Fast Pseudorandom Functions Based on Expander Graphs

Benny Applebaum[*]   Pavel Raykov[*]

May 22, 2016

Abstract

We present direct constructions of pseudorandom function (PRF) families based on Goldreich's one-way function. Roughly speaking, we assume that non-trivial local mappings $f : \{0,1\}^n \to \{0,1\}^m$ whose input-output dependencies graph form an expander are hard to invert. We show that this one-wayness assumption yields PRFs with relatively low complexity. This includes weak PRFs which can be computed in linear time of $O(n)$ on a RAM machine with $O(\log n)$ word size, or by a depth-3 circuit with unbounded fan-in AND and OR gates (AC0 circuit), and standard PRFs that can be computed by a quasilinear size circuit or by a constant-depth circuit with unbounded fan-in AND, OR and Majority gates (TC0).

Our proofs are based on a new search-to-decision reduction for expander-based functions. This extends a previous reduction of the first author (STOC 2012) which was applicable for the special case of *random* local functions. Additionally, we present a new family of highly efficient hash functions whose output on exponentially many inputs jointly forms (with high probability) a good expander graph. These hash functions are based on the techniques of Miles and Viola (Crypto 2012).

Although some of the reductions provide only relatively weak security guarantees, we believe that our constructions yield novel approach for designing PRFs, and consequently enrich the study of pseudorandomness.

1 Introduction

A pseudorandom function (PRF) is a family of efficiently computable functions with the property that the input-output behavior of a random instance of the family is "computationally indistinguishable" from that of a truly random function. Abstractly, such functions provide a "direct access" to an exponentially long pseudorandom string. Since their discovery by Goldreich, Goldwasser and Micali [GGM86], PRFs have played a central role in cryptography and complexity theory. Correspondingly, the question of minimizing the complexity of PRFs has attracted a considerable amount of attention.

Indeed, apart of being a fundamental object, fast PRFs are strongly motivated by a wide range of applications. Being the core component of symmetric cryptography, highly-efficient PRFs directly imply highly-efficient implementations of Private-Key cryptosystems, Message-Authentication Codes, and Identification Schemes. Fast pseudorandom objects (PRFs and PRGs) can be also used to speed-up several expensive Cryptomania-type applications. For example, secure computation protocols, functional encryption schemes, and program obfuscators that efficiently support a PRF functionality can be bootstrapped with relatively minor cost to general functionalities (cf., [DI05, IKOS08, GVW12, App14]). Interestingly, for these applications parallel-complexity (e.g., circuit depth) seems to be the main relevant complexity measure (affecting round complexity or the number of multilinear levels), while time (e.g., circuit size) is secondary. Another somewhat different motivation comes from the theory of computational complexity. PRFs with low-complexity shed light on the power of low-complexity functions, and partially explain our inability to analyze them. For example, the existence of PRFs in a complexity class $\mathcal{C}$ can be used to show that this class is not PAC-learnable [PW88, Val84] and that certain "natural proof" techniques will fail to prove circuit lower-bounds for functions in $\mathcal{C}$ [RR97]. Last, but not least, identifying the simplest construction of PRFs may provide valuable insights regarding the nature of computational intractability and the way it is achieved by a sequence of cheap and basic operations. This "magic" of hardness which arises from highly-efficient computation can be viewed as the essence of modern cryptography.

Being relatively complicated objects, a considerable research effort has been made to put PRFs on more solid ground at the form of simpler one-wayness assumptions (cf. [GGM86, HILL99, NR95, NR97, NRR00, LW09, BMR10, BPR12]). Annoyingly, the existence of a security reduction seem to incur a cost in efficiency. Indeed, existing theoretical constructions (either based on general primitives or on concrete intractability assumptions) are relatively slow compared to "practical constructions" whose security is based on first-order cryptanalytic principles rather than on a security reduction. As a concrete example, theoretical constructions of PRFs $F_k : \{0,1\}^n \to \{0,1\}^n$ have super-linear (or even quadratic) circuit size. In contrast, Miles and Viola [MV12] presented a candidate PRF which can be computed by a quasilinear circuit of size $\tilde{O}(n)$. (The notation $\tilde{O}(n)$ subsumes polylogarithmic factors.) Similarly, Akavia et al. [ABG+14] proposed a candidate for a weak PRF[1] which can be computed by a constant-depth circuit with unbounded fan-in AND, OR and XOR gates, whereas it is unknown how to construct such a weak PRF based on one-wayness assumption.

Our goal in this paper is to narrow the gap between provably-secure constructions and highly-efficient candidates. We present several constructions of pseudorandom functions with low-complexity,

---

[1]A weak PRF is a relaxation of a PRF which is indistinguishable from a random function for an adversary whose queries are chosen uniformly at random.

and show that their security can be reduced to variants of Goldreich's one-way function. Before introducing our constructions, let us present Goldreich's one-way function. (For more details see the survey [App15].)

1.1 Goldreich's one-way function

Let n be an input length parameter, $m \geq n$ be an output length parameter and $d \ll n$ be a locality parameter. For a d-local predicate $P : \{0,1\}^d \to \{0,1\}$ and a sequence $G = (S_1, \ldots, S_m)$ of d-tuples over the set $[n] := \{1, \ldots, n\}$, we let $f_{G,P} : \{0,1\}^n \to \{0,1\}^m$ denote the mapping

$$x \mapsto (P(x[S_1]), \ldots, P(x[S_m])),$$

i.e., the i-th output bit is computed by applying the predicate P to the input bits which are indexed by the i-th tuple S_i. Goldreich [Gol00] conjectured that for $m = n$ and possibly small value of d (e.g., logarithmic or even constant), the function $f_{G,P}$ is one-way as long as the set system $(S_1, \ldots, S_m)$ is "highly expanding" and the predicate P is sufficiently "non-degenerate". We elaborate on these two requirements.

Expansion. To formalize the expansion property let us think of $G = (S_1, \ldots, S_m)$ as a d-uniform hypergraph with m hyperedges (which correspond to the outputs) over n nodes (which correspond to the inputs). The expansion property essentially requires that every not-too-large subset of hyperedges is almost pair-wise disjoint. Formally, for a threshold r, the union of every set of $\ell \leq r$ hyperedges $S_{i_1}, \ldots, S_{i_\ell}$ should contain at least $(1 - \beta)d\ell$ nodes, i.e., $|\bigcup_{j=1}^{\ell} S_{i_j}| \geq (1 - \beta)d\ell$, where β is some constant smaller than $\frac{1}{2}$ (e.g., 0.1).

Secure predicates. A noticeable amount of research was devoted to studying the properties of "secure" predicates accumulating in several algebraic criteria (cf., [Ale03, MST03, ABW10, BQ12, ABR12, OW14, FPV15]). It is known for example, that in order to support an output length of $m = n^c$ one must use k-resilient predicate P which is uncorrelated with any GF(2)-linear combination of at most $k = \Omega(c)$ of their inputs. Additionally, the predicate P must have algebraic degree (as a GF(2) polynomial) of at least c. Moreover, P must have high *rational degree* in the following sense: any polynomial Q whose roots cover the roots of P or its complement must have algebraic degree of $\Omega(c)$ [AL15]. An example for such a predicate (suggested in [AL15]) is the d-ary XOR-MAJ$_d$ predicate which partitions its input $w = (w_1, \ldots, w_d)$ into two parts $w_L = (w_1, \ldots, w_{\lfloor d/2 \rfloor})$ and $w_R = (w_{\lfloor d/2 \rfloor+1}, \ldots, w_d)$, computes the XOR of the left part and the majority of the right part, and XOR's the results together.[2] This predicate achieves resiliency of $d/2$ and rational degree of $d/4$ and therefore seems to achieve security for $m = n^{\Omega(d)}$ outputs.

Security. Intuitively, large expansion (together with high resiliency) provide security against local algorithms that employ some form of divide-and-conquer approach. Due to the expansion of the input-output hypergraph, any small subset of the outputs gives very little information on the global solution x. High rational degree provides security against more global approaches which rely on different forms of linearization and algebraic attacks. These intuitions were formalized and proved for several classes of algorithms in previous works (cf.[AHI05, ABW10, CEMT14, ABR12, BR13, OW14]). Following these works, we make the following strong version of Goldreich's conjecture:

---

[2]In fact, it seems better to allocate a larger fraction of the inputs to the Majority part. See [AL15].

Assumption 1.1 (Expander-based OWFs (Informal)). *For some universal constant $\alpha \in (0,1)$ and every d-uniform hypergraph G with n nodes and $m < n^{\alpha d}$ hyperedges which is expanding for sets of size $r = n^{\Omega(1)}$, the function $f_{G,\text{XOR-MAJ}_d}$ cannot be inverted in polynomial time.*[3]

This assumption is consistent with known attacks. In fact, hardness results (against limited families of attacks) suggest that inversion is hard even for adversaries of complexity $\exp(r)$ where r is the expansion threshold. We refer to this variant as the strong EOWF assumption. We further mention that although previous works mainly focused on the case where the locality d is constant or logarithmic in n (which is going to be our main setting here as well), it seems reasonable to conjecture that the assumption holds even for larger values of d (e.g., $d = n^{\delta}$ for constant $\delta \in (0,1)$). Finally, we note that the expansion requirement implicitly puts restrictions on the values of n, m and d. Roughly speaking, an expansion of $r = n^{1-\beta}$ requires $\Theta(1/\beta^2) \leq d \leq n^{\Theta(\beta)}$ and restricts m to be at most $n^{\Theta(d\beta^2)}$.

1.2 Results and Techniques

We present several constructions of expander-based PRFs.

1.2.1 Weak PRF

Let P be some d-ary predicate (e.g., XOR-MAJ$_d$). In our first construction F_1, we think of the input $x \in \{0,1\}^n$ as specifying a hypergraph G_x and let the output y be the value of $f_{G_x,P}$ applied to the collection key $k \in \{0,1\}^n$. Namely, we think of the data x as specifying a computation that should be applied to k. The hypergraph G_x is defined in the natural way: Partition x to $(d\log n)$-size substrings, and view each substring as a d-tuple of elements in $[n]$ where each element is given in its binary representation. An adversary that makes q queries $x_1, \ldots, x_q$ essentially sees the value of $f_{G,P}(k)$ where $G = \bigcup G_{x_i}$. When the adversary is allowed to choose the queries, the outcome cannot be pseudorandom (think of the case where G_{x_1} and G_{x_2} share the same hyperedge). However, when the queries $x_1, \ldots, x_q$ are chosen at random (as in the setting of a weak PRF), the resulting graph G is a random hypergraph which is likely to be expanding. At this point, we can employ a search-to-decision reduction from [App13], which shows that for random hypergraphs G, one-wayness implies pseudorandomness. It follows that, for a proper choice of parameters (e.g., $d = \Omega(\log n)$), our assumption implies that the function F_1 is a weak PRF.[4]

This construction can be instantiated with different locality parameters d, ranging from $O(\log n)$ to n^{δ}. In the logarithmic regime, this gives rise to a construction $F_1 : \{0,1\}^n \to \{0,1\}^{n/\log^2 n}$ which is computable in linear time of $O(n)$ on a RAM machine with $O(\log n)$ word size. Additionally, this function can be computed, for any fixed key k, by a depth-3 circuit with unbounded fan-in AND and OR gates (i.e., an $\mathbf{AC^0}$ circuit).[5] To the best of our knowledge this is the first construction of a weak PRF that achieves such efficiency guarantees.

---

[3]In Section 2 we provide a more general assumption which allows the hypergraph to be non-uniform, and is parameterized by an expansion parameter, by a predicate family $\mathcal{P}$ and by a concrete bound on the security of the function in terms of the (circuit) size of the adversary and its success probability. The above assumption is given here in a simplified form for ease of presentation.

[4]Formally, Assumption 1.1 implies that for a random hypergraph G, the function $f_{G,P}$ is one-way (since such a graph is likely to be expanding). Then, we can apply the result of [App13].

[5]When analyzing parallel-complexity it is common to restrict the attention for the case where the key is fixed, cf. [NR95, NR97, NRR00, LW09, BMR10, MV12, ABG$^+$14].

Concrete security and application to learning. The (strong) EOWF assumption implies that F_1 resists almost-exponential size adversaries (computable by circuits of size $t = \exp(n^{1-\beta})$ for any $\beta > 0$) as long as they make only $q = n^{O(d)}$ queries to the function. Hence, logarithmic locality provides only security against a quasi-polynomial number of queries (e.g., $\exp(\text{polylog}(n))$. Similarly, the distinguishing advantage of the adversary is only quasi-polynomial $\varepsilon = \exp(-\text{polylog}(n))$. While this setting of parameters may seem too weak for many cryptographic applications, it provides a useful theoretical insight. The classical learning algorithm of Linial, Mansour and Nisan [LMN93] shows that any $\mathbf{AC^0}$-computable weak PRF can be broken either with quasipolynomial distinguishing advantage or by making quasipolynomial number of queries. (In the computational learning terminology, $\mathbf{AC^0}$ functions are PAC-learnable under the uniform distribution using a quasipolynomial number of samples and time, or weakly learnable in polynomial-time with advantage $1/\text{polylog}(n)$ over $\frac{1}{2}$.) The LMN algorithm relies on the Fourier spectrum of $\mathbf{AC^0}$ functions, and the possibility of improving it to a polynomial-time algorithm is considered to be an important open problem in learning theory. Our construction suggests that this is impossible even for depth-3 circuits, and so the Fourier-based algorithm of [LMN93] is essentially optimal. To the best of our knowledge, this is the first hardness result for learning depth-3 $\mathbf{AC^0}$ circuits over the uniform distribution. Previous hardness results either apply to $\mathbf{AC^0}$ circuits of depth d for *large* (unspecified) constant depth d [Kha93], to depth-3 *arithmetic* circuits [KS09], or to depth-2 $\mathbf{AC^0}$ circuits but over a *non-uniform* distribution [ABW10, DLS14].

1.2.2 Reducing the distinguishing advantage

Our second construction attempts to strengthen the distinguishing advantage ε of F_1. In F_1 the hypergraph $G = \bigcup G_{x_i}$ fails to be expanding with quasipolynomial probability, and in this case pseudorandomness may be easily violated. As a concrete example note that, with probability $\Omega(n^{-d})$, the hypergraph G contains a pair of identical hyperedges $S_i = S_j$, and so the corresponding outputs will be identical, and distinguishing (with constant advantage) becomes trivial.

Following [CEMT14], we observe that, although expansion is violated with quasipolynomial small probability, not all is lost, and, except for a tiny (almost exponentially small) probability, the hypergraph G is *almost expanding* in the sense that after removing a small (say sub-linear) amount of hyperedges the remaining hypergraph is expanding. We use this combinatorial structure to argue that $f_{G,P}(k)$ can be partitioned into two functions f_1 and f_2, where the input-output hypergraph G_1 of f_1 is highly expanding and the function f_2 depends only on a relatively small (sub-linear) number of inputs. As a result we can show that, for such an almost-expander G, the distribution $f_{G,P}(U_n)$ is pseudorandom except for small number of "bad outputs".[6] In fact, the number of "bad outputs" is small enough to argue that each block of $f_{G,P}(U_n)$ (corresponding to the i-th query) has a large amount of "pseudoentropy". Hence, we can get a pseudorandom output (even for almost expanding graphs) by adding a postprocessing stage in which a randomness extractor is applied to the output of F_1 (i.e., extraction is performed separately per each block of $f_{G,P}(U_n)$).

Formally, our second construction F_2 is keyed by a pair of n-bit strings (k, s), and for a given input x, we output the value $\text{Ext}_s(f_{G_x,P}(k))$ where Ext is a strong seeded randomness extractor. Since there are linear-time computable extractors [IKOS08], the construction can be still imple-

---

[6]Technically, this requires an extension of our assumption to the case of non-uniform hypergraphs, and the ability to analyze the function with respect to new predicates (obtained by restricting some of the inputs of the original predicate).

mented by a linear-time RAM machine. Moreover, since the extractor can be computed by a linear function (and therefore by a single layer of unbounded fan-in parity gates), the function F_2 can be computed by a constant-depth circuit with unbounded fan-in AND, OR and XOR gates (or even in $\mathbf{MOD}_2 \circ \mathbf{AC^0}$). We prove that the distinguishing advantage of the construction is almost exponentially-small. We do not know whether F_2 provides security against larger (say subexponential) number of queries, and leave it as an open question.

1.2.3 Handling non-random inputs

Our next goal is to move from the weak PRF setting in which the function is evaluated only over random inputs, to the standard setting where the queries can be chosen by the adversary. It is natural to try to achieve this goal by introducing a preprocessing mapping M that maps an input x to a hypergraph $M(x)$ with the property that every set of q queries $x_1 \ldots, x_q$ form together a hypergraph $G = \bigcup_i M(x_i)$ with good expansion properties. This approach faces two challenges. First, it is not clear at all how to implement the mapping M (let alone in a very efficient way). Second, we can no longer rely on the standard search-to-decision reduction from [App13] since it applies only to randomly chosen hypergraphs (as opposed to arbitrary expanders).

Search-to-decision reduction for expander-based functions. We solve the second challenge, by proving a new search-to-decision reduction that applies directly to expander hypergraphs. Namely, we show that if $f_{G,P}$ is one-way for every expander hypergraph G (as conjectured by in Assumption 1.1) then it is also pseudorandom for every expander hypergraph. Technically, the original reduction of [App13] shows that if an adversary A can distinguish $f_{G,P}(U_n)$ from a truly random string, then there exists an adversary B that inverts $f_{H,P}(U_n)$ where G and H are random hypergraphs (with polynomially related parameters). This reduction strongly exploits the ability of A to attack many different hypergraphs G. Roughly speaking, every attack on a hypergraph G_i is translated into a small piece of information on the input x (i.e., a noisy estimation on some bit x_i), and by accumulating the information gathered from different G_i's the input x is fully recovered.[7]

In contrast, in the new search-to-decision theorem we are given a distinguisher A_G which succeeds only over some *fixed* expanding hypergraph G. First, we observe that one can slightly modify G and define, for every index $i \in [n]$, a hypergraph G_i such that given $y = f_{G_i}(x)$ the attacker A_G can be used to obtain an estimation for the i-th bit of x. (This is already implicit in [App13].) One may therefore try to argue that the function $f_{\bigcup_i G_i, P}(x) = (f_{G_1}(x), \ldots, f_{G_n}(x))$ can be inverted by calling A_G for each block separately. This is problematic for two reasons: (1) inversion may fail miserably since the calls to A_G are all over statistically-dependent inputs (the same x is being used); and (2) the resulting hypergraph $H = \bigcup_i G_i$ is non-expanding (due to the use of almost identical copies of the same hypergraph G), and so inversion over H does not contradict the theorem.

Fortunately, both problems can be solved by randomizing each of the G_i's (essentially by permuting the names of the inputs). By concatenating the randomized G_i's, we get a probability distribution $\mathcal{D}(G)$ over hypergraphs which satisfies the following two properties: (1) a random hypergraph $H \overset{R}{\leftarrow} \mathcal{D}(G)$ is typically a good expander; and (2) Inverting $f_{H,P}$ for a random $H \overset{R}{\leftarrow} \mathcal{D}(G)$ reduces to inverting $f_{G,P}$. Since we work in a non-uniform model of adversaries (circuits), this suffices to prove the theorem. (See Section 3 for details.)

---

[7]An analogous use of public randomness appears in the seminal Goldreich-Levin theorem [GL89] which can be viewed as search-to-decision reduction for the keyed function $f_k(x) = (g(x), \langle x, k \rangle)$.

Mapping inputs to expanders. Going back to the first challenge, we still need to provide a mapping $M(x)$ which, when accumulated over different inputs, results in a highly expanding hypergraph. Note that although M operates on n-bit inputs, it should satisfy a global property that applies to collection of super-polynomial (or even exponential) number of inputs. Unfortunately, we do not know how to obtain such a mapping deterministically with a low computational cost. Instead, we show how to provide a family of mappings M_σ with the property that for every fixed sequence of inputs $x_1, \ldots, x_q$ and for a random σ, the hypergraph $G = \bigcup_i M_\sigma(x_i)$ is highly expanding with all but exponentially small probability. The key idea is to note that in order to guarantee expansion for r-size sets, it suffices to make sure that each set of r hyperedges of G is (almost) uniformly distributed. This means that M_σ should be almost *R-wise independent* in the following sense: For a random σ, every subset of $R = rd\log(n)$ bits of the random variable $(M_\sigma(x))_{x \in \{0,1\}^n}$ should be statistically-close to uniform. This setting is somewhat non-standard: Efficiency is measured with respect to a single invocation of M_σ (i.e., the complexity of generating a block of m hyperedges), but pseudorandomness should hold for any set of r hyperedges (R bits) across different invocations.

We construct such a mapping M_σ by tweaking a construction of low-bias generator due to Miles and Viola [MV12]. We view $\sigma \in \{0,1\}^{2n}$ as a pair of $\mathrm{GF}(2^n)$ elements σ_1, σ_2, and map an input $x \in \mathrm{GF}(2^n)$ to the $\mathrm{GF}(2^n)$-element $(x + \sigma_1)^{-1} \cdot \sigma_2$.[8] The resulting function F_3 is keyed by (k, σ, s) and for an input x it outputs the value $\mathrm{Ext}_s(f_{M_\sigma(x), P}(k))$ where $M_\sigma(x)$ is parsed as a d-uniform hypergraph with $m = n/(d\log n)$ hyperedges and d is treated as a parameter. Due to the high efficiency of M (which consists of a single multiplication and a single inversion over $\mathrm{GF}(2^n)$), the function F_3 can be computed by a quasilinear circuit $\tilde{O}(n)$ or by a constant-depth circuit with unbounded fan-in AND, OR, and Majority gates (i.e., $\mathbf{TC^0}$ circuit), for any choice of the locality parameter d.

The use of keyed mapping, allows us to prove security against a non-adaptive adversary whose i-th query is independent of the answers for the previous queries. We do not know whether the construction remains secure for adaptive adversaries, however, using the non-adaptive to adaptive transformation of [BH15], we can turn our function into a standard PRF without increasing the asymptotic cost of the construction (in terms of size and depth). We mention that the parallel complexity (i.e., $\mathbf{TC^0}$) seems essentially optimal for PRF and it matches the complexity of the best known PRF constructions based on number-theoretic or lattice assumptions [NR95, NR97, NRR00, BPR12].

Concrete security. Recall that the locality parameter d can vary from logarithmic to n^δ for some $\delta \in (0, 1)$. To get an expansion for sets of size $n^{1-\beta}$ (and therefore security against $\exp(n^{1-\beta})$-size circuits), we must restrict the number of queries q to be smaller than $n^{d\beta^2}$. In addition, the locality d should satisfy $4/\beta^2 < d < n^{\beta/4}$. Hence, polynomial locality $d = n^\delta$ allows to support sub-exponential number of queries while providing security against sub-exponential size circuits with respect to sub-exponential distinguishing advantage. Note that polynomial locality has also some effect on efficiency: The number of output bits per invocations decreases to $\tilde{O}(n/d)$ and so the computational cost per output bit is $\tilde{O}(d) = \tilde{O}(n^\delta)$. On the other extreme, a logarithmic value of d achieves an almost-optimal complexity per bit (i.e., $\tilde{O}(1)$), and provides security against circuits of almost-exponential size ($\exp(n^{1-\beta})$ for every $\beta > 0$) which make a quasipolynomial number of queries.

---

[8]The generator of [MV12] uses binary inner-product instead of field multiplication and accordingly outputs a single bit; We observe that the multi-output version still achieves low-bias. See Section 5.

Security beyond expansion. We do not know whether our analysis is tight. To the best of our knowledge, F_3 with logarithmic locality may achieve security even in the presence of sub-exponentially many queries. We remark that our analysis is somewhat pessimistic since it essentially assumes that the seed s of the extractor and the seed σ of the preprocessing mapping are both given to the adversary. Indeed, in this case the adversary sees the underlying hypergraph and, after sufficiently many queries, it can exploit its non-expanding properties. In contrast, when s and, more importantly, σ are not given, the adversary does not get a direct access to the hypergraph. One may assume that as long as M somewhat hides the hypergraph G, lack of expansion cannot be used to break the system. The question of identifying the right (and minimal) notion of hiding remains open for future research.[9]

1.3 Conclusion

We presented several elementary constructions of pseudorandom functions. All our constructions follow a similar template: The input x is mapped to a hypergraph G_x, which represents a simple (essentially single-layered) circuit $f_{G_x,P}$, the resulting circuit is applied to the key k, and the output is fed through some randomness extractor. We believe that this structure provides a new methodology for constructing pseudorandom functions which deserves to be further studied.

Following Goldreich, we conjecture that as long as the input-output relations is expanding the computation is hard to invert. We further show that such one-wayness leads to pseudorandomness by extending the techniques of [App13]. We believe that understanding this assumption, or more generally, relating the combinatorial structure of circuits to their cryptographic properties is a key question, which may eventually lead to faster and highly secure PRFs. Our proofs, which fall short of providing optimal security (in some cases they are very far from that), should be viewed as a first step in this direction.

Finally, we believe that the tools developed here (e.g., pseudoranodmness over imperfect expanders, the expander-based search-to-decision reduction, and the use of low-bias generators for generating expanders with "direct-access") will turn out to be useful for future works in the field.

Acknowledgement. We thank Adam Klivans and Shai Shalev-Shwartz for helpful discussions.

Organization. In Section 2 we provide necessary preliminaries and set-up the formal setting for expander based functions. The reduction from pseudorandomness to one-wayness appears in Section 3, and Section 4 presents the three basic constructions of PRFs. Section 5 describes the construction of the hypergraph mapper based on low-bias generator.

2 Preliminaries

General preliminaries. We let $[n]$ denote the set $\{1, \ldots, n\}$. For a string $x \in \{0,1\}^n$ and $i \in [n]$, we let $x[i]$ denote the ith bit of x. For a tuple $S = (i_1, \ldots, i_d)$, we let $x[S] = x[i_1, \ldots, i_d]$ denote the restriction of x to indices in S, i.e., the string $x[i_1] \ldots x[i_d]$. For strings $x_1, \ldots, x_q$ we write $(x_i)_{i=1}^q$ to denote the concatenation of the strings $x_1 || \cdots || x_q$. We write $\log_d n$ to denote the logarithm of n

---

[9]It is not hard to show that if M by itself is a PRF then security holds for F_3. The hope is to get somewhat weaker form of hiding, ideally, one which can be satisfied by some concrete and highly-efficient mapping M such as the one proposed here.

base d, if $d = 2$ we omit writing it explicitly. A function $\varepsilon(\cdot)$ is said to be negligible if $\varepsilon(n) < n^{-c}$ for any constant $c > 0$ and sufficiently large n. We will sometimes use $\text{neg}(\cdot)$ to denote an unspecified negligible function. For a function $t(\cdot)$, we write $t = \tilde{O}(n)$, if $t = O(n \log^k(n))$ for some $k \in \mathbb{N}$.

Probabilistic notation. For a probability distribution or random variable X (resp., set), we write $x \stackrel{R}{\leftarrow} X$ to denote the operation of sampling a random x according to X (resp., sampled uniformly from X). We let U_n (resp., U_S) denote a random variable uniformly distributed over $\{0,1\}^n$ (resp., over the set S). We write $\text{supp}(X)$ to denote the support of the random variable X, i.e., $\text{supp}(X) = \{x \mid \Pr[X = x] > 0\}$. The statistical distance between two probability distributions X and Y, denoted $\Delta(X;Y)$, is defined as the maximum, over all functions A, of the distinguishing advantage $\Delta_A(X,Y) := |\Pr[A(X) = 1] - \Pr[A(Y) = 1]|$. We say that X is ε-*statistically indistinguishable* from Y if $\Delta(X;Y) \leq \varepsilon$ and write $X \stackrel{s}{\equiv}_\varepsilon Y$. The random variable X is (t,ε)-*computationally indistinguishable* from Y if for every circuit A of size t, the distinguishing advantage $\Delta_A(X,Y)$ is at most ε, and we write $X \stackrel{c}{\equiv}_{t,\varepsilon} Y$.

Cryptographic primitives. A random variable X over n-bit strings is called (t,ε)-*pseudorandom* if $X \stackrel{c}{\equiv}_{t,\varepsilon} U_n$. A function $f : \{0,1\}^n \to \{0,1\}^m$ is (t,ε) one-way if for every t-size adversary A it holds that $\Pr_x[A(f(x)) \in f^{-1}(f(x))] < \varepsilon$.

Definition 2.1 (PRF). *A keyed function $f : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is called (q,t,ε)-pseudorandom if for any t-size circuit $D^{(\cdot)}$ aided with q oracle gates, the distinguishing advantage*

$$\left| \Pr_{k \stackrel{R}{\leftarrow} \mathcal{K}} [D^{f_k} = 1] - \Pr_{h \stackrel{R}{\leftarrow} \mathcal{H}} [D^h = 1] \right| \leq \varepsilon,$$

where $\mathcal{H}$ is a set of all functions mapping inputs from $\mathcal{X}$ to $\mathcal{Y}$. An adversary is called non-adaptive *if it generates all the queries at the beginning independently of the received responses from the oracle gates.*

A (q,t,ε)-PRF family is a sequence of keyed functions $\mathcal{F} = \{f_n : \mathcal{K}_n \times \mathcal{X}_n \to \mathcal{Y}_n\}$ equipped with an efficient key sampling algorithm and an efficient evaluation algorithm where each f_n is $(q(n), t(n), \varepsilon(n))$-pseudorandom. We say that $\mathcal{F}$ is a (q,t,ε) non-adaptive PRF (resp., weak PRF) if the above holds for non-adaptive adversaries (resp., for adversaries such that each of their queries is chosen independently and uniformly from $\mathcal{X}_n$).

Low-bias generators. We employ the following notions of low-bias and bitwise-independence generators. As in the case of PRFs, we view a two-argument function $f(k,x)$ as a keyed function whose first argument k serves as a key. We emphasize this distinction by writing $f_k(x)$ for $f(k,x)$.

Definition 2.2. *Let $g : \{0,1\}^\kappa \times \{0,1\}^m \to \{0,1\}^n$ be a keyed function. For $x \in \{0,1\}^m$, let $Y(x)$ denote the random variable $g_k(x)$ induced by $k \stackrel{R}{\leftarrow} \{0,1\}^\kappa$, and let $\mathbf{Y}$ denote the random variable $(Y(x))_{x \in \{0,1\}^n}$ where the same random key is used for all x's. We say that g is:*

- (t,ε)-bitwise independent *if every t-bit subset of $\mathbf{Y}$ is ε-close to uniform (in statistical distance), i.e., for every $\ell \leq t$ distinct indices $i_1, \ldots, i_\ell$ we have that*

$$\Delta(U_\ell; (\mathbf{Y}[i_j])_{j=1}^\ell) \leq \varepsilon.$$

– (t, ε)-biased over $\mathrm{GF}(2)$ *if for every $\ell \leq t$ distinct indices $\{i_1, \ldots, i_\ell\}$, we have that*

$$\left| \Pr\left[\sum_{j=1}^{\ell} \mathbf{Y}[i_j] = 1 \right] - \frac{1}{2} \right| \leq \varepsilon,$$

where the sum is computed over $\mathrm{GF}(2)$.

– (t, ε)-linear-fooling over $\mathrm{GF}(2^n)$ *if for every t outputs $Y(x_1), \ldots, Y(x_t)$ (parsed as elements of $\mathrm{GF}(2^n)$) of distinct $x_1, \ldots, x_t$, every t constants $b_1, \ldots, b_t$ from $\mathrm{GF}(2^n)$ (that are not all equal to zero), we have that*

$$\Delta\left(\sum_{i=1}^{t} b_i Y(x_i) \; ; \; U_{\mathrm{GF}(2^n)} \right) \leq \varepsilon.$$

Sources and Extractors. The *min-entropy* of a random variable X (denoted by $H_\infty(X)$) is defined to be $\min_{x \in \mathrm{supp}(X)} \log \frac{1}{\Pr[X=x]}$. A keyed function $E : \mathcal{S} \times \mathcal{X} \to \mathcal{Y}$ is a *strong (k, ε)-extractor* if for every distribution X over $\mathcal{X}$ with $H_\infty(X) \geq k$, it holds that $\Delta((s, \mathrm{Ext}_s(x)) \; ; \; (s, U(\mathcal{Y}))) \leq \varepsilon$, where $s \xleftarrow{R} \mathcal{S}$, $x \xleftarrow{R} X$ and $\Delta(\cdot; \cdot)$ stands for statistical distance.

We consider the following notion of random sources that can be viewed as a convex combination of the traditional bit-fixing sources [CGH+85].

Definition 2.3 (Generalized Bit-Fixing Source)**.** *A distribution X over $\{0,1\}^n$ is a generalized k-bit-fixing source if there exist k distinct indices S such that $X[S]$ is distributed like U_k and $X[[n] \setminus S]$ is independent from $X[S]$.*

We use the following simple lemma (whose proof is deferred to Appendix A.1).

Lemma 2.4. *Let Ext be a strong $(m - r, \delta)$-extractor for m-bit sources. Let $Z = Z_1||\cdots||Z_q$ be a generalized $(qm - r)$-bit-fixing source, where each $|Z_i| = m$. Then for a uniformly chosen seed s, the random variable $(s, \mathrm{Ext}_s(Z_1), \ldots, \mathrm{Ext}_s(Z_q))$ is $(q \cdot \delta)$-statistically indistinguishable from uniform.*

Hypergraphs. An (n, m)-hypergraph G is a hypergraph over vertices $[n]$ with m hyperedges $(S_1, \ldots, S_m)$ where each hyperedge is viewed as a tuple $(i_1, \ldots, i_k)$, i.e., it is ordered and may contain duplications. It is sometimes convenient to think of a hypergraph G as a bipartite graph, where the n vertices represent the lower layer of the graph, the hyperedges represent the upper layer of the graph such that each hyperedge $S = (i_1, \ldots, i_k)$ is connected to the vertices $i_1, \ldots, i_k$. We say that G is d-uniform (denoted by (n, m, d)-hypergraph) if all the hyperedges are of the same cardinality d. G is *almost d-uniform* (denoted by $[n, m, d]$-hypergraph) if $d/2 < |S_i| \leq d$ for all $i \in [m]$. We let $\mathcal{G}_{n,m,d}$ denote the probability distribution over (n, m, d)-hypergraphs in which each of the m hyperedges is chosen independently and uniformly at random from $[n]^d$. We say that a distribution over (n, m, d)-hypergraphs is (k, ε)-*random* if any k hyperedges are ε-close (in statistical distance) to the uniform distribution $\mathcal{G}_{n,k,d}$. A distribution over hypergraphs is (r, d, ε)-*random* if any $s \leq r$ hyperedges $S_1, \ldots, S_s$ contain at least sd entries that are ε-close to uniform.

For a set of hyperedges $T = \{S_1, \ldots, S_k\}$ we write $\Gamma(T)$ to denote the union of tuples $S_1, \ldots, S_k$ (where the union of tuples is naturally defined to be the set of all indices occuring in $S_1, \ldots, S_k$). Let $G \setminus T$ denote the hypergraph obtained from G by removing hyperedges T and updating the

remaining hyperedges by deleting from them vertices that belong to $\Gamma(T)$. A hypergraph G is an (r, c)-expander if for any set I of hyperedges of size at most r we have $\Gamma(I) \geq c|I|$. We refer to r as "the expansion threshold" and to c as "the expansion factor". A hypergraph G is an r_{bad}-imperfect (r, c)-expander if there exists a subset of G's hyperedges I_{bad} of size $|I_{\text{bad}}| \leq r_{\text{bad}}$ such that $G \setminus I_{\text{bad}}$ is an (r, c)-expander.

It is well known that a random hypergraph is likely to be highly expanding. The following lemma (whose proof is deferred to Appendix A.3) generalizes this fact to the case of (r, d, ε)-random hypergraphs and to the case of imperfect expansion. (Note that the failure probability drops down exponentially with the size of the imperfectness parameter t.)

Lemma 2.5. *Let β be a constant in $(0, 1)$ and $d \in \mathbb{N}$ such that $4/\beta^2 \leq d \leq n^{\beta/4}$. Let $r = n^{1-\beta}$ and $m \leq n^{d\beta^2/4}$. Let $t = t(n)$ be a non-negative function such that $t \leq r$. Then, a $(r + t, d, 2^{-\Omega(n)})$-random (n, m)-hypergraph G is t-imperfect $(r, (1-\beta)d)$-expander except with probability $n^{-(t+1)d\beta^2/10}$.*

The *union* of an (n, m_1)-hypergraph $G = (S_1, \ldots, S_{m_1})$ and (n, m_2)-hypergraph $H = (R_1, \ldots, R_{m_2})$ is the $(n, m_1 + m_2)$-hypergraph $J = G \cup H$ whose hyperedges are $(S_1, \ldots, S_{m_1}, R_1, \ldots, R_{m_2})$. Since union is an associative operation, the union of q hypergraphs $G_1 \cup \cdots \cup G_q$ is defined unambiguously.

2.1 Expander-based Functions

For an (n, m)-hypergraph $G = (S_1, \ldots, S_m)$, a sequence of m predicates $P = (P_1, \ldots, P_m)$ where $P_i : \{0, 1\}^{|S_i|} \to \{0, 1\}$, we let $f_{G,P} : \{0, 1\}^n \to \{0, 1\}^m$ denote the function that takes an input $x \in \{0, 1\}^n$ and maps it to the m-bit string $(P_1(x[S_1]), \ldots, P_m(x[S_m]))$. (If all predicates are identical we simply write $f_{G,P}$.) In its most abstract form, our assumption is parameterized by an expansion parameter β (that quantifies the "expansion loss"), and by a (possibly infinite) predicate family $\mathcal{P}$. Formally, the Expander-based OWF assumption (EOWF) and Expander-based PRG assumption (EPRG) are defined as follows.

Definition 2.6 (EOWF and EPRG). *The $\mathsf{EOWF}(\mathcal{P}, m, \beta, t, \varepsilon)$ assumption asserts that for every $[n, m, d]$-hypergraph $G = (S_1, \ldots, S_m)$ that is $(n^{1-\beta}, (1 - \beta)d)$-expanding, and every sequence of predicates[10] $P = (P_i)_{i \in [m]}$ taken from $\mathcal{P}$, the function $f_{G,P}$ is (t, ε) one-way. The $\mathsf{EPRG}(\mathcal{P}, m, \beta, t, \varepsilon)$ is defined similarly except that $f_{G,P}(U_n)$ is (t, ε) pseudorandom.*

A considerable amount of research was devoted to studying the properties of "secure" predicates. (See [App15] and references therein.) These results suggest that for some predicates of logarithmic arity $d = \Theta(\log n)$, and some constant $\beta < \frac{1}{2}$, the $\mathsf{EOWF}(\mathcal{P}, m, \beta, t, \varepsilon)$ assumption holds for every polynomial m, t and every inverse polynomial ε. We adopt this setting as our main intractability assumption and abbreviate this assumption by $\mathsf{EOWF}(\mathcal{P})$. Similarly, we let $\mathsf{EPRG}(\mathcal{P})$ denote the analogous assumption for pseudorandomness. In fact, known results suggest that for a proper family of predicates $\mathcal{P}$, every $d = d(n)$ and every $\beta < \frac{1}{2}$, the assumption holds against adversaries whose size t and success probability ε are exponential in the expansion threshold, i.e., $t = \exp(\Omega(n^{1-\beta}))$ and $\varepsilon = 1/t$, as long as the output length satisfies $m < n^{o(d)}$ or even $m < n^{\alpha d}$ for some constant α. We refer to this variant of the assumption as the *strong* $\mathsf{EOWF}(\mathcal{P})$ and *strong* $\mathsf{EPRG}(\mathcal{P})$.

---

[10]Here and through the paper, we implicitly assume that for all $i \in [m]$ the arity of the i-th predicate P_i matches the cardinality of the i-th hyperedge S_i of G.

Concrete instantiation. A candidate for such a secure predicate (suggested in [AL15]) is the d-ary XOR-MAJ$_d$ predicate which partitions its input $w = (w_1, \ldots, w_d)$ into two parts $w_L = (w_1, \ldots, w_{\lfloor d/2 \rfloor})$ and $w_R = (w_{\lfloor d/2 \rfloor + 1}, \ldots, w_d)$, computes the XOR of the left part and the majority of the right part, and XOR's the results together. This predicate satisfies several useful properties such as high resiliency, high algebraic degree and high rational degree (see Section 1.1). In fact, these properties hold for the more general case of XOR-Threshold predicates defined by:

$$\text{XOR-TH}_{d,\alpha,\tau}(w_1, \ldots, w_d) = \left(\sum_{j=1}^{\lfloor \alpha d \rfloor} w_j > \tau \lfloor \alpha d \rfloor \right) \oplus \left(\bigoplus_{i=\lfloor \alpha d \rfloor + 1}^{d} w_i \right),$$

where the first term evaluates to one if $w_1 + \cdots + w_{\lfloor \alpha d \rfloor} > \tau$ and to zero otherwise. We define[11] $\text{XOR-TH}_d = \{\text{XOR-TH}_{d,\alpha,\tau} : \forall \alpha, \tau \in (1/3, 2/3)\}$ and let $\text{XOR-TH} = \bigcup_{d \in \mathbb{N}} \text{XOR-TH}_d$. We conjecture that strong EOWF holds for this family of predicates.

3 From One-Wayness to Pseudorandomness

In this section, we show that EPRG reduces to EOWF as long as the predicate family $\mathcal{P}$ is *sensitive*. The latter condition means that every d-ary predicate $P \in \mathcal{P}$ can be written as $P(w) = w_i \oplus P'(w)$ where i is some input variable and P' does not depend on w_i. (Namely, the predicate is fully sensitive to one of its coordinates.)

Theorem 3.1. *Let β be a constant in $(0, 1)$; and $d = d(n)$, $m = m(n)$ and $\varepsilon = \varepsilon(n)$ be such that:*

$$\frac{4}{\beta} \leq d(1 - \beta) \leq n^{\beta/4} \quad and \quad \frac{4nm^3 \ln n}{\varepsilon^2} \leq n^{(\beta^2/4)(1-\beta)d},$$

and $\mathcal{P}$ be a sensitive predicate family. Then, the $\text{EPRG}(\mathcal{P}, m, \beta, t, \varepsilon)$ assumption follows from the $\text{EOWF}(\mathcal{P}, m', \beta', t', \varepsilon')$ assumption where $m' = m \cdot O(n \ln nm^2/\varepsilon^2)$, $\beta' = 3\beta$, $t' = t \cdot O(n \ln nm^2/\varepsilon^2)$ and $\varepsilon' = \Omega(\varepsilon/(mn))$.

Note that once $d(n)$ is logarithmic in n, the conditions in the theorem are satisfied for every polynomial $m = \text{poly}(n)$, every inverse polynomial $\varepsilon(n)$, and every constant β. We conclude the following corollary.

Corollary 3.2. *For every sensitive family of predicates $\mathcal{P}$, if $\text{EOWF}(\mathcal{P})$ holds then so does $\text{EPRG}(\mathcal{P})$. In particular, this holds for the special case of $\mathcal{P} = \text{XOR-TH}$.*

Note that if we plug in larger (super logarithmic) values of d in Theorem 3.1, we can support larger (super-polynomial) values of m and smaller values of ε (at the expense of decreasing β to some concrete constant).

---

[11] The constants $(1/3, 2/3)$ in the definition are somewhat arbitrary and it seems that any constants bounded away from 0 and 1 will do.

3.1 Proof of Theorem 3.1

Assume, towards a contradiction, that there exists a t-size adversary that breaks the pseudorandomness of $f_{G,P}$ with advantage ε for some $[n,m,d]$-hypergraph G which is $(n^{1-\beta}, (1-\beta)d)$-expanding and some sequence of sensitive predicates $P = (P_1, \ldots, P_m) \in \mathcal{P}^m$. Then, due to Yao's theorem [Yao82], there exists an adversary A_G of similar complexity that predicts some bit of $f_{G,P}$ with advantage $\varepsilon_p = \varepsilon/m$. To simplify notation, we assume that A_G predicts the last bit[12] of $f_{G,P}$. That is,

$$\Pr_{x \xleftarrow{R} \{0,1\}^n, y = f_{G,P}(x)} [A_G(y[1, \ldots, m-1]) = y[m]] - \frac{1}{2} \geq \varepsilon_p. \tag{1}$$

We will prove the following lemma.

Lemma 3.3. *Let $\kappa = 4\ln n/\varepsilon_p^2$, $m' = \kappa \cdot m \cdot n$ and $P' = P^{\kappa n} = (P_1, \ldots, P_m)^{\kappa n}$. There exists a distribution $\mathcal{D}$ over (n, m', d)-hypergraphs such that:*

1. *A hypergraph H sampled from $\mathcal{D}$ is $(n^{1-3\beta}, (1-3\beta)d)$-expanding with probability $1 - 1/(n \ln n)$.*

2. *There exists an adversary B of size $t' = O(\kappa \cdot n \cdot t)$ and a set of inputs $\text{Good} \subseteq \{0,1\}^n$ which contains at least $\varepsilon_p/2$-fraction of all n-bit strings, such that for every string $x \in \text{Good}$,*

$$\Pr_{H \xleftarrow{R} \mathcal{D}} [B(H, f_{H,P'}(x)) = x] \geq 1/(2n).$$

We show that Theorem 3.1 follows from Lemma 3.3. Call H good if

$$\Pr_{x \xleftarrow{R} \{0,1\}^n} [B(H, f_{H,P'}(x)) = x | x \in \text{Good}] \geq 1/(3n).$$

By a Markov argument, a random $H \xleftarrow{R} \mathcal{D}$ is likely to be good with probability $\Omega(1/n)$. Combing this with the first item, it follows, by a union bound, that there exists a good H which is also $(n^{1-3\beta}, (1-3\beta)d)$-expanding. By hardwiring H to B, we get an adversary B_H which inverts $f_{H,P'}$ with probability of at least

$$\Pr_{x \xleftarrow{R} \{0,1\}^n} [x \in \text{Good}] \cdot \Pr_{x \xleftarrow{R} \{0,1\}^n} [B_H(f_{H,P'}(x)) = x | x \in \text{Good}] \geq \Omega(\varepsilon_p/n) = \Omega(\varepsilon/(mn)),$$

contradicting the $\mathsf{EOWF}(\mathcal{P}, m', 3\beta, t', \varepsilon/(mn))$ assumption. We move on to prove Lemma 3.3.

Proof of Lemma 3.3. Before describing the distribution $\mathcal{D}$, we need some additional notation. For a permutation $\pi : [n] \to [n]$ and a tuple $S = (i_1, \ldots, i_d) \subseteq [n]^d$, let $\pi(S)$ denote the tuple $(\pi(i_1), \ldots, \pi(i_d))$. For an $[n,m,d]$-hypergraph G with the hyperedges $(S_1, \ldots, S_m)$, let $\pi(G)$ denote a $[n,m,d]$-hypergraph with the hyperedges $(\pi(S_1), \ldots, \pi(S_m))$. For a string $x \in \{0,1\}^n$, let $\pi(x)$ denote the bit-string whose coordinates are permuted under π. We define the distribution $\mathcal{D}$ based on the hypergraph G via the following procedure:

---

[12]The choice of the last bit unpredictability is without loss of generality since we can permute the order of the bits of $f_{G,P}$ (see [App13]).

> **Figure 1: The Distribution $\mathcal{D}$**
>
> 1. Take $[n, m, d]$-hypergraph G as an input. Let $\ell^* \in [n]$ denote the first index of the last hyperedge of G.
> 2. Sample a random index $\tau \xleftarrow{R} [n]$. For each $j \in [n]$, let $\pi_1^j, \ldots, \pi_\kappa^j$ be $\kappa = 4 \ln n / \varepsilon_p^2$ random permutations over $[n]$ subject to $\pi_i^j(\ell^*) = \tau$.
> 3. For each $j \in [n]$ and $i \in [\kappa]$, let G_i^j be the hypergraph $\pi_i^j(G)$ modified such that the first entry of its last hyperedge is set to j.
> 4. The output of $\mathcal{D}$ is the graph $H = \bigcup_{j \in [n], i \in [\kappa]} G_i^j$.

We start by proving the first item of Lemma 3.3. Consider the distribution $\mathcal{D}'$ resulting from generating $\kappa \cdot n$ uniform and independent permutations ϕ_i^j ($j \in [n], i \in [\kappa]$), and outputting the graph $H' = \cup_{i,j} H'_{i,j}$ where $H'_{i,j} = \phi_i^j(G)$. Observe that $\mathcal{D}$ can be viewed as a two step process in which: (1) H' is sampled from $\mathcal{D}'$; and (2) We modify at most two nodes in every hyperedge of H' based on some random process.[13] Since the second step can reduce the expansion of a set T by at most $2|T|$, and since our setting of parameters implies that $\beta d > 2$, it suffices to show that $\Pr_{H'}[H'$ is $(n^{1-2\beta}, (1 - 2\beta)d)$-expanding$] \geq 1 - 1/(n \ln n)$.

To see this, recall that G is $(r, d' = (1-\beta)d)$-expanding and therefore, for every i, j, the random variable $\phi_i^j(G)$ is $(r, d', 0)$-random. Moreover, the permutations ϕ_i^j are sampled independently at random, and therefore $H' = \bigcup_{i,j} \phi_i^j(G)$ is a $(r, d', 0)$-random $(n, \kappa mn)$-hypergraph. Observe that our parameters satisfy the requirements of Lemma 2.5 (i.e., $4/\beta^2 \leq d' \leq n^{\beta/4}$ and $\kappa mn \leq n^{\beta^2 d'/4}$). By applying the lemma with $t = 0$, we conclude that H' is $(n^{1-\beta}, (1 - \beta)^2 d)$-expanding (and thus also $(n^{1-2\beta}, (1 - 2\beta)d)$-expanding), except with failure probability of at most $n^{-(\beta^2/4)(1-\beta)d}$. The latter quantity is upper-bounded by $1/(n \ln n)$ since $\frac{4nm^3 \ln n}{\varepsilon^2} \leq n^{(\beta^2/4)(1-\beta)d}$. This completes the proof of the first part of Lemma 3.3.

We proceed with the proof of the second item of Lemma 3.3. Let $S = (\ell^*, i_2, \ldots, i_d)$ be the last hyperedge of G. Let S' denote the $d - 1$ tuple $(i_2, \ldots, i_d)$ and let $P_m : \{0,1\}^d \rightarrow \{0,1\}$ be the predicate computed by the last output of $f_{G,P}$. We assume (WLOG) that the first input of P_m is sensitive and so it can be written as $P_m(w_1, \ldots, w_d) = w_1 \oplus Q(w_2, \ldots, w_d)$ for some $(d - 1)$-ary predicate Q.

The algorithm B is a variant of the inversion algorithms given in [App13]. The input is a hypergraph $H = \bigcup_{j \in [n], i \in [\kappa]} G_i^j$ and a string $y \in \{0,1\}^{\kappa \cdot n \cdot m}$ such that $y = f_{H,P'}(x)$. Let y be parsed as $(y_i^j)_{j \in [n], i \in [\kappa]}$ where each $y_i^j = f_{G_i^j, P}(x)$. For each $j \in [n]$ and $i \in [\kappa]$, the algorithm B runs A_G on input $y_i^j[1, \ldots, m-1]$ and gets a prediction bit e_i^j. Let σ_i^j be the inverse permutation of π_i^j, and $x_i^j = \sigma_i^j(x)$; then, we get that $y_i^j = f_{\sigma_i(G_i^j), P}(x_i^j)$. By construction, this means that $y_i^j[1, \ldots, m-1] = f_{G,P}(x_i^j)[1, \ldots, m-1]$ and so A_G attempts to predict the value $P_m(x_i^j[S]) = x_i^j[\ell^*] \oplus Q(x_i^j[S'])$. Note that the bit $y_i^j[m]$ equals to $x_i^j[\sigma_i^j(j)] \oplus Q(x_i^j[S'])$, and so

$$P_m(x_i^j[S]) \oplus y_i^j[m] = x_i^j[\ell^*] \oplus x_i^j[\sigma_i^j(j)] = x[\pi_i^j(\ell^*)] \oplus x[j] = x[\tau] \oplus x[j].$$

Assuming that $x[\tau]$ is known (indeed, we can either guess it or try both values), the above equation provides an estimation for $x[j]$. Since our predictor may err, this estimation is "noisy", i.e., it

---

[13]Specifically, sample a random index $\tau \in [n]$, and for every sub-hypergraph $H'_{i,j}$ and hyperedge $S \in H'_{i,j}$ swap the node $\phi_i^j(\ell^*)$ with the node τ, except for the first entry in the last hyperedge of $H'_{i,j}$ which $\phi_i^j(\ell^*)$ is replaced by j.

equals to $x[j]$ only with probability $\frac{1}{2} + \Omega(\varepsilon_p)$. After collecting κ such votes (and arguing that these votes are "independent enough") we eventually recover the input x bit by bit by deciding on the majority of the votes for each $x[j]$. We proceed by formally describing the algorithm B.

Figure 2: The inverter B

- **Input:** A graph $H = \bigcup_{j \in [n], i \in [\kappa]} G_i^j$ and $y_i^j = f_{G_i^j, P}(x)$.

- Initialize $v_1, \ldots, v_n$ to 0.

- For $j \in [n]$ and $i \in [\kappa]$:
  1. Compute $e_i^j := A_G(y_i^j[1, \ldots, m-1])$, and let $b_i^j = e_i^j \oplus y_i^j[m]$.
  2. If $b_i^j = 1$, then increase v_j by 1, otherwise decrease v_j by 1.

- For $j \in [n]$, set z_j to 1 if $v_j > 0$, otherwise set it to 0. Let $s_0 = z_1 \cdots z_n$ and $s_1 = \overline{z_1} \cdots \overline{z_n}$.

- **Output:** s_0 if $y = f_{H,P'}(s_0)$ and s_1 if $y = f_{H,P'}(s_1)$. Otherwise, output $\bot$.

We now prove that B inverts $f_{H,P'}$ well. Let $\mathrm{wt}(x)$ be the hamming weight of $x \in \{0,1\}^n$ and for $w \in [n]$, let $X_w = \{x \in \{0,1\}^n | \mathrm{wt}(x) = w\}$. Call x *good* if A_G predicts with advantage $\varepsilon_p/2$ the last bit of $f_{G,P}(x')$ for $x' \xleftarrow{R} X_{\mathrm{wt}(x)}$, i.e.,

$$\Pr_{x' \xleftarrow{R} X_{\mathrm{wt}(x)}, y = f_{G,P}(x')} [A_G(y[1 \ldots m-1]) = y[m]] - 1/2 \geq \varepsilon_p/2.$$

We let Good denote the set of good x's and show that this set is $\varepsilon_p/2$-dense.

Claim 3.4. $\Pr_{x \xleftarrow{R} \{0,1\}^n} [x \in \mathrm{Good}] \geq \varepsilon_p/2.$

Proof. Recall that our predictor A_G has an advantage of ε_p when it is invoked on $f_{G,P}(x')$ where $x' \xleftarrow{R} U_n$. Note that we can sample a uniform vector $x' \xleftarrow{R} \{0,1\}^n$ by first selecting $x \xleftarrow{R} U_n$ and then selecting $x' \xleftarrow{R} X_{\mathrm{wt}(x)}$. Hence, the claim follows from Markov's inequality. $\square$

Now fix a good x. Let S_n denote the set of all permutations from $[n]$ to $[n]$. Observe that sampling $x' \xleftarrow{R} X_{\mathrm{wt}(x)}$ is equivalent to taking a random permutation $\sigma \xleftarrow{R} S_n$ and computing $x' = \sigma(x)$. Hence, it holds that

$$\Pr_{\sigma \xleftarrow{R} S_n, y = f_{G,P}(\sigma(x))} [A_G(y[1 \ldots m-1]) = y[m]] - 1/2 \geq \varepsilon_p/2.$$

By an averaging argument, we get that there exists an index $\tau_x \in [n]$ such that

$$\Pr_{\sigma \xleftarrow{R} \{\pi \in S_n | \pi(\tau_x) = \ell^*\}, y = f_{G,P}(\sigma(x))} [A_G(y[1 \ldots m-1]) = y[m]] - 1/2 \geq \varepsilon_p/2.$$

Next, we show that the algorithm B recovers x with probability at least $\frac{1}{2}$ when invoked with a good input x and with a graph H generated under condition that $\tau = \tau_x$. Since τ is generated uniformly at random this implies that $\Pr_H[B(H, f_{H,P'}(x)) = x] \geq 1/(2n)$.

Claim 3.5. *For every good x, it holds that $\Pr_H[B(H, f_{H,P'}(x)) = x | \tau = \tau_x] \geq \frac{1}{2}$.*

Proof. We assume that $x[\tau] = 0$ and show that, with high probability, s_0 is likely to be x. (A similar argument shows that when $x[\tau] = 1$, s_1 is likely to be x). We prove that for each $j \in [n]$ the value z_j equals to $x[j]$ with probability $1 - 1/(2n)$. The theorem then follows by applying a union bound over all n indices.

Fix some index $j \in [n]$. Call a vote b_i^j good if it is equal to $x[j]$. Our goal is to show that with high probability a majority of the votes are good. Observe that in each iteration $i \in [\kappa]$, the predictor A_G is invoked on $y_i^j[1, \ldots, m-1] = f_{G,P}(x_i^j)[1, \ldots, m-1]$ where $x_i^j = \sigma_i^j(x)$ and that the vote b_i^j is good if the predictor succeeds in predicting $P_m(x_i^j[S])$. Since the permutations σ_i^j's (that are the inverses of π_i^j's) are independent and are uniform subject to $\sigma_i^j(\tau) = \ell^*$, and since x is good, each call to the predictor succeeds independently with probability $\frac{1}{2} + \varepsilon_p/2$. Hence, by an additive Chernoff bound, the majority of the votes are good except with probability $\exp(-2\kappa \cdot (\varepsilon_p/2)^2) = \exp(-2\ln n) < 1/(2n)$. $\qquad\square$

This completes the proof of Lemma 3.3. $\qquad\square$

4 PRF constructions

We describe a general template for constructing pseudorandom functions. The template is parameterized with a predicate family $\mathcal{P} = \{P_d\}$ where P_d is a d-ary predicate[14] and two (possibly keyed) algorithms: *mapper M* and *extractor E*. Let $n \in \mathbb{N}$ denote the security parameter and let $d = d(n)$ be a locality parameter. Given an input $x \in \{0,1\}^n$ and a uniformly chosen key $k \in \{0,1\}^n$ we define the output of the function as follows. First, we use the mapper M to map x to an $(n, n/(d\log n), d)$-hypergraph G_x. Second, given the key k we compute a pseudorandom string $y = f_{G_x,P}(k)$, where $P = P_d$. Finally, we apply a randomness extractor E to y in order to produce the final output. (The keys of E and M are appended to the key k and are treated as part of the key of the construction.) The main intuition behind this template is that if the graph G_x has good expanding properties, the string y contains enough pseudoentropy which once extracted via E looks pseudorandom.

In the following we describe several instantiations of the template by choosing different M and E.

Notation switch. Through this section, the symbol x denotes a query to the PRF while k denotes the PRF's key. Due to the structure of our construction, this means that the input to the function $f_{G,P}$ is denoted by k (the key) and the graph G is computed based on the input x. (Unlike the notation used in Section 3.)

4.1 Instantiation F_1

The first instantiation F_1 can be seen as a "plain" instantiation of the template, where the inputs are mapped to the graphs directly and no extractor is applied in the end.

---

[14]The construction can be easily generalized to handle non-uniform hypergraphs and/or different predicates d-ary predicates for each output.

> **Figure 3: Instantiation F_1**
>
> - **Parameters:** Let $\mathcal{K} = \{0,1\}^n$ be the key space, $\mathcal{X} = \{0,1\}^n$ be the input space, and $\mathcal{Y} = \{0,1\}^{n/(d \log n)}$ be the output space of F_1. Let $d = \Theta(\log n)$ and let $P \in \mathcal{P}$ be a d-ary predicate.
>
> - **Mapper M:** The input x is parsed into $n/(\log n)$ indices, then each consecutive group of d indices is interpreted as a hyperedge of the hypergraph G.
>
> - **Extractor E:** No extractor is applied in the end.
>
> - **Code of F_1:** The function $F_1 : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is defined as $F_1(k,x) := f_{M(x),P}(k)$.

Theorem 4.1. *Let n be the security parameter. For every $q = n^{o(\log n)}$, every $t(n), \varepsilon(n)$, and every constant $\beta \in (0,1)$ the function F_1 is a $(q, t, \varepsilon + n^{-\Omega(\log n)})$ weak PRF under assumption $\mathsf{EPRG}(\mathcal{P}, n \cdot q, \beta, t, \varepsilon)$.*

Proof. Fix some constant β and let $d = \Theta(\log n)$. Let $x_1, \ldots, x_q$ be $q = n^{o(\log n)}$ random strings from $\{0,1\}^n$ asked by the adversary. For $i \in [q]$, let $G_i = M(x_i)$. Since the x_i's are uniformly distributed, the graph $H := \bigcup_{i=1}^q G_i$ is a $(n^{1-\beta}, 0)$-random (n, m, d)-hypergraph with $m = qn/(d \log n) < n^{d\beta^2/4}$. Hence, by Lemma 2.5 (with imperfectness parameter $t = 0$), H is $(n^{1-\beta}, (1-\beta)d)$-expanding except with probability $\varepsilon_{\mathrm{EXP}} = n^{-\Omega(\log n)}$. (The condition $4/\beta^2 \le d \le n^{\beta/4}$ required for Lemma 2.5 holds since β is constant and $d = \Theta(\log n)$.) The theorem follows by noting that conditioned on H being $(n^{1-\beta}, (1-\beta)d)$-expanding, the $\mathsf{EPRG}(\mathcal{P}, n \cdot q, \beta, t, \varepsilon)$ assumption implies that the random variable $V = (F_1(k, x_i))_{i=1}^q$, induced by a uniformly chosen $k \in \{0,1\}^n$, is (t, ε)-pseudorandom. $\square$

Remark 4.2. *We note that the theorem extends to the case where $\log_n q + 1 < \beta^2 d/4$.*

Corollary 4.3. *Suppose that $\mathsf{EOWF}(\text{XOR-MAJ})$ holds. Then, there exists a weak PRF $F_1 : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{n/\log^2 n}$ which is computable in linear time of $O(n)$ on a RAM machine with $O(\log n)$ word size, or by a boolean circuit of size $\tilde{O}(n)$. Moreover, for every fixed key k, the function $F_1(k, \cdot)$ can be computed by a depth-3 $\mathbf{AC^0}$ circuit.*

Proof. By Corollary 3.2, $\mathsf{EOWF}(\text{XOR-MAJ})$ implies $\mathsf{EPRG}(\text{XOR-MAJ})$, which in turn, implies, by Theorem 4.1, that F_1 is a weak PRF.

Observe that the computation of F_1 consists of two steps. (1) Access the key k in the $n/\log n$ addresses specified by the input x and retrieve the corresponding content. Namely, for $1 \le i \le \ell$ where $\ell = n/\log n$, output the bits $z_i = k[x[(i-1)\log n + 1 : i \log n]]$ where $x[i : i+j]$ denotes the address represented by the substring $(x[i] \cdots x[i+j])$ under the standard binary representation. (2) Partition the bits $z_1, \ldots, z_\ell$ to d-size ℓ/d blocks, and compute for each block $1 \le i \le \ell/d$ the bit $y_i = \text{XOR-MAJ}_d(z_{(i-1)d+1}, \ldots, z_{id})$.

Time. On a RAM machine with $\log n$ word size, the first step is implemented in time $O(n)$ (these are just accesses to an array) and the second step takes $O(n/\log n)$ time.

Size. In Appendix B we show that the first step can be implemented by a circuit of quasilinear size $O(n \log^2 n \log \log n)$. In the second part, each computation of z_i consists of computing two symmetric functions (XOR and Majority) over $d/2$-long inputs. The classical result of [MP75] (see also [Weg87]) shows that every d-ary symmetric predicate can computed by a linear-size circuit (of size $O(d)$) and so the overall complexity of the second step is linear in n.

Depth. Fix some key k. Observe that both the first part and the second part of the computation have logarithmic locality (each bit z_i depends on at most $O(\log n)$ bits of x and each y_i depends on at most $O(\log n)$ bits of the z_i's). Observe that any such function can be computed by a polynomial size DNF (OR of AND's) and a polynomial size CNF (AND of OR's). Hence, the overall computation can be naively computed by a depth-4 circuit. In fact, by using DNF for the first part and CNF for the second part we can collapse the two middle layers of OR gates and implement F_1 by a depth-3 $\mathbf{AC^0}$ circuit. $\qquad\square$

We note that, under strong EOWF(XOR-MAJ), F_1 achieves security against adversaries of almost-exponential size ($\exp(n^{1-\beta})$ for every $\beta > 0$) who make polynomially many queries (or even slightly super-polynomial number of queries q) with quasipolynomial distinguishing advantage of $\varepsilon = n^{-\Omega(\log n)}$. As mentioned in the introduction, the quasipolynomial value of ε is inherent for $\mathbf{AC^0}$ constructions.

We also remark that one can extend the output length of F_1 to $\{0,1\}^n$ by stretching the output using a pseudorandom generator $G : \{0,1\}^{n/\log n} \to \{0,1\}^n$. Using fast constructions of PRGs (e.g., [App13]) one can do this while keeping the efficiency guarantees stated in the theorem.

4.2 Instantiation F_2

The second instantiation F_2 is a modification of F_1, where an extractor is applied in the end. As explained in the introduction, this allows us to reduce the distinguishing advantage ε.

Figure 4: Instantiation F_2

- **Parameters:** Let $\mathcal{K} = \mathcal{K}_f \times \mathcal{K}_e = \{0,1\}^n \times \{0,1\}^{O(n)}$ be the key space, $\mathcal{X} = \{0,1\}^n$ be the input space, and $\mathcal{Y} = \{0,1\}^{n/(2d\log n)}$ be the output space of F_2. Let $P \in \mathcal{P}$ be a d-ary predicate.

- **Mapper M:** As in F_1, $M(x)$ parses x as $(n, n/(d\log n), d)$-hypergraph.

- **Extractor:** Let $\text{Ext} : \mathcal{K}_e \times \{0,1\}^{n/(d\log n)} \to \{0,1\}^{n/(2d\log n)}$ be a strong $(\ell, \varepsilon_{\text{Ext}})$-extractor where $\ell = 0.9 \cdot n/(d\log n)$ and $\varepsilon_{\text{Ext}} = 2^{-\Omega(n)}$).

- **Code of F_2:** The function $F_2 : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is defined as $F_2((k,s),x) := \text{Ext}_s(f_{M_2(x),P}(k))$, where $(k,s) \in \mathcal{K}_f \times \mathcal{K}_e$.

Our goal is to provide a tight security reduction from breaking F_2 to the EPRG assumption. For this, we will have to rely on the security of EPRG over a predicate family $\mathcal{P}_\beta$ containing all predicates which can be obtained by selecting some d-ary predicate $P \in \mathcal{P}$ and arbitrarily fixing at most βd of its inputs. Although the security of EOWF with respect to $\mathcal{P}_\beta$ may seem like a strong assumption, we will later show that natural candidates for EOWF already satisfy it.

Theorem 4.4. *Let n be the security parameter. Let β be a constant in $(0,1)$, $q = q(n)$ and $d = d(n)$ such that $4/\beta^2 \le d \le n^{\beta/4}$ and $q \le n^{d\beta^2/4-1}$. Let $t = t(n)$, $\varepsilon = \varepsilon(n)$ be arbitrary functions. Then, the function F_2 is a $(q, t, \varepsilon + n^{-\Omega(dn^{1-\beta})} + q \cdot 2^{-\Omega(n)})$ weak PRF, under assumption* EPRG$(\mathcal{P}_\beta, n \cdot q, \beta, t, \varepsilon)$.

Proof. Let $x_1, \ldots, x_q$ be q random strings from $\{0,1\}^n$ asked by the adversary. For $i \in [q]$, let $G_i = M_2(x_i)$. Consider the (n, m, d)-hypergraph $H := \bigcup_{i=1}^q G_i$ where $m = nq/(d\log n) < n^{d\beta^2/4}$. Since

18

the G_i's are random, the hypergraph H is $(2r, 0)$-random for $r = n^{1-\beta}$. By applying Lemma 2.5 with $t = r = n^{1-\beta}$ and d-uniform hypergraphs, we conclude that, except with probability $\varepsilon_{\mathrm{EXP}} = n^{-\Omega(dr)}$, the hypergraph H is r-imperfect $(r, (1-\beta)d)$-expander.

From now on we fix a sequence of queries $(x_1, \ldots, x_q)$ which leads to such an imperfect expander H. It suffices to prove that, for a uniformly chosen $(k, s) \in \mathcal{K}$, the random variable $V := (F_2((k, s), x_i))_{i=1}^q$ is $(t, \varepsilon + q \cdot \varepsilon_{\mathrm{Ext}})$-pseudorandom for $\varepsilon_{\mathrm{Ext}} = 2^{-\Omega(n)}$.

By construction, V can be rewritten as $\overline{\mathrm{Ext}}_s(f_{H,P}(k))$ where $\overline{\mathrm{Ext}}_s(y_1, \ldots, y_q) := (\mathrm{Ext}_s(y_1), \ldots, \mathrm{Ext}_s(y_q))$. First, we show that the distribution of $f_{H,P}(k)$ is computationally indistinguishable from a generalized bit-fixing source (the proof is deferred to Appendix A.2).

Lemma 4.5. *Let G be a $[n, m, d]$-hypergraph which is $n^{1-\beta}$-imperfect $(n^{1-\beta}, (1-\beta)d)$-expander for some constant $\beta \in (0, 1)$. Then, under assumption $\mathsf{EPRG}(\mathcal{P}_\beta, m, \beta, t, \varepsilon)$, the random variable $f_{G,P}(U_n)$ is (t, ε)-computationally indistinguishable from a generalized $(m - n^{1-\beta})$ bit-fixing source.*

It follows that $f_{H,P}(k)$ is (t, ε)-computationally indistinguishable from some generalized $(\frac{qn}{d \log n} - r)$ bit-fixing source Y. We therefore conclude that $V = \overline{\mathrm{Ext}}_s(f_{H,P}(k))$ is (t, ε)-indistinguishable from $\overline{\mathrm{Ext}}_s(Y)$. By Lemma 2.4 (Section 2), the latter distribution is $(q \cdot \varepsilon_{\mathrm{Ext}})$-statistically indistinguishable from uniform. Hence, conditioned on H being an almost expander, V is $(t, \varepsilon + q\varepsilon_{\mathrm{Ext}})$-indistinguishable from uniform. Overall, we conclude that for q random queries, V is $(t, \varepsilon + q\varepsilon_{\mathrm{Ext}} + \varepsilon_{\mathrm{EXP}})$-pseudorandom, as required. $\qquad \square$

Corollary 4.6. *Suppose that strong $\mathsf{EPRG}(\text{XOR-TH})$ holds. Then, there exists a weak PRF $F_2 : \{0, 1\}^{O(n)} \times \{0, 1\}^n \to \{0, 1\}^{n/2 \log^2 n}$ which is $(q, t = \exp(n^{1-\beta}), \varepsilon = \exp(-n^{1-\beta}))$ for every polynomial q and every constant β, and can be computed in linear time of $O(n)$ on a RAM machine with $O(\log n)$ word size, and by a boolean circuit of size $\tilde{O}(n)$. Moreover, for every fixed key k, the function $F_2(k, \cdot)$ can be computed by an $\mathbf{MOD_2} \circ \mathbf{AC^0}$ circuit.*

Proof. Instantiate F_2 with $\mathcal{P} = \text{XOR-MAJ}$ and observe that $\mathcal{P}_\beta = \text{XOR-TH}$ for sufficiently small β (e.g., every $\beta < 1/6$). By Theorem 4.4, the strong $\mathsf{EPRG}(\text{XOR-TH})$ assumption implies that, for every polynomial q and constant $\beta > 0$, F_2 is $(q, t = \exp(n^{1-\beta}), \varepsilon = \exp(-n^{1-\beta}))$ weak PRF.

The efficiency analysis is identical to the analysis of F_1 except that we need to add the complexity of the extractor. Ishai et al. [IKOS08, Theorem 3.3] constructed a strong $(0.9 \cdot N, 2^{-\Omega(N)})$-extractor for N-bit sources outputting an $(N/2)$-bit string using a seed of length $O(N)$ that can be computed by a linear function (over the binary field) whose circuit is of size $O(N)$. By employing this extractor we get a linear-time implementation in the RAM model and quasilinear-size circuit implementation. Furthermore, since the extractor is a linear function it can be implemented by a single layer of XOR gates and so the overall computation is in $\mathbf{MOD_2} \circ \mathbf{AC^0}$. $\qquad \square$

4.3   Instantiation F_3

The third instantiation, F_3, is a modification of F_2, where the input x is mapped to a hypergraph using an $(n, 2^{-\Omega(n)})$-bitwise independent generator $M : \mathcal{K}_m \times \mathcal{X} \to \mathcal{X}$. An efficient construction of such a $(n, 2^{-\Omega(n)})$-bias generator (with $\mathcal{K}_m = \{0, 1\}^{2n}$) is presented in Section 5.

Theorem 4.7. *Let n be the security parameter. Let β be a constant in $(0,1)$, $q = q(n)$ and $d = d(n)$ such that $4/\beta^2 \leq d \leq n^{\beta/4}$ and $q \leq n^{d\beta^2/4-1}$. Let $t = t(n)$, $\varepsilon = \varepsilon(n)$ be arbitrary functions. Then, the function F_3 is a non-adaptive $(q, t, \varepsilon + n^{-\Omega(dn^{1-\beta})} + q \cdot 2^{-\Omega(n)})$-PRF, under assumption $\text{EPRG}(\mathcal{P}_\beta, n \cdot q, \beta, t, \varepsilon)$.*

Proof. Fix a sequence of q distinct non-adaptive queries $x_1, \ldots, x_q$. For $i \in [q]$, let $G_i := M_\sigma(x_i)$. Since M is $(n, 2^{-\Omega(n)})$-biased, the hypergraph $H := \bigcup_{i=1}^q G_i$ is $(\ell, 2^{-\Omega(n)})$-random hypergraph for $\ell = n/(d\log n) \geq 2n^{1-\beta}$. Recall also that H has at most $n \cdot q \leq n^{d\beta^2/4}$ hyperedges and d is chosen such that $4/\beta^2 \leq d \leq n^{\beta/4}$. By applying Lemma 2.5 with $t = r = n^{1-\beta}$ and d-uniform hypergraphs, we conclude that, except with probability $\varepsilon_{\text{EXP}} = n^{-\Omega(dr)}$, the hypergraph H is r-imperfect $(r, (1-\beta)d)$-expander (where the probability is taken over $\sigma \xleftarrow{R} \mathcal{K}_m$).

From now on we fix a good σ which leads to such an imperfect expander H. It suffices to prove that, for a uniformly chosen (k, s), the random variable $V := (F_3((k, \sigma, s), x_i))_{i=1}^q$ is $(t, \varepsilon + q \cdot \varepsilon_{\text{Ext}})$-pseudorandom for $\varepsilon_{\text{Ext}} = 2^{-\Omega(n)}$. By construction, V can be rewritten as $\overline{\text{Ext}}_s(f_{H,P}(k))$ where $\overline{\text{Ext}}_s(y_1, \ldots, y_q)$ stands for $(\text{Ext}_s(y_1), \ldots, \text{Ext}_s(y_q))$. Lemma 4.5 shows that the random variable $f_{H,P}(k)$ is (t, ε)-computationally close to some generalized $(qn/(d\log n) - r)$ bit-fixing source Y, and Lemma 2.4 shows that $\overline{\text{Ext}}_s(Y)$ is $q \cdot \varepsilon_{\text{Ext}}$-close to uniform. The theorem follows. $\qquad\square$

In Section 5 (Theorem 5.1) we show that there exists a $(n, 2^{-\Omega(n)})$-bias generator $M : \{0,1\}^{2n} \times \{0,1\}^n \to \{0,1\}^n$ which can be computed in quasilinear time $\tilde{O}(n)$ or by a $\mathbf{TC^0}$ circuit (i.e., a constant-depth circuit with unbounded fan-in AND, OR and Majority gates). The following corollary follows.

Corollary 4.8. *Suppose that $\text{EOWF}(\text{XOR-MAJ})$ holds. Then, there exists a non-adaptive PRF $F_3 : \{0,1\}^{3n} \times \{0,1\}^n \to \{0,1\}^{n/\log^2 n}$ which is computable by a boolean circuit of size $\tilde{O}(n)$. Moreover, for every fixed key k, the function $F_3(k, \cdot)$ can be computed by a $\mathbf{TC^0}$ circuit.*

Proof. Let $\mathcal{P} = \text{XOR-MAJ}$ and observe that $\mathcal{P}_\beta = \text{XOR-TH}$ for sufficiently small β (e.g., every $\beta < 1/6$). By Corollary 3.2, $\text{EOWF}(\text{XOR-TH})$ implies $\text{EPRG}(\text{XOR-TH})$, which in turn, implies, by Theorem 4.7, that F_3 is a non-adaptive PRF.

The efficiency analysis is identical to the analysis of F_2 except that we need to add the complexity of M which can be computed in quasilinear time $\tilde{O}(n)$ or by a $\mathbf{TC^0}$ circuit. (See Theorem 5.1). $\qquad\square$

Under the strong EPRG(XOR-TH) assumption, a logarithmic d implies that F_3 is $(q, t = \exp(n^{1-\beta}), \varepsilon = \exp(-n^{1-\beta}))$ secure for every polynomial q and every constant β. For polynomial locality $d = n^\delta$, for some constant $\delta > 0$, we get $q = \exp(n^{\Omega(\delta)})$, $t = \exp(n^{1-\Omega(\delta)})$ and $\varepsilon = \exp(-n^{1-\Omega(\beta)})$.

5 A Bitwise Independent Generator Construction

In this section we construct an efficient generator that is (t, ε)-bitwise independent in the regime of $t = n$ and negligible ε.

Theorem 5.1. *Let k_0, k_1 be two keys chosen uniformly from $\mathrm{GF}(2^n)$. For $x \in \mathrm{GF}(2^n)$, define the generator $\mathcal{V}_{k_0,k_1}(x) := \frac{k_1}{k_0+x}$. Then, $\mathcal{V}$ is $(d, d \cdot 2^{d/2+1-n})$-bitwise independent for any $d \leq 2^n$. Furthermore, the generator $\mathcal{V}$ can be computed by a circuit of quasilinear size $O(n \log^2 n \log \log n)$ and by a $\mathbf{TC^0}$ circuit.*

Proof. We observe that in order to prove that $\mathcal{V}$ is $(d, d \cdot 2^{d/2+1-n})$-bitwise independent, it is sufficient to prove that $\mathcal{V}$ is $(d, \frac{d}{2^{n-1}})$-linear-fooling over $\mathrm{GF}(2^n)$. Indeed, we know that (t, ε)-linear-fooling over $\mathrm{GF}(2^n)$ implies (t, ε)-bias over $\mathrm{GF}(2)$ [Tzu09, Theorem 4.5], which in turn implies $(t, 2^{t/2} \cdot \varepsilon)$-bitwise independence [NN93, Corollary 2.1].

We now turn to showing that $\mathcal{V}$ is $(d, \frac{d}{2^{n-1}})$-linear-fooling over $\mathrm{GF}(2^n)$ for any $d \leq 2^n$. The proof is based on the work of [MV12, Theorem 3.5]. We prove that $\mathcal{V}$ is $(d, \frac{d}{2^{n-1}})$-linear-fooling over $\mathrm{GF}(2^n)$, i.e., for any distinct $a_1, \ldots, a_d \in \mathrm{GF}(2^n)$, any d constants $b_1, \ldots, b_t$ from $\mathrm{GF}(2^n)$ (that are not all equal to zero), we have that

$$\Delta\left(\sum_{i=1}^d b_i \mathcal{V}_{k_0,k_1}(a_i) \; ; \; U_{\mathrm{GF}(2^n)} \right) \leq \frac{d}{2^{n-1}}.$$

After letting $p(x)$ denote the polynomial $\sum_{i=1}^d \frac{b_i}{x+a_i} = \sum_{i=1}^d b_i(x+a_i)^{2^n-2}$, we get that $\sum_{i=1}^d b_i \mathcal{V}_{k_0,k_1}(a_i)$ can be rewritten as $k_1 \cdot p(k_0)$. Observe that conditioned on $p(k_0) \neq 0$, we have that $k_1 \cdot p(k_0)$ is uniformly distributed over $\mathrm{GF}(2^n)$. Hence, it suffices to show that $p(x)$ has at most $2d-1$ distinct roots. First, we define auxiliary polynomials:

$$\bar{p}(x) := p(x) \cdot \prod_{j=1}^d (a_j + x) = \sum_{i=1}^d \left[b_i(x + a_i)^{2^n-1} \prod_{j \neq i}(a_j + x) \right],$$

and

$$\bar{p}_*(x) := \sum_{i=1}^d b_i \prod_{j \neq i}(a_j + x).$$

Observe that any root y of $p(x)$ is also a root of $\bar{p}(x)$. Moreover, note that for any $y \notin \{a_1, \ldots, a_d\}$ we have that $\bar{p}(y) = \bar{p}_*(y)$ (since $y^{2^n-1} = 1$ for any non-zero y). Hence, the only possible roots of $p(x)$ are the roots of $\bar{p}_*(x)$ and $\{a_1, \ldots, a_d\}$. This means that in order to show that $p(x)$ has at most $2d-1$ distinct roots, it is sufficient to show that $\bar{p}_*(x)$ has at most $d-1$ distinct roots. Because $\bar{p}_*(x)$ is a degree $d-1$ polynomial, this will always be the case unless $\bar{p}_*(x)$ is identically

zero. This is ruled out by observing that $\overline{p}_*(a_i) \neq 0$, where i is chosen such that $b_i \neq 0$. Indeed, $\overline{p}_*(a_i) = b_i \prod_{j \neq i}(a_j + a_i)$ which is non-zero because $a_1, \ldots, a_d$ are distinct.

(Complexity of $\mathcal{V}$) Finally, we turn to the analysis of the circuit complexity of $\mathcal{V}$. The complexity of $\mathcal{V}$ equals to the complexity of the division and summation circuits (dividing k_1 by $k_0 + x$). As stated in [MV12] this can be done by a $\mathbf{TC^0}$ circuit or by a circuit of size $O(n \log^2 n \log \log n)$ using the techniques of [GvzGPS00]. $\qquad\square$

References

[ABG+14]   A. Akavia, A. Bogdanov, S. Guo, A. Kamath, and A. Rosen. Candidate weak pseudorandom functions in AC^0 MOD_2. In M. Naor, editor, *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*, pages 251–260. ACM, 2014.

[ABR12]   B. Applebaum, A. Bogdanov, and A. Rosen. A dichotomy for local small-bias generators. In R. Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 600–617. Springer, 2012.

[ABW10]   B. Applebaum, B. Barak, and A. Wigderson. Public-key cryptography from different assumptions. In L. J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 171–180. ACM, 2010.

[AHI05]   M. Alekhnovich, E. A. Hirsch, and D. Itsykson. Exponential lower bounds for the running time of DPLL algorithms on satisfiable formulas. *J. Autom. Reasoning*, 35(1-3):51–72, 2005.

[AKS83]   M. Ajtai, J. Komlós, and E. Szemerédi. An o(n log n) sorting network. In D. S. Johnson, R. Fagin, M. L. Fredman, D. Harel, R. M. Karp, N. A. Lynch, C. H. Papadimitriou, R. L. Rivest, W. L. Ruzzo, and J. I. Seiferas, editors, *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 1–9. ACM, 1983.

[AL15]   B. Applebaum and S. Lovett. Algebraic attacks against random local functions and their countermeasures. *Electronic Colloquium on Computational Complexity (ECCC)*, 22:172, 2015. To appear in STOC 2016.

[Ale03]   M. Alekhnovich. More on average case vs approximation complexity. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings* [DBL03], pages 298–307.

[App13]   B. Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM J. Comput.*, 42(5):2008–2037, 2013. Preliminary version in STOC 2012.

[App14]   B. Applebaum. Bootstrapping obfuscators via fast pseudorandom functions. In P. Sarkar and T. Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th*

International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II, volume 8874 of Lecture Notes in Computer Science, pages 162–172. Springer, 2014.

[App15]   B. Applebaum. Cryptographic hardness of random local functions - survey. Electronic Colloquium on Computational Complexity (ECCC), 22:27, 2015.

[BH15]   I. Berman and I. Haitner. From non-adaptive to adaptive pseudorandom functions. J. Cryptology, 28(2):297–311, 2015.

[BMR10]   D. Boneh, H. W. Montgomery, and A. Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, editors, Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010, pages 131–140. ACM, 2010.

[BPR12]   A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In D. Pointcheval and T. Johansson, editors, Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings, volume 7237 of Lecture Notes in Computer Science, pages 719–737. Springer, 2012.

[BQ12]   A. Bogdanov and Y. Qiao. On the security of goldreich's one-way function. Computational Complexity, 21(1):83–127, 2012.

[BR13]   A. Bogdanov and A. Rosen. Input locality and hardness amplification. J. Cryptology, 26(1):144–171, 2013.

[CEMT14]   J. Cook, O. Etesami, R. Miller, and L. Trevisan. On the one-way function candidate proposed by Goldreich. ACM Trans. Comput. Theory, 6(3):14:1–14:35, July 2014.

[CGH$^+$85]   B. Chor, O. Goldreich, J. Håstad, J. Friedman, S. Rudich, and R. Smolensky. The bit extraction problem of t-resilient functions (preliminary version). In 26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985, pages 396–407. IEEE Computer Society, 1985.

[DBL03]   44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings. IEEE Computer Society, 2003.

[DI05]   I. Damgård and Y. Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In V. Shoup, editor, Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings, volume 3621 of Lecture Notes in Computer Science, pages 378–394. Springer, 2005.

[DLS14]   A. Daniely, N. Linial, and S. Shalev-Shwartz. From average case complexity to improper learning complexity. In D. B. Shmoys, editor, Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014, pages 441–448. ACM, 2014.

[FPV15]   V. Feldman, W. Perkins, and S. Vempala. On the complexity of random satisfiability problems with planted solutions. In R. A. Servedio and R. Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 77–86. ACM, 2015.

[GGM86]   O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GL89]   O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In D. S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989.

[Gol00]   O. Goldreich. Candidate one-way functions based on expander graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(90), 2000.

[GVW12]   S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In Safavi-Naini and Canetti [SC12], pages 162–179.

[GvzGPS00] S. Gao, J. von zur Gathen, D. Panario, and V. Shoup. Algorithms for exponentiation in finite fields. *J. Symb. Comput.*, 29(6):879–889, 2000.

[HILL99]   J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. Preliminary versions in *STOC'89* and *STOC'90*.

[IKOS08]   Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In C. Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 433–442. ACM, 2008.

[Kha93]   M. Kharitonov. Cryptographic hardness of distribution-specific learning. In S. R. Kosaraju, D. S. Johnson, and A. Aggarwal, editors, *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 372–381. ACM, 1993.

[KS09]   A. R. Klivans and A. A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. *J. Comput. Syst. Sci.*, 75(1):2–12, 2009.

[LF80]   R. E. Ladner and M. J. Fischer. Parallel prefix computation. *J. ACM*, 27(4):831–838, 1980.

[LMN93]   N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993.

[LW09]   A. B. Lewko and B. Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In E. Al-Shaer, S. Jha, and A. D. Keromytis, editors, *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009*, pages 112–120. ACM, 2009.

[MP75]   D. E. Muller and F. P. Preparata. Bounds to complexities of networks for sorting and for switching. *J. ACM*, 22(2):195–201, 1975.

[MST03]   E. Mossel, A. Shpilka, and L. Trevisan. On e-biased generators in NC0. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings* [DBL03], pages 136–145.

[MV12]   E. Miles and E. Viola. Substitution-permutation networks, pseudorandom functions, and natural proofs. In Safavi-Naini and Canetti [SC12], pages 68–85.

[NN93]   J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[NR95]   M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of psuedo-random functions. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, 23-25 October 1995*, pages 170–181. IEEE Computer Society, 1995.

[NR97]   M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 458–467. IEEE Computer Society, 1997.

[NRR00]   M. Naor, O. Reingold, and A. Rosen. Pseudo-random functions and factoring (extended abstract). In F. F. Yao and E. M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 11–20. ACM, 2000.

[OW14]   R. O'Donnell and D. Witmer. Goldreich's PRG: evidence for near-optimal polynomial stretch. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 1–12. IEEE, 2014.

[PF79]   N. Pippenger and M. J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.

[PW88]   L. Pitt and M. K. Warmuth. Reductions among prediction problems: on the difficulty of predicting automata. In *Proceedings: Third Annual Structure in Complexity Theory Conference, Georgetown University, Washington, D. C., USA, June 14-17, 1988*, pages 60–69. IEEE Computer Society, 1988.

[RR97]   A. A. Razborov and S. Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

[SC12]   R. Safavi-Naini and R. Canetti, editors. *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*. Springer, 2012.

[Tzu09]   Y. Tzur. Notions of weak pseudorandomness and $GF(2^n)$-polynomials. Master's thesis, Weizmann Institute of Science, 2009.

[Val84]   L. G. Valiant. A theory of the learnable. In R. A. DeMillo, editor, *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 436–445. ACM, 1984.

[Weg87]   I. Wegener. *The Complexity of Boolean Functions*. Teubner / Wiley, 1987.

[Yao82]   A. C. 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.

A Omitted Proofs

A.1 Proof of Lemma 2.4

Lemma 2.4 (Restated). *Let* Ext *be a strong* $(m - r, \delta)$-*extractor for m-bit sources. Let* $Z = Z_1 || \cdots || Z_q$ *be a generalized* $(qm - r)$-*bit-fixing source, where each* $|Z_i| = m$. *Then for a uniformly chosen seed* s, *the random variable* $(s, \mathrm{Ext}_s(Z_1), \ldots, \mathrm{Ext}_s(Z_q))$ *is* $(q \cdot \delta)$-*statistically indistinguishable from uniform.*

Proof. Because $Z = Z_1 || \cdots || Z_q$ is a generalized $(qm - r)$-bit-fixing source, there exist at least $(qm - r)$ bits of Z that are uniform, while the remaining ones are independent from them. This implies that the bits of each Z_i can be split into two parts G_i and B_i such that G_i's are independent uniform bit strings of length $m - r$ and B_i's are independent of them. To simplify the notation we assume that G_i is the prefix of Z_i and B_i is the suffix of Z_i, i.e., one can rewrite Z_i as (G_i, B_i). We have that

$$(s, \mathrm{Ext}_s(Z_1), \ldots, \mathrm{Ext}_s(Z_q)) = (s, \mathrm{Ext}_s(G_1, B_1), \ldots, \mathrm{Ext}_s(G_q, B_q)).$$

Consider now the distribution $T(b_1, \ldots, b_q)$ obtained from the distribution

$$(s, \mathrm{Ext}_s(G_1, B_1), \ldots, \mathrm{Ext}_s(G_q, B_q))$$

by fixing $B_1, \ldots, B_q$ to some $b_1, \ldots, b_q$:

$$T(b_1, \ldots, b_q) := (s, \mathrm{Ext}_s(G_1, B_1), \ldots, \mathrm{Ext}_s(G_q, B_q]) \mid (B_1 = b_1, \ldots, B_q = b_q).$$

We have that each (G_i, b_i) is a bit-fixing source, such that (G_i, b_i)'s are mutually independent. We now employ the following fact about strong extractors.

Fact A.1. *Let* Ext *be a strong* $(m - r, \delta)$-*extractor for m-bit sources. Let* $X_1, \ldots, X_q$ *be* q *m-bit random variables such that each* X_i *is a* $(m - r)$-*bit-fixing source and* X_i's *are mutually independent. Then, for a uniformly chosen seed* s, *the random variable* $(s, \mathrm{Ext}_s(X_1), \ldots, \mathrm{Ext}_s(X_q))$ *is* $(q \cdot \delta)$-*statistically indistinguishable from uniform.*

After applying the fact, we get that each $T(b_1, \ldots, b_q)$ is $(q \cdot \delta)$-statistically close to uniform. It remains to notice that the original random variable $(s, \mathrm{Ext}_s(Z_1), \ldots, \mathrm{Ext}_s(Z_q))$ is a convex combination of $T(b_1, \ldots, b_q)$'s and hence is also $(q \cdot \delta)$-statistically close to uniform (see the fact below).

Fact A.2. *For any convex combination of distributions* $X = \sum_{i=1}^{k} w_i D_i$ *and any probability distribution* Y, *it holds that* $\Delta(X; Y) \leq \sum_{i=1}^{k} w_i \Delta(D_i; Y)$.

<div align="right">□</div>

A.2 PRG with Imperfect Expansion

Lemma 4.5 (Restated). *Let G be a $[n, m, d]$-hypergraph which is $n^{1-\beta}$-imperfect $(n^{1-\beta}, (1-\beta)d)$-expander for some constant $\beta \in (0,1)$. Then, under assumption $\mathsf{EPRG}(\mathcal{P}_\beta, m, \beta, t, \varepsilon)$, the random variable $f_{G,P}(U_n)$ is (t, ε)-computationally indistinguishable from a generalized $(m-n^{1-\beta})$ bit-fixing source.*

Proof. Let A denote the set of hyperedges that make G imperfect and let B be the set of the remaining hyperedges from G. Let C denote the set $\Gamma(A)$ and let D be the set of the remaining indices in $[n]$. Let $J := G \setminus A$ and $r := |A|$. Without loss of generality, assume that all indices of hyperedges in A are smaller than indices in B and all indices in C are smaller than indices in D. Let $G(i,j)$ denote the jth index in the ith hyperedge from G. For a $|C|$-bit string z, consider an n-bit random variable X_z such that $X_z[D] = U_{|D|}$ and $X_z[C]$ is fixed to z. Now define a random variable $Y_z := f_{G,P}(X_z)$, where $P = (P_1, \ldots, P_m)$. By construction we have that $Y_z[B] = f_{J,P'}(X[D])$, where $P' = (P'_{r+1}, \ldots, P'_m)$ such that for each $i \in [r+1, \ldots, m]$ the predicate P'_i is obtained from P_i by fixing its jth input bit to $z_{G(i,j)}$ if $G(i,j) \in [|C|]$. Given that J is $(n, m-r)$-hypergraph which is $(n^{1-\beta}, (1-\beta)d)$-expanding and P' is an $(1-\beta)$-restriction of P, after applying assumption $\mathsf{EPRG}(\mathcal{P}_\beta, m, \beta, t, \varepsilon)$, we get that the random variable $Y_z[B] = f_{J,P'}(X[D])$ is (t, ε)-indistinguishable from uniform, while $Y_z[A]$ is fixed. This means that Y_z is (t, ε)-indistinguishable from an $(m-r)$ bit-fixing source R_z whose A entries are fixed. Since the random variable $Y = f_{G,P}(U_n)$ is a convex combination of $\{Y_z\}_{z \in \{0,1\}^{|C|}}$, we conclude that the random variable $Y = f_{G,P}(U_n)$ is computationally indistinguishable from a generalized $(m-r)$ bit-fixing source. $\square$

A.3 Proof of Generalized Expansion Lemma (Lemma 2.5)

Let $a := (1-\beta)d$. Let $\varepsilon = 2^{-\Omega(n)}$, and assume that the hypergraph G is $(r+t, d, \varepsilon)$-random. The proof follows by a standard calculation. Details follow.

First, we upper bound the probability that a fixed set S of size $s \leq r+t$ is non-expanding which we denote with $p(s)$. We show an upper bound on $p(s)$ in two steps. In the first step, we compute the probability $p'(s)$ that a set S' with exactly sd random outgoing edges is non-expanding. In the second step, we use the fact that G is $(r+t, d, \varepsilon)$-random and therefore $p(s) \leq p'(s) + \varepsilon$.

Consider a set S' of with exactly sd random outgoing edges. By definition $p'(s) = \Pr[|\Gamma(S')| \leq as]$. Now fix any $T \subseteq [n]$ of size as. The probability that S' is mapped inside T is exactly $\left(\frac{|T|}{n}\right)^{sd} = (\frac{as}{n})^{sd}$. By applying a union bound over all T's of size as, we get that $p'(s) \leq \binom{n}{as} \left(\frac{as}{n}\right)^{ds}$. It remains to notice that since G is $(r+t, d, \varepsilon)$-random every set of hyperedges S in G of size $s \leq r+t$ has at least sd outgoing edges that are ε-close to uniform. Hence, we have that $p(s) \leq p'(s) + \varepsilon$ for $s \leq r+t$. Finally, we conclude that

$$p(s) \leq p'(s) + \varepsilon \leq \binom{n}{as} \left(\frac{as}{n}\right)^{ds} + \varepsilon \leq \left(\frac{en}{as}\right)^{as} \left(\frac{as}{n}\right)^{ds} + \varepsilon, \tag{2}$$

where the last inequality holds because for all n, k we have that $\binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$.

Now we turn to proving the main statement of the lemma. Let I be a largest set of hyperedges of size at most $r+t$ such that $\Gamma(I) \leq a|I|$. Let $G' := G \setminus I$. Then any set S of hyperedges from G' of size at most $r+t-|I|$ has $\Gamma(S) \geq a|S|$. Thus, in order to show that G is an t-imperfect

(r,a)-expander, we just need to show that $|I| \leq t$ except with probability $n^{-(t+1)d\beta^2/10}$. Consider a maximum non-expanding I, with $|I| \leq r + t$. Then, by a union-bound,

$$\Pr[|I| > t] = \sum_{s=t+1}^{t+r} \Pr[|I| = s] \leq \sum_{s=t+1}^{t+r} \left(\frac{me}{s}\right)^s p(s).$$

Plugging in the upper bound for $p(s)$ from Eq. 2, we get that

$$\Pr[|I| > t] \leq \underbrace{\sum_{s=t+1}^{t+r} \left(\frac{me}{s}\right)^s \left(\frac{ne}{as}\right)^{as} \left(\frac{as}{n}\right)^{ds}}_{S_1} + \underbrace{\varepsilon \sum_{s=t+1}^{t+r} \left(\frac{me}{s}\right)^s}_{S_2}.$$

We now give upper bounds for S_1 and S_2 separately. We start with giving an upper bound for S_1. After regrouping the summands, S_1 can be rewritten as

$$S_1 = \sum_{s=t+1}^{t+r} \left(e^{a+1} \cdot a^{d-a} \cdot m \cdot \frac{s^{d-a-1}}{n^{d-a}}\right)^s.$$

Let $u_s := \left(e^{a+1} \cdot a^{d-a} \cdot m \cdot \frac{s^{d-a-1}}{n^{d-a}}\right)$, denote the sth term inside the brackets of S_1. We now give an upper bound for u_s by splitting it into a product of four terms e^{a+1}, a^{d-a}, $\frac{s^{d-a-1}}{n^{d-b}}$, m, and then giving an upper bound for each of the terms.

First, we have that

$$e^{a+1} = e^{(1-\beta)d+1} \overset{(\star)}{\leq} e^d = n^{d/\ln n}. \tag{3}$$

where $(\star)$ holds because of the assumption $d \geq 4/\beta^2$. Second, we have that

$$a^{d-a} = a^{\beta d} \leq d^{\beta d} \leq n^{(\beta^2/4)d}, \tag{4}$$

where the last inequality holds because of the assumption $d \leq n^{\beta/4}$. Third, we have that

$$\frac{s^{d-a-1}}{n^{d-a}} \leq \left(\frac{s}{n}\right)^{d-a} \overset{(\star)}{\leq} \left(\frac{2n^{1-\beta}}{n}\right)^{d-a} = 2^{\beta d} \cdot n^{-\beta^2 d} \leq n^{d/\ln n} \cdot n^{-\beta^2 d}, \tag{5}$$

where $(\star)$ holds because $s \leq t + r \leq 2n^{1-\beta}$. Finally, after combining Equations 3, 4, 5, and the fact that $m \leq n^{(\beta^2/4)d}$, we get that

$$u_s \leq n^{(2/\ln n)d} \cdot n^{(\beta^2/4)d} \cdot n^{(\beta^2/4)d} \cdot n^{-\beta^2 d} = n^{-(\beta^2/2+2/\ln n)d} \leq n^{-(0.4\beta^2)d},$$

where the last inequality holds because β is a constant and $2/\ln n$ is $o(1)$. Given the upper bound for u_s, we can upper bound the sum S_1 as follows.

$$S_1 = \sum_{s=t+1}^{t+r} (u_s)^s \leq \sum_{s=t+1}^{t+r} \left(n^{-(0.4\beta^2)d}\right)^s \leq rn^{-(0.4\beta^2)\cdot d\cdot(t+1)}.$$

28

where the last inequality holds since there are r summands and each is maximized when $s = t + 1$. Finally, it remains to notice that since $\beta^2 d/4 \geq 1$, we have $r \leq n \leq n^{(\beta^2 d/4)(t+1)}$. Hence, $S_1 \leq n^{-0.15\beta^2 d \cdot (t+1)}$.

We complete the proof by showing that $S_2 < 2^{-\Omega(n)}$. Indeed, each summand $\left(\frac{me}{s}\right)^s$ is upper bounded by $(me)^{t+r}$, and therefore

$$S_2 \leq \varepsilon r (me)^{t+r} \leq 2^{-\Omega(n)} \cdot n^{1-\beta} \cdot e^{2n^{1-\beta}} \cdot n^{(\beta^2 d/4) \cdot 2n^{1-\beta}} \leq 2^{-\Omega(n)},$$

where the last inequality follows by noting that $n^{(\beta^2 d/4) \cdot 2n^{1-\beta}} = 2^{o(n)}$. $\qquad\square$

B Array Multi-Access in Quasilinear time

We consider the following functionality. Given $\ell = n/\log n$ indices of length $\log n$ each $I[1], \ldots, I[\ell]$ and a data vector $K \in \{0, 1\}^n$ output $K[I[1]], \ldots, K[I[\ell]]$. We will show that this can done by $O(n \log^2 n \log \log n)$-size circuit. We assume that the input indices are sorted which is without loss of generality since t elements of bit-length $b = \log n$ can be sorted by a circuit of size $O(b\ell \log \ell) = O(n \log n)$ (e.g., using a sorting network [AKS83] where comparison is implemented via Parallel Prefix Computation [LF80]). Instead of describing an $O(n \log^2 n \log \log n)$-size circuit, we describe a Turing Machine M that solves the problem in time $T = O(n \log n)$ using a constant number of tapes. The latter can be simulated by a circuit of size $O(T \log T)$ (e.g., by turning the computation M into an oblivious Turing machine M' of complexity $O(T \log T)$ [PF79] and then moving to a circuit of size $O(T \log T)$). We sketch the description of the machine M. The machine M places the indices I on one tape, the data K on another tape and places the output on a special output tape. During its run, M maintains two counters i and j which are initialized to 1. At each step, M checks if the index $I[i]$ equals to j if this is the case then $K[j]$ is written to the current position in the output tape. Also, the head of the output tape is moved one step and the head of the index tape is moved to the next index. In case of inequality, the head of the data tape is moved forward by one step, and the counter j is increased by one. Since each step costs $O(\log n)$ operations and there are at most n steps, the overall complexity is $O(n \log n)$.