JLS Construction of IO

In this lecture, we will see a construction of "slow XIO" (more accurately, slow, weakly succinct functional encryption) due to the breakthrough work of Jain, Lin, and Sahai [JLS21, JLS22]. This construction can be directly modified to get "fast XIO", and thus IO (as seen in last lecture), but this gets quite technical. We present the core ideas in [JLS22] today without getting needlessly detailed.

At a high level, the construction has three steps:

1. Starting point: degree-2 functional encryption ($\mathsf{D^2FE}$) over $\mathbb{Z}_p$, under bilinear maps assumptions. Today, we will assume this without proof.
2. Boosting degree-2 functional encryption to *local*, *slow* functional encryption (LSFE) (or NC^0 slow functional encryption), assuming (large field) LPN.
3. Boosting local slow functional encryption to slow functional encryption for all circuits (SFE), assuming PRGs in NC^0.

In the previous lecture, we saw how to bootstrap fast XIO into IO, and in the next lecture, we will see how to construct $\mathsf{D^2FE}$.

1 Starting Point: Degree-2 Functional Encryption

Thankfully, bilinear maps assumptions imply functional encryption for degree-2 computations over $\mathbb{Z}_p$ ($\mathsf{D^2FE}$), where p is a prime. In fact, something more is true, which allows degree-$O(1)$ computations in some "public" input PI, which does *not* need to be hidden, and degree-2 computations in some "secret" input SI, which is hidden as in the usual definition of functional encryption [Wee20].

One might ask: what do degree-2 computations (over $\mathbb{Z}_p$) really buy us? Consider the following example.

Example. Let $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_p^{\ell \times \ell}$ be matrices. For all $(i, j) \in [\ell]^2$, the function $\mathsf{mult}_{i,j} : \mathbb{Z}_p^{2\ell^2} \to \mathbb{Z}_p$ given by

$$\mathsf{mult}_{i,j}(\mathbf{A}, \mathbf{B}) = (\mathbf{A} \cdot \mathbf{B})_{i,j}$$

is degree-2 in its input, where $\cdot$ denotes matrix multiplication over $\mathbb{Z}_p$.

1.1 Definition of $\mathsf{D^2FE}$

Definition 1 (Degree-2 Functional Encryption)**.** *A* degree-2 functional encryption *scheme* ($\mathsf{D^2FE}$) *consists of three algorithms:*

- $\mathsf{D^2FE.Setup}(1^\lambda, f : \mathbb{Z}_p^{|\mathsf{PI}|} \times \mathbb{Z}_p^{|\mathsf{SI}|} \to \mathbb{Z}_p^m)$*: Outputs a master public key* mpk *and a functional secret key* sk_f*, which is assumed to include a description of* f *without loss of generality. We require that each output coordinate of the function* f *has total degree* $d = O(1)$ *in* PI *and total degree* 2 *in* SI*.*
- $\mathsf{D^2FE.Enc}(\mathsf{mpk}, \mathsf{PI}, \mathsf{SI})$*: Outputs a ciphertext* ct*.*
- $\mathsf{D^2FE.Dec}(\mathsf{sk}_f, \mathsf{ct})$*: Outputs a value in* $\mathbb{Z}_p^m$*.*

We require a few properties of these algorithms.

***Correctness**: For any* PI, SI*, and any function* $f : \mathbb{Z}_p^{|\mathsf{PI}|} \times \mathbb{Z}_p^{|\mathsf{SI}|} \to \mathbb{Z}_p^m$*, as long as* $f(\mathsf{PI}, \mathsf{SI}) \in \{0,1\}^m$*, then for* $(\mathsf{mpk}, \mathsf{sk}_f) \leftarrow \mathsf{D}^2\mathsf{FE.Setup}(1^\lambda, f)$*,*

$$\Pr\left[\mathsf{D}^2\mathsf{FE.Dec}(\mathsf{sk}_f, \mathsf{D}^2\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{PI}, \mathsf{SI})) = f(\mathsf{PI}, \mathsf{SI})\right] = 1.$$

***Linear Efficiency**: The size of the circuit* $\mathsf{D}^2\mathsf{FE.Enc}(\mathsf{mpk}, \cdot, \cdot)$ *is* $(|\mathsf{PI}| + |\mathsf{SI}|) \cdot \mathrm{poly}(\lambda)$*. In particular, for* $\mathsf{ct} \leftarrow \mathsf{D}^2\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{PI}, \mathsf{SI})$*, we have* $|\mathsf{ct}| \le (|\mathsf{PI}| + |\mathsf{SI}|) \cdot \mathrm{poly}(\lambda)$*.*

***Security**: There is a constant* $c > 0$ *and a p.p.t. simulator* S *such that we have the following indistingiushability, with advantage at most* $2^{-\Omega(\lambda^c)}$ *to all p.p.t. adversaries: for all* PI, SI *and all* $f : \mathbb{Z}_p^{|\mathsf{PI}|} \times \mathbb{Z}_p^{|\mathsf{SI}|} \to \mathbb{Z}_p^m$ *of degree* $(O(1), 2)$*,*

$$(\mathsf{mpk}, \mathsf{sk}_f, \mathsf{PI}, \mathsf{D}^2\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{PI}, \mathsf{SI})) \approx_c S(f, \mathsf{PI}, f(\mathsf{PI}, \mathsf{SI})),$$

where $(\mathsf{mpk}, \mathsf{sk}_f) \leftarrow \mathsf{D}^2\mathsf{FE.Setup}(1^\lambda, f)$*.*

Throughout this whole lecture, we will assume that all input and output lengths of the various functional encryption schemes are upper- and lower-bounded by a polynomial in λ.

2 From $\mathsf{D}^2\mathsf{FE}$ to LSFE

The key assumption in this transformation is Learning Parity with Noise (LPN).

2.1 Learning Parity with Noise (LPN)

LPN is very similar to LWE; the only difference is the distribution of the noise. In LWE, the noise is small (e.g., in ℓ_2 or ℓ_∞ norm), but here, the noise is *sparse* (with high probability), i.e., small in the ℓ_0 "norm."[1] This sparsity will be critical in the construction.

Let $p \in \mathbb{N}$. Let $\mathsf{Bern}_p(\eta)$, supported over $\mathbb{Z}_p$, denote the distribution that is 0 with probability $1 - \eta$, and uniformly random over $\mathbb{Z}_p \setminus \{0\}$ with probability η. Using this notation, we define the learning parity with noise (LPN) assumption over large fields as follows.

Assumption 2 (Learning Parity with Noise over large fields)**.** *There exists a constant* $\delta > 0$ *such that for all* $c_1, c_2 > 0$*, where* $n = n(\ell) = \ell^{c_1}$ *and prime* $p \approx 2^{\ell^{c_2}}$*, we have the following indistinguishability. For* $\mathbf{A} \leftarrow \mathbb{Z}_p^{n \times \ell}$*,* $\mathbf{s} \leftarrow \mathbb{Z}_p^\ell$*,* $\mathbf{e} \leftarrow \mathsf{Bern}_p(\ell^{-\delta})^n$*, and* $\mathbf{b} \leftarrow \mathbb{Z}_p^n$*,*

$$(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{b}),$$

where addition is done over $\mathbb{Z}_p$*. The* sub-exponential *version of this assumption holds if the above indistinguishability means that all non-uniform p.p.t. distinguishers have at most sub-exponential advantage.*

2.2 Algebraic Compression

We now show a general way to compress sparse errors, in such a way that expanding the errors back is a degree-2 operation. This is arguably the main trick in the [JLS21, JLS22] construction. We say a vector $z \in \mathbb{Z}_p^m$ is k-sparse if it has at most k non-zero entries.

[1] The ℓ_0 "norm" is not truly a norm, as it does not satisfy homogeneity.

Lemma 3. *Let $\tau \in (0,1)$ be an arbitrary constant. There are efficient algorithms* $\mathsf{Compress} : \mathbb{Z}_p^m \to \mathbb{Z}_p^{m^{1-\tau/5}}$ *and* $\mathsf{Expand} : \mathbb{Z}_p^{m^{1-\tau/5}} \to \mathbb{Z}_p^m$ *(using shared randomness) with the following properties:*

1. *For all $m^{1-\tau}$-sparse $z \in \mathbb{Z}_p^m$, $\mathsf{Expand}(\mathsf{Compress}(z)) = z$ (with all but sub-exponentially small failure probability);*
2. *The function* $\mathsf{Expand} : \mathbb{Z}_p^{m^{1-\tau/5}} \to \mathbb{Z}_p^m$ *is degree-2 over $\mathbb{Z}_p$.*

Proof. We start with a "warm-up" version where $\tau = 1/2 + \alpha$ for some $\alpha > 0$. By breaking up z into chunks, can interpret $z \in \mathbb{Z}_p^m$ as a matrix $\mathsf{mat}(z) \in \mathbb{Z}_p^{\sqrt{m} \times \sqrt{m}}$. Since $\tau = 1/2 + \alpha$, we know that any input z has sparsity $m^{1-\tau} = m^{1/2-\alpha}$. In particular, since $\mathsf{mat}(z)$ has at most $m^{1/2-\alpha}$ non-zero entries, $\mathsf{rank}(\mathsf{mat}(z)) \leq m^{1/2-\alpha} \ll \sqrt{m}$. By standard linear algebra facts, this means we have the matrix decomposition

$$\mathsf{mat}(z) = UV, \quad U \in \mathbb{Z}_p^{\sqrt{m} \times m^{1/2-\alpha}}, \; V \in \mathbb{Z}_p^{m^{1/2-\alpha} \times \sqrt{m}}.$$

This directly leads to Compress and Expand algorithms:

$$\begin{aligned} &\mathsf{Compress}(z) = (U, V), \\ &\mathsf{Expand}(U, V) = \mathsf{vec}(U \cdot V), \end{aligned}$$

where $\cdot$ is matrix multiplication and vec() denotes interpreting a matrix as a list of entries. We now verify all of the properties. Item 1 is clear by construction. For the compression size, we have

$$|\mathsf{Compress}(z)| = |U| + |V| = 2\sqrt{m} \cdot m^{1/2-\alpha} = 2m^{1-\alpha} = m^{1-\Omega(1)}.$$

For Item 2, note that Expand() is just multiplying two input matrices. In particular, each entry is degree-2 in the input.

Now we do the general case. Instead of directly interpreting $z \in \mathbb{Z}_p^m$ as one matrix, we randomly split up z into *many* matrices. Specifically, we randomly partition $[m]$ into $m^{1-\tau}$ sets, each of size m^τ. By appropriate Chernoff bounds (and a union bound), one can show that with all but sub-exponentially small probability, all sets will have at most $m^{\tau/5}$ non-zero entries of z in them. Then, for each of these sets, we can apply our "warm up" version since $\tau/5 < \tau/2$, and concatenate all the results together. A bit more explicitly, if we split up z into $z^{(j)} \in \mathbb{Z}_p^{m^\tau}$ for $j \in [m^{1-\tau}]$, we have

$$\begin{aligned} &\mathsf{Compress}(z) = \left(\left\{ \left(U^{(j)}, V^{(j)} \right) \right\}_{j \in [m^{1-\tau}]} \right), \\ &\mathsf{Expand}\left(\left\{ \left(U^{(j)}, V^{(j)} \right) \right\}_{j \in [m^{1-\tau}]} \right) = \mathsf{vec}\left(\left\{ U^{(j)} \cdot V^{(j)} \right\}_{j \in [m^{1-\tau}]} \right), \end{aligned}$$

where the grouping into random sets is implicit in the notation. For all $j \in [m^{1-\tau}]$, $U^{(j)} \in \mathbb{Z}_p^{m^{\tau/2} \times m^{\tau/5}}$ and $V^{(j)} \in \mathbb{Z}_p^{m^{\tau/5} \times m^{\tau/2}}$. As before, Item 1 is clear by construction. To see the output size, we have

$$|\mathsf{Compress}(z)| = m^{1-\tau} \left(\left| U^{(j)} \right| + \left| V^{(j)} \right| \right) = 2 \cdot m^{1-\tau} \cdot m^{\tau/2} \cdot m^{\tau/5} = 2m^{1-3\tau/10} \leq m^{1-\tau/5},$$

for sufficiently large m. Lastly, to see Item 2, Expand() is once again just matrix multiplication of input matrices, so it has degree 2. □

2.3 Preliminaries on NC^0

The complexity class NC^0 corresponds to problems solvable by $O(1)$-depth circuits using gates with fan-in at most 2. Since a depth bound of d implies a locality bound of 2^d, $O(1)$ depth and $O(1)$ locality are equivalent. Thus, we say a function (family) $G_n : \{0,1\}^n \to \{0,1\}^m$ is in NC^0 if there exists a constant $\ell = O(1)$ such that every output bit of G_n is a function of at most ℓ input bits.

One fact we will use about constant-locality Boolean functions is that they can be easily *arithmetized* into constant-degree functions over $\mathbb{Z}_p$ for any p. To see this, suppose $f : \{0,1\}^n \to \{0,1\}$ depends only on d coordinates, say, without loss of generality, the first d coordinates. That is, there is some $g : \{0,1\}^d \to \{0,1\}$ such that

$$f(x_1, \cdots, x_n) = g(x_1, \cdots, x_d).$$

We can write this as a multilinear function, which, in particular, has total degree at most d:

$$f(x_1, \cdots, x_n) = g(x_1, \cdots, x_d) = \sum_{y \in \{0,1\}^d} g(y) \mathbb{1}[x = y] = \sum_{y \in \{0,1\}^d} g(y) \left(\prod_{i \,:\, y_i = 1} x_i \right) \left(\prod_{i \,:\, y_i = 0} (1 - x_i) \right).$$

That is, there is a function $\hat{f} : \mathbb{Z}_p^n \to \mathbb{Z}_p$ such that:

- $\hat{f}$ is a polynomial of degree d over $\mathbb{Z}_p^n$,
- $\hat{f}$ is d-local, in the sense that $\hat{f}$ depends only on d input coordinates, and
- $\hat{f}(x) = f(x)$ for all $x \in \{0,1\}^n \subseteq \mathbb{Z}_p^n$.

For a function $f : \{0,1\}^n \to \{0,1\}$ of locality d, we denote $\hat{f} : \mathbb{Z}_p^n \to \mathbb{Z}_p$ to be its multilinear arithmetization over $\mathbb{Z}_p^n$ of degree d. More generally, we extend this notation to $f : \{0,1\}^n \to \{0,1\}^m$ of locality d, where we denote $\hat{f} : \mathbb{Z}_p^n \to \mathbb{Z}_p^m$ to be its multilinear arithmetization applied to each output coordinate.

2.4 Definition of LSFE

Definition 4 (Local Slow Functional Encryption)**.** *A* local slow functional encryption *scheme* (LSFE) *consists of three algorithms:*

- $\mathsf{LSFE.Setup}(1^\lambda, f : \{0,1\}^n \to \{0,1\}^m)$*: Outputs a master public key* mpk *and a functional secret key* sk_f*, which is assumed to include a description of* f *without loss of generality. We require that* f *has locality* $d = O(1)$*.*
- $\mathsf{LSFE.Enc}(\mathsf{mpk}, x \in \{0,1\}^n)$*: Outputs a ciphertext* ct*.*
- $\mathsf{LSFE.Dec}(\mathsf{sk}_f, \mathsf{ct})$*: Outputs a value in* $\{0,1\}^m$*.*

We require a few properties of these algorithms.

Correctness*: For any* $x \in \{0,1\}^n$ *and any* d*-local function* $f : \{0,1\}^n \to \{0,1\}^m$*, and* $(\mathsf{mpk}, \mathsf{sk}_f) \leftarrow \mathsf{LSFE.Setup}(1^\lambda, f)$*,*

$$\Pr\left[\mathsf{LSFE.Dec}(\mathsf{sk}_f, \mathsf{LSFE.Enc}(\mathsf{mpk}, x)) = f(x)\right] = 1 - \mathrm{negl}(\lambda).$$

***Weak Succinctness**: There exists some $\epsilon > 0$ such that for all $m \geq n^2$, for $\mathsf{ct} \leftarrow \mathsf{LSFE.Enc}(\mathsf{mpk}, x)$, we have $|\mathsf{ct}| \leq m^{1-\epsilon} \cdot \mathrm{poly}(\lambda)$.*

***Security**: There is a constant $c > 0$ and a p.p.t. simulator S such that we have the following indistingiushability, with advantage at most $2^{-\Omega(\lambda^c)}$ to all p.p.t. adversaries: for all x and all d-local $f : \{0,1\}^n \to \{0,1\}^m$,*

$$(\mathsf{mpk}, \mathsf{sk}_f, \mathsf{LSFE.Enc}(\mathsf{mpk}, x)) \approx_c S(f, f(x)),$$

where $(\mathsf{mpk}, \mathsf{sk}_f) \leftarrow \mathsf{LSFE.Setup}(1^\lambda, f)$.

The main differences between this definition (LSFE) and the previous one ($\mathsf{D}^2\mathsf{FE}$) is that here we allow $O(1)$-locality functions on (secret) inputs, while in $\mathsf{D}^2\mathsf{FE}$, we allow degree-2 functions on secret inputs (SI) and $O(1)$-degree functions on *public* inputs (PI).

2.5 Construction of LSFE

Now, we give a construction of LSFE, assuming a $\mathsf{D}^2\mathsf{FE}$ scheme and the hardness of (large field) LPN.

$\mathsf{LSFE.Enc}(\mathsf{mpk}, x \in \{0,1\}^n)$:

- Let $\ell = m^{1/d}$.
- Sample $\mathbf{A} \leftarrow \mathbb{Z}_p^{n \times \ell}$, $\mathbf{s} \leftarrow \mathbb{Z}_p^\ell$, $\mathbf{e} \leftarrow \mathsf{Bern}_p(\ell^{-\delta})^n$.
- Let $\mathsf{PI} = (\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} + x \pmod p)$, where we interpret $x \in \{0,1\}^n \subseteq \mathbb{Z}_p^n$.
- Parse f from mpk. Let $\mathsf{errs} = \mathsf{Compress}(\hat{f}(x + \mathbf{e}) - \hat{f}(x))$.[a]
- Let $\mathsf{SI} = \left((\mathbf{s}, 1)^{\otimes d/2}, \mathsf{errs}\right)$.
- Output $\mathsf{ct} \leftarrow \mathsf{D}^2\mathsf{FE.Enc}(\mathsf{mpk}, \mathsf{PI}, \mathsf{SI})$.

$\mathsf{LSFE.Setup}(1^\lambda, f : \{0,1\}^n \to \{0,1\}^m)$:

- Let $g = g(\mathsf{PI}, \mathsf{SI})$ be a function defined as follows:
  - Parse $\mathsf{PI} = (\mathbf{A}, \mathbf{b})$, $\mathsf{SI} = \left((\mathbf{s}, 1)^{\otimes d/2}, \mathsf{errs}\right)$.
  - Output $\hat{f}(\mathbf{b} - \mathbf{A}\mathbf{s}) - \mathsf{Expand}(\mathsf{errs})$.
- $(\mathsf{mpk}, \mathsf{sk}_g) \leftarrow \mathsf{D}^2\mathsf{FE.Setup}(1^\lambda, g)$.
- Output $(\mathsf{mpk}, \mathsf{sk}_f := \mathsf{sk}_g)$.

$\mathsf{LSFE.Dec}(\mathsf{sk}_f, \mathsf{ct})$:

- Output $\mathsf{D}^2\mathsf{FE.Dec}(\mathsf{sk}_f, \mathsf{ct})$.

[a] There is a subtlety here that Compress and Expand are randomized and need the same shared random string. To simplify notation, we omit this, but since this random string does not need to be hidden, we can put this randomness in mpk.

Figure 1: Construction of LSFE from $\mathsf{D}^2\mathsf{FE}$ and (large-field) LPN.

Security. Exercise.

Correctness. Let $x \in \{0,1\}^n$ and let $f : \{0,1\}^n \to \{0,1\}^m$ be a d-local function. Assuming $\hat{f}(x+\mathbf{e}) - \hat{f}(x)$ is sufficiently sparse, we know

$$\begin{aligned} g(\mathsf{PI}, \mathsf{SI}) &= \hat{f}(\mathbf{b} - \mathbf{A}\mathbf{s}) - \mathsf{Expand}(\mathsf{errs}) \\ &= \hat{f}(x+\mathbf{e}) - \mathsf{Expand}(\mathsf{Compress}(\hat{f}(x+\mathbf{e}) - \hat{f}(x)) \\ &= \hat{f}(x+\mathbf{e}) - (\hat{f}(x+\mathbf{e}) - \hat{f}(x)) \\ &= \hat{f}(x) \\ &= f(x), \end{aligned}$$

where the last equality holds since $x \in \{0,1\}^n$. It remains to show that $\hat{f}(x+\mathbf{e}) - \hat{f}(x)$ is sparse, and that g is degree $O(1)$ in PI and degree 2 in SI (so that we can apply correctness of $\mathsf{D}^2\mathsf{FE}$).

To see that $\hat{f}(x+\mathbf{e}) - \hat{f}(x)$ is sparse, for $j \in [m]$, let $\hat{f}_j$ denote the jth output coordinate of $\hat{f}$, and for $i \in [n]$, let $e_i \in \mathbb{Z}_p$ denote the ith coordinate of $\mathbf{e}$. If $\hat{f}_j(x+\mathbf{e}) - \hat{f}_j(x) \neq 0$, it must be the case that one of the $d = O(1)$ input coordinates $i \in [n]$ touched by $\hat{f}_j$ must have $e_i \neq 0$. For a fixed $j \in [m]$, by a union bound, the probability of this event is at most $d \cdot \ell^{-\delta} = O(\ell^{-\delta})$. Therefore, by linearity of expectation, $\hat{f}(x+\mathbf{e}) - \hat{f}(x)$ has $O(m\ell^{-\delta})$ non-zero entries in expectation. Since $\ell = m^{1/d}$, this becomes $O(m \cdot m^{-\delta/d}) = O(m^{1-\delta/d})$ coordinates, so we can set $\tau = \delta/(2d)$ for our compression algorithm.[2]

Lastly, we see why g is degree-$O(1)$ in PI and degree-2 in SI. Since $\hat{f}$ has degree $O(1)$, it is clear that $\hat{f}(\mathbf{b} - \mathbf{A}\mathbf{s})$ has degree $O(1)$ in $\mathbf{b}$, $\mathbf{A}$, and $\mathbf{s}$, which is degree $O(1)$ in $\mathbf{b}$ and $\mathbf{A}$, and degree 2 in $(\mathbf{s}, 1)^{\otimes d/2}$. Therefore, $\hat{f}(\mathbf{b} - \mathbf{A}\mathbf{s})$ is degree-$O(1)$ in PI and degree-2 in SI. For the second term, $\mathsf{Expand}(\mathsf{errs})$ has degree 2 as we saw earlier, so $\mathsf{Expand}(\mathsf{errs})$ is degree-2 in SI (and trivially degree $O(1)$ in PI). Therefore, $g(\mathsf{PI}, \mathsf{SI}) = \hat{f}(\mathbf{b} - \mathbf{A}\mathbf{s}) - \mathsf{Expand}(\mathsf{errs})$ is degree-$O(1)$ in PI and degree-2 in SI, as desired.

Weak Succinctness: We first bound $|\mathsf{PI}|$. Note that PI can be written as an element of $\mathbb{Z}_p^{n \times (\ell+1)}$. Therefore, since $m \geq n^2$,

$$|\mathsf{PI}| = n(\ell+1) = O(nm^{1/d}) \leq O(nm^{1/3}) \leq O(m^{5/6}),$$

since we can assume $d \geq 3$ without loss of generality. For SI, we have

$$\begin{aligned} |\mathsf{SI}| &= \left|(\mathbf{s}, 1)^{\otimes d/2}\right| + |\mathsf{Compress}(\mathsf{errs})| \\ &\leq (\ell+1)^{d/2} + m^{1-\tau/5} \\ &\leq O(\ell^{d/2}) + m^{1-\delta/(10d)} \\ &\leq O(\sqrt{m}) + m^{1-\delta/(10d)} \\ &\leq O(m^{1-\delta/(10d)}). \end{aligned}$$

Finally, applying linear efficiency of $\mathsf{D}^2\mathsf{FE}$, we see that for $\mathsf{ct} \leftarrow \mathsf{LSFE.Enc}(\mathsf{mpk}, x)$, we have

$$|\mathsf{ct}| \leq (|\mathsf{PI}| + |\mathsf{SI}|) \cdot \mathrm{poly}(\lambda) \leq (m^{5/6} + m^{1-\delta/(10d)}) \cdot \mathrm{poly}(\lambda) \leq m^{1-\delta/(10d)} \cdot \mathrm{poly}(\lambda),$$

as desired.

[2]Note: This is *not* a "with high probability" statement, as the outputs of $\hat{f}$ may be arbitrarily correlated. Getting around this is an annoying technicality, which [JLS22] does by splitting up the computation and amortizing in the right way.

3 From LSFE to SFE

3.1 PRGs in NC^0

The key assumption in this transformation will be the existence of (polynomial-stretch) PRGs in NC^0, defined as follows.

Assumption 5 (PRGs in NC^0)**.** *There exists a constant $\gamma > 0$ and a uniformly efficiently generatable function (family) $G_m : \{0,1\}^{m^{1-\gamma}} \to \{0,1\}^m$ such that*

- *G_m is in NC^0; and*
- *G_m is a pseudorandom generator* (PRG)*, in the sense that for all non-uniform p.p.t. distinguishers D,*

$$\left| \Pr_{x \leftarrow \{0,1\}^{m^{1-\gamma}}}[D(G_m(x)) = 1] - \Pr_{y \leftarrow \{0,1\}^m}[D(y) = 1] \right| = \mathrm{negl}(m).$$

We say that the sub-exponential *version of this assumption holds if the negligible function above is sub-exponential in m.*

This assumption, as stated, is simplified. The candidate constructions instantiating this assumption sample a description of the family G_m, and only expect the (possibly sub-exponential) indistinguishability to hold with probability $1 - o(1)$ probability over the sampling algorithm. For the ultimate IO construction, one can do some work and use functional encryption combiners to get around this issue, boosting this $1 - o(1)$ probability to all but sub-exponential probability, which is sufficient. For simplicity, here, we assume the function family G_m is fixed and uniformly efficiently generatable.

Moreover, the polynomial stretch can be amplified to an arbitrary polynomial by composing G with itself many times. That is, we can use the following equivalent formulation of the assumption.

Assumption 6 (PRGs in NC^0)**.** *For all $\gamma > 0$, there exists a uniformly efficiently generatable function (family) $G_m : \{0,1\}^{m^\gamma} \to \{0,1\}^m$ such that*

- *G_m is in NC^0 (where the locality can depend on γ); and*
- *G_m is a pseudorandom generator* (PRG)*, in the sense that for all non-uniform p.p.t. distinguishers D,*

$$\left| \Pr_{x \leftarrow \{0,1\}^{m^\gamma}}[D(G_m(x)) = 1] - \Pr_{y \leftarrow \{0,1\}^m}[D(y) = 1] \right| = \mathrm{negl}(m).$$

We say that the sub-exponential *version of this assumption holds if the negligible function above is sub-exponential in m.*

We will use this form of the assumption, with $\gamma = 1/3$, i.e., so the PRG is length-cubing.

3.2 Randomized Encodings

Definition 7. *A* local randomized encoding (LRE) *scheme consists of two algorithms:*

- $\mathsf{LRE.Encode}(1^\lambda, C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*, x \in \{0,1\}^{\tilde{n}}; r)$*: Outputs some randomized encoding π.*

- $\mathsf{LRE.Decode}(\pi)$: *Outputs a value* $y \in \{0,1\}^*$.

We require a few properties of these algorithms.

Correctness*: For all* $x \in \{0,1\}^{\tilde{n}}$ *and all circuits* $C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*$,

$$\Pr_r \left[\mathsf{LRE.Decode}(\mathsf{LRE.Encode}(1^\lambda, C, x; r)) = C(x)\right] = 1.$$

Security*: There is a constant* $c > 0$ *and a p.p.t. simulator* S *such that we have the following indistinguishability, with advantage at most* $2^{-\Omega(\lambda^c)}$ *to all p.p.t. adversaries: for all* $x \in \{0,1\}^{\tilde{n}}$ *and all circuits* $C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*$,

$$\mathsf{LRE.Encode}(1^\lambda, C, x; r) \approx_c S(C, C(x)).$$

Locality*: The function* $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ *is* $O(1)$*-local in* x *and* r.

Efficiency*: The output length of* $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ *and its randomness complexity (i.e.,* $|r|$*) can both be upper-bounded by* $|C| \cdot \mathrm{poly}(\lambda)$.

Below, we will show the following fact.

Theorem 8. *Assume the existence of (linear-stretch)* PRG*s in* NC^0*. Then, there exists a local randomized encoding scheme.*

3.2.1 Construction of LRE Using Garbled Circuits

In the previous class, we saw a construction of randomized encodings using Yao's garbled circuits with "double" encryption. Here, we will give a slightly different construction of garbled circuits, called "point-and-permute", using PRGs directly and a GGM-style construction instead of secret-key encryption. One benefit of this construction is that it satisfies *perfect correctness*. For $G : \{0,1\}^\lambda \to \{0,1\}^{2\lambda+2}$, let $G_0, G_1 : \{0,1\}^\lambda \to \{0,1\}^{\lambda+1}$ be defined as the prefix and suffix of G, i.e., $G(\sigma) = (G_0(\sigma), G_1(\sigma))$.

Encoding. For a circuit $C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*$ and input $x \in \{0,1\}^{\tilde{n}}$, we let $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ be the following algorithm, where r denotes a random string. For each wire w in the circuit C, we will take two seeds $\sigma_w^0, \sigma_w^1 \in \{0,1\}^\lambda$ and one *permutation bit* $b_w \in \{0,1\}$ from the random string r. For each NAND gate g in the circuit, consisting of input wires i, j and output wire k, the table T_g for gate g will be an ordered list of these four values:

$$T_g := \left\{ G_z\left(\sigma_i^{y \oplus b_i}\right) \oplus G_y\left(\sigma_j^{z \oplus b_j}\right) \oplus \left(\sigma_k^{\mathsf{NAND}(y \oplus b_i, z \oplus b_j)}, \mathsf{NAND}(y \oplus b_i, z \oplus b_j) \oplus b_k\right) \right\}_{y,z \in \{0,1\}}.$$

While this expression may look ominous, the point of this table is as follows.

- Let v_i, v_j, v_k be the true binary values of wires i, j, k, respectively (for some implicit input x). This table preserves the invariant that the $(v_i \oplus b_i, v_j \oplus b_j)$th entry in the list "encrypts" the string $(\sigma_k^{v_k}, v_k \oplus b_k)$ under the "keys" $\sigma_i^{v_i}, \sigma_j^{v_j}$. (It is worth double checking this yourself by hand.)
- The reason for splitting up G into two parts is to ensure joint pseudorandomness of the three remaining entries in the table, when given $\sigma_i^{v_i}$ and $\sigma_j^{v_j}$. Otherwise, mix-and-match attacks could be possible.

The input wire labels for x will be $(\sigma_{\mathsf{inp}_i}^{x_i}, x_i \oplus b_i)$ for $i \in [\tilde{n}]$, and the output table will consist of $\{(z, \sigma_{\mathsf{outp}_j}^z)\}_{z \in \{0,1\}}$ for all output indices j. A proof of security is left as an exercise.

Decoding. We let $\mathsf{LRE.Decode}(\pi)$ be as follows: starting from the input wires, use the last bits of the previous labels (namely, $v_i \oplus b_i$ and $v_j \oplus b_j$) to index into the table. By plugging in keys $\sigma_i^{v_i}$ and $\sigma_j^{v_j}$ into the appropriate PRG, we recover a key $\sigma_k^{v_k}$ and $v_k \oplus b_k$. This proceeds until the output gates, from which the values are directly read from the output table.

Locality. The following observation will be critical to us. If G is in NC^0 (i.e., has $O(1)$ locality), then $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ also has $O(1)$ locality in x and r. For the input wire labels and output table, each output bit π_i of the encoding $\pi = \mathsf{LRE.Encode}(1^\lambda, C, x; r)$ is either a constant value, a bit of the random tape r, or the $\oplus$ of a bit of x and a bit of r. For the tables T_g, since G is $O(1)$-local, computing T_g is $O(1)$-local in x and r, by construction of T_g.

Efficiency. Lastly, we note that both the output length of $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ and the randomness complexity (i.e., $|r|$) can be upper-bounded by $|C| \cdot \mathrm{poly}(\lambda)$.

3.3 Definition of SFE

Definition 9 (Slow Functional Encryption)**.** *A* slow functional encryption *scheme* (SFE) *consists of three algorithms:*

- $\mathsf{SFE.Setup}(1^\lambda, C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*)$*: Outputs a master public key* mpk *and a functional secret key* sk_C*.*
- $\mathsf{SFE.Enc}(\mathsf{mpk}, x \in \{0,1\}^{\tilde{n}})$*: Outputs a ciphertext* ct*.*
- $\mathsf{SFE.Dec}(\mathsf{sk}_C, \mathsf{ct})$*: Outputs a value in* $\{0,1\}^*$*.*

We require a few properties of these algorithms.

Correctness*: For any* $x \in \{0,1\}^{\tilde{n}}$ *and any circuit* $C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*$*, and* $(\mathsf{mpk}, \mathsf{sk}_C) \leftarrow \mathsf{SFE.Setup}(1^\lambda, C)$*,*

$$\Pr\left[\mathsf{SFE.Dec}(\mathsf{sk}_C, \mathsf{SFE.Enc}(\mathsf{mpk}, x)) = C(x)\right] = 1 - \mathrm{negl}(\lambda).$$

Weak Succinctness*: There exists some* $\epsilon > 0$ *such that for* $\mathsf{ct} \leftarrow \mathsf{SFE.Enc}(\mathsf{mpk}, x)$*, we have* $|\mathsf{ct}| \leq |C|^{1-\epsilon} \cdot \mathrm{poly}(\tilde{n}, \lambda)$*.*

Security*: There is a constant* $c > 0$ *and a p.p.t. simulator* S *such that we have the following indistingiushability, with advantage at most* $2^{-\Omega(\lambda^c)}$ *to all p.p.t. adversaries: for all* $x \in \{0,1\}^{\tilde{n}}$*, and all circuits* $C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*$*,*

$$(\mathsf{mpk}, \mathsf{sk}_C, \mathsf{SFE.Enc}(\mathsf{mpk}, x)) \approx_c S(C, C(x)),$$

where $(\mathsf{mpk}, \mathsf{sk}_C) \leftarrow \mathsf{SFE.Setup}(1^\lambda, C)$*.*

3.4 Construction of SFE

Parameters:

- Let m denote the maximum of the following quantities: the output length of $\mathsf{LRE.Encode}$, the randomness complexity of $\mathsf{LRE.Encode}$, and $\tilde{n}^3$ (where $\tilde{n}$ is the input message length of our construction). From above, we know $m \leq |C| \cdot \mathrm{poly}(\lambda, \tilde{n})$. Moreover, without loss of generality, we can pad the output length and randomness length to be exactly m, in case $\tilde{n}^3$ is larger a priori.

$\mathsf{SFE.Enc}(\mathsf{mpk}, x \in \{0,1\}^{\tilde{n}})$:

- Sample $\sigma \leftarrow \{0,1\}^{m^{1/3}}$.
- Output $\mathsf{ct} \leftarrow \mathsf{LSFE.Enc}(\mathsf{mpk}, (x, \sigma))$, where the n in LSFE is defined by $n = \tilde{n} + m^{1/3}$.

$\mathsf{SFE.Setup}(1^\lambda, C : \{0,1\}^{\tilde{n}} \to \{0,1\}^*)$:

- Define the function $h(x, \sigma) = \mathsf{LRE.Encode}(1^\lambda, C, x; G(\sigma))$, where $G : \{0,1\}^{m^{1/3}} \to \{0,1\}^m$ is a PRG in NC^0.
- $(\mathsf{mpk}, \mathsf{sk}_h) \leftarrow \mathsf{LSFE.Setup}(1^\lambda, h)$.
- Output $(\mathsf{mpk}, \mathsf{sk}_C := \mathsf{sk}_h)$.

$\mathsf{SFE.Dec}(\mathsf{sk}_C, \mathsf{ct})$:

- Recover $\pi \leftarrow \mathsf{LSFE.Dec}(\mathsf{sk}_h, \mathsf{ct})$
- Output $\mathsf{LRE}_G\mathsf{.Decode}(\pi)$.

Figure 2: Construction of SFE from LSFE and (polynomial-stretch) PRGs in NC^0.

Correctness. Assuming that h is indeed $O(1)$-local, this follows from correctness of LSFE and LRE. That is, $\mathsf{LSFE.Dec}$ returns $\pi = \mathsf{LRE.Encode}(1^\lambda, C, x; G(\sigma))$, and $\mathsf{LRE.Decode}(\pi) = C(x)$, as desired. To see that h is $O(1)$-local, recall that $\mathsf{LRE.Encode}(1^\lambda, C, x; r)$ is $O(1)$-local in x and r, and G is $O(1)$-local, so their composition is still $O(1)$-local.

Weak Succinctness. Recall that $n = |(x, \sigma)| = \tilde{n} + m^{1/3}$. To apply weak succinctness of LSFE, we need the output length of h, namely m, to be at least n^2. To see this, we have

$$n^2 = (\tilde{n} + m^{1/3})^2 \leq O(\tilde{n}^2 + m^{2/3}) \leq O(m^{2/3}) \leq o(m),$$

where the penultimate bound holds since $m \geq \tilde{n}^3$. By weak succinctness of LSFE, we know $|\mathsf{ct}| \leq m^{1-\epsilon} \cdot \mathrm{poly}(\lambda)$. Expanding this out, we have

$$|\mathsf{ct}| \leq m^{1-\epsilon} \cdot \mathrm{poly}(\lambda) \leq (|C| \cdot \mathrm{poly}(\tilde{n}, \lambda))^{1-\epsilon} \cdot \mathrm{poly}(\lambda) \leq |C|^{1-\epsilon} \cdot \mathrm{poly}(\tilde{n}, \lambda),$$

as desired.

Security. This follows by combining LRE security, the PRG security of G, and LSFE security.

References

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021.

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from lpn over f p, dlin, and prgs in nc 0. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 670–699. Springer, 2022.

[Wee20] Hoeteck Wee. Functional encryption for quadratic functions from k-lin, revisited. In *Theory of Cryptography: 18th International Conference, TCC 2020, Durham, NC, USA, November 16–19, 2020, Proceedings, Part I 18*, pages 210–228. Springer, 2020.