambda)}$ term for the statistical argument.

Note that this construction loses parallelism.

Open Problem 5.1**.** Construct an LWE-based pseudorandom function that can be computed in NC1 and is based on LWE with polynomial modulus.

The computation in NC1 is satisfied by the BLMR construction (and by a construction of [BPR12] using "synthesizers"), and the polynomial modulus is satisfied by the a direct construction based on GGM (also in [BPR12]). We refer to [Kim20] for a detailed taxonomy of the existing PRF constructions as of Feb 2020.

Open Problem 5.2**.** Come up with a "direct" construction of a SIS-based PRG and PRF.

Of course, SIS gives us a one-way function (as described below) and can be used to construct a PRG by the result of Hastad-Impagliazzo-Levin-Luby and then a PRF by Goldreich-Goldwasser-Micali. But the resulting construction is very complex, and in particular, does not have the parallel evaluation property. A concrete question is to construct a PRF from SIS with parallel evaluation.

4.1 Collision-Resistant Hashing

We finish by describing a simple collision-resistant hash function based on SIS.

A collision resistant hashing scheme $\mathcal{H}$ consists of an ensemble of hash functions $\{\mathcal{H}_n\}_{n\in\mathbb{N}}$ where each $\mathcal{H}_n$ consists of a collection of functions that map n bits to $m < n$ bits. So, each hash function compresses its input, and by pigeonhole principle, it has collisions. That is, inputs $x \neq y$ such that $h(x) = h(y)$. Collision-resistance requires that every p.p.t. adversary who gets a hash function $h \leftarrow \mathcal{H}_n$ chosen at random fails to find a collision except with negligible probability.

Collision-Resistant Hashing from SIS. Here is a hash family $\mathcal{H}_n$ that is secure under $\mathsf{SIS}(n, m, q, B)$ where $n \log q > m \log(B+1)$. Each hash function $h_{\mathbf{A}}$ is parameterized by a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, takes as input $\mathbf{e} \in [0, \dots, B]^m$ and outputs

$$h_{\mathbf{A}}(\mathbf{e}) = \mathbf{A}\mathbf{e} \bmod q$$

A collision gives us $\mathbf{e}, \mathbf{e}' \in [0, \dots, B]^m$ where $\mathbf{A}\mathbf{e} = \mathbf{A}\mathbf{e}' \bmod q$ which in turn says that $\mathbf{A}(\mathbf{e} - \mathbf{e}') = 0 \bmod q$. Since each entry of $\mathbf{e} - \mathbf{e}'$ is in $[-B, \dots, B]$, this gives us a solution to $\mathsf{SIS}(n, m, q, B)$.

References

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EURO-CRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737. Springer, 2012.

[Kim20] Sam Kim. Key-homomorphic pseudorandom functions from lwe with a small modulus. Cryptology ePrint Archive, Report 2020/233, 2020. https://eprint.iacr.org/2020/233.