Sampling Graphs without Forbidden Subgraphs and Unbalanced Expanders with Negligible Error
(Extended Abstract)

Benny Applebaum, Eliran Kachlon
School of Electrical Engineering
Tel-Aviv University
Tel-Aviv, Israel
{*benny.applebaum, elirn.chalon*}*@gmail.com*

***Abstract*—Suppose that you wish to sample a random graph G over n vertices and m edges conditioned on the event that G does not contain a "small" t-size graph H (e.g., clique) as a subgraph. Assuming that most such graphs are H-free, the problem can be solved by a simple rejected-sampling algorithm (that tests for t-cliques) with an expected running time of $n^{O(t)}$. Is it possible to solve the problem in running time that does not grow polynomially with n^t?**

In this paper, we introduce the general problem of sampling a "random looking" graph G with a given edge density that avoids some arbitrary predefined t-size subgraph H. As our main result, we show that the problem is solvable with respect to some specially crafted k-wise independent distribution over graphs. That is, we design a sampling algorithm for k-wise independent graphs that supports efficient testing for subgraph-freeness in time $f(t) \cdot n^c$ where f is a function of t and the constant c in the exponent is independent of t. Our solution extends to the case where both G and H are d-uniform hypergraphs.

We use these algorithms to obtain the first probabilistic construction of constant-degree polynomially-unbalanced expander graphs whose failure probability is *negligible* in n (i.e., $n^{-\omega(1)}$). In particular, given constants $d > c$, we output a bipartite graph that has n left nodes, n^c right nodes with right-degree of d so that any right set of size at most $n^{\Omega(1)}$ expands by factor of $\Omega(d)$. This result is extended to the setting of unique expansion as well.

We observe that such a negligible-error construction can be employed in many useful settings, and present applications in coding theory (batch codes and LDPC codes), pseudo-randomness (low-bias generators and randomness extractors) and cryptography. Notably, we show that our constructions yield a collection of polynomial-stretch locally-computable cryptographic pseudorandom generators based on Goldreich's one-wayness assumption resolving a central open problem in parallel-cryptography (cf., Applebaum-Ishai-Kushilevitz, FOCS 2004; and Ishai-Kushilevitz-Ostrovsky-Sahai, STOC 2008).

***Keywords*-Expander Graph, LDPC Codes, Local Cryptography**

I. Introduction

Many combinatorial properties of graphs and hypergraphs can be formulated as avoiding some family $\mathcal{H}$ of small subgraphs. Notable examples consist of graphs that avoid short cycles or small cliques, expander graphs (that avoid small non-expanding subgraphs) and even graphical representations of good error-correcting codes (that avoid small "stopping sets" [17]). Motivated by the wide range of applications, the computational problem of efficiently constructing $\mathcal{H}$-free graphs has attracted a huge amount of research (e.g. [1], [15], [16], [23], [33]). In this paper, we consider several natural probabilistic variants of the construction problem.

Setup: Let $\mathcal{G}_{n,m,d}$ be the set of all (n, m, d)-*hypergraphs*, i.e., d-uniform hypergraphs over n vertices with m hyperedges. We typically think of d as a constant that does not grow with n and take $m = \mathrm{poly}(n)$. Let $\mathcal{H}$ be a family of "small" d-uniform hypergraphs of size at most t for some slowly growing function $t(n)$. While our setup is defined with respect to hypergraphs (to match our applications), the following problems make sense even for simple undirected graphs (i.e., $d = 2$) and so, for now, the reader may safely focus on this special case. (Indeed, we are not aware of prior solutions that handle the case of simple graphs.)

Problem 1.1 (Zero-error/negligible-error constructions): Generate an $\mathcal{H}$-free hypergraph $G \in \mathcal{G}_{n,m,d}$ in probabilistic $\mathrm{poly}(n)$-time. The algorithm is allowed to fail with a *negligible* error probability that vanishes faster from any inverse polynomial, i.e., $n^{-\omega(1)}$. Such a construction is referred to as a *negligible-error construction*. We say that this is a *zero-error* (or $\mathbf{ZPP}$) construction if the algorithm outputs a special failure symbol whenever it fails to find an $\mathcal{H}$-free graph.

Unlike the classical de-randomization literature which typically emphasizes the distinction between *deterministic* and *probabilistic* construction, in Problem 1.1 we focus on the *error level*. We advocate the use of negligible-error constructions as a second-best alternative when explicit constructions are unknown. Indeed, for many applications a randomized construction that almost never fails is almost as good as a fully explicit construction. In particular, if one is planning to plug-in G into some randomized algorithm or system then a negligible error in the construction of G will be swallowed by the overall error probability of the

2575-8454/19/$31.00 ©2019 IEEE
DOI 10.1109/FOCS.2019.00020

algorithm.[1]

Following the standard cryptographic tradition, we insist on an error that is negligible (i.e., tends to zero faster than any polynomial), in order to guarantee a tiny failure probability even after polynomially-many repetitions.[2] Throughout the paper, we typically assume that an α-fraction of all (n, m, d)-hypergraphs are $\mathcal{H}$-free, where the density α is large but not overwhelming, i.e., $\alpha(n) = 1 - n^{-c}$ for some constant $c > 0$. In this case, the problem is non-trivial when testing $\mathcal{H}$-freeness cannot be done in polynomial-time.

While Problem 1.1 is a relaxation of the explicit-construction problem, our next problem addresses the harder task of generating a random, or pseudorandom, $\mathcal{H}$-free graph.

Problem 1.2 (Quasi-random $\mathcal{H}$-free graphs): Sample in expected probabilistic $\mathrm{poly}(n)$ time a random graph G from some "pseudorandom" distribution over $\mathcal{G}_{n,m,d}$ *conditioned* on being $\mathcal{H}$-free.

The general task of generating a pseudorandom object that always satisfies some given property was first studied by Goldreich, Goldwasser and Nussboim [21].[3] In this setting the property (i.e., $\mathcal{H}$-freeness) is viewed as a necessary *worst-case* requirement that should be satisfied by *any* sampled hypergraph G. Using the terminology of [21], the *implementation* G must be *truthful* to the $\mathcal{H}$-freeness *specification*. Conditioned on this, G should be distributed uniformly or close to uniformly under some metric.

This combination of requirements arises when one tries to understand the behavior of an $\mathcal{H}$-free random system (e.g., in simulation) or when the hypergraph G is being used as part of a system whose analysis relies on a random choice of G and, in addition, its validity depends on $\mathcal{H}$-freeness. In such a case we cannot use a single explicit construction of $\mathcal{H}$-free hypergraphs since it may fail to achieve some other property of pseudorandom hypergraphs. On the other hand, we cannot use a random sample from $\mathcal{G}_{n,m,d}$ since it fails to be $\mathcal{H}$-free with *positive* (in our case, inverse polynomial) probability.

We further mention that in some cases even a tiny positive failure probability can be problematic. This is the case, for example, when the sampling procedure is invoked by an untrusted party who can benefit from the existence of $\mathcal{H}$-subgraphs. If our sampling algorithm has a positive failure probability, then a cheating party can cheat by selecting "bad coins" that lead to hypergraphs with $\mathcal{H}$-subgraphs. Since general subgraph-testing seems to be computationally-hard such a cheating may be left undetected.[4]

The testing barrier: A natural way to sample $\mathcal{H}$-free random hypergraphs is via rejected sampling. That is, repeatedly sample G until an $\mathcal{H}$-free hypergraph is chosen. Since we work in a regime where most hypergraphs are $\mathcal{H}$-free, the expected number of iterations will be polynomial. This approach reduces the sampling problem to the subgraph testing problem. If the largest hypergraph in $\mathcal{H}$ is of constant size t, then the problem can be trivially solved in time $f(t)n^{O(t)}$. However, we think of t as a large constant, or as a slowly increasing function of n, and so we would like to have a running time of $f(t)n^c$ where the exponent c is independent of t. Unfortunately, such a running time cannot be achieved for general subgraph-testing (even for simple cases such as cliques) unless the exponential-time hypothesis (ETH) fails (cf. [18]). We refer to this hardness-of-testing as the *testing barrier*. Jumping ahead, we will show that some variant of this barrier arises if one tries to sample a hypergraph that is *uniformly* distributed over all $\mathcal{H}$-free hypergraph in $\mathcal{G}_{n,m,d}$.

Summary: The problem of constructing $\mathcal{H}$-free hypergraphs can be roughly ranked from easy to hard as follows: negligible-error constructions, zero-error constructions, explicit constructions, pseudorandom constructions.

II. Our Results

We partially resolve Problems 1.1 and 1.2. Our main results consist of two main parts. We begin by studying pseudorandom constructions of $\mathcal{H}$-free hypergraphs (Sections II-A and II-B). Then we focus on the concrete case of unbalanced expanders, describe negligible-error constructions of such graphs (Section II-C), and use them to derive various applications (Section II-D).

A. Sampling k-Wise Independent Graphs Conditioned on H-Freeness

We show that Problem 1.2 can be solved with respect to some *k-wise independent* distribution over $\mathcal{G}_{n,m,d}$. Here k-wise independence means that every k-subset of the hyperedges are distributed uniformly over all possible d-uniform hyperedges. The use of k-wise independent distributions as a good model for pseudorandom graphs was advocated by Naor, Nussboim and Tromer [36] and by Alon and Nussboim [2]. These works further show that a large family of natural graph-theoretic properties that hold whp over random graphs (with a given edge density) also hold whp over $\mathrm{polylog}(n)$-wise independent distributions with the same density.

[1] This view is implicitly used in other contexts. For example, although the problem of deterministically generating n-bit primes is wide open, there are randomized algorithms that generate such primes with negligible (or even zero) error probability. Consequently, applications which employ prime numbers (or prime-order finite fields) rely on negligible-error constructions.

[2] Observe that in our context it is not clear how to reduce the error probability from constant or even inverse polynomial $1/n^{\Omega(1)}$ to negligible.

[3] The work of [21] focuses on *huge* exponential-size random objects. However, the problem remains non-trivial even for polynomial-size objects as long as the required property cannot be tested in polynomial-time. See Section II-B for further discussion regarding the applicability of our results to the GGN setting.

[4] The work of [14] provides a good example for such a case in the context of *financial derivatives*.

We bypass the "testing barrier" by designing a concrete k-wise independent probability distribution $\mathcal{G}_{n,m,d,k}$ in a way that allows us to efficiently test whether a given sample G is H-free. That is, our distribution is amenable to subgraph testing *by design*. To formalize this strategy, we introduce a new notion of *sampler/tester* pair of algorithms. Roughly speaking, the sampler S samples an object according to some given distribution D, and the tester T examines the *coins* of the tester and checks whether the corresponding object avoids some *bad* event E. The combination of the two allows us to sample the conditional distribution $[D|\neg E]$. (See full version of this paper [12] for more details on the sampler/tester framework.)

We prove the following key theorem. Below we define the log-density of an (n, m, d) hypergraph as $c = \log_n m$, and define the size of a hypergraph as the sum of its vertices and hyperedges.

Theorem 2.1 (key theorem): For every log-density parameter $c > 1$, edge-uniformity parameter $d \geq 2$, subgraph-size function $t(n) \leq O(\frac{\log\log\log n}{\log\log\log\log n})$ and independence parameter $k(n)$ that satisfies $k(n) \leq O(n^{1/t^{c't}})$ where c' is a constant that depends on c, there exists a $\mathrm{poly}(n)$-time sampler/tester pair (S, T) with the following properties:

- Given 1^n, the randomized sampler S uses its internal random coins r to sample an $(n, m = n^c, d)$ hypergraph G_r whose hyperedges are $k(n)$-wise independent.
- The deterministic tester takes as input a d-uniform hypergraph H of size at most $t(n)$ and a fixed sequence of coin tosses r and accepts the input if and only if H is a subgraph of the hypergraph $G_r = S(1^n, r)$ that is generated by S using coin tosses r.

Although the size t of the tested subgraph is relatively small, it is still *super-constant*. This property will be crucial for our applications. We further note that the independence parameter $k(n)$ is super-logarithmic (or even "almost" polynomial) in n and so the pseudorandomness properties established by [2], [36] hold. (See the full version [12] for a more detailed version of Theorem 2.1 as well as all other theorems that are stated here.)

Sampling $\mathcal{H}$-free graphs: It is important to note that our sampler S is independent of the subgraph H, and that the tester T gets H as an input. These properties allow us to partially solve the sampling problem (Problem 1.2) with respect to a *family* of small hypergraphs $\mathcal{H}$. Indeed, we can use the sampler S to sample a k-wise independent (n, m, d)-hypergraph G and use the basic tester to test that G is H-free for all subgraphs $H \in \mathcal{H}$. If one of the tests fails, we repeat the process from the beginning. Since $\mathcal{H}$ contains at most $\exp(t^d) < \mathrm{poly}(n)$ hypergraphs, the expected running time will be polynomial, assuming that a random k-wise independent (n, m, d)-hypergraph is $\mathcal{H}$-free with noticeable probability.

Corollary 2.2 (pseudorandom $\mathcal{H}$-free hypergraphs): Let $c, d, t(n), k(n)$ and $m = n^c$ be as in Theorem 2.1. Let $\mathcal{H}$ be an efficiently constructible family of hypergraphs each of size at most $t(n)$ such that a $k(n)$-wise independent (n, m, d)-hypergraph is $\mathcal{H}$-free with noticeable probability of $1/\mathrm{poly}(n)$. Then, there exists a probabilistic algorithm that runs in expected $\mathrm{poly}(n)$-time and samples an $\mathcal{H}$-free hypergraph from some k-wise independent distribution over (n, m, d)-hypergraphs.

Remark 2.3: It is natural to try and sample a *uniform* $\mathcal{H}$-free (n, m, d)-hypergraph, i.e., to replace the k-wise independent distribution in Corollary 2.2 with the uniform distribution over $\mathcal{G}_{n,m,d}$. We conjecture that sampling uniform $\mathcal{H}$-free hypergraphs is computationally infeasible and present some evidence towards this conjecture. In particular, suppose that:

> ($\star$) For some families of hypergraphs, it is infeasible to *certify* $\mathcal{H}$-freeness over the uniform distribution. That is, there is no 1-sided error tester that accepts most (n, m, d)-hypergraphs and never accepts a hypergraph that is not $\mathcal{H}$-free.

We show that, under the ($\star$) assumption, sampling uniform $\mathcal{H}$-free hypergraphs implies the existence of *one-way functions*. Put differently, a sampler would allow us to convert average-case hardness (of testing) to one-wayness, or, in the language of Impagliazzo [25], to move from Pessiland to Minicrypt.

The ($\star$) assumption (hardness of certifying $\mathcal{H}$-freeness) is closely related to previous intractability assumptions (cf. [7], [9], [14]). We further relate this assumption to the problem of certifying that a random low-density parity-check code has a high distance. (See the full version of this paper [12] for details.)

Remark 2.4 (On k-wise independence): It is instructive to note that Theorem 2.1 employs k-wise independence in an unconventional way. Typically, the notion of k-wise independence is useful due to the combination of pseudorandomness with computationally-cheap and randomness-efficient implementations. In contrast, the proof of Theorem 2.1 exploits the simple algebraic structure of k-wise independence constructions to force a structure on the sampled object (the hypergraph G) in a way that makes it amenable to efficient analysis (i.e., subgraph testing). The fact that such implementation is computationally-cheap or randomness-efficient is not really needed. (Nevertheless, these properties will be used in the next subsection.)

ZPP *and explicit constructions.:* Corollary 2.2 immediately leads to a **ZPP**-construction of $\mathcal{H}$-free hypergraphs. We further observe that, under standard worst-case derandomization assumptions, any **ZPP**-construction implies an explicit construction.

Corollary 2.5 (explicit $\mathcal{H}$-free hypergraphs): Let $c, d, t(n), k(n)$ and $m = n^c$ and $\mathcal{H}$ be as in Corollary 2.2. Assuming that the class of functions computable in $2^{O(n)}$ uniform-time requires $2^{\Omega(n)}$-size circuits, there exists a

deterministic $\mathrm{poly}(n)$-time algorithm that always outputs an $\mathcal{H}$-free (n, m, d)-hypergraph.

The above assumption is known to imply, for any constant a, a pseudorandom generator prg that fools n^a-time algorithms with logarithmic size seed [26]. Such a generator can be used to fully de-randomize the **ZPP** construction A and derive a fully explicit construction A'. (The algorithm A' just outputs the first seed s for which $A(\mathsf{prg}(s))$ does not output "failure".) This makes a crucial use of the ability to recognize bad outputs (which trivially holds for **ZPP** samplers). We are not aware of a similar transformation that applies to "Monte-Carlo" constructions that have a positive failure probability.[5]

B. The Succinct Setting

So far we assumed that the computational complexity of the sampler is allowed to grow polynomially in the size of the hypergraph G. In some scenarios, it is more natural to think of the hypergraph as a huge object and require a running time that is polynomial in $\log n$. In particular, we say that an (n, m, d) hypergraph G has a succinct representation if it can be represented by an identifier z of length $\mathrm{polylog}(n)$ such that given z, a hyperedge $e \in [m]$, and an index $i \in [d]$, it is possible to compute the i-th member of the hyperedge e in time $\mathrm{polylog}(n)$.[6] (Here we assume that the hyperedges are ordered and can be represented by d-tuples.) We prove a succinct version of Theorem 2.1 that applies to constant-size subgraphs H and $\mathrm{polylog}(n)$-wise independence.

Theorem 2.6: For every log-density parameter $c > 1$, edge-uniformity parameter $d \geq 2$, constant subgraph size t and independence parameter $k(n) \leq \mathrm{polylog}(n)$, there exists a $\mathrm{polylog}(n)$-time sampler/tester pair (S, T) with the following properties:

- Given n (in binary representation), the randomized sampler uses its internal random coins r to sample a succinct $(n, m = n^c, d)$ hypergraph G_r whose hyperedges are $k(n)$-wise independent.
- The deterministic tester takes as input a d-uniform hypergraph H of size at most t and a fixed sequence of coin tosses r and accepts the input if and only if H is a subgraph of the hypergraph $G_r = S(n, r)$ that is generated by S using coin tosses r.

Theorem 2.6 leads to the following succinct version of Corollary 2.2.

Corollary 2.7: Let $c, d, t, k(n)$ and $m = n^c$ be as in Theorem 2.6. Let $\mathcal{H}$ be a family of hypergraphs each of size at most t, such that a $k(n)$-wise independent (n, m, d)-hypergraph is $\mathcal{H}$-free with probability of $1/\mathrm{polylog}(n)$. Then, there exists a probabilistic algorithm that runs in expected $\mathrm{polylog}(n)$-time and samples a succinct $\mathcal{H}$-free hypergraph from some k-wise independent distribution over (n, m, d)-hypergraphs.

As already mentioned the problem of constructing huge k-wise independent graphs that satisfy some given property (specification) was studied in [2], [21], [35], [36]. Corollary 2.7 provides a zero-error (aka "truthful") solution for this problem with respect to $\mathcal{H}$-free hypergraphs of given density. To the best of our knowledge, prior to our work no solution was known even for the case of undirected graphs and concrete fixed-size forbidden subgraphs.

C. Negligible-Error Construction of Constant-Degree Unbalanced Expanders

We move back to the non-succinct setting, and consider the problem of explicitly constructing a single, not necessarily random, *expander* graph. We say that an (n, m, d)-hypergraph is an (α, t)-expander if every set S of hyperedges of size at most t "touches" at least $\alpha|S|$ vertices.[7] Equivalently, an (α, t)-expander is an (n, m, d)-hypergraph that avoids small "dense" subgraphs, i.e., (n', m', d)-hypergraphs with $n' \leq \alpha m'$ for $m' \leq t$.

We focus on the setting of *constant-degree highly unbalanced* expanders. That is, we let d be a constant, and assume that the number of hyperedges m is polynomially larger than n, i.e., $m = n^c$ for some constant log-density $1 < c < d$. A standard probabilistic calculation shows that in this regime most (n, m, d)-hypergraphs achieve a good expansion factor of $\alpha = \Omega(d)$ (or even $\alpha = d - O(1)$) for polynomially-small subsets of size at most $t = n^{1-\delta}$ where δ is a constant that depends on α, c and d. Unfortunately, the problem of *efficiently constructing* highly-unbalanced constant-degree expanders is wide open. Existing constructions either have only linearly many hyperedges $m = O(n)$ [16] or suffer from a super-constant (actually polylogarithmic) degree [23]. Motivated by the numerous applications of constant-degree highly-unbalanced expanders (to be discussed later), we present a negligible-error construction of such graphs.

We begin by giving a **ZPP**-construction of constant-degree highly-unbalanced hypergraphs that expand well for

[5]There are cases in which derandomization assumptions can be easily used to turn a negligible-error construction into an explicit construction [29]. This typically happens when one of the following holds: (1) It is "easy" to recognize a "bad" object (i.e., to detect a violation of the desired property) in polynomial-time; or (2) There is an efficient way to combine a "bad" instance with several "good" instances into a single "good" instance. As far as we know, in general, both conditions fail for $\mathcal{H}$-freeness.

[6]This is in contrast to the (more common) notion of succinctness (in the context of standard graphs), where, given a vertex v and an index i, we can compute the i-th neighbor of v in time $\mathrm{poly}\log n$. Our notion of succinctness is better suited for our applications, in which a hypergraph represents the dependencies graph of some function $f : \{0,1\}^n \to \{0,1\}^m$ (e.g., low-biased generator) where inputs correspond to vertices, outputs correspond to hyperedges, and the i-th hyperedge contains the vertices on which the i-th output depends. Our notion of succinctness guarantees that each output of f can be computed in $\mathrm{polylog}(n)$-time (e.g., in the RAM model).

[7]This formulation is equivalent to the more standard notion of bipartite expanders over n left vertices and m right vertices where the degree of each right vertex is d, and every set S of right vertices of size at most t is connected to at least $\alpha|S|$ left vertices.

small sets of super-constant size. The following theorem follows from Corollary 2.2 by instantiating the class $\mathcal{H}$ of forbidden subgraphs with the class of small non-expanding hypergraphs.

*Theorem 2.8 (**ZPP**-construction of small-set expanders):* For every log-density parameter $c > 1$, edge-uniformity parameter $d > c$, and $\alpha < d - c$ there exists a **ZPP**-construction of $(n, m = n^c, d)$-hypergraph with (α, t)-expansion where $t = O(\frac{\log\log\log n}{\log\log\log\log n})$.

Next, we show that Theorem 2.8 gives rise to a negligible-error construction of hypergraphs that expand well for polynomial-size subsets. That is, we downgrade the level of explicitness (from zero-error construction to negligible-error construction) and upgrade the expansion threshold t to polynomial.

Theorem 2.9: For every log-density parameter $c > 1$, edge-uniformity parameter $d > c$, and $\alpha < d - c$, there exists a negligible-error construction of $(n, m = n^c, 2d)$-hypergraph with (α, t)-expansion where $t = \Omega(n^{1-\delta})$ and $\delta = (c-1)/(d-\alpha-1)$.

Recall that a negligible-error construction guarantees the existence of a $\mathrm{poly}(n)$-time randomized algorithm that outputs, except with negligible probability of $n^{-\omega(1)}$, an (α, t)-expanding $(n, m, 2d)$-hypergraph.

Theorem 2.9 provides an $(n, m, D = 2d)$-hypergraph whose expansion parameters (α, t) match the parameters of a random (n, m, d)-hypergraph. While this factor-2 gap in the degree has a relatively minor effect on the expansion threshold t (which is still polynomial in n), it limits the expansion factor α to be at most $D/2 - O(1)$. Such an expansion factor suffices for many applications, but in some cases it is useful to expand by a factor larger than $D/2$. Notably, expansion beyond half the degree guarantees the useful *unique expansion* property. Formally, a hypergraph is a (β, t)-*unique expander* if for every set S of at most t hyperedges there exists a set U of at least $\beta|S|$ vertices such that each vertex in U appears in a unique hyperedge e in S.

Perhaps surprisingly, although we cannot expand by a factor better than $D/2$, we can still get a negligible-error construction of unique expanders.

Theorem 2.10: For every log-density parameter $c > 1$, edge-uniformity parameter $d > 2c$, and $\beta < d - 2c$, there exists a negligible-error construction of $(n, m = \Omega(n^c), 2d)$-hypergraph with (β, t) unique-expansion where $t = \Omega(n^{1-\delta})$ and $\delta = 2(c-1)/(d-\beta-2)$.

Theorems 2.8, 2.9 and 2.10 (whose proofs appear in the full version of this paper [12]) provide the first negligible-error constructions of highly-unbalanced constant-degree expanders.

D. Applications

We use our negligible-error construction of unbalanced expanders to obtain the first negligible-error constructions of several useful objects including batch codes, and locally-computable k-wise independent generators, low-bias generators and randomness extractors. These applications follow immediately from our expanders via standard techniques. (See full version of this paper [12] for more details.) Below we briefly describe two non-trivial applications: high-rate low-density parity-check (LDPC) codes, and locally-computable cryptographic pseudorandom generators (PRGs) with polynomial stretch.

1) High-Rate LDPC Codes: LDPC codes [19] (see also [32], [37], [38]) are $[m, k]$ linear error-correcting codes whose $(m-k) \times m$ parity check matrix is *sparse* in the sense that it contains only dm non-zero entries for some *sparsity constant* $d = O(1)$.[8] Any (n, m, d)-hypergraph G defines an $[m, m-n]$-binary LDPC by letting the parity-check matrix be the $n \times m$ incidence matrix of G. The parity-check matrix has md ones, and is therefore sparse when $d = O(1)$. Moreover, it is well known that if, for some $\beta > 0$, the hypergraph G achieves unique-neighbor expansion of (β, γ) then the resulting code has a distance of γ.

Theorem 2.10 leads to the first negligible-error construction of high-rate LDPC code that tolerates polynomially small number of errors.[9] In particular, for every constants $1 < c < d/2$ we get an LDPC with sparsity $2d$ that maps k bits of information into $k + O(k^{1/c})$-bit codeword with a distance of $n^{1-O(c/d)}$.

Sipser and Spielman showed that an LDPC code whose underlying graph has a very good expansion factor (well beyond half the degree) can be efficiently decoded by a linear time decoding algorithm with $O(\log n)$ parallel steps [37]. Unfortunately, the hypergraph given by Theorem 2.10 does not satisfy such a strong expansion property. Nevertheless, we show that our construction can be tweaked in a way that still allows for highly efficient decoding via a variant of the Sipser-Spielman decoder. In particular, we prove the following theorem.

Theorem 2.11: For every constant $c > 1$, integer $d > 10c$ and constant $0.9d < \alpha < d - c$, there exists a negligible-error construction of an $[m, m - 2m^{1/c}]$-LDPC code with sparsity of $2d$ that admits a decoder that runs in quasi-linear time $O(m \log^2 m)$ and $O(\log^2 m)$ parallel steps and corrects up to $\Omega(n^{1-\delta})$ errors where $\delta = (c-1)/(d-\alpha-1)$.

[8] Recall that an $[m, k]$-code is a linear code with codewords of length m and information words of length k, and an $[m, k, \Delta]$-code has, in addition, an absolute distance of Δ.

[9] The status of existing explicit/negligible-error constructions is the same as the status of unbalanced expanders. In fact, any $[m, m-n, t]$ LDPC with sparsity md implies an (n, m)-hypergraph with average rank of d such that any set of t hyperedges has some "odd-expansion" property. We do not have better ways to construct such expanders compared to standard expanders. The situation is similar for all the applications discussed in this paper. That is, unbalanced constant-degree hypergraphs with some expansion property for polynomial-size subsets of hyperedges are also necessary for all these applications, and accordingly so far we had no explicit or negligible-error constructions.

2) Polynomial-Stretch Locally-Computable PRGs: A cryptographic pseudorandom generator stretches a short random n-bit seed into a longer m-bit pseudorandom string that is computationally indistinguishable from a truly random string. We say that a PRG is locally-computable if each of its output bits depends on at most $d = O(1)$ input bits. Locally-computable PRGs were extensively studied in the past two decades. In particular, locally-computable PRGs that *polynomially* stretch their input (i.e., $m = n^c$ for $c > 1$) have shown to have remarkable applications. This includes secure-computation with constant computational overhead [8], [27] and general-purpose obfuscation based on constant-degree multilinear maps (cf. [30], [31]).

Unfortunately, constructing locally-computable PRGs with polynomial-stretch turns out to be a challenging task. Indeed, while there are good constructions of local PRGs with sub-linear stretch $m = n + o(n)$ [10], and even linear stretch $m = n + \Omega(n)$ [4], [6], [11] under standard assumptions, we currently have only partial solutions to the polynomial-stretch regime. In particular, in [4] the first author constructed a locally-computable polynomial-stretch *weak-PRG*. Here *weak* means that the distinguishing advantage ε of any polynomial-time adversary is upper-bounded by some fixed inverse polynomial $1/\mathrm{poly}(n)$, whereas the standard cryptographic definition requires a negligible distinguishing advantage of $n^{-\omega(1)}$. The construction of [4] is based on the one-wayness of random local functions with polynomially-long output length – a variant of Goldreich's one-wayness assumption [20].

We show that our negligible-error construction of expanders can be used to upgrade any weak-PRG into standard PRG while preserving constant locality and polynomial stretch.

Theorem 2.12: For every constants $d \in \mathbb{N}, a > 0$ and $c, c' > 1$ there exists a constant d' for which the following holds. Any ensemble of d-local PRGs that stretches n bits to n^c bits and achieves indistinguishability parameter of $\varepsilon = 1/n^a$ can be converted into an ensemble of d'-local (standard) PRGs that stretches n bits to $n^{c'}$ bits.

The term ensemble here means that, given 1^n, we can sample in polynomial-time a circuit that implements a locally computable function f from n-bits to m bits so that except with negligible probability f is a PRG. This use of ensembles is standard in the context of parallel cryptography and typically has at most a minor effect on the applications.

Combined with the weak-PRG of [4], Theorem 2.12 yields the first construction of local PRG with polynomial stretch based on a one-wayness assumption, resolving an important open question in the theory of parallel cryptography [4], [10], [27], [34]. We mention that there is a second heuristic approach for constructing such pseudorandom generators, due to [27] (see also [13], [34] and the survey [5]). This approach also requires the existence of explicit (or negligible-error) construction of highly-unbalanced constant degree expanders, and one can instantiate it using our constructions as well. In fact, it is known that such expanders are *necessary* for *any* construction of locally-computable PRG with large-stretch [11].[10] Theorem 2.12 shows that, up to some extent, such expanders are also sufficient for this task.

III. Technical Overview

We briefly sketch some of the main techniques.

A. Sampler/Tester for H-free hypergraphs

We present a k-wise independent distribution over (n, m, d) hypergraphs that admits efficient subgraph-testing for hypergraphs of size $t = O(\frac{\log\log\log n}{\log\log\log\log n})$ (as in Theorem 2.1). For simplicity let us focus on the case of directed *graphs* ($d = 2$). Let us further assume that the number of vertices n is prime, and that the number of edges m is an integer power of n, i.e., $m = n^c$ for some integer $c \geq 1$.

We identify every vertex with an element of the field $\mathbb{F} = \mathrm{GF}(n)$, and index the edges with c-tuples of elements of $\mathbb{F}$. We sample the graph by uniformly sampling a pair (A, B) of c-variate polynomials over $\mathbb{F}$ of total degree k. For every tuple $h = (h_1, \ldots, h_c) \in \mathbb{F}^c$, we define the h-th edge to be $(A(h), B(h))$. That is, h leaves the source vertex $A(h)$ and enters the target vertex $B(h)$.

It is not hard to show that every set of k edges are uniformly distributed. (This follows by a simple extension of the well-known fact that random degree-k univariate polynomials are k-wise independent.) We reduce the problem of subgraph testing to the following polynomial satisfiability problem: Check whether a system of $O(t)$ polynomial equations of degree $D = O(k + t)$ and $O(t)$ variables over the field $\mathbb{F}$ has a solution. The latter problem can be solved by an algorithm of Kayal [28, Theorem 6.1.1] in time $\mathrm{poly}(D^{t^{O(t)}} t \log|\mathbb{F}|)$ which is polynomial in n for our choice of parameters.[11]

We describe a simplified version of the reduction for the special case of detecting a directed rectangle (4-cycle). First observe that any sequence of edges indexed by $x_1, x_2, x_3, x_4 \in \mathbb{F}^c$ that form a rectangle must satisfy the system $\mathcal{L}_1$ of equations

$$B(X_1) = A(X_2), \quad B(X_2) = A(X_3),$$
$$B(X_3) = A(X_4), \quad B(X_4) = A(X_1),$$

[10]Indeed, prior works on expander-based cryptography (cf. [3], [4], [8], [13], [20], [27], [30], [31]) assumed, either explicitly or implicitly, the existence of explicit constant-degree unbalanced vertex-expander, or at least that such expanders can be sampled efficiently with negligible error, even though it was unknown how to do so, cf. [27, Remark 5.7]).

[11]On a high-level, Kayal's algorithm decomposes the algebraic closed set X, defined by the input polynomials, into closed sets X_i, such that each X_i is birational to a hypersurface Y_i. If some Y_i has an absolutely irreducible $\mathbb{F}_q$-factor, then by Weil's theorem there exist rational points in Y_i, and by the birational correspondence also in X_i, so the algorithm outputs "yes". Otherwise, if X_i contains a rational point it has to lie on a closed proper subset of X_i. The subset is computed and the algorithm is applied on it recursively. See [28] for full details.

where the formal variables X_1, X_2, X_3 and X_4 correspond to indices of edges and so they take values from $\mathbb{F}^c$. However, a moment of inspection suggests that the system $\mathcal{L}_1$ can be also solved by a 2-cycle: Assign the first edge to X_1 and X_3 and the second edge to X_2 and X_4. We therefore need a mechanism for excluding solutions that assign the same value to different variables. Fortunately, this can be achieved by introducing few more auxiliary variables and few more low-degree equations.

In particular, we add four new variables $Y = (Y_0, Y_1, Y_2, Y_3)$ which take values from $\mathbb{F}^c$ and define a new system $\mathcal{L}_2$ of four equations

$$\sum_{j=0}^{3} Y_j X_1^j = 1, \quad \sum_{j=0}^{3} Y_j X_2^j = 2,$$
$$\sum_{j=0}^{3} Y_j X_3^j = 3, \quad \sum_{j=0}^{3} Y_j X_4^j = 4,$$

where arithmetic is over the extension field $\mathrm{GF}(n^c)$ and 1,2,3,4 represent four distinct constants from this field. Observe that the variables Y define a degree-3 univariate polynomial $P_Y(\cdot)$, and the system is satisfiable if this polynomial evaluates to i over the input X_i. Clearly, any solution to $\mathcal{L}_1$ that assigns non-distinct values to the X variables violates $\mathcal{L}_2$. On the other hand, any solution to $\mathcal{L}_1$ that assigns distinct values to the X variables can be extended by an assignment to Y in a way that satisfies $\mathcal{L}_2$. (Such an assignment can be found via polynomial interpolation). Hence, by combining $\mathcal{L}_2$ with $\mathcal{L}_1$ we get a new system that excludes solutions in which the same edge is being used twice.

To complete the reduction one has to deal with few additional minor technicalities. Firstly, the system $\mathcal{L}_1$ is over the field $\mathbb{F} = \mathrm{GF}(n)$ whereas the second system is over the extension field $\mathrm{GF}(n^c)$. This is solved by projecting down the second system to the base field, and checking the satisfiability of the combined system over $\mathbb{F}$. Secondly, an additional distinctness gadget should be used to further force distinct vertices. (Otherwise, a system for detecting a 4-path will be fooled by a 4-cycle.)

The construction extends to d-uniform hypergraphs in a straightforward way (use d polynomials instead of 2), and to the case of *non-integral* log-density $c = \log_n m$ by working over appropriate extension fields. (See full version of this paper [12] for full details.) Finally, observe that the sampled graph has a *succinct* representation: An edge query can be implemented by evaluating a low-degree polynomial. Moreover, for polylogarithmic k and constant t, the polynomial-satisfiability algorithm can be implemented in polylogarithmic time, and so we get a succinct version of the theorem.

B. Expanding the Expansion: From Small-Sets to Large Sets

Theorem 2.8 converts a zero-error construction of (n, m, d)-hypergraphs G_1 with (α, t)-expansion for small threshold $t = O(\frac{\log\log\log n}{\log\log\log\log n})$ into a negligible-error construction of (α, T)-expander with polynomial threshold of $T = n^{1-\delta}$. The proof is based on two observations.

First, suppose that, in addition to G_1, we are given an (n, m, d)-hypergraph G_2 with the property that any "medium-size" set S of hyperedges $t \leq |S| \leq T$ expands by a factor of α. Then, we can combine G_1 and G_2 into a single $(n, m, 2d)$-hypergraph G by letting the i-th hyperedge of G be the union of the i-th hyperedge of G_1 and G_2. (Both hyperedges should be viewed as d-size multisets over $[n]$.) Every hyperedge set S of size at most T now expands by a factor of α, either due to the expansion of G_1 (when $|S| \leq t$) or due to the expansion of G_2 (when $t < |S| \leq T$). As a result, the expansion factor remains unchanged, but the overall degree doubles.

The second observation is that a random (n, m, d)-hypergraph forms a negligible-error construction of medium-size expanders. Indeed, in our regime of parameters, the probability that a random (n, m, d)-hypergraph contains an s-tuple of hyperedges that violate expansion (touch less than αs vertices) is $n^{-\Omega(s)}$ which is negligible when $s \geq t > \omega(1)$. (The constants in the big-Omega depend on $d, c = \log_n m$ and the exponent of the expansion threshold $\log_n T$.) Indeed, the only reason for which a random (n, m, d)-hypergraph does not qualify as a negligible-error expander is the existence of small non-expanding sets which appear with *noticeable* probability.

Unique-expansion: Unique-expansion is achieved via a similar approach except that the merging procedure is slightly different. As before we merge a pair of (n, m, d)-hypergraphs G_1 and G_2 into a single hypergraph G by defining the i-hyperedge of G to be the union of the i-th hyperedge of G_1 and G_2. However, now we treat the vertices of G_1 and the vertices of G_2 as distinct sets. (E.g., the vertices of G_1 are indexed from 1 to n and the vertices of G_2 are indexed by $n + 1$ to $2n$.) As a result, G is a $(2n, m, 2d)$-hypergraph. It is not hard to verify that if G_1 is a (β, t) unique-expander and G_2 expands by β for sets of size $t < s \leq T$, then G is a (β, T) unique-expander.

Coding perspective: Recall that unbalanced hypergraphs can be viewed as parity-check matrices of error-correcting codes where t-weight codewords correspond to "bad" t-size subgraphs (that violate unique expansion). Using this terminology the above transformation defines a code by taking the intersection of the code G_1 (that has no nontrivial codewords of weight smaller than t) with the code G_2 (that has no codewords of weight $s \in [t, T]$). The efficient decoding algorithm, presented in the full version of this paper [12], further exploits this view, and shows that, in our setting, a noisy codeword of the intersection code

G can be decoded by combining the decoders of G_1 and G_2. In particular, the G_2 decoder "reduces" the number of noisy coordinates from T to (roughly) t, and the G_1 decoder further reduces the noise from t to zero.

C. Local Hardness Amplification: From weak-PRGs to strong-PRGs

Theorem 2.12 converts a weak-PRG $g : \{0,1\}^n \to \{0,1\}^m$ into a standard PRG while preserving polynomial stretch and constant locality. Such hardness amplification theorems are typically based on a direct sum construction: Apply g on ℓ independent copies of the seed and XOR the results. By Yao's XOR-lemma (cf. [22]), if we start with an inverse polynomial indistinguishability, it suffices to take a super-constant number of copies $\ell = \omega(1)$. Unfortunately, this leads to a super-constant growth in the locality. We therefore take a different approach based on randomness extractors.

We generate polynomially-many pseudorandom strings (using independent seeds) and place them as rows of a $k \times m$ matrix. Since the rows are independent and the indistinguishability parameter is a small inverse polynomial, one can guarantee that each column has an almost full pseudo-entropy of $k - 1/\mathrm{poly}(k)$. Finally, we extract the randomness from each column using randomness extractor. This approach was used by [4] (following a more general transformation from [24]) to obtain a linear-stretch local-PRG.

The success of this approach depends, however, on the existence of a suitable locally-computable randomness extractor. The extractor should take a k-bit source with an almost-full entropy of $k - 1/\mathrm{poly}(k)$ and a polynomially-short random seed of length $k^{1-\varepsilon}$ and output an almost-uniform k-bit string with negligible statistical error. The main new observation is that such extractors exist, and a negligible-error construction can be achieved based on negligible-error construction of highly-unbalanced constant-degree expanders. (Similar connections between expanders and locally-computable extractors were established, for a different regime of parameters, in related contexts [4], [11]). See full version of this paper [12] for more details.

References

[1] Noga Alon. Explicit ramsey graphs and orthonormal labelings. *the electronic journal of combinatorics*, 1(1):12, 1994.

[2] Noga Alon and Asaf Nussboim. k-wise independent random graphs. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 813–822. IEEE Computer Society, 2008.

[3] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 152–181. Springer, 2017.

[4] Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM Journal on Computing*, 42(5):2008–2037, 2013.

[5] Benny Applebaum. Cryptographic hardness of random local functions - survey. *Computational Complexity*, 25(3):667–722, 2016.

[6] Benny Applebaum. Exponentially-hard gap-csp and local PRG via local hardcore functions. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 836–847. IEEE Computer Society, 2017.

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

[8] Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In *Annual International Cryptology Conference (CRYPTO)*, pages 223–254. Springer, 2017.

[9] Benny Applebaum, Naama Haramaty, Yuval Ishai, Eyal Kushilevitz, and Vinod Vaikuntanathan. Low-complexity cryptographic hash functions. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 7:1–7:31. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[10] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006.

[11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in NC^0. *Computational Complexity*, 17(1):38–69, 2008.

[12] Benny Applebaum and Eliran Kachlon. Sampling graphs without forbidden subgraphs and almost-explicit unbalanced expanders. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:11, 2019. Full version of this paper.

[13] Benny Applebaum and Shachar Lovett. Algebraic attacks against random local functions and their countermeasures. *SIAM Journal on Computing*, 47(1):52–79, 2018.

[14] Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge. Computational complexity and information asymmetry in financial products. *Commun. ACM*, 54(5):101–107, 2011.

[15] Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 671–680. ACM, 2006.

[16] Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 659–668. ACM, 2002.

[17] Changyan Di, David Proietti, I. Emre Telatar, Thomas J. Richardson, and Rüdiger L. Urbanke. Finite-length analysis of low-density parity-check codes on the binary erasure channel. *IEEE Trans. Information Theory*, 48(6):1570–1579, 2002.

[18] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

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

[20] Oded Goldreich. Candidate one-way functions based on expander graphs. *IACR Cryptology ePrint Archive*, 2000:63, 2000.

[21] Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the implementation of huge random objects. *SIAM J. Comput.*, 39(7):2761–2822, 2010.

[22] Oded Goldreich, Noam Nisan, and Avi Wigderson. On yao's xor-lemma. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(50), 1995.

[23] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009.

[24] Iftach Haitner, Omer Reingold, and Salil Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. *SIAM Journal on Computing*, 42(3):1405–1430, 2013.

[25] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*, pages 134–147. IEEE Computer Society, 1995.

[26] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997.

[27] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 433–442. ACM, 2008.

[28] Neeraj Kayal. *Derandomizing some number-theoretic and algebraic algorithms*. Phd, Indian Institute of Technology, 2007. Available in https://eccc.weizmann.ac.il/resources/pdf/kayal.pdf.

[29] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

[30] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 28–57. Springer, 2016.

[31] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 11–20. IEEE, 2016.

[32] Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, Daniel A Spielman, et al. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on information Theory*, 47(2):585–598, 2001.

[33] Grigorii A Margulis. Explicit constructions of graphs without short cycles and low density codes. *Combinatorica*, 2(1):71–78, 1982.

[34] Elchanan Mossel, Amir Shpilka, and Luca Trevisan. On ε-biased generators in NC^0. *Random Structures & Algorithms*, 29(1):56–81, 2006.

[35] Moni Naor and Asaf Nussboim. Implementing huge sparse random graphs. In Moses Charikar, Klaus Jansen, Omer Reingold, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, volume 4627 of *Lecture Notes in Computer Science*, pages 596–608. Springer, 2007.

[36] Moni Naor, Asaf Nussboim, and Eran Tromer. Efficiently constructible huge graphs that preserve first order properties of random graphs. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 66–85. Springer, 2005.

[37] Michael Sipser and Daniel A Spielman. Expander codes. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 566–576. IEEE, 1994.

[38] Daniel A Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996.