Fine-grained Cryptography[⋆]

Akshay Degwekar, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan

MIT, CSAIL

Abstract. *Fine-grained cryptographic primitives* are ones that are secure against adversaries with an a-priori bounded polynomial amount of resources (time, space or parallel-time), where the honest algorithms use less resources than the adversaries they are designed to fool. Such primitives were previously studied in the context of time-bounded adversaries (Merkle, CACM 1978), space-bounded adversaries (Cachin and Maurer, CRYPTO 1997) and parallel-time-bounded adversaries (Håstad, IPL 1987). Our goal is come up with fine-grained primitives (in the setting of parallel-time-bounded adversaries) and to show unconditional security of these constructions when possible, or base security on widely believed separation of worst-case complexity classes. We show:

1. NC^1-cryptography: Under the assumption that $\mathsf{NC}^1 \neq \oplus\mathsf{L}/\mathsf{poly}$, we construct one-way functions, pseudo-random generators (with sub-linear stretch), collision-resistant hash functions and most importantly, *public-key encryption schemes*, all computable in NC^1 and secure against all NC^1 circuits. Our results rely heavily on the notion of randomized encodings pioneered by Applebaum, Ishai and Kushilevitz, and crucially, make *non-black-box* use of randomized encodings for logspace classes.
2. AC^0-cryptography: We construct (unconditionally secure) pseudo-random generators with arbitrary polynomial stretch, weak pseudo-random functions, secret-key encryption and perhaps most interestingly, *collision-resistant hash functions*, computable in AC^0 and secure against all AC^0 circuits. Previously, one-way permutations and pseudo-random generators (with linear stretch) computable in AC^0 and secure against AC^0 circuits were known from the works of Håstad and Braverman.

1 Introduction

The last four decades of research in the theory of cryptography has produced a host of fantastic notions, from public-key encryption [DH76, RSA78, GM82]

[⋆] MIT. E-mail: {akshayd,vinodv,prashvas}@mit.edu. Research supported in part by NSF Grants CNS-1350619 and CNS-1414119, Alfred P. Sloan Research Fellowship, Microsoft Faculty Fellowship, the NEC Corporation, a Steven and Renee Finn Career Development Chair from MIT. This work was also sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226.

and zero-knowledge proofs [GMR85] in the 1980s, to fully homomorphic encryption [RAD78, Gen09, BV11] and program obfuscation [BGI$^+$01, GGH$^+$13, SW14] in the modern day. Complexity theory is at the heart of these developments, playing a key role in coming up with precise mathematical definitions as well as constructions whose security can be reduced to precisely stated computational hardness assumptions.

However, the uncomfortable fact remains that a vast majority of cryptographic constructions rely on *unproven assumptions.* At the very least, one requires that $\mathsf{NP} \not\subseteq \mathsf{BPP}$ [IL89], but that is hardly ever enough — when designing advanced cryptographic objects, cryptographers assume the existence of one-way functions as a given, move up a notch to assuming the hardness of specific problems such as factoring, discrete logarithms and the approximate shortest vector problem for lattices, and, more recently, even more exotic assumptions. While there are some generic transformations between primitives, such as from one-way functions to pseudo-random generators and symmetric encryption (e.g., [HILL99]), there are large gaps in our understanding of relationships between most others. In particular, it is a wide open question whether $\mathsf{NP} \not\subseteq \mathsf{BPP}$ suffices to construct even the most basic cryptographic objects such as one-way functions, or whether it is possible to construct public-key encryption assuming only the existence of one-way functions (for some partial negative results, see [BT03, AGGM06, BB15, IR88]).

In this work, we ask if a weaker version of these cryptographic primitives can be constructed, with security against a *bounded* class of adversaries, based on either *mild complexity-theoretic assumptions* or *no assumptions* at all. Indeed, this question has been asked by several researchers in the past.

1. Merkle [Mer78] constructed a non-interactive key exchange protocol (and thus, a public-key encryption scheme) where the honest parties run in linear time $O(n)$ and security is shown against adversaries that run in time $o(n^2)$. His assumption was the existence of a random function that both the honest parties and the adversary can access (essentially, the random oracle model [BR93]). Later, the assumption was improved to exponentially strong one-way functions [BGI08]. This work is timeless, not only because it jump-started public-key cryptography, but also because it showed how to obtain a primitive with much structure (trapdoors) from one that apparently has none (namely, random oracles and exponentially strong one-way functions).
2. Maurer [Mau92] introduced the bounded storage model, which considers adversaries that have an a-priori bounded amount of space but unbounded computation time. Cachin and Maurer constructed symmetric-key encryption and key-exchange protocols that are *unconditionally secure* in this model [CM97] assuming that the honest parties have storage $O(s)$ and the adversary has storage $o(s^2)$ for some parameter s. There has been a rich line of work on this model [CM97, AR99, DM04] following [Mau92].
3. Implicit in the work of Håstad [Has87] is a beautiful construction of a one-way permutation that can be computed in NC^0 (constant-depth circuits with AND and OR gates of unbounded fan-in and NOT gates), but inverting

which is hard for any AC^0 circuit. Here is the function:

$$f(x_1, x_2, \ldots, x_n) = \big(x_1, x_1 \oplus x_2, x_2 \oplus x_3, \ldots, x_{n-1} \oplus x_n\big)$$

Clearly, each output bit of this function depends on at most two input bits. Inverting the function implies in particular the ability to compute x_n, which is the parity of all the output bits. This is hard for AC^0 circuits as per [FSS84, Ajt83, Hås86].

All these works share two common features. First, security is achieved against a class of adversaries with bounded resources (time, space and parallel time, respectively, in the three works above). Secondly, the honest algorithms use less resources than the class of adversaries they are trying to fool. We propose to call the broad study of such cryptographic constructions *fine-grained cryptography*, and construct several fine-grained cryptographic primitives secure against parallel-time-bounded adversaries.

We study two classes of low-depth circuits (as adversaries). The first is AC^0, which is the class of functions computable by *constant-depth* polynomial-sized circuits consisting of AND, OR, and NOT gates of *unbounded fan-in*, and the second is NC^1, the class of functions computable by *logarithmic-depth* polynomial-sized circuits consisting of AND, OR, and NOT gates of *fan-in 2*. In both cases, we mean the non-uniform versions of these classes. Note that this also covers the case of adversaries that are randomized circuits with these respective restrictions. This is because for any such randomized adversary $\mathcal{A}$ there is a non-uniform adversary $\mathcal{A}'$ that performs as well as $\mathcal{A}$ – $\mathcal{A}'$ is simply $\mathcal{A}$ hard-coded with the randomness that worked best for it.

Early developments in circuit lower bounds [FSS84, Ajt83, Hås86] showed progressively better and *average-case* and *exponential* lower bounds for the PARITY function against AC^0 circuits. This has recently been sharpened to an average-case depth hierarchy theorem [RST15]. We already saw how these lower bounds translate to meaningful cryptography, namely one-way permutations against AC^0 adversaries. Extending this a little further, a reader familiar with Braverman's breakthrough result [Bra10] (regarding the pseudorandomness of n^{ϵ}-wise independent distributions against AC^0) will notice that his result can be used to construct large-stretch pseudo-random generators that are computable by fixed-depth AC^0 circuits and are pseudo-random against arbitrary constant-depth AC^0 circuits. Can we do more? *Can we construct secret-key encryption, collision-resistant hash functions, and even trapdoor functions, starting from known lower bounds against* AC^0 *circuits*? Our first contribution is a positive answer to some of these questions.

Our second contribution is to study adversaries that live in NC^1. In this setting, as we do not know any lower bounds against NC^1, we are forced to rely on an unproven complexity-theoretic assumption; however, we aim to limit this to a worst-case, widely believed, separation of complexity classes. Here, we construct several cryptographic primitives from the *worst-case* hardness assumption that $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$, the most notable being an additively-homomorphic public-key encryption scheme where the key generation, encryption and decryption algorithms are all computable in $\mathsf{AC}^0[2]$ (constant-depth circuits with $\mathsf{MOD2}$ gates; note that $\mathsf{AC}^0[2] \subsetneq \mathsf{NC}^1$ [Raz87, Smo87]), and the scheme is semantically secure against NC^1 adversaries. ($\oplus\mathsf{L}/\mathsf{poly}$ can be thought of as the class of languages with polynomial-sized branching programs. Note that by Barrington's Theorem [Bar86], all languages in NC^1 have polynomial-sized branching programs of constant width.)

Apart from theoretical interest stemming from the fact that these are rather natural objects, one possible application of such constructions (that was suggested to us independently by Ron Rothblum and Yuval Ishai) is in using them in conjunction with other constructions that are secure against polynomial-time adversaries under stronger assumptions. This could be done to get hybrids that are secure against polynomial-time adversaries under these stronger assumptions while also being secure against bounded adversaries unconditionally (or under minimal assumptions). For instance, consider an encryption scheme where the message is first encrypted using the AC^0-encryption scheme from Section 5.3, and the resultant ciphertext is then encrypted using a scheme that works in AC^0 and is secure against polynomial-time adversaries under some standard assumptions (see [AIK04] for such schemes). This resultant scheme can be shown to be secure against polynomial-time adversaries under the same assumptions while being unconditionally secure against AC^0 adversaries.

We now briefly describe the relation between our results and the related work on randomized encodings [IK00, AIK04], and move on to describing the results in detail.

Relation to Randomized Encodings and Cryptography in NC^0*.* Randomized encodings of Ishai and Kushilevitz [IK00, AIK04] are a key tool in our results against NC^1 adversaries. Using randomized encodings, Applebaum, Ishai and Kushilevitz [AIK04] showed how to convert several cryptographic primitives computable in logspace classes into ones that are computable in NC^0. The difference between their work and ours is two-fold: (1) Their starting points are cryptographic schemes secure against arbitrary polynomial-time adversaries, which rely on average-case hardness assumptions, whereas in our work, the focus is on achieving security either with no unproven assumptions or only worst-case assumptions; of course, our schemes are not secure against polynomial-time adversaries, but rather, limited adversarial classes; (2) In the case of public-key encryption, they manage to construct key generation and encryption algorithms that run in NC^0, but the decryption algorithm retains its high complexity. In contrast, in this work, we can construct public key encryption (against NC^1 adversaries) where the encryption algorithm can be computed in NC^0 and the key generation and decryption in $\mathsf{AC}^0[2]$.

A Remark on Cryptographic vs. Non-Cryptographic Constructions An important *desideratum* for us is that the (honest) algorithms in our constructions can be implemented with fewer resources than the adversary that they are trying to fool. We call such constructions *cryptographic* in contrast to *non-cryptographic* constructions where this is not necessarily the case. Perhaps the clearest and the most well-known example of this distinction is the case of pseudo-random generators (PRGs) [BM84, Yao82, NW94]. Cryptographic PRGs, pioneered in the works of Blum, Micali and Yao [BM84, Yao82] are functions computable in a *fixed* polynomial time that produce outputs that are indistinguishable from random against *any* polynomial-time machine. The designer of the PRG does not know the precise power of the adversary: he knows that the adversary is polynomial-time, but not *which* polynomial. On the other hand, non-cryptographic ("Nisan-Wigderson type") PRGs [NW94] take more time to compute than the adversaries they are designed to fool.

Our constructions will be exclusively in the *cryptographic* regime. For example, our one-way functions, pseudo-random generators and collision-resistant hash functions against AC^0 are computable by circuits of fixed polynomial size $q(\lambda)$ and fixed (constant) depth d, and maintain security (in the appropriate sense) against adversarial circuits of size $p'(\lambda)$ and depth d' for any polynomial function p' and any constant d'.

1.1 Our Results and Techniques

Our results are grouped into two classes — primitives secure against NC^1 circuits based on minimal worst-case assumptions, and those that are unconditionally secure against AC^0 circuits. In the description below and throughout the rest of the paper, all algebra is over $\mathbb{F}_2$.

Constructions against NC^1 Adversaries We construct one-way functions (OWFs), pseudo-random generators (PRGs), additively homomorphic public-key encryption (PKE), and collision-resistant hash functions (CRHFs) that are computable in NC^1 and are secure against NC^1 adversaries, based on the worst-case assumption that $\oplus\mathsf{L/poly} \not\subseteq \mathsf{NC}^1$. An important tool we use for these constructions is the notion of *randomized encodings* of functions introduced in [IK00].

A randomized encoding of a function f is a randomized function $\hat{f}$ that is such that for any input x, the distribution of $\hat{f}(x)$ reveals $f(x)$, but nothing more about x. We know through the work of [IK00, AIK04] that there are randomized encodings for the class $\oplus\mathsf{L/poly}$ that can be computed in (randomized, uniform) NC^0. Randomized encodings naturally offer a flavor of worst-to-average case reductions as they reduce the problem of evaluating a function on a given input to deciding some property of the distribution of its encoding. Our starting point is the observation, implicit in [AIK04, AR15], that they can be used to generically construct infinitely-often one-way functions and pseudo-random generators with additive stretch, computable in NC^0 and secure against NC^1 adversaries (assuming, again, that $\oplus\mathsf{L/poly} \not\subseteq \mathsf{NC}^1$). We start with the following general theorem.

Theorem 1.1 (Informal). *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two classes such that $\mathcal{C}_2 \not\subseteq \mathcal{C}_1$ and $\mathcal{C}_2$ has perfect randomized encodings computable in $\mathcal{C}_1$. Then, there are OWFs* *and PRGs that are computable in $\mathcal{C}_1$ and are secure against arbitrary adversarial functions in $\mathcal{C}_1$.*

Informally, the argument for Theorem 1.1 is the following: Let L be the language in $\mathcal{C}_2$ but not $\mathcal{C}_1$. The PRG is a function that takes an input r and outputs the randomized encoding of the indicator function for membership in L on the input 0^λ, using r as the randomness (where λ is a security parameter). Any adversary that can distinguish the output of this function from random can be used to decide if a given x is in the language L by computing the randomized encoding of x and feeding it to the adversary. This gives us a PRG with a non-zero additive stretch (and also a OWF) if the randomized encoding has certain properties (they need to be *perfect*) — see Section 3 for details.

While we have one way functions and pseudorandom generators, a blackbox construction of public key cryptosystems from randomized encodings seems elusive. Our first contribution in this work is to use the algebraic structure of the randomized encodings for $\oplus\mathsf{L}/\mathsf{poly}$ to construct an additively homomorphic public key encryption scheme secure against NC^1 circuits (assuming that $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$).

Additively Homomorphic Public-Key Encryption. The key attribute of the randomized encodings of [IK00, AIK04] for $\oplus\mathsf{L}/\mathsf{poly}$ is that the encoding is not a structureless string. Rather, the randomized encodings of computations are matrices whose rank corresponds to the result of the computation. Our public-key encryption construction uses two observations:

- Under the assumption $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$, there exist, for an infinite number of values of n, distributions D_0^n and D_1^n over $n \times n$ matrices of rank $(n-1)$ and n, respectively, that are indistinguishable to NC^1 circuits.
- It is possible to sample a matrix $\mathbf{M}$ from D_0^n along with the non-zero vector $\mathbf{k}$ in its kernel. The sampling can be accomplished in NC^1 or even $\mathsf{AC}^0[2]$.

The public key in our scheme is such a matrix $\mathbf{M}$, and the secret key is the corresponding $\mathbf{k}$. Encryption of a bit b is a vector $\mathbf{r}^T\mathbf{M} + b\mathbf{t}^T$, where $\mathbf{r}$ is a random vector[1] and $\mathbf{t}$ is a vector such that $\langle \mathbf{t}, \mathbf{k} \rangle = 1$. In effect, the encryption of 0 is a random vector in the row-span of $\mathbf{M}$ while the encryption of 1 is a random vector outside. Decryption of a ciphertext $\mathbf{c}$ is simply the inner product $\langle \mathbf{c}, \mathbf{k} \rangle$. Semantic security against NC^1 adversaries follows from the fact that D_0^n and D_1^n are indistinguishable to NC^1 circuits. In particular, (1) We can indistinguishably replace the public key by a random full rank matrix $\mathbf{M}'$ chosen from D_n^1; and (2) with $\mathbf{M}'$ as the public key, encryptions of 0 are identically distributed to the encryptions of 1. The following is an informal restatement of Theorem 4.1.

Theorem 1.2 (Informal). *If $\oplus\mathsf{L}/\mathsf{poly} \neq \mathsf{NC}^1$, there is a public-key encryption scheme secure against NC^1 adversaries where key generation, encryption and decryption are all computable in (randomized) $\mathsf{AC}^0[2]$.*

[1] We maintain the convention that all vectors are by default column vectors. For a vector $\mathbf{r}$, the notation $\mathbf{r}^T$ denotes the row vector that is the transpose of $\mathbf{r}$.

The scheme above is additively homomorphic, and thus, collision-resistant hash functions (CRHF) against NC^1 follow immediately from the known generic transformations [IKO05] which work in NC^1.

Theorem 1.3 (Informal). *If $\oplus\mathsf{L/poly} \neq \mathsf{NC}^1$, there is a family of collision-resistant hash functions that is secure against NC^1 adversaries where both sampling hash functions and evaluating them can be performed in (randomized) $\mathsf{AC}^0[2]$.*

We remark that in a recent work, Applebaum and Raykov [AR15] construct CRHFs against polynomial-time adversaries under the assumption that there are average-case hard functions with perfect randomized encodings. Their techniques also carry over to our setting and imply, for instance, the existence of CRHFs against NC^1 under the assumption that there is a language that is average-case hard for NC^1 that has perfect randomized encodings that can be computed in NC^1. This does not require any additional structure on the encodings apart from perfectness, but does require average-case hardness in place of our worst-case assumptions.

Constructions against AC^0 Adversaries We construct one-way functions (OWFs), pseudo-random generators (PRGs), weak pseudo-random functions (weak PRFs), symmetric-key encryption (SKE) and collision-resistant hash functions (CRHFs) that are computable in AC^0 and are unconditionally secure against arbitrary AC^0 circuits. While some constructions for OWFs and PRGs against AC^0 were already known [Hås86, Bra10], and the existence of weak PRFs and SKE, being minicrypt primitives, is not that surprising, the possibility of unconditionally secure CRHFs against AC^0 is somewhat surprising, and we consider this to be our primary contribution in this section. We also present a candidate construction for public-key encryption, but we are unable to prove its unconditional security against AC^0 circuits.

As we saw earlier, Håstad [Has87] constructed one-way permutations secure against AC^0 circuits based on the hardness of computing PARITY. When allowed polynomial running time, we have black-box constructions of pseudorandom generators [HILL99] and pseudorandom functions [GGM86] from one-way functions. But because these reductions are not implementable in AC^0, getting primitives computable in AC^0 requires more effort.

Our constructions against AC^0 adversaries are primarily based on the theorem of Braverman [Bra10] (and its recent sharpening by Tal [Tal14]) regarding the pseudo-randomness of polylog-wise independent distributions against constant depth circuits. We use this to show that AC^0 circuits cannot distinguish between the distribution $(\mathbf{A}, \mathbf{Ak})$, where $\mathbf{A}$ is a random "sparse" matrix of dimension $\mathsf{poly}(n) \times n$ and $\mathbf{k}$ is a uniformly random secret vector, from the distribution $(\mathbf{A}, \mathbf{r})$, where $\mathbf{r}$ is a uniformly random vector. Sparse here means that each row of $\mathbf{A}$ has at most $\mathsf{polylog}(n)$ many ones.

(This is shown as follows. It turns out that with high probability, a matrix chosen in this manner is such that any set of $\mathsf{polylog}(n)$ rows is linearly independent (Lemma 2.5). Note that when a set of rows of $\mathbf{A}$ is linearly independent, the corresponding set of bits in $\mathbf{Ak}$ are uniformly distributed. This implies that if all $\mathsf{polylog}(n)$-sized sets of rows of $\mathbf{A}$ are linearly independent, then $\mathbf{Ak}$ is $\mathsf{polylog}(n)$-wise independent. This fact, along with the theorems regarding pseudo-randomness mentioned above prove the indistinguishability by AC^0.)

We also crucially use the fact, from [AB84], that the inner product of an arbitrary vector with a sparse vector can be computed in constant depth.

OWFs and PRGs This enables us to construct PRGs in NC^0 with constant multiplicative stretch and in AC^0 with polynomial multiplicative stretch. The construction is to fix a sparse matrix $\mathbf{A}$ with the linear independence properties mentioned above, and the PRG output on seed $\mathbf{k}$ is $\mathbf{Ak}$. Pseudo-randomness follows from the indistinguishability arguments above. This is stated in the following informal restatement of Theorem 5.1. We need to show that there exist such matrices $\mathbf{A}$ in which any $\mathsf{polylog}$-sized set of rows are linearly independent, and yet are sparse. As we show in Section 2.3, there are indeed matrices that have these properties.

Theorem 1.4 (Informal). *For any constant c, there is a family of circuits $\left\{C_n : \{0,1\}^n \to \{0,1\}^{n^c}\right\}$ such that for any n, each output bit of C_n depends on at most $O(c)$ input bits. Further, for large enough n, AC^0 circuits cannot distinguish the output distribution $C_n(U_n)$ from U_{n^c}.*

We note that similar techniques have been used in the past to construct PRGs that fool circuit families of a fixed constant depth - see, for instance, [Vio12].

Weak PRFs against AC^0. A Pseudo-Random Function family (PRF) is a collection of functions such that a function chosen at random from this collection is indistinguishable from a function chosen at random from the set of all functions (with the appropriate domain and range), based on just a polynomial number of evaluations of the respective functions. In a *Strong* PRF, the distinguisher is allowed to specify (even adaptively) the input points at which it wants the function to be evaluated. In a *Weak* PRF, the distinguisher is given function evaluations at input points chosen uniformly at random.

We construct Weak PRFs against AC^0 that are unconditionally secure. In our construction, each function in the family is described by a vector $\mathbf{k}$. The computation of the pseudo-random function proceeds by mapping its input $\mathbf{x}$ to a sparse vector $\mathbf{a}$ and computing the inner product $\langle \mathbf{a}, \mathbf{k} \rangle$ over $\mathbb{F}_2$. Given polynomially many samples of the form $(\mathbf{a}, \langle \mathbf{a}, \mathbf{k} \rangle)$, one can write this as $(\mathbf{A}, \mathbf{Ak})$, where $\mathbf{A}$ is a matrix with random sparse rows. Our mapping of $\mathbf{x}$'s to $\mathbf{a}$'s is such that $(\mathbf{A}, \mathbf{Ak})$ is in some sense the only useful information contained in a set of random function evaluations. This is now indistinguishable from $(\mathbf{A}, \mathbf{r})$ where $\mathbf{r}$ is uniformly random, via the arguments mentioned earlier in this section. The following is an informal restatement of Theorem 5.2.

Theorem 1.5 (Informal). *There is a Weak Pseudo-Random Function family secure against* AC^0 *adversaries and is such that both sampling a function at random and evaluating it can be performed in* AC^0.

We note that while our construction only gives us quasi-polynomial security (that is, an adversary might be able to achieve an inverse quasi-polynomial advantage in telling our functions from random) as opposed to exponential security, we show that this is an inherent limitation of weak PRFs computable in AC^0. Roughly speaking, due to the work of [LMN93], we know that a constant fraction of the Fourier mass of any function on n inputs computable in AC^0 is concentrated on Fourier coefficients upto some $\mathsf{polylog}(n)$. So there is at least one coefficient in the case of such a function that is at least $\Omega\left(\frac{1}{2^{\mathsf{polylog}(n)}}\right)$ in absolute value, whereas in a random function any coefficient would be exponentially small. So, by guessing and estimating a Fourier coefficient of degree at most $\mathsf{polylog}(n)$ (which can be done in AC^0), one can distinguish functions computed in AC^0 from a random function with some $\Omega\left(\frac{1}{2^{\mathsf{polylog}(n)}}\right)$ advantage.

Symmetric Key Encryption against AC^0. In the case of polynomial-time adversaries and constructions, weak PRFs generically yield symmetric key encryption schemes, and this continues to hold in our setting. However, we present an alternative construction that has certain properties that make it useful in the construction of collision-resistant hash functions later on. The key in our scheme is again a random vector $\mathbf{k}$. The encryption of a bit b is a random sparse vector $\mathbf{c}$ such that $\langle \mathbf{c}, \mathbf{k} \rangle = b$ over $\mathbb{F}_2$. (Similar schemes, albeit without the sparsity, have been employed in the past in the leakage-resilience literature — see [GR12] and references therein.)

Encryption is performed by rejection sampling to find such a $\mathbf{c}$, and decryption is performed by computing $\langle \mathbf{c}, \mathbf{k} \rangle$, which can be done in constant depth owing to the sparsity of $\mathbf{c}$. We reduce the semantic security of this construction to the indistinguishability of the distributions $(\mathbf{A}, \mathbf{A}\mathbf{k})$ and $(\mathbf{A}, \mathbf{r})$ mentioned earlier. Note that this scheme is additively homomorphic, a property that will be useful later. The following is an informal restatement of Theorem 5.3.

Theorem 1.6 (Informal). *There is a Symmetric Key Encryption scheme that is secure against* AC^0 *adversaries and is such that key generation, encryption and decryption are all computable in (randomized)* AC^0.

Collision Resistance against AC^0. Our most important construction against AC^0, which is what our encryption scheme was designed for, is that of Collision Resistant Hash Functions. Note that while there are generic transformations from additively homomorphic encryption schemes to CRHFs ([IKO05]), these transformations do not work in AC^0 and hence do not yield the construction we desire.

Our hash functions are described by matrices where each column is the encryption of a random bit under the above symmetric encryption scheme. Given such a matrix $\mathbf{M}$ that consists of encryptions of the bits of a string m, the evaluation of the function on input $\mathbf{x}$ is $\mathbf{M}\mathbf{x}$. Note that we wish to perform this computation in constant depth, and this turns out to be possible to do correctly for most keys because of the sparsity of our ciphertexts.

Finding a collision for a given key $\mathbf{M}$ is equivalent to finding a vector $\mathbf{u}$ such that $\mathbf{M}\mathbf{u} = 0$. By the additive homomorphism of the encryption scheme, and the fact that $\mathbf{0}$ is a valid encryption of 0, this implies that $\langle m, \mathbf{u} \rangle = 0$. But this is non-trivial information about m, and so should violate semantic security. However showing that this is indeed the case turns out to be somewhat non-trivial.

In order to do so, given an AC^0 adversary A that finds collisions for the hash function with some non-negligible probability, we will need to construct another AC^0 adversary, B, that breaks semantic security of the encryption scheme. The most straightforward attempt at this would be as follows. B selects messages $\mathbf{m}_0$ and $\mathbf{m}_1$ at random and sends them to the challenger who responds with $\mathbf{M} = \mathsf{Enc}(\mathbf{m}_b)$. B then forwards this to A who would then return, with non-negligible probability, a vector $\mathbf{u}$ such that $\langle \mathbf{u}, \mathbf{m}_b \rangle = 0$. If B could compute $\langle \mathbf{u}, \mathbf{m}_0 \rangle$ and $\langle \mathbf{u}, \mathbf{m}_1 \rangle$, B would then be able to guess b correctly with non-negligible advantage. The problem with this approach is that $\mathbf{u}$, $\mathbf{m}_0$ and $\mathbf{m}_1$ might all be of high Hamming weight, and this being the case, B would not be able to compute the above inner products.

The solution to this is to choose $\mathbf{m}_0$ to be a sparse vector and $\mathbf{m}_1$ to be a random vector and repeat the same procedure. This way, B can compute $\langle \mathbf{u}, \mathbf{m}_0 \rangle$, and while it still cannot check whether $\langle \mathbf{u}, \mathbf{m}_1 \rangle = 0$, it can instead check whether $\mathbf{M}\mathbf{u} = \mathbf{0}$ and use this information. If it turns out that $\mathbf{M}\mathbf{u} = \mathbf{0}$ and $\langle \mathbf{u}, \mathbf{m}_0 \rangle \neq 0$, then B knows that $b = 1$, due to the additive homomorphism of the encryption scheme. Also, when $b = 1$, since $\mathbf{m}_0$ is independent of $\mathbf{m}_1$, the probability that A outputs $\mathbf{u}$ such that $\langle \mathbf{u}, \mathbf{m}_0 \rangle \neq 0$ is non-negligible. Hence, by guessing $b = 1$ when this happens and by guessing b at random otherwise, B can gain non-negligible advantage against semantic security. This achieves the desired contradiction and demonstrates the collision resistance of our construction. The following is an informal restatement of Theorem 5.4.

Theorem 1.7 (Informal). *There is a family of Collision Resistant Hash Functions that is secure against AC^0 adversaries and is such that both sampling a hash function at random and evaluating it can be performed in (randomized) AC^0.*

Candidate Public Key Encryption against AC^0 We also propose a candidate Public Key Encryption scheme whose security we cannot show. It is similar to the LPN-based cryptosystem in [Ale03]. The public key is a matrix of the form $\mathbf{M} = (\mathbf{A}, \mathbf{A}\mathbf{k})$ where $\mathbf{A}$ is a random $n \times n$ matrix and $\mathbf{k}$, which is also the secret key, is a random sparse vector of length n. To encrypt 0, we choose a random sparse vector $\mathbf{r}$ and output $\mathbf{c}^T = \mathbf{r}^T\mathbf{M}$, and to encrypt 1 we just output a random vector $\mathbf{c}^T$ of length $(n+1)$. Decryption is simply the inner product of $\mathbf{c}$ and the vector $(\mathbf{k}\ 1)^T$, and can be done in AC^0 because $\mathbf{k}$ is sparse.

1.2 Other Related Work: Cryptography against Bounded Adversaries

The big bang of public-key cryptography was the result of Merkle [Mer78] who constructed a key exchange protocol where the honest parties run in linear time $O(n)$ and security is obtained against adversaries that run in time $o(n^2)$. His assumption was the existence of a random function that both the honest parties and the adversary can access. Later, the assumption was improved to strong one-way functions [BGI08]. This is, indeed, a fine-grained cryptographic protocol in our sense.

The study of ϵ-biased generators [AGHP93, MST06] is related to this work. In particular, ϵ-biased generators with exponentially small ϵ give us almost k-wise independent generators for large k, which in turn fool AC^0 circuits by a result of Braverman [Bra10]. This and other techniques have been used in the past to construct PRGs that fool circuits of a fixed constant depth, with the focus generally being on optimising the seed length [Vio12, TX13].

The notion of precise cryptography introduced by Micali and Pass [MP06] studies reductions between cryptographic primitives that can be computed in linear time. That is, they show constructions of primitive B from primitive A such that if there is a $\mathsf{TIME}(f(n))$ algorithm that breaks primitive B, there is a $\mathsf{TIME}(O(f(n)))$ algorithm that breaks A.

Maurer [Mau92] introduced the bounded storage model, which considers adversaries that have a bounded amount of space and unbounded computation time. There are many results known here [DM04, Vad04, AR99, CM97] and in particular, it is possible to construct Symmetric Key Encryption and Key Agreement protocols unconditionally in this model[CM97].

2 Preliminaries

In this section we establish notation that shall be used throughout the rest of the presentation and recall the notion of randomized encodings of functions. We state and prove some results about certain kinds of random matrices that turn out to be useful in Section 5. In Sections 2.4 and 2.5, we present formal definitions of a general notion of adversaries with restricted computational power and also of several standard cryptographic primitives against such restricted adversaries (as opposed to the usual definitions, which are specific to probabilistic polynomial time adversaries).

2.1 Notation

For a distribution D, by $x \leftarrow D$ we denote x being sampled according to D. Abusing notation, we denote by $D(x)$ the probability mass of D on the element x. For a set S, by $x \leftarrow S$, we denote x being sampled uniformly from S. We also denote the uniform distribution over S by U_S, and the uniform distribution over $\{0,1\}^\lambda$ by U_λ. We use the notion of total variational distance between distributions, given by:

$$\varDelta(D_1, D_2) = \frac{1}{2}\sum_x |D_1(x) - D_2(x)|$$

For distributions D_1 and D_2 over the same domain, by $D_1 \equiv D_2$ we mean that the distributions are the same, and by $D_1 \approx D_2$, we mean that $\varDelta(D_1, D_2)$ is a negligible function of some parameter that will be clear from the context. Abusing notation, we also sometimes use random variables instead of their distributions in the above expressions.

For any $n \in \mathbb{N}$, we denote by $\lfloor n \rfloor_2$ the greatest power of 2 that is not more than n. For any n, k, and $d \leq k$, we denote by $SpR_{k,d}$ the uniform distribution over the set of vectors in $\{0,1\}^k$ with exactly d non-zero entries, and by $SpM_{n,k,d}$ the distribution over the set of matrices in $\{0,1\}^{n\times k}$ where each row is distributed independently according to $SpR_{k,d}$.

We identify strings in $\{0,1\}^n$ with vectors in $\mathbb{F}_2^n$ in the natural manner. For a string (vector) x, $\|x\|$ denotes its Hamming weight. Finally, we note that all arithmetic computations (such as inner products, matrix products, etc.) in this work will be over $\mathbb{F}_2$, unless specified otherwise.

2.2 Constant-Depth Circuits

Here we state a few known results on the computational power of constant depth circuits that shall be useful in our constructions against AC^0 adversaries.

Theorem 2.1 (Hardness of Parity, [Hås14]). *For any circuit C with n inputs, size s and depth d,*

$$\Pr_{x\leftarrow\{0,1\}^n}[C(x) = \mathsf{PARITY}(x)] \leq \frac{1}{2} + 2^{-\varOmega(n/(\log s)^{d-1})}$$

Theorem 2.2 (Partial Independence, [Bra10, Tal14]). *Let D be a k-wise independent distribution over $\{0,1\}^n$. For any circuit C with n inputs, size s and depth d,*

$$\left|\Pr_{x\leftarrow D}[C(x) = 1] - \Pr_{x\leftarrow\{0,1\}^n}[C(x) = 1]\right| \leq \frac{s}{2^{\varOmega(k^{1/(3d+3)})}}$$

The following lemma is implied by theorems proven in [AB84, RW91] regarding the computability of polylog thresholds by constant-depth circuits.

Lemma 2.3 (Polylog Inner Products). *For any constant c and for any function $t : \mathbb{N} \to \mathbb{N}$ such that $t(\lambda) = O(\log^c \lambda)$, there is an AC^0 family $\mathcal{I}^t = \{ip_\lambda^t\}$ such that for any λ,*

- *ip_λ^t takes inputs from $\{0,1\}^\lambda \times \{0,1\}^\lambda$.*
- *For any $x, y \in \{0,1\}^\lambda$ such that* $\min(\|x\|, \|y\|) \leq t(\lambda)$, *$ip_\lambda^t(x, y) = \langle x, y\rangle$.*

2.3 Sparse Matrices and Linear Codes

In this section we describe and prove some properties of a sampling procedure for random matrices. In interest of space, we will defer the proofs of the lemmas stated in this section to the full version.

We describe the following two sampling procedures that we shall use later. SRSamp and SMSamp abbreviate *Sparse Row Sampler* and *Sparse Matrix Sampler*, respectively. $\mathsf{SRSamp}(k, d, r)$ samples unformly at random a vector from $\{0,1\}^k$ with exactly d non-zero entries, using r for randomness – it chooses a set of d distinct indices between 0 to $k-1$ (via rejection sampling) and outputs the vector in which the entries at those indices are 1 and the rest are 0. When we don't specifically need to argue about the randomness, we drop the explicitly written r. $\mathsf{SMSamp}(n, k, d)$ samples an $n \times k$ matrix whose rows are samples from $\mathsf{SRSamp}(k, d, r)$ using randomly and independently chosen r's.

Construction 2.1 Sparse row and matrix sampling.

$\mathsf{SRSamp}(k, d, r)$: Samples a vector with exactly d non-zero entries.

1. If r is not specified or $|r| < d^2 \lceil \log(k) \rceil$, sample $r \leftarrow \{0,1\}^{d^2 \lceil \log(k) \rceil}$ anew.
2. For $l \in [d]$ and $j \in [d]$, set $u_j^l = r_{((l-1)d+j-1)\lceil \log(k) \rceil + 1} \cdots r_{((l-1)d+j)\lceil \log(k) \rceil}$.
3. If there is no l such that for all distinct $j_1, j_2 \in [d]$, $u_{j_1}^l \neq u_{j_2}^l$, output 0^k.
4. Else, let l_0 be the least such l.
5. For $i \in [k]$, set $v_i = 1$ if there is a $j \in [d]$ such that $u_j^{l_0} = i$ (when interpreted in binary), or $v_i = 0$ otherwise.
6. Output $v = (v_1, \ldots, v_k)$.

$\mathsf{SMSamp}(n, k, d)$: Samples a matrix where each row has d non-zero entries.

1. For $i \in [n]$, sample $r_i \leftarrow \{0,1\}^{d^2 \lceil \log(k) \rceil}$ and $a_i \leftarrow \mathsf{SRSamp}(k, d, r_i)$.
2. Output the $n \times k$ matrix whose i-th row is a_i.

For any fixed k and $d < k$, note that the function $S_{k,d} : \{0,1\}^{d^2 \lceil \log(k) \rceil} \to \{0,1\}^k$ given by $S_{k,d}(x) = \mathsf{SRSamp}(k, d, x)$ can be easily seen to be computed by a circuit of size $O((d^3 + kd^2)\log(k))$ and depth 8. And so the family $\mathcal{S} = \left\{ S_{\lambda, d(\lambda)} \right\}$ is in AC^0. When, in our constructions, we require computing $\mathsf{SRSamp}(k, d, x)$, this is to be understood as being performed by the circuit for $S_{k,d}$ that is given as input the prefix of x of length $d^2 \lceil \log(k) \rceil$. So if the rest of the construction is computed by polynomial-sized constant depth circuits, the calls to SRSamp do not break this property.

Recall that we denote by $SpR_{k,d}$ the uniform distribution over the set of vectors in $\{0,1\}^k$ with exactly d non-zero entries, and by $SpM_{n,k,d}$ the distribution over the set of matrices in $\{0,1\}^{n \times k}$ where each row is sampled independently according to $SpR_{k,d}$. The following lemma states that the above sampling procedures produce something close to these distributions.

Lemma 2.4 (Uniform Sparse Sampling). *For any n, and $d = \log^2(k)$, there is a negligible function ν such that for any k that is a power of two, when $r \leftarrow \{0,1\}^{\log^5(k)}$,*

1. $\Delta(\mathsf{SRSamp}(k,d,r), SpR_{k,d}) \leq \nu(k)$
2. $\Delta(\mathsf{SMSamp}(n,k,d), SpM_{n,k,d}) \leq n\nu(k)$

The following property of the sampling procedures above is easiest proven in terms of expansion properties of bipartite graphs represented by the matrices sampled. The analysis closely follows that of Gallager ([Gal62]) from his early work on Low-Density Parity Check codes.

Lemma 2.5 (Sampling codes). *For any constant $c > 0$, set $n = k^c$, and $d = \log^2(k)$. For a matrix $\mathbf{H}$, let $\delta(\mathbf{H})$ denote the minimum distance of the code whose parity check matrix is $\mathbf{H}$. Then, there is a negligible function ν such that for any k that is a power of two,*

$$\Pr_{\mathbf{H} \leftarrow \mathsf{SMSamp}(n,k,d)} \left[\delta(\mathbf{H}) \geq \frac{k}{\log^3(k)} \right] \geq 1 - \nu(k)$$

Recall that a *δ-wise independent* distribution over n bits is a distribution whose marginal distribution on any set of δ bits is the uniform distribution.

Lemma 2.6 (Distance and Independence). *Let $\mathbf{H}$ (of dimension $n \times k$) be the parity check matrix of an $[n,(n-k)]_2$ linear code of minimum distance more than δ. Then, the distribution of $\mathbf{Hx}$ is δ-wise independent when $\mathbf{x}$ is chosen uniformly at random from $\{0,1\}^k$.*

The following is immediately implied by Lemmas 2.5, 2.6 and Theorem 2.2. It effectively says that AC^0 circuits cannot distinguish between $(\mathbf{A}, \mathbf{As})$ and $(\mathbf{A}, \mathbf{r})$ when $\mathbf{A}$ is sampled using SRSamp and $\mathbf{s}$ and $\mathbf{r}$ are chosen uniformly at random.

Lemma 2.7. *For any polynomial n, there is a negligible function ν such that for any Boolean family $\mathcal{G} = \{g_\lambda\} \in \mathsf{AC}^0$, and for any k that is a power of 2, when $\mathbf{A} \leftarrow \mathsf{SMSamp}(n(k), k, \log^2(k))$, $\mathbf{s} \leftarrow \{0,1\}^k$ and $\mathbf{r} \leftarrow \{0,1\}^{n(k)}$,*

$$|\Pr[g_\lambda(\mathbf{A}, \mathbf{As}) = 1] - \Pr[g_\lambda(\mathbf{A}, \mathbf{r}) = 1]| \leq \nu(\lambda)$$

2.4 Adversaries

Definition 2.8 (Function Family). *A* function family *is a family of (possibly randomized) functions $\mathcal{F} = \{f_\lambda\}_{\lambda \in \mathbb{N}}$, where for each λ, f_λ has domain D_λ^f and co-domain R_λ^f.*

In most of our considerations, D_λ^f and R_λ^f will be $\{0,1\}^{d_\lambda^f}$ and $\{0,1\}^{r_\lambda^f}$ for some sequences $\{d_\lambda^f\}_{\lambda \in \mathbb{N}}$ and $\{r_\lambda^f\}_{\lambda \in \mathbb{N}}$. Wherever function families are seen to act as adversaries to cryptographic objects, we shall use the terms *adversary* and *function family* interchangeably. The following are some examples of natural classes of function families.

Definition 2.9 (AC^0)**.** *The class of (non-uniform)* AC^0 *function families is the set of all function families* $\mathcal{F} = \{f_\lambda\}$ *for which there is a polynomial* p *and constant* d *such that for each* λ*,* f_λ *can be computed by a (randomized) circuit of size* $p(\lambda)$*, depth* d *and unbounded fan-in using* AND*,* OR *and* NOT *gates.*

Definition 2.10 (NC^1)**.** *The class of (non-uniform)* NC^1 *function families is the set of all function families* $\mathcal{F} = \{f_\lambda\}$ *for which there is a polynomial* p *and constant* c *such that for each* λ*,* f_λ *can be computed by a (randomized) circuit of size* $p(\lambda)$*, depth* $c\log(\lambda)$ *and fan-in* 2 *using* AND*,* OR *and* NOT *gates*

2.5 Primitives Against Bounded Adversaries

In this section, we generalize the standard definitions of several standard cryptographic primitives to talk about security against different classes of adversaries. In the following definitions, $\mathcal{C}_1$ and $\mathcal{C}_2$ are two function classes, and $l, s : \mathbb{N} \to \mathbb{N}$ are some functions. Due to space constraints, we do not define all the primitives we talk about in the paper here, but the samples below illustrate how our definitions relate to the standard ones, and the rest are analogous. All definitions are present in the full version of the paper.

Implicit (and hence left unmentioned) in each definition are the following conditions:

- *Computability*, which says that the function families that are part of the primitive are in the class $\mathcal{C}_1$. Additional restrictions are specified when they apply.
- *Non-triviality*, which says that the security condition in each definition is not vacuously satisfied – that there is at least one function family in $\mathcal{C}_2$ whose input space corresponds to the output space of the appropriate function family in the primitive.

Definition 2.11 (One-Way Function). *Let* $\mathcal{F} = \left\{f_\lambda : \{0,1\}^\lambda \to \{0,1\}^{l(\lambda)}\right\}$ *be a function family.* $\mathcal{F}$ *is a* $\mathcal{C}_1$-One-Way Function (OWF) against $\mathcal{C}_2$ *if:*

- ***Computability:*** *For each* λ*,* f_λ *is deterministic.*
- ***One-wayness:*** *For any* $\mathcal{G} = \left\{g_\lambda : \{0,1\}^{l(\lambda)} \to \{0,1\}^\lambda\right\} \in \mathcal{C}_2$*, there is a negligible function* ν *such that for any* $\lambda \in \mathbb{N}$*:*

$$\Pr_{x \leftarrow U_\lambda}\left[f_\lambda(g_\lambda(y)) = y \mid y \leftarrow f_\lambda(x)\right] \leq \nu(\lambda)$$

For a function class $\mathcal{C}$, we sometimes refer to a $\mathcal{C}$-OWF or an OWF against $\mathcal{C}$. In both these cases, both $\mathcal{C}_1$ and $\mathcal{C}_2$ from the above definition are to be taken to be $\mathcal{C}$. To be clear, this implies that there is a family $\mathcal{F} \in \mathcal{C}$ that realizes the primitive and is secure against all $\mathcal{G} \in \mathcal{C}$. We shall adopt this abbreviation also for other primitives defined in the above manner.

Definition 2.12 (Symmetric Key Encryption). *Consider function families $\mathcal{K}eyGen = \{\mathsf{KeyGen}_\lambda : \varnothing \to K_\lambda\}$, $\mathcal{E}nc = \{\mathsf{Enc}_\lambda : K_\lambda \times \{0,1\} \to C_\lambda\}$, and $\mathcal{D}ec = \{\mathsf{Dec}_\lambda : K_\lambda \times C_\lambda \to \{0,1\}\}$. $(\mathcal{K}eyGen, \mathcal{E}nc, \mathcal{D}ec)$ is a* $\mathcal{C}_1$-Symmetric Key Encryption Scheme against $\mathcal{C}_2$ *if:*

- ***Correctness:*** *There is a negligible function ν such that for any $\lambda \in \mathbb{N}$ and any $b \in \{0,1\}$:*

$$\Pr\left[\mathsf{Dec}_\lambda(k,c) = b \,\middle|\, \begin{array}{c} k \leftarrow \mathsf{KeyGen}_\lambda \\ c \leftarrow \mathsf{Enc}_\lambda(k,b) \end{array}\right] \geq 1 - \nu(\lambda)$$

- ***Semantic Security:*** *For any polynomials $n_0, n_1 : \mathbb{N} \to \mathbb{N}$, and any family $\mathcal{G} = \left\{g_\lambda : C_\lambda^{n_0(\lambda)+n_1(\lambda)+1} \to \{0,1\}\right\} \in \mathcal{C}_2$, there is a negligible function ν' such that for any $\lambda \in \mathbb{N}$:*

$$\Pr\left[g_\lambda\left(\left\{c_i^0\right\}, \left\{c_i^1\right\}, c\right) = b \,\middle|\, \begin{array}{c} k \leftarrow \mathsf{KeyGen}_\lambda, b \leftarrow U_1 \\ c_1^0, \ldots, c_{n_0(\lambda)}^0 \leftarrow \mathsf{Enc}_\lambda(k,0) \\ c_1^1, \ldots, c_{n_1(\lambda)}^1 \leftarrow \mathsf{Enc}_\lambda(k,1) \\ c \leftarrow \mathsf{Enc}_\lambda(k,b) \end{array}\right] \leq \frac{1}{2} + \nu'(\lambda)$$

2.6 Randomized Encodings

The notion of randomized encodings of functions was introduced by Ishai and Kushilevitz [IK00] in the context of secure multi-party computation. Roughly, a randomized encoding of a deterministic function f is another deterministic function $\widehat{f}$ that is easier to compute by some measure, and is such that for any input x, the distribution of $\widehat{f}(x,r)$ (when r is chosen uniformly at random) reveals the value of $f(x)$ and nothing more. This reduces the computation of $f(x)$ to determining some property of the distribution of $\widehat{f}(x,r)$. Hence, randomized encodings offer a flavor of worst-to-average case reduction — from computing $f(x)$ from x to that of computing $f(x)$ from random samples of $\widehat{f}(x,r)$.

We work with the following definition of *Perfect Randomized Encodings* from [App14]. We note that constructions of such encodings for $\oplus\mathsf{L/poly}$ which are computable in NC^0 were presented in [IK00].

Definition 2.13 (Perfect Randomized Encodings). *Consider a deterministic function $f : \{0,1\}^n \to \{0,1\}^t$. We say that the deterministic function $\widehat{f} : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^s$ is a* Perfect Randomized Encoding (PRE) *of f if the following conditions are satisfied.*

- ***Input independence:*** *For every $x, x' \in \{0,1\}^n$ such that $f(x) = f(x')$, the random variables $\widehat{f}(x, U_m)$ and $\widehat{f}(x', U_m)$ are identically distributed.*
- ***Output disjointness:*** *For every $x, x' \in \{0,1\}^n$ such that $f(x) \neq f(x')$, $Supp(\widehat{f}(x,U_m)) \cap Supp(\widehat{f}(x',U_m)) = \phi$.*
- ***Uniformity:*** *For every x, $\widehat{f}(x,U_m)$ is uniform on its support.*
- ***Balance:*** *For every $x, x' \in \{0,1\}^n$, $\left|Supp(\widehat{f}(x,U_m))\right| = \left|Supp(\widehat{f}(x',U_m))\right|$*

– ***Stretch preservation:*** $s - (n + m) = t - n$

Additionally, the PRE is said to be surjective *if it also has the following property.*

– ***Surjectivity:*** *For every $y \in \{0,1\}^s$, there exist x and r such that $\hat{f}(x,r) = y$.*

We naturally extend the definition of PREs to function families – a family $\widehat{\mathcal{F}} = \left\{\widehat{f_\lambda}\right\}$ is a PRE of another family $\mathcal{F} = \{f_\lambda\}$ if for all large enough λ, $\widehat{f_\lambda}$ is a PRE of f_λ. Note that this notion only makes sense for deterministic functions, and the functions and families we assume or claim to have PREs are to be taken to be deterministic.

3 OWFs from worst-case assumptions

In this section and in Section 4, we describe some constructions of cryptographic primitives against bounded adversaries starting from worst-case hardness assumptions. The existence of Perfect Randomized Encodings (PREs) can be leveraged to construct one-way functions and pseudo-random generators against bounded adversaries starting from a function that is hard in the worst-case for these adversaries. We describe this construction below.

Remark 3.1 (Infinitely often primitives). For a class $\mathcal{C}$, the statement $\mathcal{F} = \{f_\lambda\} \notin \mathcal{C}$ implies that for any family $\mathcal{G} = \{g_\lambda\}$ in $\mathcal{C}$, there are an infinite number of values of λ such that $f_\lambda \not\equiv g_\lambda$. Using such a worst case assumption, we only know how to obtain primitives whose security holds for an infinite number of values of λ, as opposed to holding for all large enough λ. Such primitives are called *infinitely-often*, and all primitives constructed in this section and Section 4 are infinitely-often primitives.

On the other hand, if we assume that for every $\mathcal{G} \in \mathcal{C}$, there exists λ_0 such that for all $\lambda > \lambda_0$, $f_\lambda \not\equiv g_\lambda$ we can achieve the regular stronger notion of security (that holds for all large enough security parameters) in each case by the same techniques.

Theorem 3.2 (OWFs, PRGs from PREs). *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two function classes satisfying the following conditions:*

1. *Any function family in $\mathcal{C}_2$ has a surjective PRE computable in $\mathcal{C}_1$.*
2. *$\mathcal{C}_2 \nsubseteq \mathcal{C}_1$.*
3. *$\mathcal{C}_1$ is closed under a constant number of compositions.*
4. *$\mathcal{C}_1$ is non-uniform or randomized.*
5. *$\mathcal{C}_1$ can compute arbitrary thresholds.*

Then:

1. *There is a $\mathcal{C}_1$-OWF against $\mathcal{C}_1$.*
2. *There is a $\mathcal{C}_1$-PRG against $\mathcal{C}_1$ with non-zero additive stretch.*

Theorem 3.2 in effect shows that the existence of a language with PREs outside $\mathcal{C}_1$ implies the existence of one way functions and pseudorandom generators computable in $\mathcal{C}_1$ secure against $\mathcal{C}_1$. Instances of classes that satisfy its hypothesis (apart from $\mathcal{C}_2 \nsubseteq \mathcal{C}_1$) include NC^1 and BPP. Note that this theorem does not provide constructions against AC^0 because AC^0 cannot compute arbitrary thresholds.

Proof Sketch. We start with a language in $\mathcal{C}_2 \setminus \mathcal{C}_1$ described by a function family $\mathcal{F} = \{f_\lambda\}$. Let $\widehat{\mathcal{F}} = \left\{\widehat{f_\lambda}\right\}$ be its randomized encoding. Say f_λ takes inputs from $\{0,1\}^\lambda$. Then the PRG/OWF for parameter λ is the function $g_\lambda(r) = \widehat{f_\lambda}(0^\lambda, r)$.

Without loss of generality, say $f_\lambda(0^\lambda) = 0$ and $f_\lambda(z_1) = 1$ for some z_1. To show pseudorandomness, we first observe that, by the perfectness of the randomized encoding, the uniform distribution can be generated as an equal convex combination of $\widehat{f_\lambda}(0^\lambda, r)$ and $\widehat{f_\lambda}(z_1, r)$. The advantage in distinguishing $g_\lambda(r) = \widehat{f_\lambda}(0^\lambda, r)$ from uniform can hence be used to decide if a given input x is in the language because an equal convex combination of $\widehat{f_\lambda}(0^\lambda, r)$ and $\widehat{f_\lambda}(x, r)$ will be identical to $\widehat{f_\lambda}(0^\lambda, r)$ if $f_\lambda(x) = f_\lambda(0) = 0$, and otherwise will be identical to uniform.

We require the class to be closed under composition and to be able to compute thresholds in order to be able to amplify the success probability. The non-zero additive stretch comes from the fact that the PRE is stretch-preserving.

4 PKE against NC^1 from worst-case assumptions

In Theorem 3.2 we saw that we can construct one way functions and PRGs with a small stretch generically from Perfect Randomized Encodings (PREs) starting from worst-case hardness assumptions. We do not know how to construct Public Key Encryption (PKE) in a similar black-box fashion. In this section, we use certain algebraic properties of a specific construction of PREs for functions in $\oplus\mathsf{L/poly}$ due to Ishai-Kushilevitz [IK00] to construct Public Key Encryption and Collision Resistant Hash Functions against NC^1 that are computable in $\mathsf{AC}^0[2]$ under the assumption that $\oplus\mathsf{L/poly} \not\subseteq \mathsf{NC}^1$. We state the necessary implications of their work here. We start by describing sampling procedures for some relevant distributions in Construction 4.1.

In the randomized encodings of [IK00], the output of the encoding of a function f on input x is a matrix $\mathbf{M}$ sampled identically to $\mathbf{R}_1\mathbf{M}_0^\lambda\mathbf{R}_2$ when $f(x) = 0$ and identically to $\mathbf{R}_1\mathbf{M}_1^\lambda\mathbf{R}_2$ when $f(x) = 1$, where $\mathbf{R}_1 \leftarrow \mathsf{LSamp}(\lambda)$ and $\mathbf{R}_2 \leftarrow \mathsf{RSamp}(\lambda)$. Notice that $\mathbf{R}_1\mathbf{M}_1^\lambda\mathbf{R}_2$ is full rank, while $\mathbf{R}_1\mathbf{M}_0^\lambda\mathbf{R}_2$ has rank $(\lambda - 1)$. The public key in our encryption scheme is a sample $\mathbf{M}$ from $\mathbf{R}_1\mathbf{M}_0^\lambda\mathbf{R}_2$, and the secret key is a vector $\mathbf{k}$ in the kernel of $\mathbf{M}$. An encryption of 0 is a random vector in the row-span of $\mathbf{M}$ (whose inner product with $\mathbf{k}$ is hence 0), and an encryption of 1 is a random vector that is not in the row-span of $\mathbf{M}$ (whose inner product with $\mathbf{k}$ is non-zero). Decryption is simply inner product with $\mathbf{k}$. (This is very similar to the cryptosystem in [ABW10] albeit without the noise that is added there.)

Security follows from the fact that under our hardness assumption $\mathbf{M}$ is indistinguishable from $\mathbf{R}_1\mathbf{M}_1^\lambda\mathbf{R}_2$ (see Theorem 4.2), which has an empty kernel, and so when used as the public key results in identical distributions of encryptions of 0 and 1.

Construction 4.1 Sampling distributions from [IK00]

Let $\mathbf{M}_0^n$ and $\mathbf{M}_1^n$ be the following $n \times n$ matrices:

$$\mathbf{M}_0 = \begin{pmatrix} 0 & & \cdots & 0 & 0 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}, \mathbf{M}_1 = \begin{pmatrix} 0 & & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & 0 & \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

$\mathsf{LSamp}(n)$:

1. Output an $n \times n$ upper triangular matrix where all entries in the diagonal are 1 and all other entries in the upper triangular part are chosen at random.

$\mathsf{RSamp}(n)$:

1. Sample at random $\mathbf{r} \leftarrow \{0,1\}^{n-1}$.
2. Output the following $n \times n$ matrix:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & \Big| \\ 0 & 1 & \ddots & \vdots & r \\ \vdots & \ddots & \ddots & 0 & \Big| \\ 0 & \cdots & 0 & 1 & \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix}$$

Theorem 4.1 (Public Key Encryption Against NC^1). *Assume $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$. Then, the tuple of families $(\mathcal{K}eyGen, \mathcal{E}nc, \mathcal{D}ec)$ defined in Construction 4.2 is an $\mathsf{AC}^0[2]$-Public Key Encryption Scheme against NC^1.*

Before beginning with the proof, we describe some properties of the construction. We first begin with two sampling procedures that correspond to sampling from $\widehat{f}(x,\cdot)$ when $f(x) = 0$ or $f(x) = 1$ as described earlier. We describe these again in Construction 4.3.

Theorem 4.2 ([IK00, AIK04]). *For any boolean function family $\mathcal{F} = \{f_\lambda\}$ in $\oplus\mathsf{L}/\mathsf{poly}$, there is a polynomial n such that for any λ, f_λ has a PRE $\widehat{f_\lambda}$ such that the distribution of $\widehat{f_\lambda}(x)$ is identical to $\mathsf{ZeroSamp}(n(\lambda))$ when $f_\lambda(x) = 0$ and is identical to $\mathsf{OneSamp}(n(\lambda))$ when $f_\lambda(x) = 1$.*

This implies that if some function in $\oplus\mathsf{L}/\mathsf{poly}$ is hard to compute on the worst-case then it is hard to distinguish between samples from $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$. In particular, the following lemma follows immediately from the observation that $\mathsf{ZeroSamp}$ and $\mathsf{OneSamp}$ can be computed in NC^1.

Lemma 4.3. *If $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$, then there is a polynomial n and a negligible function ν such that for any family $\mathcal{F} = \{f_\lambda\}$ in NC^1, for an infinite number of*

Construction 4.2 Public Key Encryption

Let λ be the security parameter. Let $\mathbf{M}_0^\lambda$ be the $\lambda \times \lambda$ matrix described in Construction 4.1. Define the families $\mathcal{K}eyGen = \{\mathsf{KeyGen}_\lambda\}$, $\mathcal{E}nc = \{\mathsf{Enc}_\lambda\}$, and $\mathcal{D}ec = \{\mathsf{Dec}_\lambda\}$ as follows.

```
$\mathsf{KeyGen}_\lambda$:
  1. Sample $\mathbf{R}_1 \leftarrow \mathsf{LSamp}(\lambda)$ and $\mathbf{R}_2 \leftarrow \mathsf{RSamp}(\lambda)$.
  2. Let $\mathbf{k} = (\mathbf{r}\ 1)^T$ be the last column of $\mathbf{R}_2$.
  3. Compute $\mathbf{M} = \mathbf{R}_1\mathbf{M}_0^\lambda\mathbf{R}_2$.
  4. Output $(\mathsf{pk} = \mathbf{M}, \mathsf{sk} = \mathbf{k})$.

$\mathsf{Enc}_\lambda(\mathsf{pk} = \mathbf{M}, b)$:
  1. Sample $\mathbf{r} \in \{0,1\}^\lambda$.
  2. Let $\mathbf{t}^T = (0\ \dots\ 0\ 1)$, of length $\lambda$.
  3. Output $\mathbf{c}^T = \mathbf{r}^T\mathbf{M} + b\mathbf{t}^T$.

$\mathsf{Dec}_\lambda(\mathsf{sk} = \mathbf{k}, \mathbf{c})$:
  1. Output $\langle \mathbf{c}, \mathbf{k} \rangle$.
```

Construction 4.3 Sampling procedures

```
$\mathsf{ZeroSamp}(n)$: $\widehat{f}(x,r)$ where $f(x) = 0$
  1. Sample $\mathbf{R}_1 \leftarrow \mathsf{LSamp}(n)$ and $\mathbf{R}_2 \leftarrow \mathsf{RSamp}(n)$.
  2. Output $\mathbf{R}_1\mathbf{M}_0\mathbf{R}_2$.

$\mathsf{OneSamp}(n)$: $\widehat{f}(x,r)$ where $f(x) = 1$
  1. Sample $\mathbf{R}_1 \leftarrow \mathsf{LSamp}(n)$ and $\mathbf{R}_2 \leftarrow \mathsf{RSamp}(n)$.
  2. Output $\mathbf{R}_1\mathbf{M}_1\mathbf{R}_2$.
```

values of λ,

$$\left| \Pr_{\mathbf{M} \leftarrow \mathsf{ZeroSamp}(n(\lambda))} [f_\lambda(\mathbf{M}) = 1] - \Pr_{\mathbf{M} \leftarrow \mathsf{OneSamp}(n(\lambda))} [f_\lambda(\mathbf{M}) = 1] \right| \leq \nu(\lambda)$$

Lemma 4.3 can now be used to prove Theorem 4.1 as described in Section 1.1. We defer the details to the full version.

Remark 4.4. The computation of the PRE from [IK00] can be moved to NC^0 by techniques noted in [IK00] itself. Using similar techniques with Construction 4.2 gives us a Public Key Encryption scheme with encryption in NC^0 and decryption and key generation in $\mathsf{AC}^0[2]$. The impossibility of decryption in NC^0, as noted in [AIK04], continues to hold in our setting.

Remark 4.5. (This was pointed out to us by Abhishek Jain.) The above PKE scheme has what are called, in the terminology of [PVW08], "message-lossy" public keys – in this case, this is simply $\mathbf{M}$ when sampled from $\mathsf{OneSamp}$, as in the proof above. Such schemes may be used, again by results from [PVW08], to construct protocols for Oblivious Transfer where the honest parties are computable in NC^1 and which are secure against semi-honest NC^1 adversaries under the same assumptions (that $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$).

4.1 Collision Resistant Hashing

Note that again, due to the linearity of decryption, Construction 4.2 is additively homomorphic – if c_1 and c_2 are valid encryptions of m_1 and m_2, $(c_1 \oplus c_2)$ is a valid encryption of $(m_1 \oplus m_2)$. Furthermore, the size of ciphertexts does not increase when this operation is performed. Given these properties, we can use the generic transformation from additively homomorphic encryption to collision resistance due to [IKO05], along with the observation that all operations involved in the transformation can still be performed in $\mathsf{AC}^0[2]$, to get the following.

Theorem 4.6. *Assume $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \mathsf{NC}^1$. Then, for any constant $c < 1$ and function s such that $s(n) = O(n^c)$, there exists an $\mathsf{AC}^0[2]$-CRHF against NC^1 with compression s.*

5 Cryptography Without Assumptions

In this section, we present some constructions of primitives unconditionally secure against AC^0 adversaries that are computable in AC^0. This is almost the largest complexity class (after AC^0 with MOD gates) for which we can hope to get such unconditional results owing to a lack of better lower bounds. In this section, we present constructions of PRGs with arbitrary polynomial stretch, Weak PRFs, Symmetric Key Encryption, and Collision Resistant Hash Functions. We end with a candidate for Public Key Encryption against AC^0 that we are unable to prove secure, but also do not have an attack against.

5.1 High-Stretch Pseudo-Random Generators

We present here a construction of Pseudo-Random Generators against AC^0 with arbitrary polynomial stretch that can be computed in AC^0. In fact, the same techniques can be used to obtain constant stretch generators computable in NC^0.

The key idea behind the construction is the following: [Bra10] implies that for any constant ϵ, an n^ϵ-wise independent distribution will fool AC^0 circuits of arbitrary constant depth. So, being able to sample such distributions in AC^0 suffices to construct good PRGs. As shown in Section 2.3, if $\mathbf{H}$ is the parity-check matrix of a code with large distance d, then the distribution $\mathbf{Hx}$ is d-wise independent for $\mathbf{x}$ being a uniformly random vector (by Lemma 2.6). Further, as was also shown in Section 2.3, even for rather large d there are such matrices $\mathbf{H}$ that are sparse, allowing us to compute the product $\mathbf{Hx}$ in AC^0.

Theorem 5.1 (PRGs against AC^0). *For any polynomial l, the family $\mathcal{F}^l$ from Construction 5.1 is an AC^0-PRG with multiplicative stretch $\left(\frac{l(\lambda)}{\lambda}\right)$.*

Construction 5.1 AC^0-PRG against AC^0

For any polynomial l, we define the family $\mathcal{F}^l = \left\{ f^l_\lambda : \{0,1\}^\lambda \to \{0,1\}^{l(\lambda)} \right\}$ as follows. Lemma 2.5 implies for large λ, there is an $[l(\lambda), (l(\lambda) - \lambda)]_2$ linear code with minimum distance at least $\frac{\lambda}{\log^3(\lambda)}$ whose parity check matrix has $\log^2(\lambda)$ non-zero entries in each row. Denote this parity check matrix by $\mathbf{H}_{l,\lambda}$. The dimensions of $\mathbf{H}_{l,\lambda}$ are $l(\lambda) \times \lambda$.

$$f^l_\lambda(\mathbf{x}) = \mathbf{H}_{l,\lambda}\mathbf{x}$$

Proof. For any l, the most that needs to be done to compute $f^l_\lambda(x)$ is computing the product $\mathbf{H}_{l,\lambda}\mathbf{x}$. We know that each row of $\mathbf{H}_{l,\lambda}$ contains at most $\log^2(\lambda)$ non-zero entries. Hence, by Lemma 2.3, $\mathcal{F}^l$ is in AC^0. The multiplicative stretch being $\left(\frac{l(\lambda)}{\lambda}\right)$ is also easily verified.

For pseudo-randomness, we observe that the product $\mathbf{H}_{l,\lambda}\mathbf{x}$ is $\Omega\left(\frac{\lambda}{\log^3(\lambda)}\right)$-wise independent, by Lemma 2.6. And hence, Theorem 2.2 implies that this distribution is pseudo-random to adversaries in AC^0.

5.2 Weak Pseudo-Random Functions

In this section, we describe our construction of Weak Pseudo-Random Functions against AC^0 computable in AC^0 (Construction 5.2). Roughly, we know that for a random sparse matrix $\mathbf{H}$, $(\mathbf{H}, \mathbf{Hk})$ is indistinguishable from $(\mathbf{H}, \mathbf{r})$ where $\mathbf{r}$ and $\mathbf{k}$ are chosen uniformly at random. We choose the key of the PRF to be a random vector $\mathbf{k}$. On an input $\mathbf{x}$, the strategy is to use the input $\mathbf{x}$ to generate a sparse vector $\mathbf{y}$ and then take the inner product $\langle \mathbf{y}, \mathbf{k} \rangle$.

Construction 5.2 AC^0-PRF against AC^0

Let $\mathcal{I}^t = \left\{ ip^t_\lambda \right\}$ be the inner product family with threshold promise t described in Lemma 2.3. Define families $\mathcal{K}ey\mathcal{G}en = \{\mathsf{KeyGen}_\lambda\}$ and $\mathcal{E}val = \{\mathsf{Eval}_\lambda\}$ as follows.

KeyGen_λ:

1. Output a random vector $\mathbf{k} \leftarrow \{0,1\}^{\lfloor \lambda \rfloor_2}$.

$\mathsf{Eval}_\lambda(\mathbf{k}, \mathbf{r})$:

1. Compute $\mathbf{y} \leftarrow \mathsf{SRSamp}(\lfloor \lambda \rfloor_2, \log^2(\lfloor \lambda \rfloor_2), \mathbf{r})$.
2. Output $ip^{\log^2(\lambda)}_{\lfloor \lambda \rfloor_2}(\mathbf{k}, \mathbf{y})$.

Theorem 5.2 (PRFs against AC^0). *The pair of families $(\mathcal{K}ey\mathcal{G}en, \mathcal{E}val)$ defined in Construction 5.2 is a Weak AC^0-PRF against AC^0.*

The intuitive reason one would think Construction 5.2 might be pseudo-random is that a collection of random function values from a randomly sampled key seems to contain the same information as $(\mathbf{H}, \mathbf{H}\mathbf{k})$ where $\mathbf{k}$ is sampled uniformly at random and $\mathbf{H}$ is sampled using SMSamp: a matrix with sparse rows. We know from Lemma 2.5 that except with negligible probability, $\mathbf{H}$ is going to be the parity check matrix of a code with large distance, and when it is, the arguments from Section 5.1 show that $(\mathbf{H}, \mathbf{H}\mathbf{k})$ is indistinguishable from $(\mathbf{H}, \mathbf{r})$, where $\mathbf{r}$ is sampled uniformly at random.

The only fact that prevents this from functioning as a proof is that what the adversary gets is not $(\mathbf{y}, \langle \mathbf{y}, \mathbf{k} \rangle)$ where $\mathbf{y}$ is an output of SRSamp, but rather $(\mathbf{r}, \langle \mathbf{y}, \mathbf{k} \rangle)$, where $\mathbf{r}$ is randomness that when used in SRSamp gives $\mathbf{y}$. One way to show that this is still pseudo-random is to reduce the case where the input is $(\mathbf{y}, \langle \mathbf{y}, \mathbf{x} \rangle)$ to the case where the input is $(\mathbf{r}, \langle \mathbf{y}, \mathbf{x} \rangle)$ using an AC^0-reduction. To do this, one would need an AC^0 circuit that would, given $\mathbf{y}$, sample from a distribution close to the uniform distribution over $\mathbf{r}$'s that cause SRSamp to output $\mathbf{y}$ when used as randomness. We implement this proof strategy in the full version.

Construction 5.2 of Weak PRFs achieves only quasi-polynomial security — that is, there is no guarantee that some AC^0 adversary may not have an inverse quasi-polynomial advantage in distinguishing between the PRF and a random function. Due to the seminal work of Linial-Mansour-Nisan [LMN93] and subsequent improvements in [Tal14], we know that this barrier is inherent and we cannot hope for exponential security – see the full version for details.

5.3 Symmetric Key Encryption

In this section, we present a Symmetric Key Encryption scheme against AC^0 computable in AC^0, which is also additively homomorphic – a property that shall be useful in constructing Collision Resistant Hash Functions later on.

In Section 5.2, we saw a construction of Weak PRFs. And Weak PRFs give us Symmetric Key Encryption generically (where $\mathsf{Enc}(\mathbf{k}, b) = (\mathbf{r}, \mathsf{PRF}(\mathbf{k}, \mathbf{r}) \oplus b)$). For the Weak PRF construction from Section 5.2, this implied scheme also happens to be additively homomorphic. But it has the issue that the last bit of the ciphertext is an almost unbiased bit and hence it is not feasible to do more than $\mathsf{polylog}(\lambda)$ homomorphic evaluations on collections of ciphertexts in AC^0. So, we construct a different Symmetric Key Encryption scheme that does not suffer from this drawback and is still additively homomorphic. Then we will use this scheme to construct Collision Resistant Hash Functions. This scheme is described in Construction 5.3. In this scheme we choose the ciphertext by performing rejection sampling in parallel. For encrypting a bit b, we sample a ciphertext $\mathbf{c}$ such that $\mathbf{c}$ is sparse and $\langle \mathbf{c}, \mathbf{k} \rangle = b$. This ensures that the we have an additively homomorphic scheme where all the bits are sparse.

Theorem 5.3 (Symmetric Encryption Against AC^0). *The tuple of families $(\mathcal{K}ey\mathcal{G}en, \mathcal{E}nc, \mathcal{D}ec)$ defined in Construction 5.3 is an AC^0-Symmetric-Key Encryption Scheme against AC^0.*

Construction 5.3 AC^0-Symmetric Key Encryption against AC^0

Let $\mathcal{I}^t = \left\{ip_\lambda^t\right\}$ be the inner product family with threshold promise t described in Lemma 2.3. Define families $\mathcal{K}eyGen = \{\mathsf{KeyGen}_\lambda\}$, $\mathcal{E}nc = \{\mathsf{Enc}_\lambda\}$, and $\mathcal{D}ec = \{\mathsf{Dec}_\lambda\}$ as below.

KeyGen_λ:

1. Output $\mathbf{k} \leftarrow \{0,1\}^{\lfloor\lambda\rfloor_2}$.

$\mathsf{Enc}_\lambda(\mathbf{k}, b)$:

1. For $i \in [\lambda]$, sample $\mathbf{c}_i \leftarrow \mathsf{SRSamp}(\lfloor\lambda\rfloor_2, \log^2(\lfloor\lambda\rfloor_2))$.
2. Choose the first i such that $\langle \mathbf{c}_i, \mathbf{k}\rangle = b$.
3. If such an i exists, output $\mathbf{c}_i$, else output $0^{\lfloor\lambda\rfloor_2}$.

$\mathsf{Dec}_\lambda(\mathbf{k}, \mathbf{c})$:

1. Output $ip_{\lfloor\lambda\rfloor_2}^{\log^2(\lambda)}(\mathbf{k}, \mathbf{c})$.

The key idea behind the proof is showing that for most keys $\mathbf{k}$, the distribution of a uniformly random bit along with its encryption, that is,

$$D_1 = \{(b, \mathsf{Enc}_\lambda(\mathbf{k}, b)) \mid b \leftarrow \{0,1\}\}$$

is statistically close to the distribution of a random sparse vector along with its inner product with $\mathbf{k}$, that is,

$$D_2 = \left\{(\langle \mathbf{r}, \mathbf{k}\rangle, \mathbf{r}) \mid \mathbf{r} \leftarrow \mathsf{SRSamp}(\lambda, \log^2 \lambda)\right\}$$

The second distribution is similar to the one that came up in the security proof of the weak PRF construction earlier, where we effectively showed that we can replace $\langle \mathbf{r}, \mathbf{k}\rangle$ with an independent random bit without being caught by AC^0 adversaries. We defer the complete proof to the full version.

5.4 Collision Resistant Hash Functions

To construct Collision Resistant Hash Functions (CRHFs), we use the additive homomorphism of the Symmetric Key Encryption scheme constructed in Section 5.3. Each function in the family of hash functions is given by a matrix whose columns are ciphertexts from the encryption scheme, and evaluation is done by treating the input as a column vector and computing its product with this matrix (effectively computing a linear combination of ciphertexts). To find collisions, the adversary needs to come up with a vector in the kernel of this matrix. We show that constant depth circuits of polynomial size cannot do this for most such matrices. This is because the all-zero vector is a valid encryption of 0 in our encryption scheme, and as this scheme is additively homomorphic, finding a subset of ciphertexts that sum to zero is roughly the same as finding a subset of the corresponding messages that sum to 0, and this is a violation of semantic security.

Construction 5.4 AC^0-CRHFs against AC^0

Let $\mathcal{I}^t = \{ip^t_\lambda\}$ be the inner product family with threshold promise t described in Lemma 2.3. Let $(\mathcal{K}eyGen^{Enc}, \mathcal{E}nc^{Enc})$ be the SKE scheme from Construction 5.3. Let $l(\lambda) = \left\lfloor \frac{\lambda}{s(\lambda)} \right\rfloor_2$.

For any $s : \mathbb{N} \to \mathbb{N}$ such that $s(\lambda) = O(\log^c(\lambda))$ for some constant c, we define the families $\mathcal{K}eyGen^s = \{\mathsf{KeyGen}^s_\lambda\}$ and $\mathcal{E}val^s = \{\mathsf{Eval}^s_\lambda\}$ as follows.

$\mathsf{KeyGen}^s_\lambda$:

1. Sample $\mathbf{k} \leftarrow \mathsf{KeyGen}^{Enc}_{l(\lambda)}$, and $b_1, \ldots, b_\lambda \leftarrow \{0,1\}$.
2. Output $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_\lambda)$, where $\mathbf{m}_i \leftarrow \mathsf{Enc}^{Enc}_{l(\lambda)}(\mathbf{k}, b_i)$.

$\mathsf{Eval}^s_\lambda(\mathbf{M}, \mathbf{x})$:

1. Note that $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_\lambda)$, where each $\mathbf{m}_i$ is of length $l(\lambda)$.
2. For $j \in [l(\lambda)]$, let $r_j = (\mathbf{m}_{1j}, \ldots, \mathbf{m}_{\lambda j})$ (the jth bit of each $\mathbf{m}_i$).
3. Output $(h_1, \ldots, h_{l(\lambda)})$, where $h_j = ip_\lambda^{4s(\lambda)\log^2(\lambda)}(r_j, \mathbf{x})$.

Theorem 5.4 (CRHFs Against AC^0). *For any polylogarithmic function s, the pair of families $(\mathcal{K}eyGen^s, \mathcal{E}val^s)$, from Construction 5.4 is an AC^0-CRHF with compression s.*

We refer the reader to the sketch of the proof of Theorem 1.7 (an informal version of Theorem 5.4) towards the end of Section 1.1 and leave the proof of Theorem 5.4 to the full version.

5.5 Candidate Public Key Encryption Scheme

In Lemma 2.7 we showed that the distribution $(\mathbf{A}, \mathbf{A}\mathbf{k})$ where $\mathbf{A}$ was sampled as a sparse matrix and $\mathbf{k}$ was a random vector is indistinguishable from $(\mathbf{A}, \mathbf{r})$ where $\mathbf{r}$ is also a random vector, for a wide range of parameters. We need at least one of the two $\mathbf{A}$ or $\mathbf{k}$ to be sparse to enable the computation of $\mathbf{A}\mathbf{k}$ in AC^0. If we make the analogous indistinguishability assumption with the key being sparse – that is, that $(\mathbf{A}, \mathbf{A}\mathbf{k})$ is indistinguishable from $(\mathbf{A}, \mathbf{r})$ where $\mathbf{A} \leftarrow \{0,1\}^{\lambda\times\lambda}$, $\mathbf{k} \leftarrow \mathsf{SRSamp}(\lambda, \log^2\lambda)$ and $\mathbf{r} \leftarrow \{0,1\}^\lambda$ – this allows us to construct a Public Key Encryption scheme against AC^0 computable in AC^0.

This is presented in Construction 5.5, and is easily seen to be secure under Assumption 5.5. This candidate is very similar to the LPN based cryptosystem due to Alekhnovich [Ale03]. Note that while the correctness of decryption in Construction 5.5 is not very good, this may be easily amplified by repetition without losing security, as the error is one-sided.

Assumption 5.5 *Distributions $D_1 = (\mathbf{A}, \mathbf{A}\mathbf{k})$ where $\mathbf{A} \leftarrow \{0,1\}^{\lambda\times\lambda}$, $\mathbf{k} \leftarrow$ $\mathsf{SRSamp}(\lambda, \log^2\lambda)$ and $D_2 = (\mathbf{A}, \mathbf{r})$ where $\mathbf{r} \leftarrow \{0,1\}^\lambda$ are indistinguishable by AC^0 adversaries with non-negligible advantage.*

Construction 5.5 Public key encryption

Let $\mathcal{I}^t = \{ip^t_\lambda\}$ be the inner product family with threshold promise t described in Lemma 2.3. Define families $\mathcal{K}ey\mathcal{G}en = \{\mathsf{KeyGen}_\lambda\}$, $\mathcal{E}nc = \{\mathsf{Enc}_\lambda\}$, and $\mathcal{D}ec = \{\mathsf{Dec}_\lambda\}$ as below.

KeyGen_λ:

1. Sample $\mathbf{A} \leftarrow \{0,1\}^{\lambda\times\lambda-1}$, $\mathbf{k} \leftarrow \mathsf{SRSamp}(\lambda-1, \log^2\lambda)$
2. Output $(\mathsf{pk}, \mathsf{sk}) = ((\mathbf{A}, \mathbf{Ak}), \mathbf{k} \circ 1)$.

$\mathsf{Enc}_\lambda(\mathsf{pk}, b)$:

1. If $b = 0$, sample $\mathbf{t} \leftarrow \mathsf{SRSamp}(\lambda, \log^2\lambda)$ and output $\mathbf{t}^T\mathsf{pk}$
2. Else if $b = 1$, output $\mathbf{t} \leftarrow \{0,1\}^\lambda$

$\mathsf{Dec}_\lambda(\mathsf{sk}, \mathbf{c})$:

1. Output $ip^{\log^2(\lambda)}_{\lfloor\lambda\rfloor_2}(\mathsf{sk}, \mathbf{c})$.

The most commonly used proof technique in this paper – showing k-wise independence for a large k – cannot be used to prove the security of this scheme because due to the sparsity of the key, the distribution $(\mathbf{A}, \mathbf{Ak})$ is not k-wise independent.

Conclusions and Open Questions. We construct various cryptographic primitives secure against parallel-time-bounded adversaries. Our constructions against AC^0 are unconditional whereas our constructions against NC^1 require the assumption that $\mathsf{NC}^1 \neq \oplus\mathsf{L/poly}$. Our constructions make use of circuit lower bounds [Bra10] and non-black-box use of randomized encodings for logspace classes [IK00].

There are several open questions that arise out of this work. Perhaps the most immediate are:

1. Unconditional lower-bounds are known for slightly larger classes like $\mathsf{AC}^0[p]$ when p is a prime power. Can we get cryptographic primitives from those lower-bounds?
2. Construct a public key encryption scheme secure against AC^0, either by proving the security of our candidate proposal (see Section 5.5) or by completely different means.
   Natural ways of doing this lead us to a fascinating question about the complexity of AC^0 circuits. Braverman [Bra10] shows that *any* n^ϵ-wise independent distribution fools all AC^0 circuits. Our candidate encryption, however, produces ciphertexts that come from a $\log^c(n)$-wise distribution for some constant c. This raises the following question: *Can we show* some *fixed* $\mathsf{polylog}$*-wise independent distribution (that is* not *n^ϵ-wise independent) that fools* AC^0 *circuits of arbitrary depth?* (This question came up during discussions with Li-Yang Tan.)
3. We relied on the assumption that $\oplus\mathsf{L/poly} \not\subseteq \mathsf{NC}^1$ to construct primitives secure against NC^1. It would be desirable to relax the assumption to $\mathsf{P} \not\subseteq \mathsf{NC}^1$.

A related extension of Merkle's work is to construct a public-key encryption scheme resistant against $O(n^c)$ time adversaries (for some $c > 2$) under worst-case hardness assumptions.

Acknowledgements

We thank Prabhanjan Ananth for several useful discussions towards the beginning of the project. We would also like to thank the anonymous reviewers for their careful comments.

References

AB84. Miklós Ajtai and Michael Ben-Or. A theorem on probabilistic constant depth computations. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 471–474, 1984.

ABW10. Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180. ACM, 2010.

AGGM06. Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. On basing one-way functions on np-hardness. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 701–710, 2006.

AGHP93. Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Addendum to "simple construction of almost k-wise independent random variables". *Random Struct. Algorithms*, 4(1):119–120, 1993.

AIK04. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nco. In *FOCS 2004: 45th Annual IEEE Symposium on Foundations of Computer Science: proceedings: 17-19 October, 2004, Rome, Italy*, page 166. IEEE Computer Society Press, 2004.

Ajt83. M. Ajtai. 11-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1 – 48, 1983.

Ale03. Michael Alekhnovich. More on average case vs approximation complexity. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 298–307. IEEE, 2003.

App14. Benny Applebaum. Cryptography in nc 0. In *Cryptography in Constant Parallel Time*, pages 33–78. Springer, 2014.

AR99. Yonatan Aumann and Michael O Rabin. Information theoretically secure communication in the limited storage space model. In *Advances in CryptologyCRYPTO99*, pages 65–79. Springer, 1999.

AR15. Benny Applebaum and Pavel Raykov. On the relationship between statistical zero-knowledge and statistical randomized encodings. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:186, 2015.

Bar86. David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in nc^1. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 1–5, 1986.

BB15. Andrej Bogdanov and Christina Brzuska. On basing size-verifiable one-way functions on np-hardness. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 2015.

BGI$^+$01. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology - CRYPTO 2001*, pages 1–18, 2001.

BGI08. Eli Biham, Yaron J. Goren, and Yuval Ishai. Basing weak public-key cryptography on strong one-way functions. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008.*, volume 4948 of *Lecture Notes in Computer Science*, pages 55–72. Springer, 2008.

BM84. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73. ACM, 1993.

Bra10. Mark Braverman. Polylogarithmic independence fools AC^0 circuits. *J. ACM*, 57(5), 2010.

BT03. Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 308–317. IEEE Computer Society, 2003.

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *FOCS*, pages 97–106. IEEE, 2011. Invited to SIAM Journal on Computing.

CM97. Christian Cachin and Ueli Maurer. Unconditional security against memory-bounded adversaries. In *Advances in CryptologyCRYPTO'97*, pages 292–306. Springer, 1997.

DH76. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

DM04. Stefan Dziembowski and Ueli Maurer. On generating the initial key in the bounded-storage model. In *Advances in Cryptology-EUROCRYPT 2004*, pages 126–137. Springer, 2004.

FSS84. Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

Gal62. Robert G. Gallager. Low-density parity-check codes. *IRE Trans. Information Theory*, 8(1):21–28, 1962.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

GGH$^+$13. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS 2013*, pages 40–49, 2013.

GGM86. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.

GM82. Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC 1982*, pages 365–377, 1982.

GMR85. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304, 1985.

GR12. Shafi Goldwasser and Guy N. Rothblum. How to compute in the presence of leakage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 31–40, 2012.

Hås86. Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20, 1986.

Has87. Johan Hastad. One-way permutations in nc 0. *Information Processing Letters*, 26(3):153–155, 1987.

Hås14. Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014.

HILL99. Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

IK00. Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 294–304. IEEE, 2000.

IKO05. Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Sufficient conditions for collision-resistant hashing. In *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, pages 445–456, 2005.

IL89. Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 230–235. IEEE Computer Society, 1989.

IR88. Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 8–26, 1988.

LMN93. Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM (JACM)*, 40(3):607–620, 1993.

Mau92. Ueli M Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

Mer78. Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, 1978.

MP06. Silvio Micali and Rafael Pass. Local zero knowledge. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 306–315, 2006.

MST06. Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On epsilon-biased generators in nc^0. *Random Struct. Algorithms*, 29(1):56–81, 2006.

NW94. Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

PVW08. Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, pages 554–571, 2008.

RAD78. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation*, pages 169–177. Academic Press, 1978.

Raz87. A. A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

RSA78. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

RST15. Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:65, 2015.

RW91. Prabhakar Ragde and Avi Wigderson. Linear-size constant-depth polylog-treshold circuits. *Inf. Process. Lett.*, 39(3):143–146, 1991.

Smo87. Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82, 1987.

SW14. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 475–484. ACM, 2014.

Tal14. Avishay Tal. Tight bounds on the fourier spectrum of ac^0. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:174, 2014.

TX13. Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of AC0. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 242–247, 2013.

Vad04. Salil P Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Journal of Cryptology*, 17(1):43–77, 2004.

Vio12. Emanuele Viola. The complexity of distributions. *SIAM Journal on Computing*, 41(1):191–218, 2012.

Yao82. Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.