Pseudorandom Generators with Long Stretch and Low Locality from Random Local One-Way Functions[*]

Benny Applebaum[†]

July 17, 2013

Abstract

We continue the study of *locally-computable* pseudorandom generators (PRG) $G : \{0,1\}^n \to \{0,1\}^m$ such that each of their outputs depend on a small number of d input bits. While it is known that such generators are likely to exist for the case of small sub-linear stretch $m = n + n^{1-\delta}$, it is less clear whether achieving larger stretch such as $m = n + \Omega(n)$, or even $m = n^{1+\delta}$ is possible. The existence of such PRGs, which was posed as an open question in previous works (e.g., [Cryan and Miltersen, MFCS 2001], [Mossel, Shpilka and Trevisan, FOCS 2003], and [Applebaum, Ishai and Kushilevitz, FOCS 2004]), has recently gained an additional motivation due to several interesting applications.

We make progress towards resolving this question by obtaining several local constructions based on the one-wayness of "random" local functions – a variant of an assumption made by Goldreich (ECCC 2000). Specifically, we construct collections of PRGs with the following parameters:

- Linear stretch $m = n + \Omega(n)$ and constant locality $d = O(1)$.
- Polynomial stretch $m = n^{1+\delta}$ and *any* (arbitrarily slowly growing) super-constant locality $d = \omega(1)$, e.g., $\log^\star n$.
- Polynomial stretch $m = n^{1+\delta}$, constant locality $d = O(1)$, and inverse polynomial distinguishing advantage (as opposed to the standard case of $n^{-\omega(1)}$).

Our constructions match the parameters achieved by previous "ad-hoc" candidates, and are the first to do this under a one-wayness assumption. At the core of our results lies a new search-to-decision reduction for random local functions. This reduction also shows that some of the previous PRG candidates can be based on one-wayness assumptions. Altogether, our results fortify the existence of local PRGs of long stretch.

As an additional contribution, we show that our constructions give rise to strong inapproximability results for the densest-subgraph problem in d-uniform hypergraphs for constant d. This allows us to improve the previous bounds of Feige (STOC 2002) and Khot (FOCS 2004) from constant inapproximability factor to n^{ε}-inapproximability, at the expense of relying on stronger assumptions.

[*]Preliminary version appeared in STOC 2012.

[†]School of Electrical Engineering, Tel Aviv University, `bennyap@post.tau.ac.il`. Work partially done while a postdoc at the Weizmann Institute of Science. Supported by Koshland and Alon Fellowship, ISF grant 1155/11, and by the Check Point Institute for Information Security.

1 Introduction

Pseudorandomness is one of the most fundamental concepts in cryptography and complexity theory. Formally, an *ε-pseudorandom generator* (ε-PRG) $f : \{0,1\}^n \to \{0,1\}^m$ maps a random n-bit seed $x = (x_1, \ldots, x_n)$ into a longer pseudorandom string $y = (y_1, \ldots, y_m)$ such that no polynomial-time adversary can distinguish y from a truly random m-bit string with *distinguishing advantage* better than ε (by default, ε is negligible $n^{-\omega(1)}$). The notion of PRGs is highly non-trivial and, in some sense, very fragile: any noticeable regularity or correlation in the output y immediately violates pseudorandomness. This is, for example, fundamentally stronger than the notion of *one-way functions* which asserts that f is hard to invert on typical images. For this reason, a considerable research effort has been made to put PRGs on more solid ground at the form of one-wayness assumptions [40, 33, 23, 16, 38, 34, 27]. In this paper we aim to establish similar firm foundations for the case of locally-computable PRGs with large stretch.

Locally-computable PRGs offer an extreme level of parallelism. In such PRGs each output y_i depends only on a small number of d input bits. Our goal is to gain many pseudorandom bits, i.e., maximize the *stretch* $m - n$, while keeping the *locality* d as small as possible. Ultimately, d is a constant that does not grow with the total input length or the level of security; In this case, the computation can be carried in *constant*-parallel time, which is captured by the complexity class $\mathbf{NC^0}$.

This strong efficiency requirement seems hard to get as, at least intuitively, such form of locality may lead to algorithmic attacks. Still, it was shown in [6] that, for the regime of sub-linear stretch $m = n + n^{1-\delta}$, PRGs with constant locality exist under standard cryptographic assumptions (e.g., the hardness of factoring, discrete log, or lattice problems, or more generally, the existence of log-space computable one-way functions [26]). Unfortunately, it is unknown how to extend this result to linear stretch ($m > (1+\delta)n$) let alone polynomial stretch ($m > n^{1+\delta}$).[1] This raises the following question which was posed by several previous works (e.g., [14, 36, 6, 7, 29]):

> How long can be the stretch of a pseudorandom generator with locality d? How large should d be to achieve linear or polynomial stretch?

Local PRGs with linear and polynomial stretch are qualitatively different than ones with sub-linear stretch. For example, as shown in [29], such PRGs allow to improve the *sequential complexity* of cryptography: a linear-stretch PRG (hereafter abbreviated by LPRG) with constant locality would lead to implementations of primitives (e.g., public-key encryption, commitment schemes) with *constant* computational overhead, and a polynomial-stretch PRG (hereafter abbreviated by PPRG) with constant locality would lead to secure computation protocols with *constant* computational overhead – a fascinating possibility which is not known to hold under any other cryptographic assumption. From a practical point of view, large stretch PRGs with low locality give rise to highly efficient stream-ciphers that can be implemented by fast parallel hardware.

Furthermore, this difference between sub-linear and linear/polynomial stretch turns out to have applications also outside cryptography. As shown in [7], local PRGs with large stretch lead to strong (average-case) inapproximability results for constraint satisfaction problems such as Max3SAT under a natural distribution: LPRGs with constant locality rule out the existence of a PTAS, whereas

[1]In fact, for the special case of 4-local functions, there is a provable separation: such functions can compute sub-linear PRGs [6] but *cannot* compute polynomial-stretch PRGs [14, 36]. For larger locality $d \geq 5$, the best upper-bound on m is $n^{d/2}$ [36].

PPRGs yield *tight* bounds that match the upper-bounds achieved by the simple "random assignment" algorithm.

Although local LPRGs and PPRGs are extremely useful, their existence is not well established. Even when $d = O(\log n)$ it is unknown how to construct LPRGs, let alone PPRGs, whose security can be reduced to some other weaker cryptographic assumption, e.g., a one-wayness assumption.[2] This state of affairs has lead to a more direct approach. Rather than trying to obtain PPRGs or LPRGs based on one-wayness assumptions, several researchers suggested concrete candidates for LPRGs with constant locality and negligible distinguishing advantage [2, 7] and PPRGs with constant locality and inverse polynomial distinguishing advantage [36, 4]. (A detailed account of this line of works is given in Section 1.3.) All of these candidates are essentially based on what we call *random local functions*, i.e., each output bit is computed by applying some fixed d-ary predicate Q to a randomly chosen d-size subset of the input bits. Formally, this can be viewed as selecting a random member from a collection $\mathcal{F}_{Q,n,m}$ of d-local functions where each member $f_{G,Q} : \{0,1\}^n \to \{0,1\}^m$ is specified by a d-uniform hypergraph G with n nodes and m hyperedges, and the i-th output of $f_{G,Q}$ is computed by applying the predicate Q to the d inputs that are indexed by the i-th hyperedge.

Remark 1.1 (**Collection vs. Single function**). *The above construction gives rise to a* collection *of local PRGs, where a poly-time preprocessing is used to publicly pick (once and for all) a random instance f from the collection. The adversary is given the description of the chosen function, and its distinguishing advantage is measured with respect to a random seed and a random member of the family. (See Section 3 and [22, Section 2.4.2], [28] for formal definitions). The use of collections is standard in the context of parallel cryptography (e.g., in number-theoretic constructions preprocessing is used to set-up the group or choose a random composite number [37, 15, 25]) and essentially has no effect on the aforementioned applications, hence we adopt it as our default setting. (See [6, Appendix A] for detailed discussion.) In our case the preprocessing typically has a parallel implementation in* $\mathbf{AC^0}$.

1.1 Our constructions

The gap between the rich applications of large-stretch PRGs in $\mathbf{NC^0}$ and the lack of provable constructions is highly unsatisfactory. Our goal in this paper is to remedy the situation by replacing the ad-hoc candidates with constructions whose security can be based on a more conservative assumption. Specifically, our results essentially show that in order to construct local PRGs with output length m it suffices to assume that random local functions are *one-way*; namely, that it is hard to *invert* a random member of the collection $\mathcal{F}_{Q,n,m'}$ for m' of the same magnitude as m. This one-wayness assumption, originally made by Goldreich [21] and further established in [39, 3, 13, 35, 10, 4, 11], is significantly weaker (i.e., more plausible) than the corresponding pseudorandomness assumption (see Section 1.3 for discussion). Let us now state our main results, starting with the case of linear stretch.

Theorem 1.2 (**Local LPRG**). *For every constant d, d-ary predicate Q and $m > \Omega_d(n)$ the following holds. If the collection $\mathcal{F}_{Q,n,m}$ is one-way, then there exists a collection of ε-LPRGs with constant locality and negligible distinguishing advantage $\varepsilon = n^{-\omega(1)}$.*

[2]To the best of our knowledge, even the class $\mathbf{AC^0}$ (which is strictly stronger than $O(\log n)$-local functions) does not contain any (provably-secure) large-stretch PRG, and only in $\mathbf{TC^0}$, which is strictly more powerful than $\mathbf{AC^0}$, such constructions are known to exist [37].

Theorem 1.2 is applicable for every choice of predicate Q, and it provides the first construction of an LPRG in $\mathbf{NC^0}$ based on a one-wayness assumption. Moving to the case of polynomial stretch, we say that the predicate Q is *sensitive* if *some* of its coordinates i has full influence (i.e., flipping the value of the i-th variable always changes the output of Q).

Theorem 1.3 (**Local PPRG**). *For every constant d, sensitive d-ary predicate Q, constant $\delta > 0$, constant $a > 0$ and function $b(n) < o(n)$ the following holds. If $\mathcal{F}_{Q,n,n^{1+\delta}}$ is one-way, then there exists a collection of ε-PRGs with output length n^a, distinguishing advantage $\varepsilon \le 1/n^b + \mathrm{neg}(n)$ and locality $d' = (bd/\delta)^{O(\frac{\log a}{\delta})}$.*

Specifically, by letting a and b be constants, we obtain polynomial stretch, fixed inverse polynomial distinguishing advantage, and *constant* locality. Alternatively, setting $b = b(n) = \omega(1)$ (e.g., $b = \log^*(n)$), yields polynomial stretch and *negligible* distinguishing advantage $n^{-\omega(1)}$ at the expense of having slightly super-constant locality.

The PPRGs constructions of Theorem 1.3 are the first to achieve constant locality and inverse-polynomial distinguishing advantage (resp., super-constant locality and negligible distinguishing advantage) under a one-wayness assumption. These parameters also match the ones achieved by known heuristic candidates for PPRGs.[3] We mention that there are sensitive predicates for which $\mathcal{F}_{Q,n,m=n^{1+\delta}}$ seems to be one-way. As a concrete example, the predicate $Q(x_1, x_2, x_3, x_4, x_5) = (x_1 \wedge x_2) \oplus x_3 \oplus x_4 \oplus x_5$, introduced by [36], provides security against a large family of "DPLL-style" algorithms [13] and against degree-1 distinguishers over $\mathbb{F}_2$ [5].

Following the previous theorems, one may ask whether it is possible to show that $\mathcal{F}_{Q,n,m}$ *itself* is pseudorandom. We show that this is indeed the case:

Theorem 1.4 (**Random Local Function is PPRG**). *For every constants a, b and d, and every sensitive d-ary predicate Q, if $\mathcal{F}_{Q,n,n^{3a+2b}}$ is one-way then $\mathcal{F}_{Q,n,n^a}$ is n^{-b}-pseudorandom.*[4]

As a corollary, we reduce the pseudorandomness of some of the previous ad-hoc constructions to a one-wayness assumption. We view Theorem 1.4 as one of the main conceptual contributions of this work. The ensemble $\mathcal{F}_{Q,n,m}$ is highly interesting for its own sake as it generalizes an important and well-studied family of random constraint satisfaction problems (e.g., random planted 3-SAT [20, 12, 1]). Indeed, the problem of inverting a random member of the ensemble $\mathcal{F}_{Q,n,m}$ boils down to solving a system of m random d-local (non-linear) equations of the form $y_i = Q(x_{i_1}, \ldots, x_{i_d})$ with a planted solution x. Theorem 1.4 yields an average-case search-to-decision reduction for this problem. Combined with the results of [7], it follows that any non-trivial approximation of the value of the system of equation (i.e., that performs better than the trivial random assignment algorithm) allows to fully recover the planted solution.

1.2 New application: Hardness of the Densest-Sub-Hypergraph Problem

We use Theorem 1.4 to derive new inapproximability results. This continues the line of research started by Feige [18] in which inapproximability follows from average-case hardness.

[3] In the heuristic constructions the dependencies graph G should satisfy non-trivial expansion properties [7], but when $m = n^{1+\Omega(1)}$ and $d = O(1)$ it is unknown how to efficiently sample such a good expander with negligible failure probability.

[4] In the case of a general predicate, we show that each bit in the output of $\mathcal{F}_{Q,n,n^a}$ cannot be predicted with probability better than $1 - \varepsilon$ for some constant $\varepsilon > 0$. (See Theorem 5.1.)

For a d-uniform hypergraph G, we say that a set of nodes S *contains* an edge $e = (v_1, \dots, v_d)$ if all the endpoints of e are in S, i.e., $v_1, \dots, v_d \in S$. For an integer d and real $p \in (0, 1)$, we define the p Densest-Sub-hypergraph Problem for d-uniform hypergraphs ($p - \mathsf{DSH}_d$) as follows.

Definition 1.5 ($p - \mathsf{DSH}_d$)**.** *Given a d-uniform hypergraph G with n nodes and m edges (hereafter referred to as an (n, m, d) hypergraph), distinguish between:*

- ***No case ("Random").*** *Every set S of nodes of density p (i.e., size pn) in G contains at most $p^d(1 + o(1))$ fraction of the edges.*
- ***Yes case ("Planted").*** *There exists a set S of nodes of density p in G that contains at least $p^{d-1}(1 - o(1))$ fraction of the edges.*

That is, the No instance corresponds to a random-looking hypergraph, while the Yes instance contains a planted dense subgraph. The parameter p controls both the approximation ratio and the gap-location (i.e., size of the dense subgraph). This formulation of $p-\mathsf{DSH}_d$ was explicitly presented by Khot [32] (under the term "Quasi-random PCP"), and was implicit in the work of Feige [18]. These works showed that, assuming that **NP** cannot be solved in probabilistic sub-exponential time, the problem is hard for $p = \frac{1}{2}$ and some constant d. The constant p can be improved by taking graph products, however, this increases the degree d. Hence, for a constant degree, the best known inapproximability ratio was constant. In contrast, even in the simplest case of $d = 2$, which corresponds to the famous "Densest Subgraph problem" [19], the best known algorithm [9] achieves an approximation ratio no better than $\Theta(n^{1/4})$.[5] Our next theorem partially closes this gap:

Theorem 1.6. *For every constant d, d-ary predicate Q and real $\delta > 0$, if $\mathcal{F}_{Q,n,n^{1+\delta}}$ is $\frac{1}{\log n}$-pseudorandom then $p - \mathsf{DSH}_d$ is intractable for every $n^{-\frac{\delta}{d+1.01}} \le p \le \frac{1}{2}$.*

Letting $p = \frac{1}{2}$, we derive the same parameters as in [18, 32]. We can obtain much stronger inapproximability ratio of, say, $p = n^{-1/(d+1.01)}$ for a *fixed* locality d, assuming that $\mathcal{F}_{Q,n,n^2}$ is $\frac{1}{\log n}$-pseudorandom. As shown in Theorem 1.4, the latter assumption follows from the one-wayness of $\mathcal{F}_{Q,n,m'}$ for sufficiently large polynomial $m'(n)$.

Interestingly, Theorem 1.6 yields average-case hardness with respect to a "planted distribution". Namely, we show that it is hard to distinguish a random hypergraph (which is likely to be a "No" instance) from a random hypergraph in which a dense random subgraph is planted. (Jumping ahead, the planted subgraph essentially encodes a preimage of the pseudorandom generator.) We also mention that such distributions are used by Arora et al. [8] to show that financial derivatives can be fraudulently mispriced without detection.

1.3 Discussion and Previous Works

1.3.1 Pseudorandomness of $\mathcal{F}_{Q,n,m}$

In [36] it was shown that, for proper choices of the predicate Q and the hypergraph G, the function $f_{G,Q}$ fools linear-tests over $\mathbb{F}_2$. A similar result holds for a *random* member in $\mathcal{F}_{Q,n,m=n^{1+\varepsilon}}$ as long as Q is sufficiently "good" [4]. Recently, a full classification of good predicates (at the form of a dichotomy theorem) was established in [5].

[5] We are unaware of any non-trivial approximation algorithm for the case of hypergraphs, i.e., $d > 2$.

Alekhnovich [2] conjectured that the collection $\mathcal{F}_{Q=\oplus_p,n,m=\Theta(n)}$ is pseudorandom where $\oplus_p$ is a *randomized* predicate which computes $z_1 \oplus z_2 \oplus z_3$ and with some small probability $p < \frac{1}{2}$ flips the result. Although this construction does not lead directly to a local PRG (due to the use of noise), it was shown in [7] that it can be derandomized and transformed into an $\mathbf{NC^0}$ construction with linear stretch. (The restriction to linear stretch holds even if one strengthens Alekhnovich's assumption to $m = \text{poly}(n)$.)

Recently, it is was shown that, for the special case of the noisy-linear predicate $\oplus_p$, one-wayness implies weak pseudorandomness [4]. Specifically, if $\mathcal{F}_{\oplus_p,n,m=O(n \log n)}$ is one-way then $\mathcal{F}_{\oplus_p,n,m=O(n)}$ is pseudorandom with distinguishing advantage $\varepsilon = 1/n$.[6] Our work is highly inspired by this result. From a technical point of view, many of the ideas used in [4] heavily rely on the linear structure of $\oplus_p$, and so part of the challenge in establishing our reductions is to find analogues which work in the general case of arbitrary predicates. (See Section 2 for an overview of our proofs.) As a byproduct, our techniques provide a simpler proof for the case of $\oplus_p$ with slightly better parameters.

1.3.2 One-wayness of $\mathcal{F}_{Q,n,m}$

The ensemble $\mathcal{F}_{Q,n,m}$ was explicitly presented by Goldreich [21] who conjectured one-wayness for the case of $m = n$ and essentially every non-trivial predicate (i.e., non-linear and non-degenerate). In [21, 3, 13, 35, 17, 30] it is shown that a large class of algorithms (including ones that capture DPLL-based heuristics) fail to invert $\mathcal{F}_{Q,n,m}$ in polynomial-time. These results are further supported by the experimental study of [39, 13] which employs, among other attacks, SAT-solvers.

Recently, a strong self-amplification theorem was proved in [11] showing that for $m = \Omega_d(n)$ if $\mathcal{F}_{Q,n,m}$ is hard-to-invert over tiny (sub-exponential small) fraction of the inputs with respect to sub-exponential time algorithm, then the same ensemble is actually hard-to-invert over almost all inputs (with respect to sub-exponential time algorithms). In addition, the one-wayness of $\mathcal{F}_{Q,n,m}$ is actively challenged by the theoretical and practical algorithmic study of random constraint satisfaction problems (e.g., Random 3-SAT, see [20, 12, 1] for surveys). The fact that this research falls short of inverting $\mathcal{F}_{Q,n,m}$ provides a good evidence to its security.[7]

To summarize, when m is linear, i.e., $m = cn$ for arbitrary constant $c > 1$, it is unknown how to invert the function (with respect to a general predicate) in complexity smaller than $2^{\Omega(n)}$. Moreover, even when m is polynomial, e.g., $m = n^{d/100}$, no polynomial-time inversion algorithm is known. In light of this, it seems reasonable to assume that for every constant $c > 1$ there exists a sufficiently large locality d and a predicate Q for which $\mathcal{F}_{Q,n,n^c}$ cannot be inverted in polynomial time.

1.3.3 One-wayness vs. pseudorandomness

The above works indicate that one-wayness is much more solid than pseudorandomness. We wish to emphasize that this is true even with respect to heuristic constructions. Indeed pseudorandomness is quite fragile, especially with low locality, as in this case even the task of avoiding simple regularities

[6] We believe that this result can be combined with the techniques of [7] to yield a local *weak* LPRG with $\varepsilon = 1/\text{poly}(n)$ based on the one-wayness of $\mathcal{F}_{\oplus_p,n,m=O(n \log n)}$. However, it falls short of providing LPRG (i.e., with standard security) or weak PPRG.

[7] This research have lead to non-trivial algorithms which allow to invert $\mathcal{F}_{Q,n,m=\Omega_d(n)}$ when the predicate is correlated with one or two of its inputs [10], however, these attacks do not generalize to other predicates.

in the output is highly challenging.[8] In contrast, it seems much easier to find a "reasonable" candidate one-way functions (i.e., one that resists all basic/known attacks). Moreover, it is not hard to come up with examples for local functions whose one-wayness may be plausible but they fail to be pseudorandom (e.g., if the hypergraph happen to have the same hyperedge twice, or if the predicate is unbalanced). The proof of our main theorems show that in this case, despite the existence of non-trivial regularities in the outputs, random local one-way functions achieve some form of pseudoentropy (i.e., weak unpredictability).

1.3.4 More on DSH

DSH is a natural generalization of the notoriously hard Densest k-Subgraph (DSG) problem (e.g., [19]) whose exact approximation ratio is an important open question. The best known algorithm achieves $O(n^{1/4})$-approximation [9], while known hardness results only rule out PTAS [32]. Naturally, DSH, which deals with hypergraphs, only seems harder. DSH has also a special role as a starting point for many other inapproximability results for problems like graph min-bisection, bipartite clique, and DSG itself [18, 32]. Recently, it was shown how to use the average-case hardness of DSH to plant a trapdoor in $\mathcal{F}_{Q,n,m}$, and obtain public-key encryption schemes [4]. This raises the exciting possibility that, for random local functions, it may be possible to go all the way from one-wayness to public-key cryptography: first assume that $\mathcal{F}_{Q,n,m}$ is one-way, then use Theorem 1.4 to argue that this collection is actually pseudorandom, then employ Theorem 1.6 to argue that DSH is hard over a planted distribution, and finally, use [4] to obtain a public-key cryptosystem. Unfortunately, the parameters given in Theorem 1.6 do not match the ones needed in [4]; still we consider the above approach as an interesting research direction.

2 Our Techniques

To illustrate some of our techniques, let us outline the proof of our main constructions starting with the case of polynomial stretch PRGs (Theorems 1.3 and 1.4).

2.1 Constructing Weak-PPRGs

Conceptually, we reduce pseudorandomness to one-wayness via the following approach: Suppose that we have an adversary which breaks the pseudorandomness properties of the function $f_{G,Q}(x)$ with respect to a random hypergraph G. Then we can collect information about x, and eventually invert the function, by invoking the adversary multiple times with respect to many *different* hypergraphs $G_1, \ldots, G_t$ which are all close variants of the original graph G. Details follow.

The basic procedure. Let us assume for now that the first input of Q has full influence. Also, assume that we have an adversary that breaks the δ-pseudorandomness of $\mathcal{F}_{Q,n,m}$. By Yao's theorem [40], this means that there exists an efficient algorithm $\mathbf{P}$ that *predicts* the next bit in the output of $\mathcal{F}_{Q,n,m}$ with probability $\frac{1}{2}+\varepsilon$ where $\varepsilon = \delta/m$. Let us assume for simplicity that $\mathbf{P}$ always attempts to predict the last-bit. Namely, the predictor $\mathbf{P}$ is given a random (n, m, d) hypergraph G and the $m-1$-bit long prefix of the string $y = f_{G,Q}(x)$, and it predicts with probability $\frac{1}{2}+\varepsilon$

[8] This even led to the belief that weak non-cryptographic forms of pseudorandomness, e.g., ε-bias, cannot be achieved [14], which was refuted in a non-trivial way by [36].

the last output $y_m = Q(x_S)$ which corresponds to the hyperedge $S = (i_1, \ldots, i_d)$. Given such a pair (G, y), let us replace the first entry i_1 of S with a random index $\ell \in [n]$ (hereafter referred to as "pivot"), and then invoke $\mathbf{P}$ on the modified pair. If the predictor succeeds and outputs $Q(x_{S'})$, then, by comparing this value to y_m, we get to learn whether the input bits x_ℓ and x_{i_1} are equal. Since the predictor may err, we can treat this piece of information as a noisy 2-LIN equation of the form $x_\ell \oplus x_{i_1} = b$ where $b \in \{0, 1\}$.

Collecting many 2-LIN equations. In order to recover x, we would like to collect many such equations and then solve a Max-2-LIN problem. To this end, we may start with a larger output length tm for some large t, i.e., attempt to invert the collection $\mathcal{F}_{Q,n,tm}$. Given a random hypergraph G with tm hyperedges, and an tm-bit string $y = f_{G,Q}(x)$, we partition the pair (G, y) to many blocks $(G^{(i)}, y^{(i)})$ of size m each, and then apply the above procedure to each block separately. This gives us a highly-noisy system of 2-LIN equations with a very large noise rate of $\frac{1}{2} - \varepsilon$ where $\varepsilon < 1/m < 1/n$ corresponds to the quality of prediction.

One may try to "purify" the noise by collecting many (say n^2/ε^2) equations, and correcting the RHS via majority vote, however, this approach is doomed to fail as the noise is not random, and can be chosen *arbitrarily* by the adversary in a way that depends on the equations. To see this, consider the trivial predictor which predicts well only when the output depends on x_1, and otherwise outputs a random guess. This predictor satisfies our condition (i.e., its prediction advantage is $1/n$) but it seems to be totally useless since it works only for equations which involve x_1. As a result, repetition will not decrease the noise.

Partial rerandomization. We fix this problem by randomizing the blocks $(G^{(i)}, y^{(i)})$. Specifically, we will permute the nodes of each $G^{(i)}$ under a random permutation $\pi^{(i)} : [n] \to [n]$, and invoke our basic procedure on the pairs $(\pi^{(i)}(G^{(i)}), y^{(j)})$. This is essentially equivalent to shuffling the coordinates of x. Furthermore, this transformation does not affect the distribution of the hypergraphs since hyperedges were chosen uniformly at random anyway. As a result, the noise (i.e., the event that $\mathbf{P}$ errs) becomes independent of the variables that participates in the equations, and the distribution of the prediction errors is "flattened" over all possible hyperedges. This transformation also yields a partial form of "random-self-reducibility": the input x is mapped to a random input of the same Hamming weight.

To show that the basic procedure succeeds well in each of the blocks, we would like to argue that the resulting pairs $(H^{(i)}, f_{H^{(i)}}(x^{(i)}))$ are uniformly and independently distributed, where $H^{(i)}$ is the permuted hypergraph $\pi^{(i)}(G^{(i)})$ and $x^{(i)}$ is the permuted string $\pi^{(i)}(x)$. This is not true as all the strings $\pi^{(i)}(x)$ share the same weight. Still we can show that this dependency does not decrease the success probability by too much. Moreover, we reduce the overhead of the reduction by applying the basic procedure with the *same* pivot ℓ. Again, the random permutation ensures that this does not affect the quality of the output too much. This optimization (and others) allow us to achieve a low overhead and take $t = O(1/\varepsilon^2)$. As a result, we derive Theorem 1.4 and obtain a PPRG with constant locality, some fixed polynomial stretch and polynomial distinguishing advantage. Standard amplification techniques now yield Theorem 1.3.

2.2 Constructing LPRGs

Let us move to the case of LPRGs (Theorem 1.2). We would like to use the "basic procedure" but our predicate is not necessarily sensitive. For concreteness, think of the majority predicate. In this case, when recovering a 2-LIN equation, we are facing two sources of noise: one due to the error of the prediction algorithm, and the other due to the possibility that the current assignment x_S is "stable" – flipping its i-location does not change the value of the predicate (e.g., in the case of majority, any assignment with less than $\lfloor d/2 \rfloor$ ones). Hence, this approach is useful only if the predictor's success probability is larger than the probability of getting a stable assignment. Otherwise, our predictor, which may act arbitrarily, may decide to predict well only when the assignments are stable, and make a random guess otherwise. Therefore, we can prove only $(1-\varepsilon)$-unpredictability for some constant $\varepsilon > 0$.[9] This seems problematic as the transformation from unpredictability to pseudorandomness (Yao's theorem) fail for this range of parameters.

The solution is to employ a different transformation. Recently, it was shown by [26] (HRV) how to convert unpredictablity into pseudorandomness via the use of randomness extractors. We further note that this transformation is local as long as the underlying extractor is locally computable. In general, one can show that randomness extractors with constant locality are deemed to have poor parameters (e.g., the seed length must be at least linear in the source length). Fortunately, it turns out that, for the special case of constant unpredictability and linear stretch, the HRV techniques can be applied with low-quality extractors for which there are (non-trivial) local implementations [36, 7]. This allows us to transform any $n + \Omega(n)$-long sequence with constant $(1-\varepsilon)$-unpredictability into an LPRG, while preserving constant locality.

Let us return to the first step in which prediction is used for inversion. In the LPRG setting we would like to base our construction on one-wayness with respect to $O(n)$ output-length (rather than super-linear length). Hence, the overhead of the reduction should be small, and we cannot apply the basic procedure to independent parts of the output as we did in the PPRG case. Our solution is to iterate the basic procedure n times where the hypergraph G, the m-bit string y, and the hyperedge S are all *fixed*, and in each iteration a different pivot $j \in [n]$ is being planted in S. We show that, whp, this allows to find a string x' which agrees with x on, say, 2/3 of the coordinates. At this point we employ the algorithm of [10] which recovers x given such an approximation x' and $f_{G,Q}(x)$.

2.3 Hardness of DSH

We move on to Theorem 1.6 in which we show that the pseudorandomness of $f_{G,Q}(x)$ for a random G implies strong inapproximability for the densest subhypergraph problem. Recall that one can amplify the inapproximability gap at the expense of increasing the cardinality of the hyperedges by taking graph product. In a nutshell, we show that the strong nature of pseudorandomness allows to apply some form of product amplification for "free" without changing the hypergraph.

Suppose that for a random d-uniform hypergraph G, the pair (G, y) is indistinguishable from the pair $(G, f_{G,Q}(x))$, where y is a random m-bit string and x is a random n-bit string. Assume,

[9] We show that the actual bound on ε depends on a new measure of "matching" sensitivity $\mu(Q)$ defined as follows: Look at the subgraph of the d-dimensional combinatorial hypercube whose nodes are the sensitive assignments of Q (i.e., the boundary and its neighbors), let M be a largest matching in the graph, and let $\mu(Q) = |M|/2^d$. For example, for majority with an odd arity d, it can be shown that all the assignments of Hamming weight $\lceil d/2 \rceil$ and $\lfloor d/2 \rfloor$ are in the matching and so the matching sensitivity is exactly $2\binom{d}{\lfloor d/2 \rfloor}/2^d$.

without loss of generality, that $Q(1^d) = 1$. (Otherwise use its complement.) We define an operator ρ that given a hypergraph G and an m-bit string z, deletes the i-th hyperedge if z_i is zero. It is not hard to see that $\rho(G, y)$ is a random sub-hypergraph with n nodes and about $m' = m/2$ hyperedges, so we expect hypergraphs on $n/2$ nodes to have $m'/2^d$ hyperedges. On the other hand, while the hypergraph $\rho(G, f_{G,Q}(x))$ is expected to have a similar number of hyperedges (otherwise, one can distinguish between the two distributions), the set of nodes $S = \{i : x_i = 1\}$ will not lose any hyperedge as $Q(1^d) = 1$. Hence, the subgraph induced by the set S is likely to have $n/2$ nodes and $m/2^d = m'/2^{d-1}$ hyperedges.

Overall, ρ maps (whp) a random instance to a No instance of $\frac{1}{2} - \mathsf{DSH}_d$ and a pseudorandom instance to a Yes-instance of $\frac{1}{2} - \mathsf{DSH}_d$. This leads to a basic hardness for $p = \frac{1}{2}$. Now, by a standard hybrid argument, one can show that the hypergraph – which is a public index – can be reused, and so it is hard to distinguish between the tuples

$$(G, y^{(1)}, \ldots, y^{(t)}) \qquad \text{and} \qquad (G, f_{G,Q}(x^{(1)}), \ldots, f_{G,Q}(x^{(t)})),$$

where the y's are random m-bit strings and the x's are random n-bit strings. Roughly speaking, each of these t copies allows us to re-apply the mapping ρ and further improve the parameter p by a factor of 2. (See full proof in Section 7.)

It is instructive to compare this to Feige's refutation assumption. The above distributions can be viewed as distributions over satisfiable and unsatisfiable CSPs where in both cases the hypergraph G is randomly chosen. In contrast, Feige's refutation assumption is weaker as it essentially asks for distinguishers that work well with respect to arbitrary (worst-case) distribution over the satisfiable instances. Hence the hypergraph cannot be reused and this form of amplification is prevented.

Organization. Some preliminaries are given in Section 3 including background on Goldreich's function and cryptographic definitions. Sections 4– 6 are devoted to the proofs of Theorems 1.2–1.4, where Sections 4 and 5 describe the reductions from inversion to prediction (for the PPRG setting and for the LPRG setting), and Section 6 completes the proofs based on additional generic transformations. Finally, in Section 7, we prove Theorem 1.6.

3 Preliminaries

Basic notation. We let $[n]$ denote the set $\{1, \ldots, n\}$ and $[i..j]$ denote the set $\{i, i+1, \ldots, j\}$ if $i \leq j$, and the empty set otherwise. For a string $x \in \{0,1\}^n$ we let $x^{\oplus i}$ denote the string x with its i-th bit flipped. We let x_i denote the i-th bit of x. For a set $S \subseteq [n]$ we let x_S denote the restriction of x to the indices in S. If S is an ordered set $(i_1, \ldots, i_d)$ then x_S is the *ordered* restriction of x, i.e., the string $x_{i_1} \ldots x_{i_d}$. The Hamming weight of x is defined by $\mathrm{wt}(x) = |\{i \in [n] | x_i = 1\}|$. The uniform distribution over n-bit strings is denoted by $\mathcal{U}_n$. For a distribution or random variable X (resp., set), we write $x \xleftarrow{R} X$ to denote the operation of sampling a random x according to X (resp., uniformly from X).

Hypergraphs. An (n, m, d) hypergraph is a hypergraph over n vertices $[n]$ with m hyperedges each of cardinality d. We assume that each hyperedge $S = (i_1, \ldots, i_d)$ is ordered, and that all the d members of an hyperedge are distinct. We also assume that the hyperedges are ordered from 1 to m. Hence, we can represent G by an ordered list $(S_1, \ldots, S_m)$ of d-sized (ordered) hyperedges. For indices $i \leq j \in [m]$ we let $G_{[i..j]}$ denote the subgraph of G which contains the hyperedges $(S_i, \ldots, S_j)$. We let $\mathcal{G}_{n,m,d}$ denote the distribution over (n, m, d) hypergraphs in which a hypergraph is chosen by picking each hyperedge uniformly and independently at random from all the possible $n \cdot (n-1) \cdot \ldots \cdot (n-d+1)$ ordered hyperedges.

Sensitivity and influence measures. Every d-ary predicate $Q : \{0,1\}^d \to \{0,1\}$ naturally partitions the d-dimensional Hamming cube into a zero-set V_0 and a one-set V_1 where $V_b = \{w \in \{0,1\}^d | Q(w) = b\}$. We let $H_Q = (V_0 \cup V_1, E)$ denote the bipartite graph induced by this partition, namely, $(u, v) \in V_0 \times V_1$ is an edge if the strings u and v differ in exactly one coordinate. A matching $M \subseteq V_0 \times V_1$ is a set of pair-wise *distinct* edges in H_Q, i.e., for every pair (u, v) and (u', v') in M we have $u \neq u'$ and $v \neq v'$. We will be interested in the probability that a randomly selected node lands inside a maximum matching:

$$\begin{aligned}\mathrm{Match}(Q) &= \max_M \Pr_{w \xleftarrow{R} \{0,1\}^d} [\exists u \text{ s.t. } (w, u) \in M \text{ or } (u, w) \in M] \\ &= \max_M 2|M|/2^d,\end{aligned}$$

where the maximum is taken over all matchings in H_Q. The *matching density* $\mathrm{Match}(Q)$ will be used to measure the "sensitivity" of Q. We also rely on more traditional measures of sensitivity as follows. The *influence* of the i-th coordinate of Q is defined by $\mathrm{Inf}_i(Q) = \Pr_{w \xleftarrow{R} \{0,1\}^d}[Q(w) \neq Q(w^{\oplus i})]$, and the maximum influence is defined by $\mathrm{Inf}_{\max}(Q) = \max_{i \in [d]} \mathrm{Inf}_i(Q)$. The following simple proposition relates the different sensitivity measures.

Proposition 3.1. *For any d-ary predicate Q we have:* $\mathrm{Inf}_{\max}(Q) \leq \mathrm{Match}(Q) \leq \sum_i \mathrm{Inf}_i(Q)$.

Proof. Consider the graph H_Q and color each edge (u, v) by the color $i \in [d]$ for which $u = v^{\oplus i}$. The inequalities follow by counting edges while noting that $\mathrm{Inf}_{\max}(Q)$ measures the cardinality of the largest monochromatic matching (in nodes), whereas $\sum_i \mathrm{Inf}_i(Q)$ measures the sum of degrees. $\square$

In [31] it is shown that if the predicate Q is balanced then $\mathrm{Inf}_{\max}(Q) \geq c \log d/d$ for some universal constant c.

3.1 Cryptographic definitions

Collection of functions. Let $s = s(n)$ and $m = m(n)$ be integer-valued functions. A collection of functions $\{F_k\}$ is formally defined via a mapping $F : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^m$ which takes an index $k \in \{0,1\}^s$ and an input point $x \in \{0,1\}^n$, and outputs the evaluation $F_k(x)$ of the point x under k-th function in the collection. We always assume that the collection is equipped with two efficient algorithms: an index-sampling algorithm K which given 1^n samples a index $k \in \{0,1\}^s$, and an evaluation algorithm which given $(1^n, k \in \{0,1\}^s, x \in \{0,1\}^n)$ outputs $F_k(x)$. We say that the collection is in $\mathbf{NC^0}$ if there exists a constant d (which does not grow with n) such that for every fixed k the function F_k has output locality of d. (In our case, k is typically the dependencies hypergraph G.) All the cryptographic primitives in this paper are modeled as collection of functions. We will always assume that the adversary that tries to break the primitive gets the collection index as a public parameter. Moreover, our constructions are all in the "public-coin" setting, and so they remain secure even if the adversary gets the coins used to sample the index of the collection.

Through this section, we let F denote a collection of functions, K denote the corresponding index-sampling algorithm, and $\varepsilon = \varepsilon(n) \in (0,1)$ be a parameter which measures security. All probabilities are taken over the explicit random variables and in addition over the internal coin tosses of the adversary algorithms.

One-way functions. Informally, a function is one-way if given a random image y it is hard to find a preimage x. We will also use a stronger variant of approximation-resilient one-wayness in which even the easier task of finding a string which approximates the preimage is infeasible. Formally, for a proximity parameter $\delta = \delta(n) \in (0, \frac{1}{2})$ and security parameter $\varepsilon = \varepsilon(n) \in (0,1)$, we say that a collection of functions $F : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^m$ is (ε, δ) *approximation-resilient one-way function* (AOWF) if for every efficient adversary $\mathcal{A}$ which outputs a list of candidates and all sufficiently large n's, we have that

$$\Pr_{\substack{k \xleftarrow{R} K(1^n) \\ x \xleftarrow{R} \mathcal{U}_n}} [\exists z \in \mathcal{A}(k, F_k(x)), z' \in F_k^{-1}(F_k(x)) \text{ s.t. } \Delta(z, z') \leq \delta(n)] < \varepsilon(n),$$

where $\Delta(\cdot,\cdot)$ denotes the relative Hamming distance. Note that the size of the list outputted by $\mathcal{A}$ is polynomially-bounded, as $\mathcal{A}$ is efficient. In the special case of $\delta = 0$, the collection F is referred to as ε *one-way*, or simply *one-way* if in addition ε is a negligible function. This is consistent with standard definitions of one-wayness (cf. [22]) as when $\delta = 0$, the algorithm can efficiently check which of the candidates (if any) is a preimage and output only a single candidate z rather than a list.

Indistinguishability. We say that a pair of distribution ensembles $Y = \{Y_n\}$ and $Z = \{Z_n\}$ is ε-indistinguishable if for every efficient adversary $\mathcal{A}$, the *distinguishing advantage*

$$|\Pr[\mathcal{A}(1^n, Y_n) = 1] - \Pr[\mathcal{A}(1^n, Z_n) = 1]|$$

is at most $\varepsilon(n)$. We say that the ensembles are ε statistically-close (or statistically-indistinguishable) if the above holds for computationally unbounded adversaries.

Pseudorandom and unpredictability generators. Let $m = m(n) > n$ be a length parameter. A collection of functions $F : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}^m$ is an ε-*pseudorandom generator* (PRG) if the ensemble $(K(1^n), F_{K(1^n)}(\mathcal{U}_n))$ is ε-indistinguishable from the ensemble $(K(1^n), \mathcal{U}_{m(n)})$. When ε is negligible, we refer to F as a *pseudorandom generator.* The collection F is ε *unpredictable generator* (UG) if for every efficient adversary $\mathcal{A}$ and every sequence of indices $\{i_n\}_{n \in \mathbb{N}}$, where $i_n \in [m(n)]$, we have that

$$\Pr_{k \xleftarrow{R} K(1^n), x \xleftarrow{R} \mathcal{U}_n} [\mathcal{A}(k, F_k(x)_{[1..i_n-1]}) = F_k(x)_{i_n}] < \varepsilon(n)$$

for all sufficiently large n's. We say that F is ε *last-bit unpredictable generator* (LUG) if the above is true for the sequence of indices $i_n = m(n)$. Observe that ε must be larger than $\frac{1}{2}$ as an adversary can always toss a random coin and guess the i_n-th bit with probability $\frac{1}{2}$. We refer to $m(n) - n$ as the *stretch* of the PRG (resp., UG), and classify it as *sublinear* if $m(n) - n = o(n)$, *linear* if $m(n) - n = \Omega(n)$ and *polynomial* if $m(n) - n > n^{1+\Omega(1)}$.

Remark 3.2 (Uniform unpredictability)**.** *Our notion of unpredictability is somewhat non-uniform as we quantify over all possible index sequences $\{i_n\}$. One may consider a uniform variant where the quantification is restricted to* efficiently samplable *sequences of indices. We prefer the non-uniform version as most of the time it will be easier to work with. However, it is not hard to show that the two definitions are essentially equivalent. Clearly, for any ε, non-uniform $(\frac{1}{2}+\varepsilon)$-unpredictability implies uniform $(\frac{1}{2}+\varepsilon)$-unpredictability. For the other direction, we claim that, for any ε and δ which are lower-bounded by some inverse polynomial, uniform $(\frac{1}{2}+\varepsilon)$-unpredictability implies non-uniform $(\frac{1}{2}+\varepsilon+\delta)$-unpredictability. To see this, consider an efficient adversary $\mathcal{A}$ that contradicts non-uniform $(\frac{1}{2}+\varepsilon+\delta)$-unpredictability, and let us construct an efficient algorithm $\mathcal{I}$ that samples a "good" sequence of indices $\{i_n\}$ for which $\mathcal{A}$ predicts the i_n-th bit of the sequence $F_{K(1^n)}(\mathcal{U}_n)$ with probability $\frac{1}{2}+\varepsilon$. The idea is to let $\mathcal{I}$ estimate, for each $i \in [m(n)]$, the success probability $p_i(n)$ of $\mathcal{A}$ in predicting the i-th bit of the sequence $F_{K(1^n)}(\mathcal{U}_n)$, and then output an index i_n which maximizes the estimated success probability. By standard Chernoff bounds, it is possible to efficiently estimate each $p_i(n)$ with an additive error of $\delta/2$ and confidence of $1-2^{-\Omega(n)}$. (Just test the empirical success probability of $\mathcal{A}$ on $O(n/\delta^2)$ random inputs). It follows that, for $i \overset{R}{\leftarrow} \mathcal{I}$, the predictor $\mathcal{A}$ predicts the i-th bit of $F_{K(1^n)}(\mathcal{U}_n)$ with probability $\frac{1}{2}+\varepsilon+\delta-\delta/2-2^{-\Omega(n)} > \frac{1}{2}+\varepsilon$, contradicting uniform $(\frac{1}{2}+\varepsilon)$-unpredictability.*

3.2 Goldreich's Function

For a predicate $Q : \{0,1\}^d \to \{0,1\}$ and an (n,m,d) hypergraph $G = ([n],(S_1,\ldots,S_m))$ we define the function $f_{G,Q} : \{0,1\}^n \to \{0,1\}^m$ as follows: Given an n-bit input x, the i-th output bit y_i is computed by applying Q to the restriction of x to the i-th hyperedge S_i, i.e., $y_i = Q(x_{S_i})$. For $m = m(n)$, the function collection $\mathcal{F}_{Q,n,m}$ is defined via the mapping

$$(G,x) \mapsto f_{G,Q}(x),$$

where (for each length parameter n) G is an $(n,m(n),d)$ hypergraph which serves as a public index, and x is an n-bit string which serves as an input. The index-sapmler selects the hypergraph G uniformly at random from $\mathcal{G}_{n,m,d}$. To simplify notation, we will often omit the input length and write $\mathcal{F}_{Q,m}$ under the convention that n always denote the input length of the collection.

In the following we show that, for the ensemble $\mathcal{F}_{Q,m}$, last-bit unpredictability and standard unpredictability are equivalent, and so are approximation-resilient one-wayness and standard one-wayness. These properties will be useful later.

Proposition 3.3. *For every constant locality $d \in \mathbb{N}$, predicate $Q : \{0,1\}^d \to \{0,1\}$, $m = \mathrm{poly}(n)$ and $\varepsilon = 1/\mathrm{poly}(n)$ the following holds: If $\mathcal{F}_{Q,m}$ is $(\frac{1}{2}+\varepsilon)$ last-bit unpredictable then $\mathcal{F}_{Q,m}$ is $(\frac{1}{2}+\varepsilon(1+1/n))$-unpredictable.*

Proof. The proof follows from the invariance of the distribution $\mathcal{G}_{n,m,d}$ under permutations of the order of the edges. Formally, assume towards a contradiction, that there exists a next-bit predictor $\mathbf{P}$ and a sequence of indices $\{i_n\}$ such that

$$\Pr_{\substack{x \overset{R}{\leftarrow} \mathcal{U}_n, G \overset{R}{\leftarrow} \mathcal{G}_{n,m,d} \\ y = f_{G,Q}(x)}} [\mathbf{P}(G, y_{[1..i_n-1]}) = y_{i_n}] \geq \frac{1}{2} + \varepsilon(n) + \varepsilon(n)/n,$$

for infinitely many n's. We construct a last-bit predictor $\mathbf{P}'$ with success probability of $\frac{1}{2}+\varepsilon$ as follows. First, use Remark 3.2 to efficiently find an index $j \in [m]$ such that, with probability $1-2^{-\Omega(n)}$ over the coins of $\mathbf{P}'$, it holds that $\Pr[\mathbf{P}(G, y_{1..j-1}) = y_j] > \frac{1}{2}+\varepsilon(n)+\frac{\varepsilon(n)}{2n}$ where the probability is taken over a random input and random coin tosses of $\mathbf{P}$. Now given an input $(G, y_{[1..m-1]})$, construct the hypergraph G' by swapping the j-th edge S_j of G with its last edge S_m. Then, $\mathbf{P}'$ invokes $\mathbf{P}$ on the input $(G', y_{[1..j-1]})$ and outputs the result. It is not hard to verify that this transformation maps the distribution $(G \xleftarrow{R} \mathcal{G}_{n,m,d}, f_{G,Q}(\mathcal{U}_n)_{[1..m-1]})$ to the distribution $(G \xleftarrow{R} \mathcal{G}_{n,m,d}, f_{G,Q}(\mathcal{U}_n)_{[1..j-1]})$, and so $\mathbf{P}'$ predicts the last bit with probability $\frac{1}{2}+\varepsilon(n)+\frac{\varepsilon(n)}{2n}-2^{-\Omega(n)} > \frac{1}{2}+\varepsilon$ as required. □

Proposition 3.4. *For every constant locality $d \in \mathbb{N}$, predicate $Q : \{0,1\}^d \to \{0,1\}$, and fixed proximity parameter $\delta \in (0, \frac{1}{2})$ (which may depend on d), there exists a constant $c = c(d, \delta)$, such that for every $m > cn$ and every inverse polynomial $\varepsilon = \varepsilon(n)$ if $\mathcal{F}_{Q,m}$ is ε one-way then $\mathcal{F}_{Q,m}$ is also (ε', δ) approximation-resilient one-way for $\varepsilon' = \varepsilon(1 + o(1))$.*

Proof. The proof is based on an algorithm of [10] which, given an approximation of x, inverts $f_{G,Q}(x)$. Formally, let $d \in \mathbb{N}$, $\delta \in (0, \frac{1}{2})$ be constants, and let Q be a d-ary predicate. In [10, Theorem 1.3] it is shown that there exists a constant $c = c(d, \delta)$ such that for every $m \geq cn$ and every inverse polynomial $p(\cdot)$, there exists an efficient algorithm $\mathcal{A} = \mathcal{A}_p$ that for a fraction of $1-p(n)$ of the hypergraph/input pairs $(G, x) \xleftarrow{R} \mathcal{G}_{n,m,d} \times \mathcal{U}_n$, and for every string x' which is δ-close to x, the output of $\mathcal{A}(G, y = f_{G,Q}(x), x')$ is a preimage of y under $f_{G,Q}$.

We can now prove the proposition. Suppose that $\mathcal{F}_{Q,m}$ is not (ε', δ) approximation-resilient one-way. That is, there exists an algorithm $\mathcal{B}$ which given $(G, y = f_{G,Q}(x))$ outputs a list of strings L such that with probability ε', over $G \xleftarrow{R} \mathcal{G}_{n,m,d}$ and $x \xleftarrow{R} \mathcal{U}_n$, the list L contains a string x' which δ-approximates x. We invert a pair (G, y) as follows: (1) First call $\mathcal{B}$ and generate a list L; (2) For each member $x' \in L$, invoke the algorithm $\mathcal{A}_{\varepsilon'/n}$ on the input (G, y, x'); (3) At the end, check whether one of the outputs is a preimage of y and output this result. By a union bound, the overall success probability is $\varepsilon' - \varepsilon'/n$ as required. □

4 Random Local Functions with $(\frac{1}{2} + 1/\mathrm{poly})$-Unpredictability

In Section 4.1 we will prove the following theorem:

Theorem 4.1 (one-way $\Rightarrow$ $(\frac{1}{2}+1/\mathrm{poly})$-unpredictable)**.** *Let $d \in \mathbb{N}$ be a constant and $Q : \{0,1\}^d \to \{0,1\}$ be a sensitive predicate. Then, for every $m \geq n$ and inverse polynomial ε, if $\mathcal{F}_{Q,m/\varepsilon^2}$ is ε one-way then $\mathcal{F}_{Q,m}$ is $(\frac{1}{2}+c\varepsilon)$-UG, for some constant $c = c(d) > 0$.*

In Section 4.3 we will show that the above theorem generalizes to variants of $\mathcal{F}_{Q,m}$ that capture some of the existing heuristic candidates.

4.1 Proof of Theorem 4.1

Let Q be some sensitive d-ary predicate, $m > n$ be polynomial, and ε be inverse polynomial. Assume, towards a contradiction, that $\mathcal{F}_{Q,m}$ is not $(\frac{1}{2}+c\varepsilon)$-UG for some universal constant c whose value will be determined later. By Proposition 3.3, this means that there exists an efficient algorithm $\mathbf{P}$ which predicts the last-bit of $\mathcal{F}_{Q,m}$ with probability $\frac{1}{2} + \delta$ where $\delta = c\varepsilon(1 - o(1))$. We will use $\mathbf{P}$ to construct an efficient algorithm Invert which approximately inverts $\mathcal{F}_{Q,m'}$ with probability $\delta/8$ and proximity $1/3$ for every $m' \geq 16m/\delta^2$. By Proposition 3.4, this implies that $\mathcal{F}_{Q,c'16m/\delta^2}$ is not $\delta/9$ one-way, where $c' = c'(d, 1/3)$ is a sufficiently large constant. Letting c be a constant larger than $\max(4c', 9)$, we derive a contradiction to the ε one-wayness of $\mathcal{F}_{Q,m/\varepsilon^2}$.

It is therefore left to describe how to transform the last-bit predictor $\mathbf{P}$ into an algorithm Invert. Syntactically, $\mathbf{P}$ takes as an input an $(m-1, n, d)$ hypergraph G, an $(m-1)$-bit string y (supposedly $y = f_{G,Q}(x)$), and an hyperedge S, and outputs its guess for $Q(x_S)$. Before we describe our approximate inverter, we will need some notation. For a permutation $\pi : [n] \to [n]$ and a tuple $S = (i_1, \ldots, i_d) \subseteq [n]^d$ we let $\pi(S)$ denote the tuple $(\pi(i_1), \ldots, \pi(i_d))$. For an (m, n, d) hypergraph $G = (S_1, \ldots, S_m)$ we let $\pi(G)$ denote the (m, n, d) hypergraph $(\pi(S_1), \ldots, \pi(S_m))$. For a string $x \in \{0,1\}^n$, the string $\pi(x)$ is the string whose coordinates are permuted under π. Also, let us assume for simplicity, and without loss of generality, that the first variable of the predicate Q has influence 1, i.e., $\mathrm{Inf}_1(Q) = 1$.

Our inversion algorithm will make use of the following sub-routine Vote (Figure 1), which essentially corresponds to the "basic procedure" described in Section 2. The subroutine takes a "small" (n, m, d) hypergraph G, a corresponding output string $y = f_{G,Q}(x)$, and uses the predictor $\mathbf{P}$ to approximate the value $x_i \oplus x_\ell$ where the indices i and ℓ are given as additional inputs. (The index ℓ is referred to as "global advice" as it will be reused among different iterations). The algorithm Invert (Figure 2) uses Vote to approximately invert a random member of $\mathcal{F}_{Q,tm}$.

- **Input**: an (n, m, d) hypergraph G, a string $y \in \{0,1\}^m$, an index $i \in [n]$.
- **Global advice**: index $\ell \in [n]$.

1. Choose a random hyperedge $S = (s_1, \ldots, s_d)$ from G subject to the constraint $s_1 = i$ and $\ell \notin \{s_2, \ldots, s_d\}$. Let s denote the index of S in G, i.e., $S = G_s$. If no such edge exists abort with a failure symbol.
2. Choose a random permutation $\pi : [n] \to [n]$. Let $G' = \pi(G_{-s})$ be the hypergraph obtained by removing S and permuting the nodes under π. Similarly, let $y' = y_{-s}$ be the string y with its s-th bit removed. Finally, define the hyperedge
$$S' = \pi(S_{1 \leftarrow \ell}) = (\pi(\ell), \pi(s_2), \ldots, \pi(s_d)).$$
3. Output $\mathbf{P}(G', y', S') \oplus y_s$.

Figure 1: Algorithm Vote.

We will prove that Invert approximately inverts $\mathcal{F}_{Q,tm}$ as follows.

Lemma 4.2. *Assume that $\mathbf{P}$ is a $\frac{1}{2} + \delta$ last-bit predictor for the collection $\mathcal{F}_{Q,m}$ and let t be an arbitrary parameter. Then,*

$$\Pr_{G \overset{R}{\leftarrow} \mathcal{G}_{n,tm,d}, x \overset{R}{\leftarrow} \mathcal{U}_n} [\mathsf{Invert}(G, y = f_{G,Q}(x)) \text{ is } 2\exp(-t\delta^2/8)\text{-close to } x] \geq \delta/8.$$

The theorem follows from the lemma by taking $t = 16/\delta^2$.

- **Input**: an (n, tm, d) hypergraph G and a string $y \in \{0,1\}^{tm}$, where t is a parmeter.

1. Partition the input (G, y) to t blocks of length m where $y^{(j)} = y_{[(j-1)m+1..jm]}$ and $G^{(j)} = G_{[(j-1)m+1..jm]}$.
2. Choose a random pivot $\ell \overset{R}{\leftarrow} [n]$, and a random bit b (our guess for x_ℓ).
3. For each $i \in [n]$ we recover the i-th bit of x as follows:
   (a) For each $j \in [t]$, invoke the subroutine Vote on the input $(G^{(j)}, y^{(j)}, i)$ with global advice ℓ, and record the output as $v_{i,j}$.
   (b) Set v_i to be the majority vote of all $v_{i,j}$'s.
4. If $b = 0$ output v; otherwise output the complement $\mathbf{1} - v$.

Figure 2: Algorithm Invert.

Proof outline of the lemma. From now on fix a sufficiently large input length n for which $\mathbf{P}$ is successful. Let us focus on the way our algorithm recovers one fixed variable $i \in [n]$. First we will show that in each call to the subroutine Vote, whenever the predictor predicts correctly, we get a "good" vote regarding whether x_i and x_ℓ agree. Hence, if our global guess b for x_ℓ is correct, and most of the predictions (in the i-th iteration of the outer loop) are good, we successfully recover x_i. In order to show that the predictor succeeds well, we should analyze the distribution on which it is invoked. In particular, we should make sure that the marginal distribution of each query to $\mathbf{P}$ is roughly uniform, and, that the dependencies between the queries (during the i-th iteration of the outer loop) are minor. This is a bit subtle, as there are some dependencies due to the common input x and common pivot ℓ. To cope with this, we will show (in Claim 4.3) that these queries can be viewed as independent samples, alas taken from a "modified" distribution which is different from the uniform. Later we will show that, whp, $\mathbf{P}$ predicts well over this non-uniform distribution.

The modified distribution. Let X_k denote the set of all n-bit strings whose Hamming weight is exactly k. For $k \in [n]$ and a bit $\sigma \in \{0,1\}$ define the distribution $D_{k,\sigma}$ over tuples (G, r, y, T) as follows: the hypergraph G is sampled from $\mathcal{G}_{n,m-1,d}$, the string r is uniformly chosen from X_k, the string y equals to $f_{Q,G}(r)$, and the hyperedge $T = (T_1, \ldots, T_d)$ is chosen uniformly at random subject to $r_{T_1} = \sigma$. In Section 4.2, we prove the following claim:

Claim 4.3. *Let (G, y, ℓ, i) be the input to* Vote *where $G \overset{R}{\leftarrow} \mathcal{G}_{n,m,d}$, the indices $\ell \in [n]$ and $i \in [n]$ are arbitrarily* fixed *and $y = f_{Q,G}(x)$ for an arbitrary* fixed *$x \in \{0,1\}^n$. Consider the random process* Vote(G, y, ℓ, i) *induced by the internal randomness and the distribution of G. Then, the following hold:*

1. *The process fails with probability at most $1/2$.*
2. *Conditioned on not failing, the random variable (G', x', y', S') is distributed according to $D_{\mathrm{wt}(x),x_\ell}$, where $x' = \pi(x)$ and $\mathrm{wt}(x)$ is the Hamming weight of x.*

3. *Conditioned on not failing, if the outcome of the predictor* $\mathbf{P}(G', y', S')$ *equals to* $Q(x'_{S'})$ *then the output of* Vote *is* $x_i \oplus x_\ell$ *(with probability 1).*

We can now prove the lemma.

Proof of Lemma 4.2. Assume that $\mathbf{P}$ predicts the last-bit of $\mathcal{F}_{Q,m}$ with probability $\frac{1}{2} + \delta$. First, we will show that with good probability over $x \in \{0,1\}^n$ and the pivot $\ell \in [n]$, the predictor $\mathbf{P}$ predicts well on the distribution $D_{\mathrm{wt}(x),x_\ell}$. Formally, we say that a pair (x, ℓ) is *good* if

$$\Pr_{(G,r,y,T) \overset{R}{\leftarrow} D_{\mathrm{wt}(x),x_\ell}} [\mathbf{P}(G, y, T) = Q(r_T)] > \frac{1}{2} + \delta/2.$$

We claim that a random pair $(x \overset{R}{\leftarrow} \mathcal{U}_n, \ell \overset{R}{\leftarrow} [n])$ is likely to be good with probability $\delta/2$. By assumption, $\Pr[\mathbf{P}(G, y, T) = Q(r_T)] > \frac{1}{2} + \delta$ where G is sampled from $\mathcal{G}_{n,m-1,d}$, $r \overset{R}{\leftarrow} \mathcal{U}_n$ and $T \overset{R}{\leftarrow} \binom{[n]}{d}$. Equivalently, we can choose the pair (r, T) as follows: (1) Sample $x \overset{R}{\leftarrow} \mathcal{U}_n$ and let $r \overset{R}{\leftarrow} X_{\mathrm{wt}(x)}$; (2) Choose a random pivot $\ell \overset{R}{\leftarrow} [n]$ and let T be uniformly chosen from $\binom{[n]}{d}$ subject to $r_{T_1} = r_\ell$. (For every fixed r, the random variable T_1 will simply be uniform over $[n]$.) Hence, our assumption can be written as

$$\Pr_{x \overset{R}{\leftarrow} \mathcal{U}_n, \ell \overset{R}{\leftarrow} [n], (G,r,y,T) \overset{R}{\leftarrow} D_{\mathrm{wt}(x),x_\ell}} [\mathbf{P}(G, y, T) = Q(r_T)] > \frac{1}{2} + \delta,$$

and so, by Markov's inequality, a random pair (x, ℓ) is good with probability $\delta/2$.

Next, let us analyze the Hamming distance between the output of $\mathsf{Invert}(G, f_{G,Q}(x))$ and x. In the following, we condition on the event that x and ℓ are good and that our guess b for x_ℓ is successful. We will show that for every such good triple (x, ℓ, b) and for every fixed index $i \in [n]$,

$$\Pr[\text{the } i\text{-th bit of } \mathsf{Invert}(G, f_{G,Q}(x)) \text{ disagrees with } x_i] \leq \exp(-t\delta^2/8), \tag{1}$$

where the probability is taken over $G \overset{R}{\leftarrow} \mathcal{G}_{n,tm,d}$ and the internal randomness of Vote. By the linearity of expectation, this means that, conditioned on (x, ℓ, b) being good, $\mathsf{Invert}(G, f_{G,Q}(x))$ is expected to be at most $\exp(-t\delta^2/8)$-far from x, and so, by Markov's inequality, the distance is upper-bounded by $2\exp(-t\delta^2/8)$ with probability $\frac{1}{2}$. Since $\delta/4$ of the triples (x, ℓ, b) are good, the lemma follows.

Fix some good triple (x, ℓ, b) and some index $i \in [n]$. We will now prove Eq. 1. Claim 4.3 guarantees that each call to the subroutine Vote fails independently with probability at most $\frac{1}{2}$. Furthermore, conditioned on not failing, in each call to Vote the predictor $\mathbf{P}$ gets an *independent* sample from $D_{\mathrm{wt}(x),x_\ell}$. Hence, the subroutine returns a correct vote, i.e., $v_{i,j} = x_i \oplus x_\ell$, with probability $\frac{1}{2} + \delta/2$. The rest of the analysis follows from standard concentration bounds.

Formally, define a random variable $\alpha_j \in [-1, 1]$ which takes the value 1 if the vote $v_{i,j}$ is good i.e., $v_{i,j} = x_i \oplus x_\ell$, takes the value -1 if the vote is incorrect, and takes the value 0 if the subroutine Vote fails. The α_j's are independent and the expectation of each one of them is at least $\frac{1}{2}(\frac{1}{2} + \delta/2) - \frac{1}{2}(\frac{1}{2} - \delta/2) = \delta/2$. Therefore, by Hoeffding's bound, the probability that their sum is negative is at most $\exp(-2t(\delta/2)^2/4) = \exp(-t\delta^2/8)$, as required. $\square$

4.2 Proof of Claim 4.3

First item. We upper-bound the probability of failure. First, the probability that G has no hyperedge whose first entry equals to i is $(1-1/n)^m < (1-1/n)^n < 2/5$. Conditioned on having an hyperedge whose first entry is i, the probability of having ℓ as one of its other entries is at most $O(d/n)$. Hence, by a union bound, the failure probability is at most $2/5 + O(d/n) < 1/2$.

Second item. Fix x and let k be its Hamming weight. Let x_+ be the support of x, i.e., set of indices j such that $x_j = 1$. Consider the distribution of the pair (G, S) defined in Step 1 of Vote. This pair can be sampled independently as follows: first choose a random hyperedge S whose first entry is i and ℓ does not appear in its other entries, then construct G by choosing a random hypergraph R from $\mathcal{G}_{n,m-1,d}$ and by planting S in a random location at R. From this view, it follows that the pair $(G_{-s}, S_{1\leftarrow\ell})$ (defined in Step 2) is independently distributed such that $G_{-s} \overset{R}{\leftarrow} \mathcal{G}_{n,m-1,d}$ and $S_{1\leftarrow\ell}$ is a random hyperedge whose first entry is ℓ. Since x is fixed and $y' = y_{-s} = f_{Q,G_{-s}}(x)$, we have now a full understanding of the distribution of the tuple $(G_{-s}, x, y', S_{1\leftarrow\ell})$.

We will now analyze the effect of the permutation π. Let $x' = \pi(x)$ and $G' = \pi(G_{-s})$. First, observe that for every fixed permutation π the tuple (G', x', y') satisfies $y' = f_{Q,G'}(x')$ since $y' = f_{Q,G_{-s}}(x)$. Moreover, since G_{-s} is taken from $\mathcal{G}_{n,m-1,d}$, so is $G' = \pi(G_{-s})$ even when π is fixed. Let us now pretend that the random permutation π is selected in two steps. First, choose a random set $A \subseteq [n]$ of size k and then, in the second step, choose a random permutation π_A subject to the constraint that $\pi(x_+) = A$.

Consider the distribution of x' which is induced by the random choice of A, i.e., before the *second* step was performed. Already in this phase we have that x' is uniformly and independently distributed according to X_k. Hence, $(G' \overset{R}{\leftarrow} \mathcal{G}_{n,m-1,d}, x' \overset{R}{\leftarrow} X_k, y' = f_{Q,G'}(x'))$.

Let us now fix both G' and A (and therefore also x') and so the only randomness left is due to the choice of π_A. We argue that the hyperedge $S' = \pi_A(S_{1\leftarrow\ell})$ is uniformly and independently distributed under the constraint that the first entry τ of S' satisfies $x'_\tau = x_\ell$. To see this, recall that the first entry of $S_{1\leftarrow\ell}$ is ℓ, and so the entry $S'_1 = \pi_A(\ell)$ is mapped to a random location in the set $\left\{j : x'_j = x_\ell\right\}$, also recall that the last $d-1$ entries of $S_{1\leftarrow\ell}$ are random indices (different than ℓ) and so for every fixing of π_A the $d-1$ last entries of S' are still random. This completes the proof as we showed that the tuple (G', x', y', S') is distributed properly.

Third item. Suppose that $\mathbf{P}$ outputs the bit $Q(x'_{S'})$. Then, since $S' = \pi(S_{1\leftarrow\ell})$ and $x' = \pi(x)$, the result equals to $Q(x_{S_{1\leftarrow\ell}})$, which, by definition, can be written as $Q(x_S) \oplus (x_\ell \oplus x_i)$. Hence, when this bit is XOR-ed with $Q(x_S)$, we get $x_\ell \oplus x_i$, as required. $\square$

4.3 Generalization to The Noisy Case

Let Q be a sensitive predicate, and let E be a "noise" distribution over m bits. Consider the collection $\mathcal{F}'_{Q,m,E}$ which is indexed by a random (n, m, d) hypergraph G, and given x it outputs $(G, f_{G,Q}(x) \oplus E)$. One-wayness is defined with respect to x, that is, we say that $\mathcal{F}_{Q,m,E}$ is ε one-way if it is hard to recover x with probability ε. We say that E is *symmetric* if it is invariant under permutations: for every $\pi : [m] \to [m]$ the random variable $\pi(E)$ is identically distributed as E. For an integer t, we let E^t denote the distribution over $\{0,1\}^{tm}$ obtained by concatenating t independent copies of E. Theorem 4.1 can be generalized to this setting as follows.

Theorem 4.4 (Theorem 4.1: generalization)**.** *Let $d \in \mathbb{N}$ be a constant locality parameter, $Q : \{0,1\}^d \to \{0,1\}$ be a sensitive predicate, $m = m(n) \geq n$ be a length parameter, and E be a symmetric distribution over $\{0,1\}^m$. Then, for every inverse polynomial ε, if $\mathcal{F}'_{Q,tm,E^t}$ is ε-one-way then $\mathcal{F}'_{Q,m,E}$ is $(\frac{1}{2} + 9\varepsilon)$-UG, where $t = O(\log n/\varepsilon^2)$.*

Proof sketch. The proof is the essentially the same as the proof of Theorem 4.1. Assume, towards a contradiction, that $\mathcal{F}'_{Q,m,E}$ is not $(\frac{1}{2} + 9\varepsilon)$-UG. Due to the symmetry of E we can generalize Proposition 3.3 to the noisy setting and conclude that there exists an efficient algorithm $\mathbf{P}$ which predicts the last-bit of $\mathcal{F}'_{Q,m,E}$ with probability $\frac{1}{2} + \delta$ where $\delta = 8\varepsilon$. We will invert a random output of $\mathcal{F}'_{Q,tm,E^t}$ using the algorithm Invert (instantiated with $\mathbf{P}$). Since the noise is symmetric and independent of the input x and the graph G, the analysis of Invert does not change, and Lemma 4.2 remains true. The only difference (compared to the non-noisy case) is that we do not know whether the reduction from approximation-resilient one-wayness to one-wayness of [10] holds. Therefore we employ the algorithm Invert with $t = 8\ln(4n)/\delta^2$ overhead, which, ensures (by Lemma 4.2) that, with probability $\varepsilon = \delta/8$, the relative Hamming distance between the output of Invert and the preimage x is at most $1/2n$. Since x has only n coordinates, we obtain an exact inverter which succeeds with probability ε, in contrast to our assumption. □

This generalization captures the case of noisy-local-parity construction ([2, 7, 4]) where Q is linear (i.e., "exclusive-or") and E is a "Bernoulli noise", i.e., each bit of E takes the value one independently with probability $p < \frac{1}{2}$. It also captures a variant of the MST construction [36], where Q is linear and E is some locally samplable noise distribution. Hence, in both cases weak pseudorandomness follows from one-wayness.

5 Random Local Functions are $(1 - \Omega(1))$-Unpredictable

We move on to the case of general predicates and prove the following theorem:

Theorem 5.1 (**one-way** $\Rightarrow$ $(1 - \Omega(1))$**-UG**)**.** *For every constant $d \in \mathbb{N}$, predicate $Q : \{0,1\}^d \to \{0,1\}$, and constant $\varepsilon \in (0, \mathrm{Match}(Q)/2)$, there exists a constant $c > 0$ such that for every polynomial $m > cn$ the following holds. If the collection $\mathcal{F}_{Q,m}$ is $\varepsilon/5$-one-way then it is also ε'-UG where $\varepsilon' = 1 - \mathrm{Match}(G)/2 + \varepsilon$.*

Recall that for a non-fixed predicate $\mathrm{Match}(Q) > 2^{-d}$, for a balanced predicate $\mathrm{Match}(Q) > \Omega(\log d/d)$, and for a sensitive predicate $\mathrm{Match}(Q) = 1$. The proof of the theorem is given in Section 5.1.

5.1 Proof of Theorem 5.1

Let Q be some d-ary predicate, and let $\mu = \mathrm{Match}(Q)$ denote its matching sensitivity. Let $\varepsilon \in (0, \mu/2)$ be a constant, $c = c(d, \varepsilon)$ be some constant whose value will be determined later, and $m > cn$ be some polynomial. Assume, towards a contradiction, that $\mathcal{F}_{Q,m}$ is not $(1 - \mu/2 + \varepsilon)$ UG. By Proposition 3.3, this means that there exists an efficient algorithm $\mathbf{P}$ which predicts the *last*-bit of $\mathcal{F}_{Q,m}$ with probability $(1 - \mu/2 + \delta)$ for $\delta = \varepsilon(1 - o(1))$. We will use $\mathbf{P}$ to construct an efficient algorithm $\mathcal{A}$ which approximately inverts the collection $\mathcal{F}_{Q,m}$ with probability $\delta/4$ and proximity $\frac{1}{2} - \delta/6$. The theorem will then follow as, by Proposition 3.4, such an approximate inverter allows to exactly invert $\mathcal{F}_{Q,m}$ with probability $\frac{\delta}{4} - o(1) > \frac{\varepsilon}{5}$, for sufficiently large constant $c = c(d, \delta) = c(d, \varepsilon)$.

Our goal now is to approximately invert the collection $\mathcal{F}_{Q,m}$. To this aim, we rely on the subroutine Approx, depicted in Figure 3. Below, we follow the convention of the previous section and view the input of the predictor $\mathbf{P}$ as being composed of an $(m-1, n, d)$ hypergraph G, an $(m-1)$-bit string y (supposedly $y = f_{G,Q}(x)$), and an hyperedge S. Under this convention, $\mathbf{P}$ outputs its guess for $Q(x_S)$.

- **Input**: (n, m, d) hypergraph G and string $y \in \{0,1\}^m$.
- **Randomness**: Choose uniformly at random a set $S = (s_1, \ldots, s_d)$, and an index $\ell \in [d]$, as well as random coins r for $\mathbf{P}$.

1. For every $i \in [n]$: Let $\hat{x}_i = \mathbf{P}(G, y, S_{\ell \leftarrow i}; r)$, where $S_{\ell \leftarrow i}$ is the set obtained by replacing the ℓ-th entry in S with the index i, and $\mathbf{P}$ is always invoked with the same fixed sequence of coins r.
2. Output the candidate $\hat{x}$ and its complement.

Figure 3: Algorithm Approx.

Analyzing Approx. For Approx to succeed, we intuitively need the following two conditions. (1) Sensitivity: flipping the ℓ-th entry of x_S should change the value of the predicate Q; and (2) Correctness: The predictor should predict well over many of the i's. We will prove that conditions of this spirit indeed guarantee success, and then argue that the conditions hold with good enough probability (taken over a random input and the random coins of the algorithm).

We begin by formalizing these conditions. We say that the tuple (x, G, r, S, ℓ) is *good* if the following two conditions hold

$$Q(x_S) \quad \neq \quad Q(x_S^{\oplus \ell}) \tag{2}$$

where $z^{\oplus i}$ denotes the string z with its i-th bit flipped, and, in addition, for at least $(\frac{1}{2} + \delta/6)$ fraction of the $i \in [n]$

$$\mathbf{P}(G, f_{G,Q}(x), S_{\ell \leftarrow i}; r) = Q(x_{S_{\ell \leftarrow i}}). \tag{3}$$

It is not hard to see that a good tuple leads to a good approximation of x.

Lemma 5.2. *If the tuple (x, G, r, S, ℓ) is good then either $\hat{x}$ or its complement agrees with x for a fraction of $(\frac{1}{2} + \delta/6)$ of the indices.*

Proof. Let s_ℓ be the ℓ-th entry of S. Then, by Eq. 2, we can write

$$Q(x_{S_{\ell \leftarrow i}}) = Q(x_S) \oplus x_{s_\ell} \oplus x_i.$$

Hence, for every $i \in [n]$ for which Eq. 3 holds we have that

$$\hat{x}_i = \mathbf{P}(G, y, S_{\ell \leftarrow i}; r) = Q(x_{S_{\ell \leftarrow i}}) = Q(x_S) \oplus x_{s_\ell} \oplus x_i = b \oplus x_i,$$

where $b = Q(x_S) \oplus x_{s_\ell}$. Hence, if $b = 0$ the output $\hat{x}$ agrees with x on a fraction of $(\frac{1}{2} + \delta/6)$ of its coordinates, and otherwise, the complement $\mathbf{1} - \hat{x}$ has such an agreement. □

In the next section, we will prove that for many of the triples (x, G, r), a randomly chosen (S, ℓ) forms a good tuple with probability $\Omega(\delta\mu/d)$.

Lemma 5.3. *For at least $\delta - \text{neg}(n)$ fraction of the pairs (x, G), we have that*

$$\Pr_{S,\ell,r}[(x, G, r, S, \ell) \text{ is good}] > \Omega(\delta\mu/d)). \tag{4}$$

We can now complete the proof of Theorem 5.1. Given an (n, m, d) hypergraph G and an m-bit string $y = f_{G,Q}(x)$, the approximate inverter $\mathcal{A}$ calls the subroutine $\mathsf{Approx}(G, y)$ for $t = O(\frac{\log(1/\delta)d}{\delta\mu})$ times (each time with a fresh choice of random coins), and outputs all the $2t$ candidates. We claim that, with probability $\delta/4$, the list contains a good candidate whose agreement with x is $(\frac{1}{2} + \delta/6)n$. Indeed, let us condition on the event that the pair (G, x) satisfies Eq. 4, which, by Lemma 5.3, happens with probability at least $\delta/2$. In this case, by Lemmas 5.2 and 5.3, in each iteration we will output, with probability $\Omega(\delta\mu/d)$, a good candidate whose agreement with x is $(\frac{1}{2} + \delta/6)n$. Since the success probability of each iteration is independent of the others, at least one iteration succeeds with probability $1 - \delta/4$, and so, by a union bound, the overall success probability is $\delta/2 - \delta/4 = \delta/4$. $\square$

5.2 Proof of Lemma 5.3

Call x *balanced* if $\text{wt}(x) \in (n/2 \pm n^{2/3})$. We call a triple (x, G, r) *good* if x is balanced and

$$\Pr_S[\mathbf{P}(G, f_{G,Q}(x), S; r) = Q(x_S)] > 1 - \mu/2 + \delta/2. \tag{5}$$

Claim 5.4. *A random triple (x, G, r) is good with probability $\delta - \text{neg}(n)$.*

Proof. By our assumption on $\mathbf{P}$ we have that

$$\Pr_{G,S,x,r}[\mathbf{P}(G, f_{G,Q}(x), S; r) = Q(x_S)] > 1 - \mu/2 + \delta.$$

Hence, by Markov's inequality and the fact that $\delta < \mu$,

$$\Pr_{G,x,r}[(x, G) \text{ satisfy Eq. 5}] > \delta/(\mu - \delta) > \delta.$$

By a Chernoff bound, a random x is balanced with probability $1 - \text{neg}(n)$, and so we can write

$$\Pr_{G,x,r}[(x, G) \text{ satisfy Eq. 5} | x \text{ is balanced}] > \delta - \text{neg}(n),$$

and the claim follows. $\square$

Fix a good triple (x, G, r). We call S *predictable* if $\mathbf{P}(G, f_{G,Q}(x), S; r) = Q(x_S)$. The pair (S, ℓ) is *neighborhood predictable* (in short NP) if

$$\Pr_{i \in [n]}[S_{\ell \leftarrow i} \text{ is predictable}] > \frac{1}{2} + \delta/6 \tag{6}$$

The pair (S, ℓ) is *sensitive* if $Q(x_S) \neq Q(x_S^{\oplus \ell})$. To prove Lemma 5.3 it suffices to show that

Lemma 5.5. *A fraction of at least $\frac{\delta\mu}{3d}\cdot(1-o(1))$ of the pairs (S,ℓ) are simultaneously sensitive and neighborhood predictable.*

Proof. We will need some definitions. For a set S let $x_S \in \{0,1\}^d$ be the "label" of the set. Fix some maximal matching M of the predicate Q whose cardinality is $\mu 2^d$. We restrict our attention to sets S for which x_S appears in some edge in M. (Abusing notation, we write $x_S \in M$.) For such S, we define the *index* $\ell(S)$ to be the single integer $\ell \in [d]$ for which the pair $(x_S, x_S^{\oplus\ell})$ is an edge in M. (Since M is a matching, S will have exactly one index.) Observe that, by definition, the pair $(S,\ell(S))$ is always sensitive. Hence, to prove the lemma, it suffices to show that the quantity

$$\Pr_{S,\ell}[x_S \in M \wedge \ell=\ell(S)\wedge (S,\ell(S)) \text{ is NP}] = \Pr_S[x_S\in M]\cdot \Pr_{\ell \xleftarrow{R}[d]}[\ell=\ell(S)]\cdot \Pr_S[(S,\ell(S)) \text{ is NP}|x_S\in M]$$

is lower bounded by $\frac{\delta\mu}{3d}\cdot(1-o(1))$. Since $\Pr_{\ell \xleftarrow{R}[d]}[\ell=\ell(S)]=1/d$ it suffices to show that

$$\Pr_S[x_S\in M] \;>\; \mu(1-o(1)) \tag{7}$$
$$\Pr_S[(S,\ell(S)) \text{ is NP}|x_S\in M] \;>\; \delta/3. \tag{8}$$

We begin with Eq. 7. Since x is balanced, the label x_S of a random set S is almost uniform over the set of d-bit strings, and therefore it hits M with probability close to its density μ. Formally, for a label $z\in\{0,1\}^d$, let p_z denote the probability that a random set S is labeled by z. Note that p_z depends only in the Hamming weight of z (and x). In particular, since x is balanced and d is constant, we have

$$p_z \in \left[\left(\frac{n/2-n^{2/3}-d}{n}\right)^d, \left(\frac{n/2+n^{2/3}}{n-d}\right)^d\right] = [2^{-d}-o(1), 2^{-d}+o(1)],$$

and so Eq. 7 is derived via

$$\Pr_S[x_S\in M] = \sum_{z\in M} p_z = \left(\mu 2^d\cdot 2^{-d}(1\pm o(1))\right) = \mu(1\pm o(1)).$$

From now on, we focus on proving Eq. 8. Define a directed graph H over d-size sets S for which $x_S\in M$, and for each node S and $i\in[n]$ add an edge $(S, S_{\ell(S)\leftarrow i})$. In these terms, Eq. 8 asserts that at least $\delta/3$ of the nodes S in H, have many $(\frac{1}{2}+\delta/6)$ predictable out-neighbors. By Markov's inequality, (8) follows from the following equation

$$\Pr_{S,i\xleftarrow{R}[n]}[S_{\ell(S)\leftarrow i} \text{ is predictable}|x_S\in M] > \frac{1}{2}+\delta/3. \tag{9}$$

We prove (9) in two steps: First (Claim 5.6), we argue that the graph H is symmetric and regular. Therefore, the set T, obtained by choosing a random starting point S and taking a single random step in H, is uniformly distributed over H; Second (Claim 5.7), we show that $\frac{1}{2}+\delta/3$ of the nodes in H are predictable. By combining these facts together we derive Eq.9.

Claim 5.6. *The graph H is symmetric and each node has exactly n distinct neighbors including one self loop.*

Proof. We show that the graph is symmetric. Fix an edge $(S, T = S_{\ell \leftarrow i})$ where $x_S = z$ and ℓ be the index of S, i.e., $(z, z^{\oplus \ell})$ is an edge in M. We claim that ℓ is also the index of T. Indeed, by definition x_T is either z or $z^{\oplus \ell}$ and therefore T's index is ℓ. It follows that for every j the pair $(T, T_{\ell \leftarrow j})$ is also an edge in H and by taking j to be the ℓ-th entry of S we get that $(T, T_{\ell \leftarrow j} = S)$ is an edge. The rest of the claim follows directly from the definition of H. □

In fact, it is not hard to verify that the edges form an equivalence relation and therefore the graph is composed of vertex-disjoint union of n-sized cliques. We now show that $\frac{1}{2} + \delta/3$ of the nodes in H are predictable.

Claim 5.7. $\Pr_S[S$ *is predictable*$|x_S \in M] > \frac{1}{2} + \delta/3$.

Proof. By Bayes' theorem and the goodness of (x, G, r) we have

$$\begin{aligned} 1 - \mu/2 + \delta &< \Pr_S[S \text{ is predictable}] \\ &= \Pr_S[x_S \notin M] \cdot \Pr_S[S \text{ is predictable}|x_S \notin M] + \Pr_S[x_S \in M] \cdot \Pr_S[S \text{ is predictable}|x_S \in M]. \end{aligned}$$

Rearranging the equation and bounding $\Pr_S[S$ is predictable$|x_S \notin M]$ by 1, gives

$$\begin{aligned} \Pr_S[S \text{ is predictable}|x_S \in M] &> \left(\Pr_S[S \text{ is predictable}] - \Pr_S[x_S \notin M]\right) \cdot \frac{1}{\Pr_S[x_S \in M]} \\ &> \frac{1 - \mu/2 + \delta - 1 + \Pr_S[x_S \in M]}{\Pr_S[x_S \in M]}. \end{aligned}$$

Recall that $\Pr_S[x_S \in M] = (\mu \pm o(1))$, hence, we conclude that

$$\begin{aligned} \Pr_S[S \text{ is predictable}|x_S \in M] &> \frac{1 - \mu/2 + \delta - 1 + \mu - o(1)}{\mu + o(1)} \\ &= \frac{\mu/2 + \delta - o(1)}{\mu + o(1)} \\ &> \frac{1}{2} + \delta/2 - o(1), \end{aligned}$$

and the claim follows. □

Eq. 9 now follows from Claims 5.6 and 5.7. This completes the proof of Lemma 5.5. □

6 From Unpredictability to Pseudorandomness

We will prove Theorems 1.2, 1.3, and 1.4 by combining the "one-wayness to unpredictability" reductions (proved in Sections 4 and 5) with several generic transformations.

First we will need the well-known theorem of Yao [40] which shows that sufficiently strong unpredictability leads to pseudorandomness:

Fact 6.1 (Good UG ⇒ PRG)**.** *A* $(\frac{1}{2} + \varepsilon)$ *UG of output length* $m(n)$ *is an* $m \cdot \varepsilon$ *PRG.*

By combining this fact with Theorem 4.1 we obtain Theorem 1.4:

Corollary 6.2 (Theorem 1.4 restated). *For every constant d, sensitive predicate $Q : \{0,1\}^d \to \{0,1\}$, length parameter $m(n) \geq n$, and an inverse polynomial $\delta(n) \in (0,1)$, if $\mathcal{F}_{Q,m^3/\delta^2}$ is one-way then $\mathcal{F}_{Q,m}$ is $c\delta$-pseudorandom, for some constant $c = c(d) > 0$.*

Proof. By Theorem 4.1, if $\mathcal{F}_{Q,m^3/\delta^2}$ is one-way then $\mathcal{F}_{Q,m}$ is $(\frac{1}{2} + \Omega_d(\delta/m))$-unpredictable, and by Yao's theorem (Fact 6.1) the latter is $\Omega_d(\delta)$-pseudorandom. □

Recall that in Theorem 5.1 we showed that for constant ε and sufficiently large $m = \Omega(n)$ if $\mathcal{F}_{Q,m}$ is ε-one-way then it is also ε'-unpredictable for some related constant $\varepsilon' < 1$. We would like to use this theorem to obtain a linear stretch PRG. However, in this case Yao's theorem (Fact 6.1) is useless as we have only constant unpredictability. For this setting of parameters we give an alternative new $\mathbf{NC^0}$ transformation from UG to PRG which preserves linear stretch.

Theorem 6.3. *For every constant $0 < \varepsilon < \frac{1}{2}$, there exists a constant $c > 0$ such that any $\mathbf{NC^0}$ unpredictable generator $G : \{0,1\}^n \to \{0,1\}^{cn}$ which is $(\frac{1}{2} + \varepsilon)$-unpredictable, can be transformed into an $\mathbf{NC^0}$ pseudorandom generator with linear stretch (e.g., that maps n bits to $2n$ bits) and negligible distinguishing advantage.*

The theorem is proved by combining the techniques of [26] with non-trivial $\mathbf{NC^0}$ randomness extractors from [7]. The proof of this theorem is deferred to Section 6.1.

As a corollary of the above theorem and Theorem 5.1 we get:

Corollary 6.4 (Theorem 1.2 restated). *Let $d \in \mathbb{N}$ be an arbitrary constant and $Q : \{0,1\}^d \to \{0,1\}$ be a predicate. Then there exist constants c and ε such that if $\mathcal{F}_{Q,cn}$ is ε one-way then there exists a collection of PRGs which doubles its input in $\mathbf{NC^0}$.*

We mention that by standard techniques (see Fact 6.5 below), we can obtain any fixed linear stretch at the expense of increasing the locality to a different constant.

We will now show that for sensitive Q if $\mathcal{F}_{Q,n^{1+\delta}}$ is one-way then one can get arbitrary polynomial stretch and arbitrary (fixed) inverse polynomial security in $\mathbf{NC^0}$ (i.e., prove Theorem 1.3). For this purpose, we will need the following amplification procedures (together with Theorem 4.1):

Fact 6.5 (Amplifying unpredictability and stretch). *For every polynomial $t = t(n)$ and constant s:*

1. *A d-local $(\frac{1}{2} + \varepsilon(n))$-UG $G : \{0,1\}^n \to \{0,1\}^{m(n)}$ can be transformed into a (td)-local $(\frac{1}{2} + \varepsilon'(n))$-UG $G' : \{0,1\}^{n \cdot t} \to \{0,1\}^{m(n)}$ where $\varepsilon'(n) = \varepsilon(n)^{\Omega(t)} + \mathrm{neg}(n)$.*

2. *A d-local $\varepsilon(n)$-PRG $G : \{0,1\}^n \to \{0,1\}^{n^b}$ can be transformed into a (d^s)-local $s\varepsilon(n)$-PRG $G' : \{0,1\}^n \to \{0,1\}^{n^{(b^s)}}$.*

The above fact also holds with respect to collections. The first part is based on Yao's XOR-lemma, and may be considered to be a folklore, and the second part is based on standard composition. A proof is given in Section A for completeness.

We can prove Theorem 1.3.

Corollary 6.6 (Theorem 1.3 restated). *For every constant d, sensitive predicate $Q : \{0,1\}^d \to \{0,1\}$, and constant $\delta > 0$. If $\mathcal{F}_{Q,n^{1+\delta}}$ is one-way then for every stretch parameter $1 < a < O(1)$ and security parameter $1 < b < o(n)$ there exists a collection of PRGs of output length n^a and pseudorandomness of $1/n^b + \mathrm{neg}(n)$ with locality $d' = (bd/\delta)^{O(\frac{\lg a}{\delta})}$.*

Note that for fixed $\delta > 0$, we can have PPRG with arbitrary fixed polynomial stretch and security with constant locality. Alternatively, by setting $b = b(n) = \omega(1)$ (e.g., $b = \log^*(n)$), we get a standard PPRG with slightly super constant locality.

Proof of Corollary 6.6. Fix d, Q and δ, and assume that $\mathcal{F}_{Q,n^{1+\delta}}$ is one-way. Without loss of generality, $\delta \leq 1$. Then, by Theorem 4.1, $\mathcal{F}_{Q,n^{1+\delta/4}}$ is $(\frac{1}{2} + n^{-\delta/4})$-unpredictable. We can now amplify unpredictability via Fact 6.5, part 1.

Specifically, by taking $t = \Omega(b/\delta)$ we get a new generator G with input length $\ell = tn$, output length $n^{1+\delta/4} = \ell^{1+\delta/5}$, locality td and unpredictability of $n^{-(b+4)} = \ell^{-(b+3)}$. By Yao's theorem (Fact 6.1) the resulting collection is pseudorandom with security $\ell^{-(b+3)} \cdot \ell^{1+\delta/5} = \ell^{-(b+1)}$ (as $\delta \leq 1$).

Finally, increase the stretch of the PRG by applying s-composition (Fact 6.5, part 2), for $s = \lg(a)/\lg(1 + \delta/5)$. This leads to a PRG which stretches ℓ-bits to $\ell^{(1+\delta/5)^s} = \ell^a$ bits, with pseudorandomness of $s \cdot \ell^{-(b+1)} < \ell^{-b}$, and locality of $(td)^s = (bd/\delta)^{O(\frac{\lg a}{\delta})}$. □

6.1 Proof of Theorem 6.3

We will prove the following (seemingly weaker) version of Theorem 6.3. For some universal constants $0 < \varepsilon_0 < \frac{1}{2}$ and $c_0 > 0$, if there exists an $(\frac{1}{2}+\varepsilon_0)$-UG (resp., collection of UG) $G : \{0,1\}^n \to \{0,1\}^{c_0 n}$ in $\mathbf{NC^0}$, then there exists a PRG (resp., collection of PRG) with linear stretch in $\mathbf{NC^0}$. Note that this version implies Theorem 6.3, as for any fixed $\varepsilon > 0$ given $(\frac{1}{2} + \varepsilon)$-unpredictable generator $G : \{0,1\}^n \to \{0,1\}^{cn}$ with sufficiently large constant $c = c_\varepsilon$, we can amplify unpredictability (via Fact 6.5, part 2) and obtain a new UG in $\mathbf{NC^0}$ and unpredictability of $(\frac{1}{2} + \varepsilon_0)$ and stretch $c_0 n$.

To prove the theorem we will employ $\mathbf{NC^0}$ randomness extractors.

Extractors. The *min-entropy* of a random variable X is at least t if for every element x in the support of X we have that $\Pr[X = x] \leq 2^{-t}$. A mapping $\mathrm{Ext} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^N$ is a (t, Δ) *randomness extractor* (or extractor in short), if for every random variable X over $\{0,1\}^n$ with min-entropy of t, we have that $\mathrm{Ext}(\mathcal{U}_\ell, X)$ is Δ statistically-close to the uniform distribution. We refer to Δ as the *extraction error*, and to the first argument of the extractor as the *seed*. We typically write $\mathrm{Ext}_r(x)$ to denote $\mathrm{Ext}(r, x)$. We will use the following fact which follows by combining Lemma 5.7 and Theorem 5.12 of [7]:

Fact 6.7 (Non-trivial extractors in $\mathbf{NC^0}$). *For some constants $\alpha, \beta < 1$ there exists an $\mathbf{NC^0}$ extractor* Ext *that extracts n bits from random sources of length n and min-entropy $\alpha \cdot n$, by using a seed of length βn. Furthermore, the error of this extractor is exponentially small in n.*

We mention that the extractor is relatively weak ($\alpha + \beta > 1$), however, it will be sufficient for our purposes.

Construction 6.8. *Let $G : \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}^{cn}$ be a UG collection and* $\mathrm{Ext} : \{0,1\}^{\beta n} \times \{0,1\}^n \to \{0,1\}^n$ *be the extractor of Fact 6.7. We define the following collection $H : \{0,1\}^{s(n)\cdot n} \times \{0,1\}^{n^2(1+c\beta)} \to \{0,1\}^{cn^2}$ as follows.*

- *Index: n independent indices $\vec{k} = (k^{(1)}, \ldots, k^{(n)}) \in (\{0,1\}^{s(n)})^n$ where each index is sampled independently by the index sampling algorithm $K(1^n)$ of G.*
- *Input: n independent seeds $\vec{x} = (x^{(1)}, \ldots, x^{(n)}) \in (\{0,1\}^n)^n$ for the generator, and cn independent seeds for the extractor $\vec{z} = (z^{(1)}, \ldots, z^{(cn)}) \in (\{0,1\}^{\beta n})^{cn}$.*

- *Output: Compute the $n \times cn$ matrix Y whose i-th row is $G_{k^{(i)}}(x^{(i)})$. Let Y_i denote the i-th* column *of Y, and output* $(\mathrm{Ext}_{z^{(1)}}(Y_1), \ldots, \mathrm{Ext}_{z^{(cn)}}(Y_{cn}))$.

Note that H has linear stretch if $c > 1/(1-\beta)$. Also, the locality of H is the product of the localities of G and Ext, and so it is constant. Let ε be a constant which is strictly smaller than $(1-\alpha)/2$.

Lemma 6.9. *If G is $(\frac{1}{2}+\varepsilon)$-unpredictable, then the mapping H is a pseudorandom generator.*

Proof. The proof follows (a special case of) the analysis of [26, Sec.5].[10] First, by Proposition 4.8 of [26], we have that, being a $(\frac{1}{2}+\varepsilon)$-UG, G has next-bit pseudo-entropy in the following sense. For every sequence of efficiently computable index family $\{i_n\}$ and efficient distinguisher $\mathcal{A}$ there is a random binary variable W, jointly distributed with $k \overset{R}{\leftarrow} K(1^n)$ and $G_k(\mathcal{U}_n)$, such that: (1) the Shannon entropy of W given k and the $i_n - 1$ prefix of $G(\mathcal{U}_n)$ is at least δ, where $\delta = 1-2\varepsilon$; and (2) $\mathcal{A}$ cannot distinguish between $(k, G_k(\mathcal{U}_n)_{[1..i_n]})$ and $(k, G_k(\mathcal{U}_n)_{[1..i_n-1]}, W)$ with more than negligible advantage, even when $\mathcal{A}$ is given an oracle which samples the joint distribution $(k, G_k(\mathcal{U}_n), W)$.

Then, we use Claim 5.3 of [26], to argue that the n-fold direct product $G^{(n)}$ which outputs the matrix Y (defined in Construction 6.8) has block pseudo-min-entropy of $n(\delta - o(1))$ in the following sense. For every sequence of efficiently computable index family $\{i_n\}$ and efficient distinguisher $\mathcal{A}$ there is a random variable $W \in \{0,1\}^n$ jointly distributed with $\vec{k} \overset{R}{\leftarrow} K^n(1^n)$ and $Y = G^{(n)}_{\vec{k}}(\mathcal{U}_{n\times n})$, such that: (1) the min-entropy of W, given $\vec{k}$ and the first $i_n - 1$ columns of Y, is at least $n(\delta - o(1))$; and (2) $\mathcal{A}$ cannot distinguish between $(\vec{k}, Y_{[1..i_n]})$ and $(\vec{k}, Y_{[1..i_n-1]}, W)$ with more than negligible advantage, even when $\mathcal{A}$ is given an oracle which samples the joint distribution $(\vec{k}, Y, W)$.

This means that for every family $\{i_n\}$ the distribution $(\vec{k}, Y_{[1..i_n-1]}, \mathrm{Ext}_{\mathcal{U}_{\beta n}}(Y_{i_n}))$ is indistinguishable from $(\vec{k}, Y_{[1..i_n-1]}, \mathcal{U}_n)$. Otherwise, an adversary $\mathcal{B}$ that contradicts this statement can be used to construct an adversary $\mathcal{A}$ which contradicts the previous claim. Specifically, $\mathcal{A}(\vec{k}, M, v)$ chooses a random seed s for the extractor and invokes $\mathcal{B}$ on $(\vec{k}, M, \mathrm{Ext}_s(v))$. If v is chosen from Y_{i_n} then $\mathcal{B}$ gets a sample from $(\vec{k}, Y_{[1..i_n-1]}, \mathrm{Ext}_{\mathcal{U}_{\beta n}}(Y_{i_n}))$, and if v is chosen from W, $\mathcal{B}$ gets a sample from $(\vec{k}, Y_{[1..i_n-1]}, \mathrm{Ext}_{\mathcal{U}_{\beta n}}(W))$ which is statistically close to $(\vec{k}, Y_{[1..i_n-1]}, \mathcal{U}_n)$, as W has min-entropy of $n(\delta - o(1)) > \alpha n$. Hence, $\mathcal{A}$ has the same distinguishing advantage as $\mathcal{B}$ (up to a negligible loss).

Finally, the above statement implies that for every family $\{i_n\}$, the distributions

$$(\vec{k}, H_{\vec{k}}(\mathcal{U}_{n^2(1+c\beta)})_{[1..i_n]}) \qquad \text{and} \qquad (\vec{k}, H_{\vec{k}}(\mathcal{U}_{n^2(1+c\beta)})_{[1..i_n-1]}, \mathcal{U}_1)$$

are indistinguishable. Therefore, H is $(\frac{1}{2} + \mathrm{neg}(n))$-unpredictable generator, and, by Yao's theorem (Fact 6.1), it is pseudorandom. □

7 Inapproximability of the Densest-Subgraph Problem

Recall that the $p - \mathsf{DSH}_d$ problem (Definition 1.5) is parameterized with a real valued function $p(n)$ and an integer $d \in \mathbb{N}$. The input to the problem is an (n, m, d) hypergraph, and the goal is

[10]The relevant reduction in [26, Sec.5] transforms *Next-Block Pseudoentropy* to pseudorandomness. Our task is much easier as we begin with *next-bit unpredictability*, which is a stronger notion of unpredictability. As a result, the adaptation of [26] to our context (described in Construction 6.8) results in a much simpler reduction which is also easier to analyze.

to distinguish between two cases: **No** case, where every set of nodes of density p contains at most $p^d(1+o(1))$ fraction of the hyperedges; and **Yes** case, in which there exists a set of nodes of density p that contains at least $p^{d-1}(1-o(1))$ fraction of the hyperedges. We show that the problem is hard assuming the pseudorandomness of random local functions. More precisely, we prove the following lemma.

Lemma 7.1. *Let $d \in \mathbb{N}$ be a constant, $Q : \{0,1\}^d \to \{0,1\}$ be a predicate, $0 < \delta < d/2$ be a constant, $\varepsilon : \mathbb{N} \to (0,1)$ be a function of n, and assume that $\mathcal{F}_{Q,n^{1+\delta}}$ is ε-pseudorandom. Then, for every inverse power of two, $p \in [n^{-\frac{\delta}{d+1.01}}, \frac{1}{2}]$, there exists a pair of efficiently samplable distribution ensembles D_{no} and D_{yes} over d-uniform hypergraphs with the following properties:*

1. *The ensembles D_{yes} and D_{no} are $\varepsilon \log(1/p)$-indistinguishable.*
2. $\Pr[D_{\mathsf{no}}$ *is No instance of* $p - \mathsf{DSH}_d] > 1 - \mathrm{neg}(n)$.
3. $\Pr[D_{\mathsf{yes}}$ *is Yes instance of* $p - \mathsf{DSH}_d] > 1 - \mathrm{neg}(n) - \varepsilon \log(1/p)$.

Letting $\varepsilon = 1/\log n$, we conclude that any efficient algorithm that solves $p - \mathsf{DSH}_d$ can be used to distinguish between D_{no} and D_{yes} with advantage of $1 - \mathrm{neg}(n)) - \varepsilon \log(1/p) > \frac{1}{2} > \varepsilon \log(1/p)$, contradicting item (1) of the lemma. Hence, Theorem 1.6 follows.

Before proving the lemma (Section 7.1) few remarks are in order:

- (The distributions.) We will fully describe D_{no} and D_{yes} in the next section. For now let us mention that D_{no} is just a random $\mathcal{G}_{n,m',d}$ hypergraph where m' is sampled from the binomial distribution $\mathrm{Bin}(p, n^{1+\delta})$, while D_{yes} is a "planted" distribution which can be described as follows. (1) A random hypergraph G is sampled from $\mathcal{G}_{n^{1+\delta},n,d}$ together with a random p-dense subset T of the nodes; (2) Each hyperedge S which is not fully contained in T is removed from the graph with some probability (whose value depends on the size of $S \cap T$ and the predicate Q). Since the number of removed hyperedges is sufficiently large, the subgraph induced on T becomes dense. Observe that the above procedure allows to efficiently sample a Yes instance together with a witness dense sub-hypergraph.

- (The value of δ.) The larger δ gets, the better inapproximaility ratio we obtain. Clearly, δ cannot be larger than $\delta(d)$ where $n^{1+\delta(d)}$ is the maximal stretch of d-local pseudorandom generators. For $d \geq 5$, the best known upper-bound on $\delta(d)$ is roughly $d/2$ due to [36]. (For smaller d's, $m < O(n)$.)

- (The value of p.) The density parameter p (which also determines the inapproximability factor) is lower-bounded by $n^{-\frac{\delta}{d+1.01}}$. We note that the constant 1.01 can be replaced by any constant larger than 1. In fact, our proof goes through as long as p is lower-bounded by $n^{-\frac{\delta}{d+1}} \cdot \mathrm{polylog}(n)$. (This restriction follows from Eq. 10 in Claim 7.3.) While we did not attempt to fully optimize the parameters, this is not far from being optimal. Indeed, when $p < n^{-\frac{\delta}{d-1}}/d$, every d-uniform hypergraph with n nodes and $m' = p \cdot n^{1+\delta}$ hyperedges is a Yes instance of $p - \mathsf{DSH}_d$, as any set of $p^{d-1}m'$ hyperedges is trivially contained in a set of $p^{d-1}m'd < pn$ nodes.

7.1 Proof of Lemma 7.1

Fix some constant $d \in \mathbb{N}$, predicate $Q : \{0,1\}^d \to \{0,1\}$, real valued function $\varepsilon(n)$, and constant $0 < \delta < d/2$. Let $m = n^{1+\delta}$. Let $p \in [n^{-\frac{\delta}{d+1.01}}, \frac{1}{2}]$ be an inverse power of two, and let $t = \log(1/p)$ be an integer in $[1, \frac{\delta \log n}{d+1.01}]$. We will prove Lemma 7.1 based on the ε-pseudorandomness of $\mathcal{F}_{Q,m}$. From now on, we will assume, without loss of generality, that $Q(1^d) = 1$, otherwise we can negate it, and use $1 - Q$ as our predicate. (It is not hard to see that pseudorandomness still holds.)

The distributions. Before we define the distribution ensembles D_{yes} and D_{no}, we will need to define an operator ρ as follows. Given an (n, m, d) hypergraph G, and a $t \times m$ binary matrix $Y \in \{0,1\}^{t \times m}$, we view the i-th column of Y as a t-bit label for the i-th hyperedge of G. Then, the operator $\rho(G, Y)$ outputs the (n, m', d) subgraph G' whose hyperedges are those hyperedges of G which are indexed under Y by the all-one string 1^t.

- **The distribution D_{no}.** Choose a random (n, m, d) hypergraph G, and a random $t \times m$ binary matrix $Y \xleftarrow{R} \mathcal{U}_{t \times m}$. Output the subgraph $G' = \rho(G, Y)$.
- **The distribution D_{yes}.** Choose a random (n, m, d) hypergraph G, and a random $t \times n$ binary matrix $X \xleftarrow{R} \mathcal{U}_{t \times n}$. Let $x^{(i)}$ be the i-th row of X, and define a $t \times m$ binary matrix Y whose i-th row is $f_{G,Q}(x^{(i)})$. Output the subgraph $G' = \rho(G, Y)$.

Note that D_{no} (rep., D_{yes}) is parameterized by an integer n, and therefore form a *distribution ensemble.* By abuse of notation, we will keep the parameter n implicit, and let D_{no} (resp., D_{yes}) denote both the ensemble and its n-th member, relying on context for the appropriate interpretation.

Claim 7.2. *If $\mathcal{F}_{Q,m}$ is ε-pseudorandom then the ensembles D_{no} and D_{yes} are $t\varepsilon$-indistinguishable.*

Proof. Assume, towards a contradiction, that $\mathcal{A}$ distinguishes D_{no} from D_{yes} with advantage $t\varepsilon$. Then the algorithm $\mathcal{B}$, which first applies ρ to its input and then applies $\mathcal{A}$ to the result, is a $t\varepsilon$-distinguisher for the distributions

$$(G, y^{(1)}, \ldots, y^{(t)}) \qquad \text{and} \qquad (G, f_{G,Q}(x^{(1)}), \ldots, f_{G,Q}(x^{(t)})),$$

where G is a random (n, m, d) hypergraph, the y's are random m-bit strings and the x's are random n-bit strings. By a standard hybrid argument (cf. [22, Section 3.2.3]), this contradicts the ε-pseudorandomness of $\mathcal{F}_{Q,m}$. $\square$

Let us analyze D_{no}. Since Y and G are independent, we can redefine D_{no} as follows: choose Y uniformly at random, and for each column of Y, which equals to the all-one string, add a random d-uniform hyperedge to G'. Hence, G' is just a random $\mathcal{G}_{n,m',d}$ hypergraph where m' is sampled from the binomial distribution $\mathrm{Bin}(p, m)$, where $p = 2^{-t}$. The following claim now follows from standard concentration bounds.

Claim 7.3. *With all but negligible probability, the hypergraph G' chosen from D_{no} satisfies the following: (1) It has $m' = mp(1 \pm 1/\log n)$ hyperedges; and (2) Every set S of nodes of density p contains at most a $p^d(1 + o(1))$ fraction of the hyperedges.*

Proof. The first item follows from a multiplicative Chernoff bound: define m independent Bernoulli random variables, where the i-th variable is 1 if the i-th hyperedge is chosen. Since each random variable succeeds with probability p, the probability of having $m' = (1 \pm 1/\log n)pm$ successes is at least $1 - \exp(-\Omega(mp/\log^2 n)) > 1 - \exp(-\Omega(n))$.

To prove the second item, let us condition on the event $m' > pm/2$ which, by the previous argument, happens with probability $1 - \mathrm{neg}(n)$. Fix such an m' and let $G' \overset{R}{\leftarrow} \mathcal{G}_{n,m',d}$. Consider a fixed set of nodes S of size pn in G'. Every hyperedge of G' falls in S with probability p^d. Hence, by a multiplicative Chernoff bound, the probability that S contains a set of hyperedges of density $p^d(1 + 1/\log n)$ is at most

$$\exp\left(-\Omega\left(\frac{p^d m'}{\log^2 n}\right)\right) = \exp\left(-\Omega\left(\frac{p^{d+1} m}{\log^2 n}\right)\right) = \exp\left(-\Omega\left(\frac{n^{1+\delta(1-\frac{d+1}{d+1.01})}}{\log^2 n}\right)\right) < \exp(-2n). \quad (10)$$

Therefore, by a union bound, the probability that this happens for some set S is at most $\exp(-2n + n) = \mathrm{neg}(n)$. □

Next, we prove that D_{yes} has a planted *dense* sub-hypergraph.

Claim 7.4. *With probability at least $1 - \mathrm{neg}(n) - \varepsilon \log(1/p)$, a hypergraph G' chosen from D_{yes} has a set of nodes S of density p that contains a fraction of at least $p^{d-1}(1 - o(1))$ hyperedges.*

Proof. Label each *node* of G by the corresponding t-bit column of the matrix X, and let T be the set of nodes which are labeled by the all-one string. Let $S \subseteq T$ be the lexicographically first pn nodes of T; if $|T| \leq pn$ then $S = T$.

Consider the following event E in which: (1) S is of density at least $q = p(1 - 1/\log n)$; (2) At least $q^d(1 - 1/\log n)$ fraction of the hyperedges of the original hypergraph G fall in S; (3) The number of remaining hyperedges m' in G' is at most $pm(1 + 1/\log n)$.

The main observation is that hyperedges which fall into S are labeled by the all-one strings as $Q(1^d) = 1$, and so they also appear in G'. Hence, if E happens, then in G' the q-dense set of nodes S contains a set of hyperedges of density at least

$$\frac{q^d(1 - 1/\log n)m}{m'} \geq \frac{p^d m(1 - 1/\log n)^{d+1}}{pm(1 + 1/\log n)} > p^{d-1}(1 - o(1)).$$

Since, $q \leq p$ we can always pad S with additional nodes and obtain a p dense set S' which contains $p^{d-1}(1 - o(1))m'$ hyperedges as required.

It remains to show that the event E happens with probability $1 - \mathrm{neg}(n) - \varepsilon \log(1/p)$. First, since each node falls in T independently with probability p, it follows from a multiplicative Chernoff bound, that T contains at least $p(1 - 1/\log n)n$ nodes with all but negligible probability $\exp(-\Omega(pn/\log^2 n)) = \exp(-n^{\Omega(1)})$. (Recall that $\delta < d/2$ and therefore $pn = n^{\Omega(1)}$.) Hence, the sub-event (1) holds with all but negligible probability. Conditioned on (1), the sub-event (2) corresponds to having at least $q^d(1 - 1/\log n)$ fraction of the hyperedges fall into a set S of density q. Since each hyperedge falls in S independently with probability q^d, (2) happens with all but $\exp(-\Omega(q^d m/\log^2 n)) < \mathrm{neg}(n)$ probability (due to a multiplicative Chernoff bound). Hence, (1) and (2) happen simultaneously with probability $1 - \mathrm{neg}(n)$.

Finally, we argue that the probability β that (3) holds is at least $1 - \mathrm{neg}(n) - t \cdot \varepsilon$. Indeed, consider the algorithm which attempts to distinguish D_{no} from D_{yes} by looking at m' and accepting if and only if $m' \leq (p + 1/\log n)m$. By Claim 7.3 this leads to a distinguisher with advantage $1 - \mathrm{neg}(n) - \beta$, which, by Claim 7.2, can be at most $t \cdot \varepsilon$.

To complete the proof, observe that, by a union bound, we have that (3) holds together with (1) and (2) with probability $1 - \mathrm{neg}(n) - t \cdot \varepsilon$. □

Lemma 7.1 follows from Claims 7.2, 7.3, and 7.4. □

Acknowledgement. The author is grateful to Oded Goldreich for closely accompanying this research, and for countless insightful comments and conversations that significantly affected the results of this paper. We also thank Uri Feige, Yuval Ishai and Alex Samorodnitsky for many valuable discussions.

References

[1] D. Achlioptas. *Handbook of Satisfiability*, chapter Random Satisfiability, pages 243–268. IOS Press, 2009.

[2] M. Alekhnovich. More on average case vs approximation complexity. In *FOCS*, pages 298–307. IEEE Computer Society, 2003.

[3] 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.

[4] B. Applebaum, B. Barak, and A. Wigderson. Public-key cryptography from different assumptions. In *Proc. of 42nd STOC*, pages 171–180, 2010.

[5] B. Applebaum, A. Bogdanov, and A. Rosen. A dichotomy for local small-bias generators. In *Proc. of 9th TCC*, pages 1–18, 2012.

[6] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. *SIAM Journal on Computing*, 36(4):845–888, 2006.

[7] B. Applebaum, Y. Ishai, and E. Kushilevitz. On pseudorandom generators with linear stretch in NC^0. *J. of Computational Complexity*, 17(1):38–69, 2008.

[8] S. Arora, B. Barak, M. Brunnermeier, and R. Ge. Computational complexity and information asymmetry in financial products. *Commun. ACM*, 54(5):101–107, 2011.

[9] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an $\mathrm{O}(n^{1/4})$ approximation for densest k-subgraph. In *Proc. of 42nd STOC*, pages 201–210, 2010.

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

[11] A. Bogdanov and A. Rosen. Input locality and hardness amplification. In *Proc. of 8th TCC*, pages 1–18, 2011.

[12] A. Coja-Oghlan. Random constraint satisfaction problems. In *Proc. 5th DCM*, 2009.

[13] J. Cook, O. Etesami, R. Miller, and L. Trevisan. Goldreich's one-way function candidate and myopic backtracking algorithms. In *Proc. of 6th TCC*, pages 521–538, 2009.

[14] M. Cryan and P. B. Miltersen. On pseudorandom generators in NC^0. In *Proc. 26th MFCS*, 2001.

[15] N. Dedic, L. Reyzin, and S. P. Vadhan. An improved pseudorandom generator based on hardness of factoring. In *Proc. 3rd SCN*, 2002.

[16] B. den Boer. Diffie-hillman is as strong as discrete log for certain primes. In *Proc. of 8th CRYPTO*, pages 530–539, 1988.

[17] S. O. Etesami. Pseudorandomness against depth-2 circuits and analysis of goldreich's candidate one-way function. Technical Report EECS-2010-180, UC Berkeley, 2010.

[18] U. Feige. Relations between average case complexity and approximation complexity. In *Proc. of 34th STOC*, pages 534–543, 2002.

[19] U. Feige, D. Peleg, and G. Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[20] A. Flaxman. Random planted 3-SAT. In M.-Y. Kao, editor, *Encyclopedia of Algorithms.* Springer, 2008.

[21] O. Goldreich. Candidate one-way functions based on expander graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(090), 2000.

[22] O. Goldreich. *Foundations of Cryptography: Basic Tools.* Cambridge University Press, 2001.

[23] O. Goldreich, H. Krawczyk, and M. Luby. On the existence of pseudorandom generators. *SIAM J. Comput.*, 22(6):1163–1175, 1993. Preliminary version in Proc. 29th FOCS, 1988.

[24] O. Goldreich, N. Nisan, and A. Wigderson. On yao's XOR-lemma. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(50), 1995.

[25] O. Goldreich and V. Rosen. On the security of modular exponentiation with application to the construction of pseudorandom generators. *J. Cryptology*, 16(2):71–93, 2003.

[26] I. Haitner, O. Reingold, and S. P. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. In *Proc. of 42nd STOC*, pages 437–446, 2010.

[27] 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.

[28] Herzberg and Luby. Public randomness in cryptography. In *Proc. of 12th CRYPTO*, 1992.

[29] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In *Proc. of 40th STOC*, pages 433–442, 2008.

[30] D. Itsykson. Lower bound on average-case complexity of inversion of goldreich's function by drunken backtracking algorithms. In *Computer Science - Theory and Applications, 5th International Computer Science Symposium in Russia*, pages 204–215, 2010.

[31] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *Proc. of 29th FOCS*, pages 68–80, 1988.

[32] S. Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proc. of 45th FOCS*, pages 136–145, 2004.

[33] L. A. Levin. One-way functions and pseudorandom generators. In *Proc. of 17th STOC*, pages 363–365, 1985.

[34] U. M. Maurer and S. Wolf. The relationship between breaking the diffie-hellman protocol and computing discrete logarithms. *SIAM J. Comput.*, 28(5):1689–1721, 1999.

[35] R. Miller. Goldreich's one-way function candidate and drunken backtracking algorithms. Distinguished major thesis, University of Virginia, 2009.

[36] E. Mossel, A. Shpilka, and L. Trevisan. On ϵ-biased generators in NC^0. In *Proc. 44th FOCS*, pages 136–145, 2003.

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

[38] N. Nisan and A. Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[39] S. K. Panjwani. An experimental evaluation of goldreich's one-way function. Technical report, IIT, Bombay, 2001.

[40] A. C. Yao. Theory and application of trapdoor functions. In *Proc. 23rd FOCS*, pages 80–91, 1982.

A Omitted proofs

A.1 Amplifying unpredictability and stretch

We will prove Fact 6.5.

Part 1: unpredictability amplification. Let $G : \{0,1\}^\sigma \times \{0,1\}^n \to \{0,1\}^m$ be a d-local $(\frac{1}{2}+\varepsilon)$-unpredictable UG. Let $t = t(n)$ and define the UG collection $H : \{0,1\}^{\sigma t} \times \{0,1\}^{nt} \to \{0,1\}^m$ to be the bit-wise XOR of t independent copies of G, i.e., for

$$\vec{k} = (k^{(1)}, \dots, k^{(t)}) \in (\{0,1\}^\sigma)^t \qquad \text{and} \qquad \vec{x} = (x^{(1)}, \dots, x^{(t)}) \in (\{0,1\}^n)^t$$

let

$$H_{\vec{k}}(\vec{x}) = G_{k^{(1)}}(x^{(1)}) \oplus \ldots \oplus G_{k^{(t)}}(x^{(t)}).$$

The index-sampling algorithm of H is defined by sampling t independent keys from the key-sampling algorithm K of G. Clearly, H is td-local, we will show that it is $(\frac{1}{2} + \delta)$-unpredictable for $\delta = \varepsilon^{\Omega(t)} + \mathrm{neg}(n)$.

Assume, towards a contradiction, that there exists an algorithm $\mathcal{A}$ and a sequence of indices $\{i_n\}$ such that

$$\Pr_{\vec{k}\xleftarrow{R}(K(1^n))^t, \vec{x}\xleftarrow{R}(\mathcal{U}_n)^t, y=H_{\vec{k}}(\vec{x})}[\mathcal{A}(\vec{k}, y_{[1..i_n-1]}) = y_{i_n}] > \frac{1}{2} + \delta,$$

for infinitely many n's. Then, there exists another adversary $\mathcal{A}'$ for which

$$\Pr_{\vec{k}\xleftarrow{R}(K(1^n))^t, \vec{x}\xleftarrow{R}(\mathcal{U}_n)^t, y^{(i)}=G_{k^{(i)}}(x^{(i)})}[\mathcal{A}((k^{(1)}, y^{(1)}_{[1..i_n-1]}), \ldots, (k^{(t)}, y^{(t)}_{[1..i_n-1]})) = y^{(1)}_{i_n} \oplus \cdots \oplus y^{(1)}_{i_n}] > \frac{1}{2} + \delta,$$

for the same input lengths. Let Y_n denote the distribution $(K(1^n), G_{K(1^n)}(\mathcal{U}_n))$, viewed as a single $(\sigma+m)$-bit vector. Define a randomized predicate P_n which, given a string w (of length $\sigma+i_n-1$), samples the $(\sigma+i_n)$-th bit of Y_n, conditioned on $Y_n[1..(\sigma+i_n-1)] = w$. Then, the last equation can be rewritten as

$$\Pr[\mathcal{A}'(w^{(1)}, \ldots, w^{(t)}) = \bigoplus_{j\in[t]} P_n(w^{(j)})] > \frac{1}{2} + \delta,$$

where each $w^{(j)}$ is sampled uniformly and independently from $Y_n[1..\sigma+i_n-1]$. By Yao's XOR lemma (cf. [24]), such an efficient adversary $\mathcal{A}'$ implies an adversary $\mathcal{A}''$ for which

$$\Pr[\mathcal{A}''(Y_n[1..(\sigma+i_n-1)]) = P_n(Y_n[1..(\sigma+i_n-1)]) = Y_{\sigma+i_n}] > \frac{1}{2} + \varepsilon,$$

for the same input lengths, in contradiction to the $(\frac{1}{2}+\varepsilon)$-unpredictability of G.

Uniformity. In order to apply the above argument in a fully uniform setting we should make sure that pairs $(Y_n[1..\sigma+i_n-1], Y_n[\sigma+i_n])$ are efficiently samplable. Since Y_n is efficiently samplable it suffices to show that the sequence $\{i_n\}$ is uniform, i.e., can be generated in time poly(n). In fact, to get our bound, it suffices to have a uniform sequence $\{i'_n\}$ for which $\mathcal{A}$ achieves prediction probability of $\frac{1}{2}+\delta-\sqrt{\delta}$. Hence, we can use Remark 3.2.

Part 2: stretch amplification. Let G be the original collection of PRGs with key sampling algorithm K. We define the s-wise composition of G as follows. The collection $G^{(s)}_{\vec{k}}(x)$ is indexed by s-tuple of "original" indices $\vec{k} = (k_0, \ldots, k_s)$ where the i-th entry is sampled uniformly and independently by invoking the original index sampling generator K on $(1^{n^{(b^i)}})$. We define $G^{(0)}_{\vec{k}}(x)$ to be $G_{k_0}(x)$, and for every $i > 0$ we let $G^{(i)}_{\vec{k}}(x) = G_{k_i}(G^{(i-1)}_{\vec{k}}(x))$. Clearly, the resulting collection has output length of $n^{(b^s)}$ and locality d^s. A standard hybrid argument shows that the security is $s\varepsilon(n)$. (See [22, Chapter 3, Exercise 19].)