Unique Decoding of Explicit ε-balanced Codes Near the Gilbert–Varshamov Bound

Fernando Granha Jeronimo* Dylan Quintana† Shashank Srivastava‡
Madhur Tulsiani§

The Gilbert–Varshamov bound (non-constructively) establishes the existence of binary codes of distance $1/2 - \varepsilon$ and rate $\Omega(\varepsilon^2)$ (where an upper bound of $O(\varepsilon^2 \log(1/\varepsilon))$ is known). Ta-Shma [STOC 2017] gave an explicit construction of ε-balanced binary codes, where any two distinct codewords are at a distance between $1/2 - \varepsilon/2$ and $1/2 + \varepsilon/2$, achieving a near optimal rate of $\Omega(\varepsilon^{2+\beta})$, where $\beta \to 0$ as $\varepsilon \to 0$.

We develop unique and list decoding algorithms for (a slight modification of) the family of codes constructed by Ta-Shma, in the adversarial error model. We prove the following results for ε-balanced codes with block length N and rate $\Omega(\varepsilon^{2+\beta})$ in this family:

- For all $\varepsilon, \beta > 0$ there are explicit codes which can be uniquely decoded up to an error of half the minimum distance in time $N^{O_{\varepsilon,\beta}(1)}$.
- For any fixed constant β independent of ε, there is an explicit construction of codes which can be uniquely decoded up to an error of half the minimum distance in time $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$.
- For any $\varepsilon > 0$, there are explicit ε-balanced codes with rate $\Omega(\varepsilon^{2+\beta})$ which can be list decoded up to error $1/2 - \varepsilon'$ in time $N^{O_{\varepsilon,\varepsilon',\beta}(1)}$, where $\varepsilon', \beta \to 0$ as $\varepsilon \to 0$.

The starting point of our algorithms is the framework for list decoding direct-sum codes develop in Alev et al. [SODA 2020], which uses the Sum-of-Squares SDP hierarchy. The rates obtained there were quasipolynomial in ε. Here, we show how to overcome the far from optimal rates of this framework obtaining *unique decoding* algorithms for explicit binary codes of near optimal rate. These codes are based on simple modifications of Ta-Shma's construction.

*`University of Chicago`. `granha@uchicago.edu`. Supported in part by NSF grant CCF-1816372.
†`University of Chicago`. `dquintana@uchicago.edu`
‡`TTIC`. `shashanks@ttic.edu`. Supported in part by NSF grant CCF-1816372.
§`TTIC`. `madhurt@ttic.edu`. Supported by NSF grant CCF-1816372.

Contents

1 Introduction

Binary error correcting codes have pervasive applications [Gur10, GRS19] and yet we are far from understanding some of their basic properties [Gur09]. For instance, until very recently no explicit binary code achieving distance $1/2 - \varepsilon/2$ with rate near $\Omega(\varepsilon^2)$ was known, even though the existence of such codes was (non-constructively) established long ago [Gil52, Var57] in what is now referred as the Gilbert–Varshamov (GV) bound. On the impossibility side, a rate upper bound of $O(\varepsilon^2 \log(1/\varepsilon))$ is known for binary codes of distance $1/2 - \varepsilon/2$ (e.g., [Del75, MRRW77, NS09]).

In a breakthrough result [TS17], Ta-Shma gave an explicit construction of binary codes achieving nearly optimal distance versus rate trade-off, namely, binary codes of distance $1/2 - \varepsilon/2$ with rate $\Omega(\varepsilon^{2+\beta})$ where β vanishes as ε vanishes [1]. Actually, Ta-Shma obtained ε-balanced binary linear codes, that is, linear binary codes with the additional property that non-zero codewords have Hamming weight bounded not only below by $1/2 - \varepsilon/2$ but also above by $1/2 + \varepsilon/2$, and this is a fundamental property in the study of pseudo-randomness [NN90, AGHP92].

While the codes constructed by Ta-Shma are explicit, they were not known to admit efficient decoding algorithms, while such results are known for codes with smaller rates. In particular, an explicit binary code due to Guruswami and Rudra [GR06] is known to be even list decodable at an error radius $1/2 - \varepsilon$ with rate $\Omega(\varepsilon^3)$. We consider the following question:

Do explicit binary codes near the GV bound admit an efficient decoding algorithm?

Here, we answer this question in the affirmative by providing an efficient [2] unique decoding algorithm for (essentially) Ta-Shma's code construction, which we refer as Ta-Shma codes. More precisely, by building on Ta-Shma's construction and using our unique decoding algorithm we have the following result.

Theorem 1.1 (Unique Decoding)**.** *For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$ where $\beta = O(1/(\log_2(1/\varepsilon))^{1/6})$, and

(iii) a unique decoding algorithm with running time $N^{O_{\varepsilon,\beta}(1)}$.

Furthermore, if instead we take $\beta > 0$ to be an arbitrary constant, the running time becomes $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$ (fixed polynomial time).

We can also perform "gentle" list decoding in the following sense (note that this partially implies Theorem 1.1).

Theorem 1.2 (Gentle List Decoding)**.** *For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

[1] In fact, Ta-Shma obtained $\beta = \beta(\varepsilon) = \Theta(((\log\log 1/\varepsilon)/\log 1/\varepsilon)^{1/3})$ and thus $\lim_{\varepsilon\to 0} \beta(\varepsilon) = 0$.

[2] By "efficient", we mean polynomial time. Given the fundamental nature of the problem of decoding nearly optimal binary codes, it is an interesting open problem to make these techniques viable in practice.

(i) distance at least $1/2 - \varepsilon/2$ *(actually* ε*-balanced),*

(ii) rate $\Omega(\varepsilon^{2+\beta})$ *where* $\beta = O(1/(\log_2(1/\varepsilon))^{1/6})$*, and*

(iii) a list decoding algorithm that decodes within radius $1/2 - 2^{-\Theta((\log_2(1/\varepsilon))^{1/6})}$ *in time* $N^{O_{\varepsilon,\beta}(1)}$*.*

We observe that the exponent in the running time $N^{O_{\varepsilon,\beta}(1)}$ appearing in Theorem 1.1 and Theorem 1.2 depends on ε. This dependence is no worse than $O(\log\log(1/\varepsilon))$, and if $\beta > 0$ is taken to be an arbitrarily constant (independent of ε), the running time becomes $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$. Avoiding this dependence in the exponent when $\beta = \beta(\varepsilon)$ is an interesting open problem. Furthermore, obtaining a list decoding radius of $1/2 - \varepsilon/2$ in Theorem 1.2 with the same rate (or even $\Omega(\varepsilon^2)$) is another very interesting open problem and related to a central open question in the adversarial error regime [Gur09].

Direct sum codes. Our work can be viewed within the broader context of developing algorithms for the decoding of direct sum codes. Given a (say linear) code $\mathcal{C} \subseteq \mathbb{F}_2^n$ and a collection of tuples $W \subseteq [n]^t$, the code $\operatorname{dsum}_W(\mathcal{C})$ with block length $|W|$ is defined as

$$\operatorname{dsum}_W(\mathcal{C}) = \{(z_{w_1} + z_{w_2} + \cdots + z_{w_t})_{w \in W} \mid z \in \mathcal{C}\}.$$

The direct sum operation has been used for several applications in coding and complexity theory [ABN+92, IW97, GI01, IKW09, DS14, DDG+15, Cha16, DK17, Aro02]. It is easy to see that if $\mathcal{C}$ is ε_0-balanced for a constant ε_0, then for any $\varepsilon > 0$, choosing W to be a random collection of tuples of size $O(n/\varepsilon^2)$ results in $\operatorname{dsum}_W(\mathcal{C})$ being an ε-balanced code. The challenge in trying to construct good codes using this approach is to find explicit constructions of (sparse) collections W which are "pseudorandom" enough to yield a similar distance amplification as above. On the other hand, the challenge in decoding such codes is to identify notions of "structure" in such collections W, which can be exploited by decoding algorithms.

In Ta-Shma's construction [TS17], such a pseudorandom collection W was constructed by considering an expanding graph G over the vertex set $[n]$, and generating t-tuples using sufficiently long walks of length $t-1$ over the so-called s-wide replacement product of G with another (small) expanding graph H. Roughly speaking, this graph product is a generalization of the celebrated zig-zag product [RVW00] but with s different steps of the zig-zag product instead of a single one. Ta-Shma's construction can also be viewed as a clever way of selecting a *sub-collection* of all walks in G, which refines an earlier construction suggested by Rozenman and Wigderson [Bog12] (and also analyzed by Ta-Shma) using *all* walks of length $t-1$.

Identifying structures to facilitate decoding. For the closely related direct product construction (where the entry corresponding to $w \in W$ is the entire t-tuple $(z_{w_1}, \ldots, z_{w_t})$) which amplifies distance but increases the alphabet size, it was proved by Alon et al. [ABN+92] that the resulting code admits a unique decoding algorithm if the incidence graph corresponding to the collection W is a good sampler. Very recently, it was proved by Dinur et al. [DHK+19] that such a direct product construction admits list decoding if the incidence graph is a "double sampler". However, these results do not apply to direct sum, which achieves distance amplification while preserving the alphabet size.

For the case of direct sum codes, the decoding task can be phrased as a maximum t-XOR problem with the additional constraint that the solution must lie in $\mathcal{C}$. More precisely, given $\tilde{y} \in \mathbb{F}_2^W$ within the unique decoding radius of $\mathrm{dsum}_W(\mathcal{C})$, we consider the following optimization problem

$$\operatorname*{argmin}_{z \in \mathcal{C}} \; \Delta(\tilde{y}, \mathrm{dsum}_W(z)),$$

where $\Delta(\cdot, \cdot)$ is the (normalized) Hamming distance. While maximum t-XOR is in general hard to solve to even any non-trivial degree of approximation [Hås97], previous work by the authors [AJQ+20] identified a structural condition on W called "splittability" under which the above constraint satisfaction problem can be solved (approximately) resulting in efficient unique and list decoding algorithms. However, by itself the splittability condition is too crude to be applicable to codes such as the ones in Ta-Shma's construction. The requirements it places on the expansion of G are too strong and the framework in [AJQ+20] is only able to obtain algorithms for direct sum codes with rate $2^{-(\log(1/\varepsilon))^2} \ll \varepsilon^{2+\beta}$.

The conceptual contribution of this work can be viewed as identifying a different recursive structure in direct sums generated by expander walks, which allows us to view the construction as giving a sequence of codes $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_\ell$. Here, $\mathcal{C}_0$ is the starting code $\mathcal{C}$ and $\mathcal{C}_\ell$ is the final desired code, and each element in the sequence can be viewed as being obtained via a direct sum operation on the preceding code. Instead of considering a "one-shot" decoding task of finding an element of $\mathcal{C}_0$, this facilitates an iterative approach where at each step we reduce the task of decoding the code $\mathcal{C}_i$ to decoding for $\mathcal{C}_{i-1}$, using the above framework from [AJQ+20]. Such an iterative approach with a sequence of codes was also used (in a very different setting) in a work of Guruswami and Indyk [GI03] constructing codes over a large alphabet which are list decodable in linear time via spectral algorithms.

Another simple and well-known (see e.g., [GI04]) observation, which is very helpful in our setting, is the use of list decoding algorithms for unique decoding. For a code with distance $1/2 - \varepsilon/2$, unique decoding can be obtained by list decoding at a much smaller error radius of (say) $1/2 - 1/8$. This permits a much more efficient application of the framework from [AJQ+20], with a milder dependence on the expansion of the graphs G and H in Ta-Shma's construction, resulting in higher rates. We give a more detailed overview of our approach in Section 3.

Known results for random ensembles. While the focus in this work is on explicit constructions, there are several known (non-explicit) constructions of random ensembles of binary codes near or achieving the Gilbert–Varshamov bound (e.g., Table 1). Although it is usually straightforward to ensure the desired rate in such constructions, the distance only holds with high probability. Given a sample code from such ensembles, certifying the minimum distance is usually not known to be polynomial time in the block length. Derandomizing such constructions is also a possible avenue for obtaining optimal codes, although such results remain elusive to this date (to the best of our knowledge).

One of the simplest constructions is that of random binary linear codes in which the generator matrix is sampled uniformly. This random ensemble achieves the GV bound with high probability, but its decoding is believed to be computationally hard [MMT11].

Much progress has been made on binary codes by using results for larger alphabet codes [Gur09]. Codes over non-binary alphabets with optimal (or nearly optimal) parameters are available [vL99, Sti08, GR06] and thanks to this availability a popular approach to constructing binary codes has been to concatenate such large alphabet codes with binary ones. Thommesen [Tho83] showed that by concatenating Reed–Solomon (RS) codes with random binary codes (one random binary code for each position of the outer RS code) it is possible to achieve the GV bound. Note that Thommesen codes arise from a more structured ensemble than random binary linear codes. This additional structure enabled Guruswami and Indyk [GI04] to obtain efficient decoding algorithms for the non-explicit Thommesen codes (whose minimum distance is not known to admit efficient certification). This kind of concatenation starting from a large alphabet code and using random binary codes, which we refer as Thommesen-like, has been an important technique in tackling binary code constructions with a variety of properties near or at the GV bound. An important drawback in several such Thommesen-like code constructions is that they end up being non-explicit (unless efficient derandomization or brute-force is viable).

Using a Thommesen-like construction, Gopi et al. [GKO$^+$17] showed non-explicit constructions of locally testable and locally correctable binary codes approaching the GV bound. More recently, again with a Thommesen-like construction, Hemenway et al. [HRW17] obtained non-explicit near linear time unique decodable codes at the GV bound improving the running time of Guruswami and Indyk [GI04] (and also the decoding rates). We summarize the results discussed so far in Table 1.

Binary Code Results near the Gilbert–Varshamov bound						
Who?	**Construction**	**GV**	**Explicit**	**Concatenated**	**Decoding**	**Local**
[Gil52, Var57]	existential	yes	no	no	no	n/a
[Tho83]	Reed–Solomon + random binary	yes	no	yes	no	n/a
[GI04]	Thommesen [Tho83]	yes	no	yes	unique decoding	n/a
[GKO$^+$17]	Thommesen-like	yes	no	yes	unique decoding	LTC/LCC
[HRW17]	Thommesen-like	yes	no	yes	near linear time unique decoding	n/a
[TS17]	Expander-based	$\Omega(\varepsilon^{2+\beta})$	yes	no	no	n/a
this paper	Ta-Shma [TS17]	$\Omega(\varepsilon^{2+\beta})$	yes	no	gentle list decoding	n/a

Table 1: GV bound related results for binary codes.

There are also non-explicit constructions known to achieve list decoding capacity [GR08, MRRZ$^+$19] (being concatenated or LDPC/Gallager [Gal62] is not an obstruction to achieve capacity). Contrary to the other results in this subsection, Guruswami and Rudra [Gur05, GR06, Gur09], also using a Thommesen-like construction, obtained explicit codes that are efficiently list decodable from radius $1/2 - \varepsilon$ with rate $\Omega(\varepsilon^3)$. This was done by concatenating the so-called folded Reed–Solomon codes with a derandomization of a binary ensemble of random codes.

Results for non-adversarial error models. All the results mentioned above are for the adversarial error model of Hamming [Ham50, Gur10]. In the setting of random corruptions (Shannon model), the situation seems to be better understood thanks to the seminal result on explicit polar codes of Arikan [Ari09]. More recently, in another breakthrough Guruswami et al. [GRY19] showed that polar codes can achieve almost linear time decoding with near optimal convergence to capacity for the binary symmetric channel. This result gives an explicit code construction achieving parameter trade-offs similar to Shannon's randomized construction [Sha48] while also admitting very efficient encoding and decoding. Explicit capacity-achieving constructions are also known for bounded memory channels [SKS19] which restrict the power of the adversary and thus interpolate between the Shannon and Hamming models.

2 Preliminaries and Notation

2.1 Codes

We briefly recall some standard code terminology. Given $z, z' \in \mathbb{F}_2^n$, recall that the relative Hamming distance between z and z' is $\Delta(z, z') := |\{i \mid z_i \neq z'_i\}| / n$. A binary code is any subset $\mathcal{C} \subseteq \mathbb{F}_2^n$. The distance of $\mathcal{C}$ is defined as $\Delta(\mathcal{C}) := \min_{z \neq z'} \Delta(z, z')$ where $z, z' \in \mathcal{C}$. We say that $\mathcal{C}$ is a linear code if $\mathcal{C}$ is a linear subspace of $\mathbb{F}_2^n$. The rate of $\mathcal{C}$ is $\log_2(|\mathcal{C}|)/n$.

Instead of discussing the distance of a binary code, it will often be more natural to phrase results in terms of its bias.

Definition 2.1 (Bias). *The* bias *of a word $z \in \mathbb{F}_2^n$ is defined as* $\operatorname{bias}(z) := \left|\mathbb{E}_{i \in [n]} (-1)^{z_i}\right|$. *The bias of a code $\mathcal{C}$ is the maximum bias of any non-zero codeword in $\mathcal{C}$.*

Definition 2.2 (ε-balanced Code). *A code $\mathcal{C}$ with* $\operatorname{bias}(\mathcal{C}) \leq \varepsilon$ *is known as an* ε-balanced code*.*

2.2 Direct Sum Lifts

Starting from a code $\mathcal{C} \subseteq \mathbb{F}_2^n$, we amplify its distance by considering the *direct sum lifting* operation based on a collection $W(k) \subseteq [n]^k$. The direct sum lifting maps each codeword of $\mathcal{C}$ to a new word in $\mathbb{F}_2^{|W(k)|}$ by taking the k-XOR of its entries on each element of $W(k)$.

Definition 2.3 (Direct Sum Lifting). *Let $W(k) \subseteq [n]^k$. For $z \in \mathbb{F}_2^n$, we define the* direct sum lifting *as* $\operatorname{dsum}_{W(k)}(z) = y$ *such that $y_{\mathfrak{s}} = \sum_{i \in \mathfrak{s}} z_i$ for all $\mathfrak{s} \in W(k)$. The direct sum lifting of a code $\mathcal{C} \subseteq \mathbb{F}_2^n$ is*

$$\operatorname{dsum}_{W(k)}(\mathcal{C}) = \{\operatorname{dsum}_{W(k)}(z) \mid z \in \mathcal{C}\}.$$

We will omit $W(k)$ from this notation when it is clear from context.

Remark 2.4. *We will be concerned with collections $W(k) \subseteq [n]^k$ arising from length-$(k-1)$ walks on expanding structures (mostly on the s-wide replacement product of two expander graphs).*

We will be interested in cases where the direct sum lifting reduces the bias of the base code; in [TS17], structures with such a property are called *parity samplers*, as they emulate the reduction in bias that occurs by taking the parity of random samples.

Definition 2.5 (Parity Sampler). *A collection $W(k) \subseteq [n]^k$ is called an $(\varepsilon_0, \varepsilon)$-*parity sampler *if for all $z \in \mathbb{F}_2^n$ with* $\operatorname{bias}(z) \leq \varepsilon_0$, *we have* $\operatorname{bias}(\operatorname{dsum}_{W(k)}(z)) \leq \varepsilon$.

2.3 Linear Algebra Conventions

All vectors considered in this paper are taken to be column vectors, and are multiplied on the left with any matrices or operators acting on them. Consequently, given an indexed sequence of operators $\mathsf{G}_{k_1}, \ldots, \mathsf{G}_{k_2}$ (with $k_1 \leq k_2$) corresponding to steps k_1 through k_2 of a walk, we expand the product $\prod_{i=k_1}^{k_2} G_i$ as

$$\prod_{i=k_1}^{k_2} G_i \;:=\; G_{k_2} \cdots G_{k_1}\,.$$

Unless otherwise stated, all inner products for vectors in coordinate spaces are taken to be with respect to the (uniform) probability measure on the coordinates. Similarly, all inner products for functions are taken to be with respect to the uniform measure on the inputs. All operators considered in this paper are normalized to have singular values at most 1.

3 Proof Overview

The starting point for our work is the framework developed in [AJQ$^+$20] for decoding direct sum codes, obtained by starting from a code $\mathcal{C} \subseteq \mathbb{F}_2^n$ and considering all parities corresponding to a set of t-tuples $W(t) \subseteq [n]^t$. Ta-Shma's near optimal ε-balanced codes are constructed by starting from a code with constant rate and constant distance and considering such a direct sum lifting. The set of tuples $W(t)$ in his construction corresponds to a set of walks of length $t-1$ on the s-wide replacement product of an expanding graph G with vertex set $[n]$ and a smaller expanding graph H. The s-wide replacement product can be thought of here as a way of constructing a much smaller pseudorandom subset of the set of all walks of length $t-1$ on G, which yields a similar distance amplification for the lifted code.

The simplified construction with expander walks. While we analyze Ta-Shma's construction later in the paper, it is instructive to first consider a $W(t)$ simply consisting of all walks of length $t-1$ on an expander. This construction, based on a suggestion of Rozenman and Wigderson [Bog12], was also analyzed by Ta-Shma [TS17] and can be used to obtain ε-balanced codes with rate $\Omega(\varepsilon^{4+o(1)})$. It helps to illustrate many of the conceptual ideas involved in our proof, while avoiding some technical issues.

Let G be a d-regular expanding graph with vertex set $[n]$ and the (normalized) second singular value of the adjacency operator A_G being λ. Let $W(t) \subseteq [n]^t$ denote the set of t-tuples corresponding to all walks of length $t-1$, with $N = |W(t)| = n \cdot d^{t-1}$. Ta-Shma proves that for all $z \in \mathbb{F}_2^n$, $W(t)$ satisfies

$$\operatorname{bias}(z) \le \varepsilon_0 \quad \Rightarrow \quad \operatorname{bias}(\operatorname{dsum}_{W(t)}(z)) \le (\varepsilon_0 + 2\lambda)^{\lfloor (t-1)/2 \rfloor},$$

i.e., $W(t)$ is an $(\varepsilon_0, \varepsilon)$-*parity sampler* for $\varepsilon = (\varepsilon_0 + 2\lambda)^{\lfloor (t-1)/2 \rfloor}$. Choosing $\varepsilon_0 = 0.1$ and $\lambda = 0.05$ (say), we can choose $d = O(1)$ and obtain the ε-balanced code $\mathcal{C}' = \operatorname{dsum}_{W(t)}(\mathcal{C})$ with rate $d^{-(t-1)} = \varepsilon^{O(1)}$ (although the right constants matter a lot for optimal rates).

Decoding as constraint satisfaction. The starting point for our work is the framework in [AJQ$^+$20] which views the task of decoding $\tilde{y}$ with $\Delta(\mathcal{C}', \tilde{y}) < (1-\varepsilon)/4 - \delta$ (where the distance of $\mathcal{C}'$ is $(1-\varepsilon)/2$) as an instance of the MAX t-XOR problem (see Fig. 1). The goal is to find

$$\operatorname*{argmin}_{z \in \mathcal{C}} \Delta\left(\operatorname{dsum}_{W(t)}(z), \tilde{y}\right),$$

which can be rephrased as

$$\operatorname*{argmax}_{z \in \mathcal{C}} \mathop{\mathbb{E}}_{w=(i_1,\ldots,i_t)\in W(t)} \left[\mathbb{1}_{\{z_{i_1} + \cdots + z_{i_t} = \tilde{y}_w\}}\right].$$

It is possible to ignore the condition that $z \in \mathcal{C}$ if the collection $W(t)$ is a slightly stronger parity sampler. For any solution $\tilde{z} \in \mathbb{F}_2^n$ (not necessarily in $\mathcal{C}$) such that

$$\Delta(\mathrm{dsum}_{W(t)}(\tilde{z}), \tilde{y}) < \frac{1-\varepsilon}{4} + \delta,$$

we have

$$\Delta(\mathrm{dsum}_{W(t)}(\tilde{z}), \mathrm{dsum}_{W(t)}(z)) < \frac{1-\varepsilon}{2}$$

by the triangle inequality, and thus $\mathrm{bias}(\mathrm{dsum}_{W(t)}(z - \tilde{z})) > \varepsilon$. If $W(t)$ is not just an $(\varepsilon_0, \varepsilon)$-parity sampler, but in fact a $((1+\varepsilon_0)/2, \varepsilon)$-parity sampler, this would imply $\mathrm{bias}(z - \tilde{z}) > (1+\varepsilon_0)/2$. Thus, $\Delta(z, \tilde{z}) < (1-\varepsilon_0)/4$ (or $\Delta(z, \overline{\tilde{z}}) < (1-\varepsilon_0)/4$) and we can use a unique decoding algorithm for $\mathcal{C}$ to find z given $\tilde{z}$.

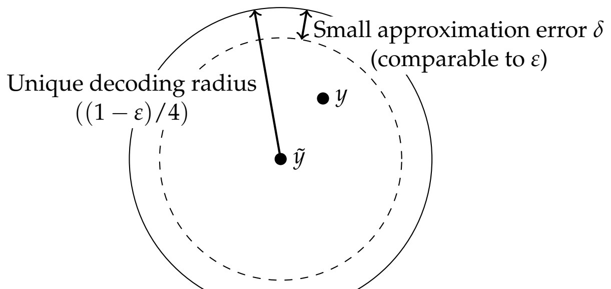
Small approximation error δ
(comparable to ε)
Unique decoding radius
$((1-\varepsilon)/4)$
y
$\tilde{y}$

Figure 1: Unique decoding ball along with error from approximation.

The task of finding such a $z \in \mathcal{C}$ boils down to finding a solution $\tilde{z} \in \mathbb{F}_2^n$ to a MAX t-XOR instance, up to a an additive loss of $O(\delta)$ in the fraction of constraints satisfied by the optimal solution. While this is hard to do in general [Hås01, Gri01], [AJQ+20] (building on [AJT19]) show that this can be done if the instance satisfies a special property called *splittability*. To define this, we let $W[t_1, t_2] \subset [n]^{t_2 - t_1 + 1}$ denote the collection of $(t_2 - t_1 + 1)$-tuples obtained by considering the indices between t_1 and t_2 for all tuples in $W(t)$. We also assume that all $w \in W[t_1, t_2]$ can be extended to the same number of tuples in $W(t)$ (which is true for walks).

Definition 3.1 (Splittability (informal)). *A collection $W(t) \subseteq [n]^t$ is said to be τ-splittable, if $t = 1$ (base case) or there exists $t' \in [t-1]$ such that:*

1. *The matrix $\mathsf{S} \in \mathbb{R}^{W[1,t'] \times W[t'+1,t]}$ defined by $\mathsf{S}(w, w') = \mathbb{1}_{\{ww' \in W\}}$ has normalized second singular value at most τ (where ww' denotes the concatenated tuple).*

2. *The collections $W[1, t']$ and $W[t'+1, t]$ are τ-splittable.*

For example, considering walks in G of length 3 ($t = 4$) and $t' = 2$, we get that $W[1,2] = W[3,4] = E$, the set of oriented edges in G. Also $\mathsf{S}(w, w') = 1$ if and only if the second vertex of w and first vertex of w' are adjacent in G. Thus, up to permutation of rows and columns, we can write the normalized version of S as $\mathsf{A}_G \otimes \mathsf{J}_d / d$ where A_G is normalized adjacency matrix of G and J_d denotes the $d \times d$ matrix of 1s. Hence such a $W(t)$ satisfies $\sigma_2(\mathsf{S}) \le \tau$ with $\tau = \sigma_2(\mathsf{A}_G)$, and a similar proof works for walks of all lengths.

The framework in [AJQ+20] and [AJT19] gives that if $W(t)$ is τ-splittable for $\tau = (\delta/2^t)^{O(1)}$, then the above instance of MAX t-XOR can be solved to additive error $O(\delta)$

using the Sum-of-Squares (SOS) SDP hierarchy. Broadly speaking, splittability allows one to (recursively) treat instances as expanding instances of problems with two "tuple variables" in each constraint, which can then be analyzed using known algorithms for 2-CSPs [BRS11, GS11] in the SOS hierarchy. Combined with parity sampling, this yields a unique decoding algorithm. Crucially, this framework can also be extended to perform *list decoding*[3] up to a radius of $1/2 - \sqrt{\varepsilon} - \delta$ under a similar condition on τ, which will be very useful for our application.

While the above can yield decoding algorithms for suitably expanding G, the requirement on τ (and hence on λ) makes the rate much worse. We need $\delta = O(\varepsilon)$ (for unique decoding) and $t = O(\log(1/\varepsilon))$ (for parity sampling), which requires $\lambda = \varepsilon^{\Omega(1)}$, yielding only a quasipolynomial rate for the code (recall that we could take $\lambda = O(1)$ earlier yielding polynomial rates).

Unique decoding: weakening the error requirement. We first observe that it is possible to get rid of the dependence $\delta = O(\varepsilon)$ above by using the *list decoding* algorithm for unique decoding. It suffices to take $\delta = 0.1$ and return the closest element from the the list of all codewords up to an error radius $1/2 - \sqrt{\varepsilon} - 0.1$, if we are promised that $\Delta(\tilde{y}, \mathcal{C})$ is within the unique decoding radius (see Fig. 2). However, this alone does not improve the rate as we still need the splittability (and hence λ) to be $2^{-\Omega(t)}$ with $t = O(\log(1/\varepsilon))$.

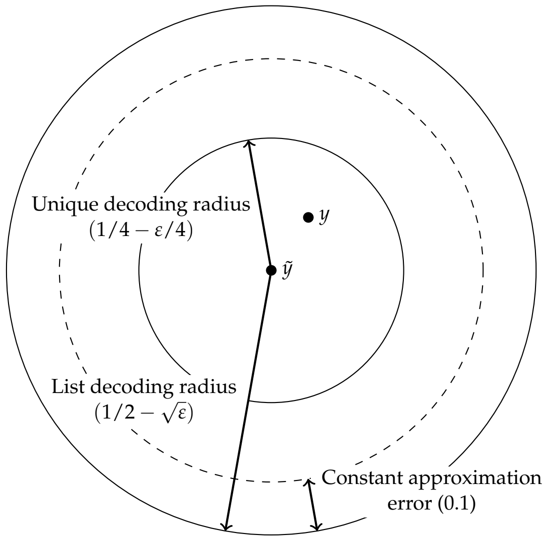

Figure 2: Unique decoding and list decoding balls along with error from approximation. Note that the list decoding ball contains the unique decoding ball even after allowing for a relatively large amount of error.

Code cascades: handling the dependence on walk length. To avoid the dependence of the expansion on the length $t - 1$ of the walk (and hence on ε), we avoid the "one-shot" decoding above, and instead consider a sequence of intermediate codes between $\mathcal{C}$

[3] While unique decoding can be thought of as recovering a single solution to a constraint satisfaction problem, the goal in the list decoding setting can be thought of as obtaining a "sufficiently rich" set of solutions which forms a good cover. This is achieved in the framework by adding an entropic term to the semidefinite program, which ensures that the SDP solution satisfies such a covering property.

and $\mathcal{C}'$. Consider the case when $t = k^2$, and instead of computing t-wise sums of bits in each $z \in \mathbb{F}_2^n$, we first compute k-wise sums according to walks of length $k-1$ on G, and then a k-wise sum of these values. In fact, the second sum can also be thought of as arising from a length $k-1$ walk on a different graph, with vertices corresponding to (directed) walks with k vertices in G, and edges connecting w and w' when the last vertex of w is connected to the first one in w' (this is similar to the matrix considered for defining splittability). We can thus think of a sequence of codes $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ with $\mathcal{C}_0 = \mathcal{C}$ and $\mathcal{C}_2 = \mathcal{C}'$, and both $\mathcal{C}_1$ and $\mathcal{C}_2$ being k-wise direct sums. More generally, when $t = k^\ell$ for an appropriate constant k we can think of a sequence $\mathcal{C} = \mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_\ell = \mathcal{C}'$, where each is an k-wise direct sum of the previous code, obtained via walks of length $k-1$ (hence k vertices) in an appropriate graph. We refer to such sequences (defined formally in Section 5) as *code cascades* (see Fig. 3).

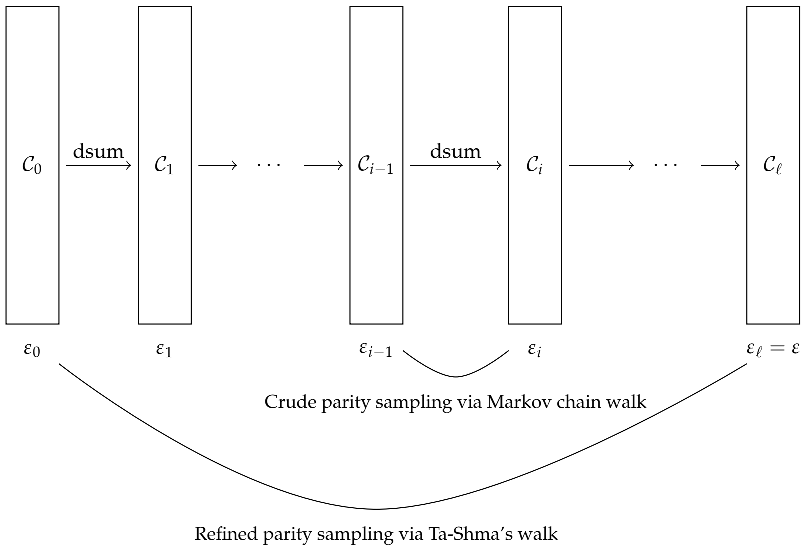

Figure 3: Code cascading.

Instead of applying the decoding framework above to directly reduce the decoding of a corrupted codeword from $\mathcal{C}'$ to the unique decoding problem in $\mathcal{C}$, we apply it at each level of a cascade, reducing the unique decoding problem in $\mathcal{C}_i$ to that in $\mathcal{C}_{i-1}$. If the direct sum at each level of the cascade is an (η_0, η)-parity sampler, the list decoding algorithm at radius $1/2 - \sqrt{\eta}$ suffices for unique decoding even if η is a (sufficiently small) constant independent of ε. This implies that we can take k to be a (suitably large) constant. This also allows the splittability (and hence λ) to be $2^{-O(k)} = \Omega(1)$, yielding polynomial rates. We present the reduction using cascades in Section 6 and the parameter choices in Section 8. The specific versions of the list decoding results from [AJQ+20] needed here are instantiated in Section 9.

While the above allows for polynomial rate, the *running time* of the algorithm is still exponential in the number of levels ℓ (which is $O(\log t) = O(\log\log(1/\varepsilon))$) since the list decoding for each level potentially produces a list of size $\mathrm{poly}(n)$, and recursively calls the decoding algorithm for the previous level on each element of the list. We obtain a fixed polynomial time algorithm by "pruning" the list at each level of the cascade before invoking the decoding algorithm for the previous level, while only slightly increasing the parity sampling requirements. The details are contained in Section 6.

Working with Ta-Shma's construction. Finally, to obtain near-optimal rates, we need to work with with Ta-Shma's construction, where the set of tuples $W(t) \subseteq [n]^t$ corresponds to walks arising from an s-wide replacement product of G with another expanding graph H. One issue that arises is that the collection of walks $W(t)$ as defined in [TS17] does not satisfy the important splittability condition required by our algorithms. However, this turns out to be easily fixable by modifying each step in Ta-Shma's construction to be exactly according to the zig-zag product of Reingold, Vadhan and Wigderson [RVW00]. We present Ta-Shma's construction and this modification in Section 4.

We also verify that the tuples given by Ta-Shma's construction satisfy the conditions for applying the list decoding framework, in Section 7. While the sketch above stated this in terms of splittability, the results in [AJQ+20] are in terms of a more technical condition called *tensoriality*. We show in Section 7 that this is indeed implied by splittability, and also prove splittability for (the modified version of) Ta-Shma's construction.

4 Ta-Shma's Construction: A Summary and Some Tweaks

In this section, we first discuss the s-wide replacement product that is central to Ta-Shma's construction of optimal ε-balanced codes, and then we describe the construction itself (we refer the reader to [TS17] for formal details beyond those we actually need here).

As mentioned before, we will also need to modify Ta-Shma's construction [TS17] a little to get *splittability* which is a notion of expansion of a collection $W(k) \subseteq [n]^k$ (and it is formally defined in Definition 7.9). The reason for this simple modification is that this *splittability* property is required by the list decoding framework. Note that we are not improving the Ta-Shma code parameters; in fact, we need to argue why with this modification we can still achieve Ta-Shma's parameters. Fortunately, this modification is simple enough that we will be able to essentially reuse Ta-Shma's original analysis. In Section 4.3, we will also have the opportunity to discuss, at an informal level, the intuition behind some parameter trade-offs in Ta-Shma codes which should provide enough motivation when we instantiate these codes in Section 8.

4.1 The s-wide Replacement Product

Ta-Shma's code construction is based on the so-called s-wide replacement product [TS17]. This is a derandomization of random walks on a graph G that will be defined via a product operation of G with another graph H (see Definition 4.2 for a formal definition). We will refer to G as the *outer* graph and H as the *inner* graph in this construction.

Let G be a d_1-regular graph on vertex set $[n]$ and H be a d_2-regular graph on vertex set $[d_1]^s$, where s is any positive integer. Suppose the neighbors of each vertex of G are labeled 1, 2, …, d_1. For $v \in V(G)$, let $v_G[j]$ be the j-th neighbor of v. The s-wide replacement product is defined by replacing each vertex of G with a copy of H, called a "cloud". While the edges within each cloud are determined by H, the edges between clouds are based on the edges of G, which we will define via operators $\mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_{s-1}$. The i-th operator G_i specifies one inter-cloud edge for each vertex $(v, (a_0, \ldots, a_{s-1})) \in V(G) \times V(H)$, which goes to the cloud whose G component is $v_G[a_i]$, the neighbor of v in G indexed by the i-th coordinate of the H component. (We will resolve the question of what happens to the H

component after taking such a step momentarily.)

Walks on the s-wide replacement product consist of steps with two different parts: an intra-cloud part followed by an inter-cloud part. All of the intra-cloud substeps simply move to a random neighbor in the current cloud, which corresponds to applying the operator $\mathsf{I} \otimes \mathsf{A}_H$, where A_H is the normalized adjacency matrix of H. The inter-cloud substeps are all deterministic, with the first moving according to G_0, the second according to G_1, and so on, returning to G_0 for step number $s+1$. The operator for such a walk taking $t-1$ steps on the s-wide replacement product is

$$\prod_{i=0}^{t-2} \mathsf{G}_{i \bmod s}(\mathsf{I} \otimes \mathsf{A}_H).$$

Observe that a walk on the s-wide replacement product yields a walk on the outer graph G by recording the G component after each step of the walk. The number of $(t-1)$-step walks on the s-wide replacement product is

$$|V(G)| \cdot |V(H)| \cdot d_2^{t-1} = n \cdot d_1^s \cdot d_2^{t-1},$$

since a walk is completely determined by its intra-cloud steps. If d_2 is much smaller than d_1 and t is large compared to s, this is less than nd_1^{t-1}, the number of $(t-1)$-step walks on G itself. Thus the s-wide replacement product will be used to simulate random walks on G while requiring a reduced amount of randomness (of course this simulation is only possible under special conditions, namely, when we are uniformly distributed on each cloud).

To formally define the s-wide replacement product, we must consider the labeling of neighbors in G more carefully.

Definition 4.1 (Rotation Map)**.** *Suppose G is a d_1-regular graph on $[n]$. For each $v \in [n]$ and $j \in [d_1]$, let $v_G[j]$ be the j-th neighbor of v in G. Based on the indexing of the neighbors of each vertex, we define the rotation map* [4] $\mathrm{rot}_G \colon [n] \times [d_1] \to [n] \times [d_1]$ *such that for every $(v, j) \in [n] \times [d_1]$,*

$$\mathrm{rot}_G((v, j)) = (v', j') \Leftrightarrow v_G[j] = v' \text{ and } v'_G[j'] = v.$$

Furthermore, if there exists a bijection $\varphi \colon [d_1] \to [d_1]$ such that for every $(v, j) \in [n] \times [d_1]$,

$$\mathrm{rot}_G((v, j)) = (v_G[j], \varphi(j)),$$

then we call rot_G locally invertible.

If G has a locally invertible rotation map, the cloud label after applying the rotation map only depends on the current cloud label, not the vertex of G. In the s-wide replacement product, this corresponds to the H component of the rotation map only depending on a vertex's H component, not its G component. We define the s-wide replacement product as described before, with the inter-cloud operator G_i using the i-th coordinate of the H component, which is a value in $[d_1]$, to determine the inter-cloud step.

Definition 4.2 (s-wide replacement product)**.** *Suppose we are given the following:*

[4]This kind of map is denoted rotation map in the zig-zag terminology [RVW00].

- *A d_1-regular graph $G = ([n], E)$ together with a locally invertible rotation map $\mathrm{rot}_G \colon [n] \times [d_1] \to [n] \times [d_1]$.*
- *A d_2-regular graph $H = ([d_1]^s, E')$.*

And we define:

- *For $i \in \{0, 1, \ldots, s-1\}$, we define $\mathrm{Rot}_i \colon [n] \times [d_1]^s \to [n] \times [d_1]^s$ as, for every $v \in [n]$ and $(a_0, \ldots, a_{s-1}) \in [d_1]^s$,*

$$\mathrm{Rot}_i((v, (a_0, \ldots, a_{s-1}))) := (v', (a_0, \ldots, a_{i-1}, a'_i, a_{i+1}, \ldots, a_{s-1})),$$

  where $(v', a'_i) = \mathrm{rot}_G(v, a_i)$.
- *Denote by G_i the operator realizing Rot_i and let A_H be the normalized random walk operator of H. Note that G_i is a permutation operator corresponding to a product of transpositions.*

Then $t-1$ steps of the s-wide replacement product are given by the operator

$$\prod_{i=0}^{t-2} \mathsf{G}_{i \bmod s} (\mathsf{I} \otimes \mathsf{A}_H).$$

Ta-Shma instantiates the s-wide replacement product with an outer graph G that is a Cayley graph, for which locally invertible rotation maps exist generically.

Remark 4.3. *Let R be a group and $A \subseteq R$ where the set A is closed under inversion. For every Cayley graph $\mathrm{Cay}(R, A)$, the map $\varphi \colon A \to A$ defined as $\varphi(g) = g^{-1}$ gives rise to the locally invertible rotation map*

$$\mathrm{rot}_{\mathrm{Cay}(R,A)}((r, a)) = (r \cdot a, a^{-1}),$$

for every $r \in R$, $a \in A$.

4.2 The Construction

Ta-Shma's code construction works by starting with a constant bias code $\mathcal{C}_0$ in $\mathbb{F}_2^n$ and boosting to arbitrarily small bias using direct sum liftings. Recall that the direct sum lifting is based on a collection $W(t) \subseteq [n]^t$, which Ta-Shma obtains using $t-1$ steps of random walk on the s-wide replacement product of two regular expander graphs G and H. The graph G is on n vertices (same as blocklength of the base code) and other parameters like degrees d_1 and d_2 of G and H respectively are chosen based on target code parameters.

To elaborate, every $t-1$ length walk on the replacement product gives a sequence of t outer vertices or G-vertices, which can be seen as an element of $[n]^t$. This gives the collection $W(t)$ with $|W(t)| = n \cdot d_1^s \cdot d_2^{t-1}$ which means the rate of lifted code is smaller than the rate of $\mathcal{C}_0$ by a factor of $d_1^s d_2^{t-1}$. However, the collection $W(t)$ is a parity sampler and this means that the bias decreases (or the distance increases). The relationship between this decrease in bias and decrease in rate with some careful parameter choices allows Ta-Shma to obtain nearly optimal ε-balanced codes.

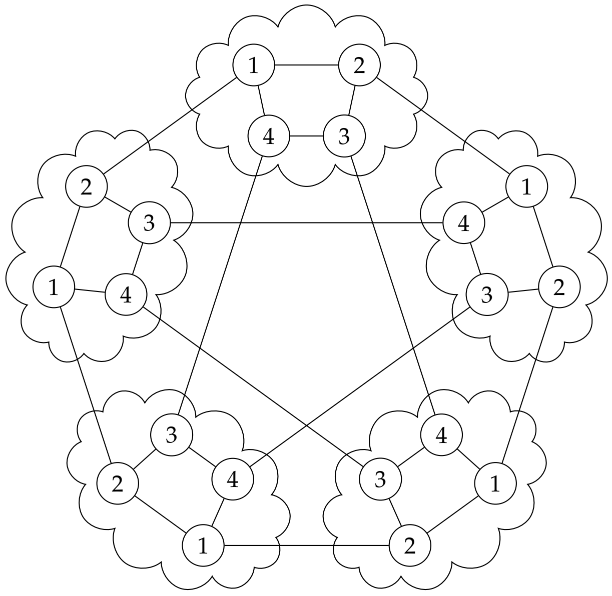

Figure 4: An example of the 1-wide replacement product with outer graph $G = K_5$ and inner graph $H = C_4$. Vertices are labeled by their H components. Note that the rotation map is locally invertible, with $\varphi(1) = 2$, $\varphi(2) = 1$, $\varphi(3) = 4$, and $\varphi(4) = 3$.

4.3 Tweaking the Construction

Recall the first s steps in Ta-Shma's construction are given by the operator

$$\mathsf{G}_{s-1}(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{s-2} \cdots G_1(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_0(\mathsf{I} \otimes \mathsf{A}_H).$$

Naively decomposing the above operator into the product of operators $\prod_{i=0}^{s-1} \mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)$ is not good enough to obtain the *splittability* property which would hold provided $\sigma_2(\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H))$ was small for every i in $\{0, \ldots, s-1\}$. However, each $\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)$ has $|V(G)|$ singular values equal to 1 since G_i is an orthogonal operator and $(\mathsf{I} \otimes \mathsf{A}_H)$ has $|V(G)|$ singular values equal to 1. To avoid this issue we will tweak the construction to be the following product

$$\prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H).$$

The operator $(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)$ is exactly the walk operator of the zig-zag product $G \circledz H$ of G and H with a rotation map given by the (rotation map) operator G_i. This tweaked construction is slightly simpler in the sense that $G \circledz H$ is an undirected graph. We know by the zig-zag analysis that $(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)$ is expanding as long G and H are themselves expanders. More precisely, we have a bound that follows from [RVW00].

Fact 4.4. *Let G be an outer graph and H be an inner graph used in the s-wide replacement product. For any integer $0 \le i \le s-1$,*

$$\sigma_2((I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)) \le \sigma_2(G) + 2 \cdot \sigma_2(H) + \sigma_2(H)^2.$$

This bound will imply *splittability* as shown in Section 7.2. We will need to argue that this modification still preserves the correctness of the parity sampling and that it can be achieved with similar parameter trade-offs.

The formal definition of a length-t walk on this slightly modified construction is given below.

Definition 4.5. *Let $t \in \mathbb{N}$, G be a d_1-regular graph and H be a d_2-regular graph on d_1^s vertices. Given a starting vertex $(v, h) \in V(G) \times V(H)$, a $(t-1)$-step walk on the tweaked s-wide replacement product of G and H is a tuple $((v_0, h_0), \ldots, (v_{t-1}, h_{t-1})) \in (V(G) \times V(H))^t$ such that*

- *$(v_0, h_0) = (v, h)$, and*
- *for every $0 \le i < t-1$, we have (v_i, h_i) adjacent to (v_{i+1}, h_{i+1}) in $(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{i \bmod s}(\mathsf{I} \otimes \mathsf{A}_H)$.*

Note that each $(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{i \bmod s}(\mathsf{I} \otimes \mathsf{A}_H)$ is a walk operator of a d_2^2-regular graph. Therefore, the starting vertex (v, h) together with a degree sequence $(m_1, \ldots, m_t) \in [d_2^2]^{t-1}$ uniquely defines a $(t-1)$-step walk.

4.3.1 Parity Sampling

We argue informally why parity sampling still holds with similar parameter trade-offs. Later in Section 4.3.2, we formalize a key result underlying parity sampling and, in Section 8, we compute the new trade-off between bias and rate in some regimes. In Section 4.1, the definition of the original s-wide replacement product as a purely graph theoretic operation was given. Now, we explain how Ta-Shma used this construction for parity sampling obtaining codes near the GV bound.

For a word $z \in \mathbb{F}_2^{V(G)}$ in the base code, let P_z be the diagonal matrix, whose rows and columns are indexed by $V(G) \times V(H)$, with $(\mathsf{P}_z)_{(v,h),(v,h)} = (-1)^{z_v}$. Proving parity sampling requires analyzing the operator norm of the following product

$$\mathsf{P}_z \prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i \mathsf{P}_z (\mathsf{I} \otimes \mathsf{A}_H), \tag{1}$$

when $\text{bias}(z) \le \varepsilon_0$. Let $\mathbf{1} \in \mathbb{R}^{V(G) \times V(H)}$ be the all-ones vector and W be the collection of all $(t-1)$-step walks on the tweaked s-wide replacement product. Ta-Shma showed (and it is not difficult to verify) that

$$\text{bias}\left(\text{dsum}_W(z)\right) = \left| \left\langle \mathbf{1}, \mathsf{P}_z \prod_{i=0}^{t-2} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{i \bmod s} \mathsf{P}_z (\mathsf{I} \otimes \mathsf{A}_H)\mathbf{1} \right\rangle \right|.$$

From the previous equation, one readily deduces that

$$\text{bias}\left(\text{dsum}_W(z)\right) \le \sigma_1 \left(\mathsf{P}_z \prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i \mathsf{P}_z (\mathsf{I} \otimes \mathsf{A}_H) \right)^{\lfloor (t-1)/s \rfloor}.$$

Set $\mathsf{B} := \mathsf{P}_z \prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i \mathsf{P}_z (\mathsf{I} \otimes \mathsf{A}_H)$. To analyze the operator norm of B, we will first need some notation. Note that B is an operator acting on the space $\mathcal{V} = \mathbb{R}^{V(G)} \otimes \mathbb{R}^{V(H)}$. Two of its subspaces play an important role in the analysis, namely,

$$\mathcal{W}^{\parallel} = \text{span}\{a \otimes b \in \mathbb{R}^{V(G)} \otimes \mathbb{R}^{V(H)} \mid b = \mathbf{1}\} \text{ and } \mathcal{W}^{\perp} = (\mathcal{W}^{\parallel})^{\perp}.$$

Note that the complement subspace is with respect to the standard inner product. Observe that $\mathcal{V} = \mathcal{W}^{\parallel} \oplus \mathcal{W}^{\perp}$. Given arbitrary unit vectors $v, w \in \mathcal{V}$, Ta-Shma considers the inner product

$$\left\langle v, \prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i\mathsf{P}_z(\mathsf{I} \otimes \mathsf{A}_H) w \right\rangle. \tag{2}$$

Each time an operator $(\mathsf{I} \otimes \mathsf{A}_H)$ appears in the above expression, the next step of the walk can take one out of d_2 possibilities and thus the rate suffers a multiplicative decrease of $1/d_2$. We think that we are "paying" d_2 for this step of the walk. The whole problem lies in the trade-off between rate and distance, so the crucial question now is how much the norm decreases as we pay d_2. For a moment, suppose that the norm always decreases by a factor of $\lambda_2 := \sigma_2(H)$ per occurrence of $(\mathsf{I} \otimes \mathsf{A}_H)$. If in this hypothetical case we could further assume $\lambda_2 = 1/\sqrt{d_2}$, then if B was a product containing $\lceil \log_{\lambda_2}(\varepsilon) \rceil$ factors of $(\mathsf{I} \otimes \mathsf{A}_H)$, the final bias would be at most ε and the rate would have suffered a multiplicative decrease of (essentially) ε^2 and we would be done.

Of course, this was an oversimplification. The general strategy is roughly the above, but a beautiful non-trivial step is needed. Going back to the bilinear form Eq. (2), if $w \in \mathcal{W}^{\perp}$ (or $v \in \mathcal{W}^{\perp}$), we pay d_2 and we do obtain a norm decrease of λ_2. More generally, note that can decompose $w = w^{\parallel} + w^{\perp}$ with $w^{\parallel} \in \mathcal{W}^{\parallel}$ and $w^{\perp} \in \mathcal{W}^{\perp}$ (decompose $v = v^{\parallel} + v^{\perp}$ similarly) and we can carry this process iteratively collecting factors of λ_2. However, we are stuck with several terms of the form for $0 \le k_1 \le k_2 < s$,

$$\left\langle v_{k_1}^{\parallel}, \prod_{i=k_1}^{k_2} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i\mathsf{P}_z(\mathsf{I} \otimes \mathsf{A}_H) w_{k_2}^{\parallel} \right\rangle,$$

with $v_{k_1}^{\parallel}, w_{k_2}^{\parallel} \in \mathcal{W}^{\parallel}$, and for which the preceding naive norm decrease argument fails. This is the point in the analysis where the structure of the s-wide replacement product is used. Since $v_{k_1}^{\parallel}, w_{k_2}^{\parallel} \in \mathcal{W}^{\parallel}$, these vectors are uniform on each "cloud", i.e., copy of H. Recall that a vertex in H is an s-tuple $(m_1, \ldots, m_s) \in [d_1]^s$. Ta-Shma leverages the fact of having a uniform such tuple to implement $k_2 - k_1 + 1$ (up to s) steps of random walk on G. More precisely, Ta-Shma obtains the following beautiful result:

Theorem 4.6 (Adapted from Ta-Shma [TS17])**.** *Let G be a locally invertible graph of degree d_1, H be a Cayley graph on $\mathbb{F}_2^{s \log d_1}$, and $0 \le k_1 \le k_2 < s$ be integers. If $v^{\parallel} = v \otimes 1$ and $w^{\parallel} = w \otimes 1$, then*

$$\left\langle v^{\parallel}, \prod_{i=k_1}^{k_2} \mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{P}_z w^{\parallel} \right\rangle = \left\langle v, (\mathsf{A}_G\mathsf{M}_z)^{k_2-k_1+1} w \right\rangle$$

where $\mathsf{M}_z \in \mathbb{R}^{V(G) \times V(G)}$ is the diagonal matrix defined as $(\mathsf{M}_z)_{v,v} := (-1)^{z_v}$ for $v \in V(G)$.

Remark 4.7. *Note that the walk operator in this theorem corresponds to the original construction. Theorem 4.6 was used by Ta-Shma to obtain Fact 4.9 whose Corollary 4.10 corresponds to the modified construction.*

Ta-Shma proved Theorem 4.6 under the more general condition that H is 0-pseudorandom. Roughly speaking, this property means that if we start with a distribution that is uniform over the clouds, and walk according to fixed H-steps $j_0, j_1, \cdots, j_{s-1}$, then the distribution of G-vertices obtained will be identical to the distribution obtained if we were doing the usual random walk on G. We will always choose H to be a Cayley graph on $\mathbb{F}_2^{s \log d_1}$, which will imply that H is also 0-pseudorandom. The proof of Theorem 4.6 crucially uses the product structure of $\mathbb{F}_2^{s \log d_1}$: every vertex of H can be represented by s registers of $\log d_1$ bits each, and both inter-cloud and intra-cloud steps can be seen as applying register-wise bijections using some canonical mapping between $[d_1]$ and $\mathbb{F}_2^{\log d_1}$.

Ta-Shma's original parity sampling proof required $\varepsilon_0 + 2\theta + 2\sigma_2(G) \leq \sigma_2(H)^2$, where ε_0 is the initial bias and θ is an error parameter arising from a number theoretic construction of Ramanujan graphs for the outer graph G. This is because $\varepsilon_0 + 2\theta + 2\sigma_2(G)$ is the reduction of bias in every two steps while taking a walk on G (see Theorem 5.2). Having $\varepsilon_0 + 2\theta + 2\sigma_2(G) \leq \sigma_2(H)^2$ ensured that after establishing Theorem 4.6, we were collecting enough reduction for d_2^2 price we paid for two steps. In the modified construction, we now have d_2^2 possibilities for each step in $(\mathsf{I} \otimes \mathsf{A}_H^2)$ (so d_2^4 price for two steps), and so if instead we have $\varepsilon_0 + 2\theta + 2\sigma_2(G) \leq \sigma_2(H)^4$ in the modified construction, we claim that the correctness of the parity sampling analysis is preserved as well as (essentially) the trade-off between walk length and norm decay. Fortunately, Ta-Shma's parameters decouple and we can choose parameters to satisfy the above requirement.

Remark 4.8. *This modification on the s-replacement product of G and H essentially* [5] *amounts to taking a different inner graph H which can be factored as* $H = \sqrt{H}\sqrt{H}$ *(and is still* 0*-pseudorandom).*

4.3.2 Spectral Analysis of the Modified Construction

We formally show that we don't loose much by going from Ta-Shma's original s-wide product construction to its tweaked version. The key technical result obtained by Ta-Shma is the following, which is used to analyze the bias reduction as a function of the total number walk steps $t-1$.

Fact 4.9 (Theorem 24 abridged [TS17])**.** *If H is a Cayley graph on* $\mathbb{F}_2^{s \log d_1}$ *and* $\varepsilon_0 + 2 \cdot \theta + 2 \cdot \sigma_2(G) \leq \sigma_2(H)^2$*, then*

$$\left\| \prod_{i=0}^{s-1} \mathsf{P}_z \mathsf{G}_i (\mathsf{I} \otimes \mathsf{A}_H) \right\|_{\text{op}} \leq \sigma_2(H)^s + s \cdot \sigma_2(H)^{s-1} + s^2 \cdot \sigma_2(H)^{s-3},$$

where $\mathsf{P}_z \in \mathbb{R}^{(V(G) \times V(H)) \times (V(G) \times V(H))}$ *is the* sign operator *of a* ε_0 *biased word* $z \in \mathbb{F}_2^{V(G)}$ *defined as a diagonal matrix with* $(P_z)_{(v,h),(v,h)} = (-1)^{z_v}$ *for every* $(v,h) \in V(G) \times V(H)$*.*

We reduce the analysis of Ta-Shma's tweaked construction to Fact 4.9. In doing so, we only lose one extra step as shown below.

Corollary 4.10. *If* H^2 *is a Cayley graph on* $\mathbb{F}_2^{s \log d_1}$ *and* $\varepsilon_0 + 2 \cdot \theta + 2 \cdot \sigma_2(G) \leq \sigma_2(H)^4$*, then*

$$\left\| \prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H) \mathsf{P}_z \mathsf{G}_i (\mathsf{I} \otimes \mathsf{A}_H) \right\|_{\text{op}} \leq \sigma_2(H^2)^{s-1} + (s-1) \cdot \sigma_2(H^2)^{s-2} + (s-1)^2 \cdot \sigma_2(H^2)^{s-4},$$

where P_z *is the* sign operator *of an* ε_0*-biased word* $z \in \mathbb{F}_2^{V(G)}$ *as in Fact 4.9.*

[5]Except at the first and last factors in the product of operators.

Proof. We have

$$\begin{aligned}\left\|\prod_{i=0}^{s-1}(\mathsf{I}\otimes \mathsf{A}_H)\mathsf{P}_z\mathsf{G}_i(\mathsf{I}\otimes \mathsf{A}_H)\right\|_{\mathrm{op}} &\le \|(\mathsf{I}\otimes \mathsf{A}_H)\|_{\mathrm{op}}\left\|\prod_{i=1}^{s-1}\mathsf{P}_z\mathsf{G}_i(\mathsf{I}\otimes \mathsf{A}_H^2)\right\|_{\mathrm{op}}\|\mathsf{P}_z\mathsf{G}_0(\mathsf{I}\otimes \mathsf{A}_H)\|_{\mathrm{op}}\\ &\le \left\|\prod_{i=1}^{s-1}\mathsf{P}_z\mathsf{G}_i(\mathsf{I}\otimes \mathsf{A}_H^2)\right\|_{\mathrm{op}}\\ &\le \sigma_2(H^2)^{s-1}+(s-1)\cdot\sigma_2(H^2)^{s-2}+(s-1)^2\cdot\sigma_2(H^2)^{s-4},\end{aligned}$$

where the last inequality follows from Fact 4.9. ■

Remark 4.11. *We know that in the modified construction H^2 is a Cayley graph since H is a Cayley graph.*

From this point onward, we will be working exclusively with the modified construction instead of using it in its original form. Any references to Ta-Shma's construction or the s-wide replacement product will actually refer to the modified versions described in this section.

5 Code Cascading

A code cascade is a sequence of codes generated by starting with a base code $\mathcal{C}_0$ and recursively applying lifting operations.

Definition 5.1. *We say that a sequence of codes $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_\ell$ is a* code cascade *provided $\mathcal{C}_i = \mathrm{dsum}_{W_i(t_i)}(\mathcal{C}_{i-1})$ for every $i \in [\ell]$. Each $W_i(t_i)$ is a subset of $[n_{i-1}]^{t_i}$, where $n_{i-1} = |W_{i-1}(t_{i-1})|$ is the block length of the code $\mathcal{C}_{i-1}$.*

Let us see how code cascades may be useful for decoding. Suppose we wish to lift the code $\mathcal{C}_0$ to $\mathcal{C}_\ell$, and there is some $W(t) \subseteq [n_0]^t$ such that $\mathcal{C}_\ell = \mathrm{dsum}_{W(t)}(\mathcal{C}_0)$. In our case of bias boosting, this t will depend on the target bias ε. However, the expansion requirement of the list-decoding framework of [AJQ+20] has a poor dependence on t. A way to work around this issue is to go from $\mathcal{C}_0$ to $\mathcal{C}_\ell$ via a code cascade as above such that each t_i is a constant independent of the final bias but $\prod_{i=1}^{\ell} t_i = t$ (which means ℓ depends on ε). The final code $\mathcal{C}_\ell$ of the cascade is the same as the code obtained from length-$(t-1)$ walks. While decoding will now become an ℓ-level recursive procedure, the gain from replacing t by t_i will outweigh this loss, as we discuss below.

5.1 Warm-up: Code Cascading Expander Walks

We now describe the code cascading construction and unique decoding algorithm in more detail. Let $G = (V, E)$ be a d-regular graph with uniform distribution over the edges. Let m be a sufficiently large positive integer, which will be the number of vertices of the walks used for the lifting between consecutive codes in the cascade. At first, it will be crucial that we can take $m = O(1)$ so that the triangle inequality arising from the analysis of the lifting between two consecutive codes involves a constant number of terms. We construct a recursive family of codes as follows.

- Start with a code $\mathcal{C}_0$ which is linear and has constant bias ε_0.

- Define the code $\mathcal{C}_1 = \mathrm{dsum}_{W(m)}(\mathcal{C}_0)$, which is the direct sum lifting over the collection $W(m)$ of all length-$(m-1)$ walks on G using the code $\mathcal{C}_0$.

- Let $\widehat{G}_i = (V_i, E_i)$ be the (directed) graph where V_i is the collection of all walks on m^i vertices on G with two walks $(v_1, \dots, v_{m^i})$ and $(u_1, \dots, u_{m^i})$ connected iff v_{m^i} is adjacent to u_1 in G.

- Define $\mathcal{C}_i$ to be the direct sum lifting on the collection $W_i(m)$ of all length-$(m-1)$ walks on G_{i-1} using the code $\mathcal{C}_{i-1}$, i.e., $\mathcal{C}_i = \mathrm{dsum}_{W_i(m)}(\mathcal{C}_{i-1})$.

- Repeat this process to yield a code cascade $\mathcal{C}_0, \dots, \mathcal{C}_\ell$.

Thanks to the definition of the graphs $\widehat{G}_i$ and the recursive nature of the construction, the final code $\mathcal{C}_\ell$ is the same as the code obtained from $\mathcal{C}_0$ by taking the direct sum lifting over all walks on $t = m^\ell$ vertices of G. We can use Ta-Shma's analysis (building on the ideas of Rozenman and Wigderson [Bog12]) for the simpler setting of walks over a single expander graph to determine the amplification in bias that occurs in going from $\mathcal{C}_0$ all the way to $\mathcal{C}_\ell$.

Theorem 5.2 (Adapted from Ta-Shma [TS17]). *Let $\mathcal{C}$ be an ε_0-balanced linear code, and let $\mathcal{C}' = \mathrm{dsum}_{W(t)}(\mathcal{C})$ be the direct sum lifting of $\mathcal{C}$ over the collection of all length-$(t-1)$ walks $W(t)$ on a graph G. Then*

$$\mathrm{bias}(\mathcal{C}') \le (\varepsilon_0 + 2\sigma_2(G))^{\lfloor (t-1)/2 \rfloor}.$$

If $\sigma_2(G) \le \varepsilon_0/2$ and $\ell = \left\lceil \log_m(2\log_{2\varepsilon_0}(\varepsilon) + 3) \right\rceil$, taking $t = m^\ell \ge 2\log_{2\varepsilon_0}(\varepsilon) + 3$ in the above theorem shows that the final code $\mathcal{C}_\ell$ is ε-balanced. Observe that the required expansion of the graph G only depends on the constant initial bias ε_0, not on the desired final bias ε. It will be important for being able to decode with better parameters that both $\sigma_2(G)$ and m are constant with respect to ε; only ℓ depends on the final bias (with more care we can make $\sigma_2(G)$ depend on ε, but we restrict this analysis to Ta-Shma's refined construction on the s-wide replacement product).

As mentioned before, to uniquely decode $\mathcal{C}_\ell$ we will inductively employ the list decoding machinery for expander walks from [AJQ+20]. The list decoding algorithm can decode a direct sum lifting $\mathcal{C}' = \mathrm{dsum}_{W(m)}(\mathcal{C})$ as long as the graph G is sufficiently expanding, the walk length $m-1$ is large enough, and the base code $\mathcal{C}$ has an efficient unique decoding algorithm (see Theorem 6.1 for details).

The expansion requirement ultimately depends on the desired list decoding radius of $\mathcal{C}'$, or more specifically, on how close the list decoding radius is to $1/2$. If the distance of $\mathcal{C}'$ is at most $1/2$, its unique decoding radius is at most $1/4$, which means list decoding at the unique decoding radius is at a constant difference from $1/2$ and thus places only a constant requirement on the expansion of G. In the case of the code cascade $\mathcal{C}_i = \mathrm{dsum}_{W_i(m)}(\mathcal{C}_{i-1})$, unique decoding of $\mathcal{C}_{i-1}$ is guaranteed by the induction hypothesis. It is not too difficult to see that each graph $\widehat{G}_i$ will have the same second singular value as G, so we can uniquely decode $\mathcal{C}_i$ if G meets the constant expansion requirement and m is sufficiently large.

5.2 Code Cascading Ta-Shma's Construction

We will now describe how to set up a code cascade based on walks on an s-wide replacement product. Consider the s-wide replacement product of the outer graph G with the inner graph H. The first s walk steps are given by the walk operator

$$\prod_{i=0}^{s-1} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H).$$

Let $\mathsf{A}_{s-1} := (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{s-2}(\mathsf{I} \otimes \mathsf{A}_H) \cdots (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_0(\mathsf{I} \otimes \mathsf{A}_H)$. If the total walk length $t-1$ is a multiple of s, the walks are generated using the operator

$$\left((\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{s-1}(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{A}_{s-1}\right)^{(t-1)/s}.$$

Here $(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{s-1}(\mathsf{I} \otimes \mathsf{A}_H)$ is used as a "binding" operator to connect two walks containing s vertices at level $\mathcal{C}_2$, s^2 vertices at level $\mathcal{C}_3$, and so on. More precisely, we form the following code cascade.

- $\mathcal{C}_0$ is an ε_0-balanced linear code efficiently uniquely decodable from a constant radius.

- $\mathcal{C}_1 = \mathrm{dsum}_{W_1(s)}(\mathcal{C}_0)$, where $W_1(s)$ is the set of length-(s-1) walks given by the operator
$$\underbrace{(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{s-2}(\mathsf{I} \otimes \mathsf{A}_H)}_{(s-2)\text{th step}} \cdots \underbrace{(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_0(\mathsf{I} \otimes \mathsf{A}_H)}_{0\text{th step}}.$$

- $\mathcal{C}_2 = \mathrm{dsum}_{W_2(s)}(\mathcal{C}_1)$, where $W_2(s)$ is the set of length-$(s-1)$ walks over the vertex set $W_1(s)$ (with the latter being the set of length-$(s-1)$ walks on the replacement product graph as mentioned above).

- $\mathcal{C}_{i+1} = \mathrm{dsum}_{W_{i+1}(s)}(\mathcal{C}_i)$, where $W_{i+1}(s)$ is the set of length-$(s-1)$ walks [6] over the vertex set $W_i(s)$. Similarly to the cascade of expander walks above, the lift can be thought of as being realized by taking walks using a suitable operator analogous to $\widehat{G}_i$. Since its description is more technical we postpone its definition (see Definition 7.2) to Section 7.2 where it is actually used.

- $\mathcal{C}_\ell$ denotes the final code in the sequence, which will later be chosen so that its bias is at most ε.

6 Unique Decoding of Ta-Shma Codes

We show how code cascading together with list decoding for each level of the cascade allow us to obtain an efficient unique decoding algorithm for Ta-Shma's construction. We obtain a sequence of results of increasing strength culminating in Theorem 1.1 (which we recall below for convenience). The approach is as follows: we use several different instantiations of Ta-Shma's construction, each yielding a value of s (for the s-wide replacement product) and expansion parameters for the family of outer and inner graphs, and show how the list decoding framework can be invoked in the associated cascade for each one.

[6]For simplicity we chose the number of vertices in all walks of the cascade to be s, but it could naturally be some $s_i \in \mathbb{N}$ depending on i.

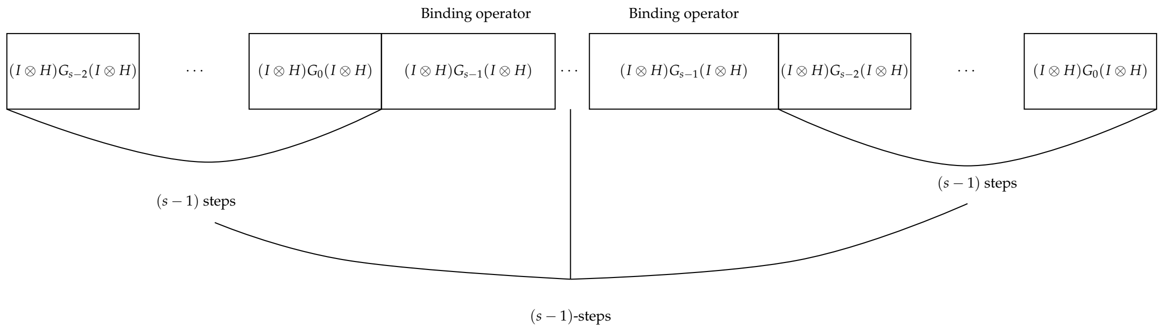
Binding operator
Binding operator
$(I \otimes H)G_{s-2}(I \otimes H)$
$\cdots$
$(I \otimes H)G_0(I \otimes H)$
$(I \otimes H)G_{s-1}(I \otimes H)$
$\cdots$
$(I \otimes H)G_{s-1}(I \otimes H)$
$(I \otimes H)G_{s-2}(I \otimes H)$
$\cdots$
$(I \otimes H)G_0(I \otimes H)$
$(s-1)$ steps
$(s-1)$ steps
$(s-1)$-steps

Figure 5: Two levels of code cascading for Ta-Shma's construction involving codes $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_2$ (to make the notation compact we used H to denote A_H).

Theorem 1.1 (Unique Decoding). *For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$ where $\beta = O(1/(\log_2(1/\varepsilon))^{1/6})$, and

(iii) a unique decoding algorithm with running time $N^{O_{\varepsilon,\beta}(1)}$.

Furthermore, if instead we take $\beta > 0$ to be an arbitrary constant, the running time becomes $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$ (fixed polynomial time).

In this section, we will fit these objects and tools together assuming the parameters are chosen to achieve the required rates and the conditions for applying the list decoding results are satisfied. The concrete instantiations of Ta-Shma codes are done in Section 8. Establishing that the list decoding framework can be applied to this construction is done in Section 7 after which the framework is finally instantiated in Section 9.

Ta-Shma uses the direct sum lifting on an s-wide replacement product graph to construct a family of ε-balanced codes $\mathcal{C}_{N,\varepsilon,\beta}$ with rate $\Omega(\varepsilon^{2+\beta})$ and finds parameters for such codes to have the required bias and rate. We will discuss unique decoding results for several versions of these codes. Throughout this section, we will use collections $W(k)$ which will always be either the set of walks with $k = s$ vertices on an s-wide replacement product graph (corresponding to the first level of the code cascade), which we denote $W[0, s-1]$, or a set of walks where the vertices are walks on a lower level of the code cascade.

6.1 Unique Decoding via Code Cascading

To perform unique decoding we will use the machinery of list decoding from Theorem 6.1 (proven later in Section 9), which relies on the list decoding framework of [AJQ+20]. Proving that this framework can be applied to Ta-Shma's construction is one of our technical contributions.

Theorem 6.1 (List decoding direct sum lifting). *Let $\eta_0 \in (0, 1/4)$ be a constant, $\eta \in (0, \eta_0)$, and*

$$k \geq k_0(\eta) := \Theta(\log(1/\eta)).$$

Suppose $\mathcal{C} \subseteq \mathbb{F}_2^n$ is an η_0-balanced linear code and $\mathcal{C}' = \operatorname{dsum}_{W(k)}(\mathcal{C})$ is the direct sum lifting of $\mathcal{C}$ on a τ-splittable collection of walks $W(k)$. There exists an absolute constant $K > 0$ such that if

$$\tau \leq \tau_0(\eta, k) := \frac{\eta^8}{K \cdot k \cdot 2^{4k}},$$

then the code $\mathcal{C}'$ is η-balanced and can be efficiently list decoded in the following sense:

If $\tilde{y}$ is $(1/2 - \sqrt{\eta})$-close to $\mathcal{C}'$, then we can compute the list

$$\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') := \left\{ (z, \operatorname{dsum}_{W(k)}(z)) \mid z \in \mathcal{C}, \Delta\Big(\operatorname{dsum}_{W(k)}(z), \tilde{y}\Big) \leq \frac{1}{2} - \sqrt{\eta} \right\}$$

in time

$$n^{O(1/\tau_0(\eta,k)^4)} \cdot f(n),$$

where $f(n)$ is the running time of a unique decoding algorithm for $\mathcal{C}$. Otherwise, we return $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') = \emptyset$ with the same running time of the preceding case.

Note that the requirement on k in the above theorem is necessary for the lifted code $\mathcal{C}'$ to be η-balanced. Splittability will imply that the walk collection $W(k)$ is expanding, which gives us parity sampling for large k. Specifically, k must be large enough for $W(k)$ to be a $(1/2 + \eta_0/2, \eta)$-parity sampler.

Applying the list decoding tool above, we can perform unique decoding in the regime of η_0, η, and k being constant. With these choices the expansion required for splittability and the parity sampling strength are only required to be constants.

Lemma 6.2 (Decoding Step)**.** *Let $\eta_0 \in (0, 1/4)$ and $\eta < \min\{\eta_0, 1/16\}$. If $W(k)$ is a walk collection on vertex set $[n]$ with $k \geq k_0(\eta)$ and splittability $\tau \leq \tau_0(\eta, k)$, where k_0 and τ_0 are as in Theorem 6.1, we have the following unique decoding property:*

If $\mathcal{C} \subseteq \mathbb{F}_2^n$ is an η_0-balanced linear code that can be uniquely decoded in time $f(n)$, then $\mathcal{C}' = \operatorname{dsum}_{W(k)}(\mathcal{C})$ is an η-balanced code that can be uniquely decoded in time $n^{O(1/\tau_0(\eta,k)^4)} \cdot f(n)$.

Proof. Using Theorem 6.1, we can list decode $\mathcal{C}'$ up to a radius of $1/2 - \sqrt{\eta}$ for any η if we have the appropriate parameters k and τ. Let $\tilde{y} \in \mathbb{F}_2^{W(k)}$ be a received word within the unique decoding radius of $\mathcal{C}'$. To perform unique decoding, we simply run the list decoding algorithm on $\tilde{y}$ and return the codeword on the resulting list which is closest to $\tilde{y}$; this will yield the correct result as long as the list decoding radius is larger than the unique decoding radius. It suffices to have $1/2 - \sqrt{\eta} > 1/4 \geq \Delta(\mathcal{C}')/2$. We choose parameters as follows:

1. Take $\eta < 1/16$ to ensure $1/2 - \sqrt{\eta} > 1/4$.
2. Let $k_0 = \Theta(\log(1/\eta))$ be the smallest integer satisfying the assumption in Theorem 6.1 with the chosen η. Take $k \geq k_0$.
3. Take $\tau \leq \tau_0(\eta, k) = \eta^8/(K \cdot k \cdot 2^{4k})$.

Note that k and τ satisfy the conditions of Theorem 6.1, so we can use this theorem to list decode a received word $\tilde{y}$ in time $n^{O(1/\tau_0(\eta,k)^4)} \cdot f(n)$. To unique decode, we return the closest y on the list to $\tilde{y}$ (or failure if the list is empty). ∎

Iteratively using the decoding step given by Lemma 6.2 above, we obtain unique decodability of all codes in a cascade (under suitable assumptions).

Lemma 6.3 (Code Cascade Decoding). *Let $\eta_0 \in (0,1/4)$ and $\eta < \min\{\eta_0, 1/16\}$. Suppose $\mathcal{C}_0 \subseteq \mathbb{F}_2^{n_0}, \mathcal{C}_1 \subseteq \mathbb{F}_2^{n_1}, \dots, \mathcal{C}_\ell \subseteq \mathbb{F}_2^{n_\ell}$ is a code cascade where $\mathcal{C}_0$ is an η_0-balanced linear code that can be uniquely decoded in time $g(n_0)$.*

If for every $i \in [\ell]$ we have that $\mathcal{C}_i$ is obtained from $\mathcal{C}_{i-1}$ by a τ_i-splittable walk collection $W_i(k_i)$ on vertex set $[n_{i-1}]$ with $k_i \geq k_0(\eta)$ and $\tau_i \leq \tau_0(\eta, k_i)$, where k_0 and τ_0 are as in Theorem 6.1, then $\mathcal{C}_\ell$ is uniquely decodable in time

$$g(n_0) \cdot \prod_{i=1}^{\ell} n_{i-1}^{O(1/\tau_0(\eta,k_i)^4)}.$$

Proof. Induct on $i \in [\ell]$ applying Lemma 6.2 as the induction step. The code $\mathcal{C}_i$ produced during each step will have bias at most $\eta < \eta_0$, so the hypothesis of Lemma 6.2 will be met at each level of the cascade. ■

We are almost ready to prove our first main theorem establishing decodability close to the Gilbert–Varshamov bound. We will need parameters for an instantiation of Ta-Shma's code that achieves the desired distance and rate (which will be developed in Section 8.1) and a lemma relating splittability to the spectral properties of the graphs used in the construction (to be proven in Section 7.2).

Lemma 6.4 (Ta-Shma Codes I). *For any $\beta > 0$, there are infinitely many values of $\varepsilon \in (0,1/2)$ (with 0 as an accumulation point) such that for infinitely many values of $N \in \mathbb{N}$, there are explicit binary Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced), and

(ii) rate $\Omega(\varepsilon^{2+\beta})$.

Furthermore, $\mathcal{C}_{N,\varepsilon,\beta}$ is the direct sum lifting of a base code $\mathcal{C}_0 \subseteq \mathbb{F}_2^{n_0}$ using the collection of walks $W[0, t-1]$ on the s-wide replacement product of two graphs G and H, with the following parameters:

- *$s \geq s_0 := \max\{128, 26/\beta\}$.*
- *The inner graph H is a regular graph with $\sigma_2(H) \leq \lambda_2$, where $\lambda_2 = (16s^3 \log s)/s^{2s^2}$.*
- *The outer graph G is a regular graph with $\sigma_2(G) \leq \lambda_1$, where $\lambda_1 = \lambda_2^4/6$.*
- *The base code $\mathcal{C}_0$ is unique decodable in time $n_0^{O(1)}$ and has bias $\varepsilon_0 \leq \lambda_2^4/3$.*
- *The number of vertices t in the walks satisfies $\lambda_2^{2(1-5/s)(1-1/s)(t-1)} \leq \varepsilon$.*

Lemma 6.5. *Let $W(k)$ be either the collection $W[0, s-1]$ of walks of length s on the s-wide replacement product with outer graph G and inner graph H or the collection of walks over the vertex set $W[0, r]$, where $r \equiv -1 \pmod s$. Then $W(k)$ is τ-splittable with $\tau = \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2$.*

The statement of this first decoding theorem is more technical than Theorem 1.1, but it will be easier to prove while the latter will build on this theorem with a more careful tuning of parameters.

Theorem 6.6 (Main I). *For every $\beta > 0$, there are infinitely many values $\varepsilon \in (0, 1/2)$ (with 0 an accumulation point) such that for infinitely many values of $N \in \mathbb{N}$ there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$, and

(iii) a unique decoding algorithm with running time $N^{O_\beta(\log(\log(1/\varepsilon)))}$.

Proof. We proceed as follows:

1. Let $\eta_0 = 1/10$ and $\eta = 1/100$ (these choices are arbitrary; we only need $\eta_0 < 1/4$, $\eta < 1/16$, and $\eta < \eta_0$). Let $k_0 = k_0(\eta)$ be the constant from Theorem 6.1 with this value of η.

2. Given $\beta > 0$, Lemma 6.4 provides a value s_0 such that the direct sum lifting on the s-wide replacement product with $s \geq s_0$ can achieve a rate of $\Omega(\varepsilon^{2+\beta})$ for infinitely many $\varepsilon \in (0, 1/2)$. Choose s to be an integer larger than both k_0 and s_0 that also satisfies
$$s^2 \cdot \left(\frac{s}{16}\right)^{-s^2} \leq \frac{\eta^8}{4K}, \tag{3}$$
where K is the constant from Theorem 6.1.

3. Use Lemma 6.4 with this value of s to obtain graphs G and H and a base code $\mathcal{C}_0$ having the specified parameters λ_1, λ_2, ε_0, and t, with the additional requirement that $t = s^\ell$ for some integer ℓ. These parameter choices ensure that the resulting code $\mathcal{C}_{N,\varepsilon,\beta}$ has the desired distance and rate. Since $s \geq 128$, we have $\lambda_2 = (16s^3 \log s)/s^{2s^2} \leq s^{-s^2}$. From the choice of t satisfying $\lambda_2^{2(1-5/s)(1-1/s)(t-1)} \leq \varepsilon$, we deduce that $\ell = O(\log(\log(1/\varepsilon)))$. Note also that the bias ε_0 of the code $\mathcal{C}_0$ is smaller than η_0.

4. Create a code cascade with ℓ levels using the s-wide replacement product of the graphs G and H as in Section 5.2, starting with $\mathcal{C}_0$ and ending with the final code $\mathcal{C}_\ell = \mathcal{C}_{N,\varepsilon,\beta}$. As the total number of vertices in a walk is $t = s^\ell$, each level of the code cascade will use walks with s vertices. Let $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_\ell$ be the sequence of codes in this cascade.

5. In order to satisfy the splittability requirement of Lemma 6.3, the walk collection $W_i(s)$ at each level of the code cascade must be τ-splittable, where $\tau \leq \tau_0(\eta, s^2)$. (We use $k = s^2$ instead of $k = s$ in the requirement for a technical reason that will be clear in Section 8.2.) The bounds on the singular values of G and H and Lemma 6.5 ensure that
$$\tau = \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2 \leq 4\lambda_2 \leq 4s^{-s^2},$$
which is smaller than $\tau_0(\eta, s^2) = \eta^8/(K \cdot s^2 \cdot 2^{4s^2})$ by Eq. (3)

6. As all hypotheses of Lemma 6.3 are satisfied by the code cascade, we apply it to conclude that $\mathcal{C}_{N,\varepsilon,\beta}$ is uniquely decodable in time

$$g(n_0) \cdot \prod_{i=1}^{\ell} n_{i-1}^{O(1/\tau_0(\eta,s)^4)} \leq N^{O(1)} \cdot \prod_{i=1}^{\ell} N^{O_\beta(1)} = N^{O_\beta(\log(\log(1/\varepsilon)))},$$

where we use that $\mathcal{C}_0$ is uniquely decodable in time $n_0^{O(1)}$, $1/\tau_0(\eta,s) = 2^{O(1/\beta)}$, $n_{i-1} < n_\ell = N$ for every $i \in [\ell]$, and $\ell = O(\log(\log(1/\varepsilon)))$.

■

In the code cascade constructed in Theorem 6.6, the final number of vertices in a walk is $t = s^\ell$, where s is a sufficiently large constant that does not depend on ε. The limited choices for t place some restrictions on the values of the final bias ε that can be achieved. To achieve any bias ε for $\mathcal{C}_\ell$ we need to choose the parameters more carefully, which will be done in Section 8.2 to yield our next main result.

Theorem 6.7 (Main II). *For every $\beta > 0$ and every $\varepsilon > 0$ with β and ε sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$, and

(iii) a unique decoding algorithm with running time $N^{O_\beta(\log(\log(1/\varepsilon)))}$.

Ta-Shma obtained codes of rate $\Omega(\varepsilon^{2+\beta})$ with vanishing β as ε goes to zero. We obtain a unique decoding algorithm for this regime (with slightly slower decreasing β as ε vanishes). More precisely, using the parameters described in Section 8.3 and the running time analysis in Section 6.2, we obtain the following theorem which is our main result for unique decoding.

Theorem 6.8 (Main Unique Decoding (restatement of Theorem 1.1)). *For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$ where $\beta = O(1/(\log_2(1/\varepsilon))^{1/6})$, and

(iii) a unique decoding algorithm with running time $N^{O_{\varepsilon,\beta}(1)}$.

Furthermore, if instead we take $\beta > 0$ to be an arbitrary constant, the running time becomes $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$ (fixed polynomial time).

Theorem 1.2 about gentle list decoding is proved in Section 8.4 after instantiating Ta-Shma codes in some parameter regimes in the preceding parts of Section 8.

6.2 Fixed Polynomial Time

In Theorem 6.7, a running time of $N^{O_\beta(\log(\log(1/\varepsilon)))}$ was obtained to decode Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta}$ of distance $1/2 - \varepsilon/2$ and rate $\Omega(\varepsilon^{2+\beta})$ for constant $\beta > 0$ and block length N. The running time contains an exponent which depends on the bias ε and is therefore not fixed polynomial time. We show how to remove this dependence in this regime of $\beta > 0$ being an arbitrary constant. More precisely, we show the following.

Theorem 6.9 (Fixed PolyTime Unique Decoding). *Let $\beta > 0$ be an arbitrary constant. For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with*

(i) distance at least $1/2 - \varepsilon/2$ (actually ε-balanced),

(ii) rate $\Omega(\varepsilon^{2+\beta})$, and

(iii) a unique decoding algorithm with fixed polynomial running time $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$.

The list decoding framework finds a list of pairs $(z, y = \mathrm{dsum}(z))$ of size at most $N^{(1/\tau_0(\eta,k))^{O(1)}}$ at each level of the code cascade and recursively issues decoding calls to all z in this list. Since the number of lifts in the cascade is $\Omega(\log(\log(1/\varepsilon)))$, we end up with an overall running time of $N^{O_\beta(\log(\log(1/\varepsilon)))}$.

We will describe a method of pruning these lists which will lead to fixed polynomial running time. Let $1/2 - \sqrt{\eta}$, where $\eta > 0$ is a constant, be the list decoding radius used for a unique decoding step in the cascade. To achieve fixed polynomial time we will prune this polynomially large list of words to a constant size at each inductive step in Lemma 6.3. As we are working with parameters within the Johnson bound, the actual list of codewords has constant size $(1/\eta)^{O(1)}$. At every step, we will be able to find a small sublist whose size only depends on η that has a certain covering property, and then issue decoding calls to this much smaller list.

Definition 6.10 (ζ-cover). *Let $W(k) \subseteq [n]^k$, $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a code, $A \subseteq \mathcal{C}$, and $\mathcal{L} = \{(z, \mathrm{dsum}_{W(k)}(z)) \mid z \in A\}$. We say that $\mathcal{L}' = \{(z^{(1)}, \mathrm{dsum}_{W(k)}(z^{(1)})), \ldots, (z^{(m)}, \mathrm{dsum}_{W(k)}(z^{(m)}))\}$ is a ζ-cover of $\mathcal{L}$ if for every $(z, y) \in \mathcal{L}$, there exists some $(z', y') \in \mathcal{L}'$ with $\mathrm{bias}(z - z') > 1 - 2\zeta$ (that is, either $\Delta(z, z') < \zeta$ or $\Delta(z, z') > 1 - \zeta$).*

Lemma 6.11 (Cover Compactness). *Let $W(k) \subseteq [n]^k$, $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a linear η_0-balanced code, $\mathcal{C}' = \mathrm{dsum}_{W(k)}(\mathcal{C})$ be an η-balanced code, and $\tilde{y} \in \mathbb{F}_2^{W(k)}$. Define*

$$\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') := \left\{ (z, \mathrm{dsum}_{W(k)}(z)) \mid z \in \mathcal{C}, \Delta\Big(\mathrm{dsum}_{W(k)}(z), \tilde{y}\Big) \le \frac{1}{2} - \sqrt{\eta} \right\}.$$

Suppose $\mathcal{L}'$ is a ζ-cover of $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}')$ for some $\zeta < 1/2$. Further, suppose that for every $(z', y') \in \mathcal{L}'$, we have $\Delta(y', \tilde{y}) \le 1/2 - \sqrt{\eta}$. If $W(k)$ is a $(1 - 2\zeta, \eta)$-parity sampler, then there exists $\mathcal{L}'' \subseteq \mathcal{L}'$ with $|\mathcal{L}''| \le 1/\eta$ which is a (2ζ)-cover of $\mathcal{L}$.

Proof. Build a graph where the vertices are pairs $(z', y') \in \mathcal{L}'$ and two vertices $(z^{(i)}, y^{(i)})$, $(z^{(j)}, y^{(j)})$ are connected iff $\mathrm{bias}(z^{(i)} - z^{(j)}) > 1 - 2\zeta$. Let $\mathcal{L}''$ be any maximal independent set of this graph. Any two vertices $(z^{(i)}, y^{(i)}), (z^{(j)}, y^{(j)}) \in \mathcal{L}''$ have $\mathrm{bias}(z^{(i)} - z^{(j)}) \le 1 - 2\zeta$ and thus $\mathrm{bias}(y^{(i)} - y^{(j)}) \le \eta$ since $W(k)$ is a $(1 - 2\zeta, \eta)$-parity sampler. This means that $\{y'' \mid (z'', y'') \in \mathcal{L}''\}$ is a code of distance at least $1/2 - \eta/2$. By the condition that $\Delta(y'', \tilde{y}) \leq 1/2 - \sqrt{\eta}$ for all $(z'', y'') \in \mathcal{L}''$ and the Johnson bound, we have $|\mathcal{L}''| \leq 1/\eta$.

Finally, we will show that $\mathcal{L}''$ is a (2ζ)-cover of $\mathcal{L}$. Let $(z, y) \in \mathcal{L}$. As $\mathcal{L}'$ is a ζ-cover of $\mathcal{L}$, there exists a pair $(z', y') \in \mathcal{L}$ with $\text{bias}(z - z') > 1 - 2\zeta$, so z is within distance ζ of either z' or its complement $\overline{z'}$. The construction of $\mathcal{L}''$ as a maximal independent set ensures that there is some $(z'', y'') \in \mathcal{L}''$ with $\text{bias}(z' - z'') > 1 - 2\zeta$, so z'' is also within distance ζ of either z' or its complement $\overline{z'}$. Applying the triangle inequality in all of the possible cases, we see that either $\Delta(z, z'') < 2\zeta$ or $\Delta(z, z'') > 1 - 2\zeta$, which implies $\mathcal{L}''$ is a (2ζ)-cover of $\mathcal{L}$. ■

To decode in fixed polynomial time, we use a modification of the list decoding result Theorem 6.1 that outputs a ζ-cover $\mathcal{L}'$ of the list of codewords $\mathcal{L}$ instead of the list itself. Theorem 6.1 recovers the list $\mathcal{L}$ by finding $\mathcal{L}'$ and unique decoding every element of it. To get $\mathcal{L}'$, we use the same algorithm, but stop before the final decoding step. This removes the unique decoding time $f(n)$ of the base code from the running time of the list decoding algorithm. We will apply Lemma 6.11 after each time we call this ζ-cover algorithm to pare the list down to a constant size before unique decoding; note that this loses a factor of 2 in the strength of the cover. To compensate for this, we will use a collection $W(k)$ with stronger parity sampling than required for Theorem 6.1. In that theorem, $W(k)$ was a $(1/2 + \eta_0/2, \eta)$-parity sampler to ensure that we obtained words within the list decoding radius $(1/4 - \eta_0/4)$ of the base code. By using a stronger parity sampler, the words in the pruned list $\mathcal{L}''$ will still be within the unique decoding radius even after accounting for the loss in the bias from cover compactness, which means decoding will still be possible at every level of the cascade. Fortunately, improving the parity sampling only requires increasing the walk length k by a constant multiplicative factor. The cover retrieval algorithm below will be proven in Section 9.

Theorem 6.12 (Cover retrieval for direct sum lifting)**.** *Let $\eta_0 \in (0, 1/4)$ be a constant, $\eta \in (0, \eta_0)$, $\zeta = 1/8 - \eta_0/8$, and*

$$k \geq k_0'(\eta) := \Theta(\log(1/\eta)).$$

Suppose $\mathcal{C} \subseteq \mathbb{F}_2^n$ is an η_0-balanced linear code and $\mathcal{C}' = \text{dsum}_{W(k)}(\mathcal{C})$ is the direct sum lifting of $\mathcal{C}$ on a τ-splittable collection of walks $W(k)$. There exists an absolute constant $K > 0$ such that if

$$\tau \leq \tau_0(\eta, k) := \frac{\eta^8}{K \cdot k \cdot 2^{4k}},$$

then the code $\mathcal{C}'$ is η-balanced, $W(k)$ is a $(1 - 2\zeta, \eta)$-parity sampler, and we have the following:

If $\tilde{y}$ is $(1/2 - \sqrt{\eta})$-close to $\mathcal{C}'$, then we can compute a ζ-cover $\mathcal{L}'$ of the list

$$\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') := \left\{ (z, \text{dsum}_{W(k)}(z)) \mid z \in \mathcal{C}, \Delta\Big(\text{dsum}_{W(k)}(z), \tilde{y}\Big) \leq \frac{1}{2} - \sqrt{\eta} \right\}$$

in which $\Delta(y', \tilde{y}) \leq 1/2 - \sqrt{\eta}$ for every $(z', y') \in \mathcal{L}'$, in time

$$n^{O(1/\tau_0(\eta, k)^4)}.$$

Otherwise, we return $\mathcal{L}' = \emptyset$ with the same running time of the preceding case.

We now have all of the pieces necessary to prove Theorem 6.9. The process is essentially the same as our earlier unique decoding algorithm, except we use the cover retrieval algorithm from Theorem 6.12 instead of the full list decoding from Theorem 6.1. This allows us to insert a list pruning step in between obtaining the ζ-cover and calling the unique decoding algorithm for the previous level of the cascade.

Proof of Theorem 6.9. We use the code $\mathcal{C}_{N,\varepsilon,\beta}$ from Theorem 6.7 to get the desired distance and rate, with the slight modification of ensuring s is larger than k_0' from Theorem 6.12 rather than k_0 from Theorem 6.1.

Each level of the code cascade between $\mathcal{C}_{i-1}$ and $\mathcal{C}_i$ uses constant $\eta_0 < 1/4$ and $\eta < \min\{\eta_0, 1/16\}$, which allows for decoding in a similar fashion to Lemma 6.2 and Lemma 6.3. The difference is that we use Theorem 6.12 as the decoding step to obtain a ζ-cover $\mathcal{L}'$ of the list $\mathcal{L}(\tilde{y}, \mathcal{C}_{i-1}, \mathcal{C}_i)$ for $\tilde{y} \in \mathbb{F}_2^{n_i}$, where $\zeta = 1/8 - \eta_0/8$. By Lemma 6.11 and the fact that the walk collection is a $(1-2\zeta, \eta)$-parity sampler, $\mathcal{L}$ has a (2ζ)-cover $\mathcal{L}'' \subseteq \mathcal{L}'$ of size at most $1/\eta$. The covering property says that for every $(z, y) \in \mathcal{L}$, there exists some $(z'', y'') \in \mathcal{L}''$ such that z is within distance $2\zeta = 1/4 - \eta_0/4$ of either z'' or its complement $\overline{z''}$. This is the unique decoding radius of the η_0-balanced code $\mathcal{C}_{i-1}$, so we can recursively decode the list

$$\mathcal{L}'' \cup \{(\overline{z''}, \mathrm{dsum}(\overline{z''})) \mid (z'', \mathrm{dsum}(z'')) \in \mathcal{L}''\}$$

to obtain the complete list of codewords in $\mathcal{C}_{i-1}$.

Now we analyze the running time. On each level of the code cascade, we run the cover retrieval algorithm once to get $\mathcal{L}'$, prune the cover to get $\mathcal{L}''$, and then feed the union of $\mathcal{L}''$ and its complement (which has size at most $2/\eta$) into the unique decoding algorithm for the next level of the cascade. Letting $T_i(n_i)$ be the running time of unique decoding a single word in the code $\mathcal{C}_i \subseteq \mathbb{F}_2^{n_i}$, we have the following recurrence:

$$T_i(n_i) \le n_i^{O(1/\tau_0(\eta,k)^4)} + \frac{2}{\eta} \cdot T_{i-1}(n_{i-1}) \quad \text{and} \quad T_0(n_0) = n_0^{O(1)}.$$

Note that the base code $\mathcal{C}_0$ has constant bias ε_0 and thus it has a fixed polynomial time decoding algorithm (e.g. Theorem 6.7). The height of the recursive call tree is the number of levels in the code cascade, which is $\ell = O(\log(\log(1/\varepsilon)))$, as in the proof of Theorem 6.6. Each node of this tree has a constant branching factor of $2/\eta$. Thus, the tree has $(\log(1/\varepsilon))^{O(1)}$ nodes, each of which costs at most $n_i^{O(1/\tau_0(\eta,k)^4)} \le N^{O(1/\tau_0(\eta,k)^4)}$ time. Furthermore, in this regime of $\beta > 0$ being a constant, k is constant as well as η, so we have $N^{O(1/\tau_0(\eta,k)^4)} = N^{O_\beta(1)}$ and the total running time is $(\log(1/\varepsilon))^{O(1)} \cdot N^{O_\beta(1)}$. ■

7 Satisfying the List Decoding Framework Requirements

The list decoding framework of [AJQ+20] is capable of decoding codes obtained from direct sum liftings, provided they satisfy a few requisite properties. The framework was originally shown to work for expander walks; we need to adapt it to our case of a code cascade based on walks on the s-wide replacement product. We will start with a broad overview of the list decoding algorithm and point out where various requirements arise.

The problem of finding a list of codewords in a direct sum lifting close to a received word can be viewed as finding approximate solutions to a k-XOR instance. This is done by solving a particular SOS program and rounding the resulting solution. The algorithm is unable to perform rounding if the k-XOR instance is based on an arbitrary collection of walks $W(k)$; it can only handle liftings in which $W(k)$ satisfies a property called *tensoriality*. If $W(k)$ is tensorial, the SOS local variables in the solution can be approximated by product distributions, which will allow us to obtain a list of solutions by independent rounding. Tensoriality for expander walks is a consequence of a simpler property known as *splittability*, which is a certain measure of the expansion of a walk collection.

Unfortunately, the list returned by the rounding process will not contain codewords directly—instead, we only get a guarantee that all of the codewords we are looking for have a weak agreement (just over $1/2$) with something on this list. We will find the desired codewords by relying on the parity sampling of $W(k)$. If $W(k)$ is a sufficiently good parity sampler, weak agreement in the lifted space corresponds to a much stronger agreement in the ground space. This will allow us to recover the codewords using the unique decoding algorithm of the base code.

To recap, applying the list decoding framework in our setting requires doing the following:

1. Proving parity sampling for the walks used in the code cascade (Section 7.1).
2. Showing that the walk collection of the s-wide replacement product is splittable (Section 7.2).
3. Making Ta-Shma's construction compatible with the Sum-of-Squares machinery (Section 7.3) and then obtaining tensoriality from splittability (Section 7.4).

An additional complication is introduced by using a code cascade instead of a single decoding step: the above requirements need to be satisfied at every level of the cascade. The details of the proofs will often differ between the first level of the cascade, which is constructed using walks on the s-wide replacement product, and higher levels, which are walks on a directed graph whose vertices are walks themselves. Once we have established all of the necessary properties, we will instantiate the list decoding framework in Section 9.

We will first define some convenient notation which will be used throughout this section.

Notation 7.1. *Let G be a d_1-regular outer graph and H be a d_2-regular inner graph used in Ta-Shma's s-wide replacement product.*

Let $0 \leq k_1 \leq k_2$ be integers. We define $W[k_1, k_2]$ to be the set of all walks starting at time k_1 and ending at time k_2 in Ta-Shma's construction. More precisely, since G and H are regular graphs, the collection $W[k_1, k_2]$ contains all walks obtained by sampling a uniform vertex $(v, h) \in V(G) \times V(H)$ and applying the operator

$$(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{k_2-1}(I \otimes \mathsf{A}_H) \cdots (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_{k_1}(I \otimes \mathsf{A}_H),$$

where the index i of each G_i is taken modulo s. Observe that when $k_1 = k_2$, we have $W[k_1, k_2] = V(G) \times V(H)$.

We define a family of Markov operators which will play a similar role to the graphs $\widehat{G}_i$ from the cascade described in Section 5.1, but for Ta-Shma's construction rather than expander walks.

Definition 7.2 (Split Operator)**.** *Let $0 \leq k_1 \leq k_2 < k_3$. We define the graph walk split operator*

$$\mathsf{S}_{k_1,k_2,k_3} \colon \mathbb{R}^{W[k_2+1,k_3]} \to \mathbb{R}^{W[k_1,k_2]}$$

such that for every $f \in \mathbb{R}^{W[k_2+1,k_3]}$,

$$(\mathsf{S}_{k_1,k_2,k_3}(f))\,(w) := \mathbb{E}_{w' : ww' \in W[k_1,k_3]}[f(w')],$$

where ww' denotes the concatenation of the walks w and w'. The operator S_{k_1,k_2,k_3} can be defined more concretely in matrix form such that for every $w \in W[k_1,k_2]$ and $w' \in W[k_2+1,k_3]$,

$$(\mathsf{S}_{k_1,k_2,k_3})_{w,w'} = \frac{\mathbb{1}_{ww' \in W[k_1,k_3]}}{|\{\tilde{w} : w\tilde{w} \in W[k_1,k_3]\}|} = \frac{\mathbb{1}_{ww' \in W[k_1,k_3]}}{d_2^{2(k_3-k_2)}}.$$

7.1 Parity Sampling for the Code Cascade

To be able to apply the list decoding machinery to the code cascade $\mathcal{C}_0 \subseteq \mathbb{F}_2^{n_0}, \mathcal{C}_1 \subseteq \mathbb{F}_2^{n_1}, \ldots, \mathcal{C}_\ell \subseteq \mathbb{F}_2^{n_\ell}$, we need the direct sum lifting at every level to be a parity sampler. The first level in the cascade uses walks directly on the s-wide replacement product, which we can show is a good parity sampler using the spectral properties proven in Section 4.3.1. However, it will be more convenient for calculating parameters later on to prove a weaker result, which will suffice for our purposes since we only need to obtain constant bias for every level of the cascade. We analyze the parity sampling of these walks with the same strategy Ta-Shma employed to show parity sampling for walks on expander graphs (which resulted in Theorem 5.2).

Claim 7.3. *Let $W[0, s-1]$ be the collection of walks on the s-wide replacement product of the graphs G and H and $z \in \mathbb{F}_2^{V(G)}$ be a word with* $\mathrm{bias}(z) \leq \eta_0$. *Let P_z be the diagonal matrix with entries $(\mathsf{P}_z)_{(v,h),(v,h)} = (-1)^{z_v}$ for $(v,h) \in V(G) \times V(H)$. If $\sigma_2((\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)) \leq \gamma$ for all $0 \leq i \leq s-2$, then*

$$\left\| \prod_{i=0}^{s-2} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{P}_z \right\|_2 \leq (\eta_0 + 2\gamma)^{\lfloor (s-1)/2 \rfloor}.$$

Proof. Let $0 \leq j < s-2$ be even. Take a vector $v \in \mathbb{R}^{V(G)\times V(H)}$ with $\|v\|_2 = 1$ and let $v^{\|}$ and $v^{\perp}$ be its parallel and orthogonal components to the all ones vector. For $0 \leq i \leq s-2$, let $\mathsf{A}_i = (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)$. Consider two terms $\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j\mathsf{P}_z$ of the product appearing in the claim. Since P_z is unitary, $\|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j\mathsf{P}_z\|_2 = \|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j\|_2$. We have

$$\begin{aligned}
\|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j v\|_2 &\leq \left\|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j v^{\|}\right\|_2 + \left\|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j v^{\perp}\right\|_2 \\
&\leq \left\|\mathsf{A}_{j+1}\mathsf{P}_z\mathsf{A}_j v^{\|}\right\|_2 + \left\|\mathsf{A}_j v^{\perp}\right\|_2 \\
&\leq \left\|\mathsf{A}_{j+1}\mathsf{P}_z v^{\|}\right\|_2 + \sigma_2(\mathsf{A}_j) \\
&\leq \left\|\mathsf{A}_{j+1}(\mathsf{P}_z v^{\|})^{\|}\right\|_2 + \left\|\mathsf{A}_{j+1}(\mathsf{P}_z v^{\|})^{\perp}\right\|_2 + \sigma_2(\mathsf{A}_j) \\
&\leq \left\|(\mathsf{P}_z v^{\|})^{\|}\right\|_2 + \sigma_2(\mathsf{A}_{j+1}) + \sigma_2(\mathsf{A}_j) \\
&\leq \eta_0 + 2\gamma.
\end{aligned}$$

Applying this inequality to every two terms of the product, the result follows. ∎

Corollary 7.4. *Let $W[0, s-1]$ be the collection of walks on the s-wide replacement product of the graphs G and H and $\eta_0 > 0$. If $\sigma_2((\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)) \leq \gamma$ for all $0 \leq i \leq s-2$, then $W[0, s-1]$ is an (η_0, η)-parity sampler, where $\eta = (\eta_0 + 2\gamma)^{\lfloor (s-1)/2 \rfloor}$.*

Proof. Let $z \in \mathbb{F}_2^n$ have bias at most η_0. The bias of $\mathrm{dsum}_{W[0,s-1]}(z)$ is given by [7]

$$\mathrm{bias}(\mathrm{dsum}_{W[0,s-1]}(z)) = \left| \left\langle \mathbf{1}, \mathsf{P}_z \left(\prod_{i=0}^{s-2} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{P}_z \right) \mathbf{1} \right\rangle \right|,$$

where P_z is the diagonal matrix with entries $(\mathsf{P}_z)_{(v,h),(v,h)} = (-1)^{z_v}$ for $(v,h) \in V(G) \times V(H)$ and $\mathbf{1}$ is the all-ones vector. Since P_z is unitary, we have

$$\mathrm{bias}(\mathrm{dsum}_{W[0,s-1]}(z)) \leq \left\| \prod_{i=0}^{s-2} (\mathsf{I} \otimes \mathsf{A}_H)\mathsf{G}_i(\mathsf{I} \otimes \mathsf{A}_H)\mathsf{P}_z \right\|_2 \leq (\eta_0 + 2\gamma)^{\lfloor (s-1)/2 \rfloor} = \eta$$

by Claim 7.3. Hence $W[0, s-1]$ is an (η_0, η)-parity sampler. ■

For higher levels of the cascade, we need to prove parity sampling for collections of walks over walks. Since the walks on the first level contain s vertices, when we take walks on higher levels, the operator linking different walks together will always use G_{s-1} as the walk operator for the G step. Thus we can consider a more specific form of the split operator where we split at a time parameter that is one less than a multiple of s.

Definition 7.5. *Let $r \equiv -1 \pmod{s}$ be a positive integer. We define the operator $\mathsf{S}^{\triangle}_{r,r}$ as*

$$\mathsf{S}^{\triangle}_{r,r} = \mathsf{S}_{k_1,k_2,k_3},$$

where $k_1 = 0$, $k_2 = r$, and $k_3 = 2r+1$. In this case, $W[k_1, k_2] = W[k_2+1, k_3]$.

All levels of the code cascade beyond the first use walks generated by the directed operator $\mathsf{S}^{\triangle}_{r,r}$. Proving parity sampling for these walks is analogous to the proof of Corollary 7.4, but slightly simpler since the walk operator doesn't change with each step.

Claim 7.6. *Let $r \equiv -1 \pmod{s}$ be a positive integer and $z \in \mathbb{F}_2^{W[0,r]}$ be a word with $\mathrm{bias}(z) \leq \eta_0$. Let $\widetilde{\mathsf{P}}_z$ be the diagonal matrix with entries $(\widetilde{\mathsf{P}}_z)_{w,w} = (-1)^{z_w}$ for $w \in W[0,r]$. For every integer $k \geq 1$, we have*

$$\left\| \left(\mathsf{S}^{\triangle}_{r,r} \widetilde{\mathsf{P}}_z \right)^{k-1} \right\|_2 \leq \left(\eta_0 + 2 \cdot \sigma_2 \left(\mathsf{S}^{\triangle}_{r,r} \right) \right)^{\lfloor (k-1)/2 \rfloor}.$$

Proof. Take a vector $v \in \mathbb{R}^{W[0,r]}$ with $\|v\|_2 = 1$ and let $v^{\parallel}$ and $v^{\perp}$ be its parallel and orthogonal components to the all ones vector. Since $\widetilde{\mathsf{P}}_z$ is unitary, $\left\| \mathsf{S}^{\triangle}_{r,r} \widetilde{\mathsf{P}}_z \mathsf{S}^{\triangle}_{r,r} \widetilde{\mathsf{P}}_z \right\|_2 = \left\| \mathsf{S}^{\triangle}_{r,r} \widetilde{\mathsf{P}}_z \mathsf{S}^{\triangle}_{r,r} \right\|_2$.

[7] This is slightly different from the expression for the bias given in Section 4.3, but both are equal since moving on the H component of the graph doesn't affect the bit assigned to a vertex.

We have

$$\begin{aligned}
\left\|\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z\mathsf{S}^{\triangle}_{r,r}v\right\|_2 &\le \left\|\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z\mathsf{S}^{\triangle}_{r,r}v^{\parallel}\right\|_2 + \left\|\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z\mathsf{S}^{\triangle}_{r,r}v^{\perp}\right\|_2\\
&\le \left\|\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z\mathsf{S}^{\triangle}_{r,r}v^{\parallel}\right\|_2 + \left\|\mathsf{S}^{\triangle}_{r,r}v^{\perp}\right\|_2\\
&\le \left\|\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z v^{\parallel}\right\|_2 + \sigma_2(\mathsf{S}^{\triangle}_{r,r})\\
&\le \left\|\mathsf{S}^{\triangle}_{r,r}(\widetilde{\mathsf{P}}_z v^{\parallel})^{\parallel}\right\|_2 + \left\|\mathsf{S}^{\triangle}_{r,r}(\widetilde{\mathsf{P}}_z v^{\parallel})^{\perp}\right\|_2 + \sigma_2(\mathsf{S}^{\triangle}_{r,r})\\
&\le \left\|(\widetilde{\mathsf{P}}_z v^{\parallel})^{\parallel}\right\|_2 + \sigma_2(\mathsf{S}^{\triangle}_{r,r}) + \sigma_2(\mathsf{S}^{\triangle}_{r,r})\\
&\le \eta_0 + 2\cdot\sigma_2(\mathsf{S}^{\triangle}_{r,r}).
\end{aligned}$$

As $\left\|(\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z)^{k-1}\right\|_2 \le \left\|(\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z)^2\right\|^{\lfloor (k-1)/2\rfloor}$, the result follows. ■

Corollary 7.7. *Let $r \equiv -1 \pmod{s}$ be a positive integer and $\eta_0 > 0$. The collection of walks $W(k)$ with k vertices over the vertex set $W[0,r]$ using random walk operator $\mathsf{S}^{\triangle}_{r,r}$ is an (η_0,η)-parity sampler, where $\eta = (\eta_0 + 2\cdot\sigma_2(\mathsf{S}^{\triangle}_{r,r}))^{\lfloor (k-1)/2\rfloor}$.*

Proof. Let $z \in \mathbb{F}_2^{W[0,r]}$ have bias at most η_0. The bias of the direct sum lifting of z is given by

$$\mathrm{bias}(\mathrm{dsum}_{W(k)}(z)) = \left|\left\langle \mathbf{1}, \widetilde{\mathsf{P}}_z(\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z)^{k-1}\mathbf{1}\right\rangle\right|,$$

where $\widetilde{\mathsf{P}}_z$ is the diagonal matrix with entries $(\widetilde{\mathsf{P}}_z)_{w,w} = (-1)^{z_w}$ for $w \in W[0,r]$ and $\mathbf{1}$ is the all-ones vector. Since $\widetilde{\mathsf{P}}_z$ is unitary, we have

$$\left|\left\langle \mathbf{1}, \widetilde{\mathsf{P}}_z(\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z)^{k-1}\mathbf{1}\right\rangle\right| \le \left\|\left(\mathsf{S}^{\triangle}_{r,r}\widetilde{\mathsf{P}}_z\right)^{k-1}\right\|_2 \le \left(\eta_0 + 2\cdot\sigma_2\left(\mathsf{S}^{\triangle}_{r,r}\right)\right)^{\lfloor (k-1)/2\rfloor} = \eta$$

by Claim 7.6. Hence $W(k)$ is an (η_0,η)-parity sampler. ■

7.2 Splittability of Ta-Shma's Construction

We investigate the splittability of the collection of walks generated by Ta-Shma's construction. In order to formally define this property, we will need the concept of an interval splitting tree, which describes how a walk is split into smaller and smaller pieces.

Definition 7.8 (Interval Splitting Tree)**.** *We say that a binary rooted tree $\mathcal{T}$ is a k-interval splitting tree if it has exactly k leaves and*

- *the root of $\mathcal{T}$ is labeled with $(0, m, k-1)$ for some $m \in \{0, 1, \ldots, k-2\}$, and*
- *each non-leaf non-root vertex v of $\mathcal{T}$ is labeled with (k_1, k_2, k_3) for some integer $k_2 \in [k_1, k_3 - 1]$. Suppose (k_1', k_2', k_3') is the label assigned to the parent of v. If v is a left child, we must have $k_1 = k_1'$ and $k_3 = k_2'$; otherwise, we must have $k_1 = k_2' + 1$ and $k_3 = k_3'$.*

Given an interval splitting tree $\mathcal{T}$, we can naturally associate a split operator S_{k_1,k_2,k_3} to each internal node (k_1, k_2, k_3). The splittability of a collection $W[0, k-1]$ of k-tuples is a notion of expansion at every node in the splitting tree.

Definition 7.9 (($\mathcal{T}, \tau$)-splittability)**.** *The collection $W[0, k-1]$ is said to be $(\mathcal{T}, \tau)$-splittable if $\mathcal{T}$ is a k-interval splitting tree and*

$$\sigma_2(\mathsf{S}_{k_1,k_2,k_3}) \leq \tau$$

for every internal node (k_1, k_2, k_3) of $\mathcal{T}$.

If there exists some k-interval splitting tree $\mathcal{T}$ such that $W[0, k-1]$ is $(\mathcal{T}, \tau)$-splittable, then $W[0, k-1]$ will be called τ-splittable.

In order to prove that the collection of walks in Ta-Shma's construction is splittable, a split operator S_{k_1,k_2,k_3} can be related to the walk operator $(I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)$ as shown below. This structural property will allow us to deduce spectral properties of S_{k_1,k_2,k_3} from the spectrum of $(I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)$.

Lemma 7.10. *Let $0 \leq k_1 \leq k_2 < k_3$. Suppose G is a d_1-regular outer graph on vertex set $[n]$ with walk operator G_{k_2} used at step k_2 of a walk on the s-wide replacement product and H is a d_2-regular inner graph on vertex set $[m]$ with normalized random walk operator A_H. Then there are orderings of the rows and columns of the representations of S_{k_1,k_2,k_3} and A_H as matrices such that*

$$\mathsf{S}_{k_1,k_2,k_3} = ((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)) \otimes \mathsf{J}/d_2^{2(k_3-k_2-1)},$$

where $\mathsf{J} \in \mathbb{R}^{[d_2]^{2(k_2-k_1)} \times [d_2]^{2(k_3-k_2-1)}}$ is the all ones matrix.

Proof. Partition the set of walks $W[k_1, k_2]$ into the sets $W_{1,1}, \ldots, W_{n,m}$, where $w \in W_{i,j}$ if the last vertex of the walk $w_{k_2} = (v_{k_2}, h_{k_2})$ satisfies $v_{k_2} = i$ and $h_{k_2} = j$. Similarly, partition $W[k_2+1, k_3]$ into the sets $W'_{1,1}, \ldots, W'_{n,m}$, where $w' \in W'_{i,j}$ if the first vertex of the walk $w'_1 = (v_1, h_1)$ satisfies $v_1 = i$ and $h_1 = j$. Note that $|W_{i,j}| = d_2^{2(k_2-k_1)}$ and $\left|W'_{i,j}\right| = d_2^{2(k_3-k_2-1)}$ for all $(i,j) \in [n] \times [m]$, since there are d_2^2 choices for each step of the walk.

Now order the rows of the matrix S_{k_1,k_2,k_3} so that all of the rows corresponding to walks in $W_{1,1}$ appear first, followed by those for walks in $W_{1,2}$, and so on in lexicographic order of the indices (i,j) of $W_{i,j}$, with an arbitrary order within each set. Do a similar re-ordering of the columns for the sets $W'_{1,1}, \ldots, W'_{1,m}$. Observe that

$$\begin{aligned}(\mathsf{S}_{k_1,k_2,k_3})_{w,w'} &= \frac{\mathbb{1}_{ww' \in W[k_1,k_3]}}{d_2^{2(k_3-k_2)}} \\ &= \frac{d_2^2 \cdot (\text{weight of transition from } (v_{k_2}, h_{k_2}) \text{ to } (v'_1, h'_1) \text{ in } (I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H))}{d_2^{2(k_3-k_2)}},\end{aligned}$$

which only depends on the adjacency of the last vertex of w and the first vertex of w'. If the vertices $w_{k_2} = (v_{k_2}, h_{k_2})$ and $w'_1 = (v_1, h_1)$ are adjacent, then

$$(\mathsf{S}_{k_1,k_2,k_3})_{w,w'} = ((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H))_{(v_{k_2},h_{k_2}),(v'_1,h'_1)} / d_2^{2(k_3-k_2-1)},$$

for every $w \in W_{w_{k_2}}$ and $w' \in W'_{w_{k_1}}$; otherwise, $(\mathsf{S}_{k_1,k_2,k_3})_{w,w'} = 0$. Since the walks in the rows and columns are sorted according to their last and first vertices, respectively, the matrix S_{k_1,k_2,k_3} exactly matches the tensor product $((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)) \otimes \mathsf{J}/d_2^{2(k_3-k_2-1)}$.
■

Corollary 7.11. *Let $0 \leq k_1 \leq k_2 < k_3$. Suppose G is a d_1-regular outer graph with walk operator G_{k_2} used at step k_2 of a walk on the s-wide replacement product and H is a d_2-regular inner graph with normalized random walk operator A_H. Then*

$$\sigma_2(\mathsf{S}_{k_1,k_2,k_3}) = \sigma_2((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)).$$

Proof. Using Lemma 7.10 and the fact that

$$\sigma_2(((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)) \otimes \mathsf{J}/d_2^{2(k_3-k_2-1)}) = \sigma_2((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)),$$

the result follows. ■

Remark 7.12. *Corollary 7.11 is what causes the splittability argument to break down for Ta-Shma's original construction, as $\sigma_2(\mathsf{G}_{k_2}(\mathsf{I} \otimes \mathsf{A}_H)) = 1$.*

By combining this result with the spectral bound from Fact 4.4, we find that the collection of walks of length s on the s-wide replacement product is $(\mathcal{T}, \tau)$-splittable for any splitting tree $\mathcal{T}$, where τ is controlled by the second singular values of the graphs G and H. This analysis can also be applied to walks on higher levels of the cascade where the vertex set is $W[0, r]$.

Corollary 7.13 (Restatement of Lemma 6.5)**.** *The collection of walks $W[0, s-1]$ on the s-wide replacement product with outer graph G and inner graph H and the collection of walks $W(k)$ on the vertex set $W[0, r]$ with random walk operator $\mathsf{S}^{\triangle}_{r,r}$ and $r \equiv -1 \pmod{s}$ are both τ-splittable with $\tau = \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2$.*

Proof. By Corollary 7.11 and Fact 4.4, the split operator S_{k_1,k_2,k_3} for any $0 \leq k_1 \leq k_2 < k_3$ satisfies

$$\sigma_2(\mathsf{S}_{k_1,k_2,k_3}) = \sigma_2((I \otimes \mathsf{A}_H)G_{k_2}(I \otimes \mathsf{A}_H)) \leq \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2,$$

so $W[0, s-1]$ is τ-splittable with $\tau = \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2$, as any internal node (k_1, k_2, k_3) of any s-interval splitting tree will have $\sigma_2(\mathsf{S}_{k_1,k_2,k_3}) \leq \tau$. The split operators of any k-interval splitting tree for the collection $W(k)$ are of the form S_{k_1,k_2,k_3} with $k_1 \equiv 0 \pmod{s}$ and $k_2, k_3 \equiv -1 \pmod{s}$, which means $W(k)$ is τ-splittable as well. ■

7.3 Integration with Sum-of-Squares

Before defining tensoriality and obtaining it in our setting, we examine how the Sum-of-Squares hierarchy is used in the list decoding algorithm in more detail.

7.3.1 SOS Preliminaries: p-local PSD Ensembles

The SOS hierarchy gives a sequence of increasingly tight semidefinite programming relaxations for several optimization problems, including CSPs. Since we will use relatively few facts about the SOS hierarchy, already developed in the analysis of Barak, Raghavendra and Steurer [BRS11], we will adapt their notation of *p-local distributions* to describe the relaxations.

Solutions to a semidefinite relaxation of a CSP on n boolean variables using p levels of the SOS hierarchy induce probability distributions μ_S over $\mathbb{F}_2^S$ for any set $S \subseteq [n]$ with $|S| \leq p$. These distributions are consistent on intersections: for $T \subseteq S \subseteq [n]$, we have $\mu_{S|T} = \mu_T$, where $\mu_{S|T}$ denotes the restriction of the distribution μ_S to the set T. We use these distributions to define a collection of random variables $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$ taking values in $\mathbb{F}_2$ such that for any set S with $|S| \leq p$, the collection of variables $\{\mathbf{Z}_i\}_{i \in S}$ has joint distribution μ_S. Note that the entire collection $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_n\}$ *may not* have a joint distribution: this property is only true for sub-collections of size at most p. We will refer to the collection $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_n\}$ as a *p-local ensemble* of random variables.

For any $T \subseteq [n]$ with $|T| \leq p - 2$ and any $\xi \in \mathbb{F}_2^T$, we can define a $(p - |T|)$-local ensemble $\{\mathbf{Z}'_1, \ldots, \mathbf{Z}'_n\}$ by "conditioning" the local distributions on the event $\mathbf{Z}_T = \xi$, where $\mathbf{Z}_T$ is shorthand for the collection $\{\mathbf{Z}_i\}_{i \in T}$. For any S with $|S| \leq p - |T|$, we define the distribution of $\mathbf{Z}'_S$ as $\mu'_S := \mu_{S \cup T} | \{\mathbf{Z}_T = \xi\}$.

Finally, the semidefinite program also ensures that for any such conditioning, the conditional covariance matrix

$$\mathsf{M}_{(S_1,\alpha_1)(S_2,\alpha_2)} \;=\; \operatorname{Cov}\left(\mathbb{1}_{[\mathbf{Z}'_{S_1}=\alpha_1]}, \mathbb{1}_{[\mathbf{Z}'_{S_2}=\alpha_2]}\right)$$

is positive semidefinite, where $|S_1|, |S_2| \leq (p - |T|)/2$. Here, for each pair S_1, S_2 the covariance is computed using the joint distribution $\mu'_{S_1 \cup S_2}$. In this paper, we will only consider p-local ensembles such that for every conditioning on a set of size at most $(p-2)$, the conditional covariance matrix is PSD. We will refer to these as *p-local PSD ensembles*. We will also need a simple corollary of the above definitions.

Fact 7.14. *Let $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_n\}$ be a p-local PSD ensemble and $W(k) \subseteq [n]^k$ For $1 \leq i < k$, define $W(i) \subseteq [n]^i$ to be the collection of tuples of size i appearing in elements of $W(k)$. For all $p' \leq p/2$, the collection $\left\{\mathbf{Z}_{\mathrm{set}(w)}\right\}_{w \in W(\leq p')}$ is a (p/p')-local PSD ensemble, where $W(\leq p') = \bigcup_{i=1}^{p'} W(i)$.*

For random variables $\mathbf{Z}_S$ in a p-local PSD ensemble, we use the notation $\{\mathbf{Z}_S\}$ to denote the distribution of $\mathbf{Z}_S$ (which exists when $|S| \leq p$). As we will work with ordered tuples of variables instead of sets, we define $\mathbf{Z}_w$ for $w \in [n]^k$ based on the set $S_w = \mathrm{set}(w)$, taking care that repeated elements of w are always assigned the same value.

Definition 7.15 (Plausible assignment)**.** *Given $w = (w_1, \ldots, w_k) \in [n]^k$ and an assignment $\alpha \in \mathbb{F}_2^w$, we say that α is plausible for w if there are no distinct $i, j \in [k]$ such that $w_i = w_j$ but $\alpha_i \neq \alpha_j$.*

The distribution $\{\mathbf{Z}_w\} = \mu_w$ is defined as $\mu_w(\alpha) = \mu_{S_w}(\alpha|_{S_w})$ if $\alpha \in \mathbb{F}_2^w$ is plausible for w, and $\mu_w(\alpha) = 0$ otherwise.

7.3.2 Tensoriality

A key algorithm in the list decoding framework is propagation rounding (Algorithm 7.16), which solves a CSP to find solutions close to a codeword. Suppose $W(k) \subseteq [n]^k$ is a collection of walks, or more generally, a collection of any k-tuples. The algorithm starts with a local PSD ensemble $\{\mathbf{Z}_1, \ldots, \mathbf{Z}_n\}$ which is the solution to an SOS program for list decoding. Propagation rounding takes this solution and conditions some of the variables according to a random assignment to these variables to yield another local PSD ensemble $\mathbf{Z}'$.

Algorithm 7.16 (Propagation Rounding Algorithm, adapted from [AJQ+20]).

```
Input   An $(L+2k)$-local PSD ensemble $\{\mathbf{Z}_1, \dots, \mathbf{Z}_n\}$ and collection $W(k) \subseteq [n]^k$.
Output  A random assignment $(\sigma_1, \dots, \sigma_n) \in \mathbb{F}_2^n$ and $2k$-local PSD ensemble $\mathbf{Z}'$.

1. Choose $m \in \{1, \dots, L/k\}$ uniformly at random.
2. For $j = 1, \dots, m$, sample a walk $w_j$ independently and uniformly from $W(k)$.
3. Write $S = \bigcup_{j=1}^{m} \mathrm{set}(w_j)$ for the set of the seed vertices.
4. Sample an assignment $\sigma : S \to \mathbb{F}_2$ according to the local distribution $\{\mathbf{Z}_S\}$.
5. Set $\mathbf{Z}' = \{\mathbf{Z}_1, \dots, \mathbf{Z}_n | \mathbf{Z}_S = \sigma\}$, i.e. the local ensemble $\mathbf{Z}$ conditioned on agreeing with $\sigma$.
6. For all $i \in [n]$, sample independently $\sigma_i \sim \{\mathbf{Z}'_i\}$.
7. Output $(\sigma_1, \dots, \sigma_n)$ and $\mathbf{Z}'$.
```

If the collection $W(k) \subseteq [n]^k$ used in the direct sum lifting is amenable to SOS rounding, the conditioned ensemble $\mathbf{Z}'$ will be able to recover a word close to some codeword on the list. This is quantified by the following *tensorial* properties. We will see shortly how splittability will be used to obtain tensoriality in our setting.

Definition 7.17 (Tensorial Walk Collection). *Let $W(k) \subseteq [n]^k$, $\mu \in [0,1]$, and $L \in \mathbb{N}$. Define Ω to be the set of all tuples (m, S, σ) obtainable in propagation rounding (Algorithm 7.16) on $W(k)$ with SOS degree parameter L. We say that $W(k)$ is (μ, L)-tensorial if the local PSD ensemble $\mathbf{Z}'$ returned by propagation rounding satisfies*

$$\mathop{\mathbb{E}}_{\Omega} \mathop{\mathbb{E}}_{w \in W(k)} \left\| \{\mathbf{Z}'_w\} - \left\{\mathbf{Z}'_{w(1)}\right\} \cdots \left\{\mathbf{Z}'_{w(k)}\right\} \right\|_1 \leq \mu. \tag{4}$$

The framework actually uses a strengthening of the above property, in which variables for pairs of walks chosen independently approximately behave as a product.

Definition 7.18 (Two-Step Tensorial Walk Collection). *Let $W(k) \subseteq [n]^k$, $\mu \in [0,1]$, and $L \in \mathbb{N}$. Define Ω to be the set of all tuples (m, S, σ) obtainable in propagation rounding (Algorithm 7.16) on $W(k)$ with SOS degree parameter L. We say that $W(k)$ is (μ, L)-two-step tensorial if it is (μ, L)-tensorial and the local PSD ensemble $\mathbf{Z}'$ returned by propagation rounding satisfies the additional condition*

$$\mathop{\mathbb{E}}_{\Omega} \mathop{\mathbb{E}}_{w, w' \in W(k)} \left\| \{\mathbf{Z}'_w \mathbf{Z}'_{w'}\} - \{\mathbf{Z}'_w\}\{\mathbf{Z}'_{w'}\} \right\|_1 \leq \mu.$$

7.3.3 From Directed to Undirected

In order to apply the list decoding framework using the directed split operator S_{k_1,k_2,k_3}, we will replace it with the symmetrized version

$$\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3}) = \begin{pmatrix} 0 & \mathsf{S}_{k_1,k_2,k_3} \\ (\mathsf{S}_{k_1,k_2,k_3})^{\dagger} & 0 \end{pmatrix}$$

and show how $\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})$ corresponds to a particular undirected graph.

Definition 7.19. *Let $0 \le k_1 \le k_2 < k_3$. We define the operator $\mathfrak{S}_{k_2,k_3,k_1} : \mathbb{R}^{W[k_1,k_2]} \to \mathbb{R}^{W[k_2+1,k_3]}$ such that for every $f \in \mathbb{R}^{W[k_1,k_2]}$,*

$$\left(\mathfrak{S}_{k_2,k_3,k_1}(f)\right)(w') := \mathbb{E}_{w:ww' \in W[k_1,k_3]}[f(w)],$$

for every $w' \in W[k_2+1,k_3]$.

The operator $\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})$ defines an undirected weighted bipartite graph on the vertices $W[k_1,k_2] \cup W[k_2+1,k_3]$. We can see that $\mathfrak{S}_{k_2,k_3,k_1}$ is the adjoint of S_{k_1,k_2,k_3}, which means that each edge ww' in this graph is weighted according to the transition probability from one walk to the other whenever one of w, w' is in $W[k_1,k_2]$ and the other is in $W[k_2+1,k_3]$.

Claim 7.20.

$$(\mathsf{S}_{k_1,k_2,k_3})^\dagger = \mathfrak{S}_{k_2,k_3,k_1}.$$

Proof. Let $f \in C^{W[k_1,k_2]}$ and $g \in C^{W[k_2+1,k_3]}$. For $i \le j$, define $\Pi_{i,j}$ to be the uniform distribution on $W[i,j]$. We show that $\langle f, \mathsf{S}_{k_1,k_2,k_3} g\rangle = \langle \mathfrak{S}_{k_2,k_3,k_1} f, g\rangle$. On one hand we have

$$\begin{aligned}
\langle f, \mathsf{S}_{k_1,k_2,k_3} g\rangle &= \mathbb{E}_{w \in W[k_1,k_2]}\left[f(w)\mathbb{E}_{w':ww' \in W[k_1,k_3]}[g(w')]\right] \\
&= \mathbb{E}_{w \in W[k_1,k_2]}\left[f(w) \sum_{w' \in W[k_2+1,k_3]} \frac{\Pi_{k_1,k_3}(ww')}{\Pi_{k_1,k_2}(w)} g(w')\right] \\
&= \sum_{w \in W[k_1,k_2]} \Pi_{k_1,k_2}(w) f(w) \sum_{w' \in W[k_2+1,k_3]} \frac{\Pi_{k_1,k_3}(ww')}{\Pi_{k_1,k_2}(w)} g(w') \\
&= \sum_{ww' \in W[k_1,k_3]} f(w) g(w') \Pi_{k_1,k_3}(ww').
\end{aligned}$$

On the other hand we have

$$\begin{aligned}
\langle \mathfrak{S}_{k_2,k_3,k_1} f, g\rangle &= \mathbb{E}_{w' \in W[k_2+1,k_3]}\left[\mathbb{E}_{w:ww' \in W[k_1,k_3]}[f(w)] g(w')\right] \\
&= \mathbb{E}_{w' \in W[k_2+1,k_3]}\left[\sum_{w \in W[k_1,k_2]} \frac{\Pi_{k_1,k_3}(ww')}{\Pi_{k_2+1,k_3}(w')} f(w) g(w')\right] \\
&= \sum_{w' \in W[k_2+1,k_3]} \Pi_{k_2+1,k_3}(w') \sum_{w \in W[k_1,k_2]} \frac{\Pi_{k_1,k_3}(ww')}{\Pi_{k_2+1,k_3}(w')} f(w) g(w') \\
&= \sum_{ww' \in W[k_1,k_3]} f(w) g(w') \Pi_{k_1,k_3}(ww').
\end{aligned}$$

Hence, $\mathfrak{S}_{k_2,k_3,k_1} = (\mathsf{S}_{k_1,k_2,k_3})^\dagger$ as claimed. ■

7.3.4 Variables for Walks on the s-wide Replacement Product

When analyzing walks on the s-wide replacement product, we actually need to use two separate, but related, local PSD ensembles. In Ta-Shma's construction, the vertices of the outer graph G correspond to positions in the base code $\mathcal{C}_0 \subseteq \mathbb{F}_2^n$, where $n = |V(G)|$. Given a vertex $(v,h) \in V(G) \times V(H)$ in the s-wide replacement product and codeword $z \in \mathcal{C}_0$, (v,h) is assigned bit z_v, regardless of the vertex h of the inner graph. We will enforce this property by working with variables in $V(G)$ rather than the full $V(G) \times V(H)$. The local PSD ensemble $\mathbf{Z} = \{\mathbf{Z}_v\}_{v \in V(G)}$ contains one variable for every vertex of G, with local distributions for sets of variables up to a given size. For a walk w on the s-wide replacement product, we will use $\mathbf{Z}_w$ as an abbreviation for $\mathbf{Z}_{S_w}$, where S_w is the set of all G-components of vertices visited on the walk.

The constraints of the CSP are placed on walks on the s-wide replacement product that do care about the H-component of the vertices, so we define a second local PSD ensemble $\mathbf{Y} = \{\mathbf{Y}_{(v,h)}\}_{(v,h) \in V(G) \times V(H)}$ with a variable for each vertex of the s-wide replacement product of G and H. It is this collection $\mathbf{Y}$ for which we need to prove tensoriality in order to use the list decoding framework. When we perform propagation rounding, we condition the ensemble $\mathbf{Z}$ on a random assignment σ to a subset $S \subseteq V(G)$, rather than conditioning $\mathbf{Y}$ on a random assignment to a subset of $V(G) \times V(H)$. Working with $\mathbf{Z}$ ensures that the rounded assignments will be consistent on each cloud of the s-wide replacement product. Since the bit assigned to a vertex (v, h) only depends on v, independent rounding of $\{\mathbf{Z} \mid \mathbf{Z}_S = \sigma\}$ will also yield the desired rounding of $\{\mathbf{Y} \mid \mathbf{Z}_S = \sigma\}$.

We can define $\mathbf{Y}$ based on the ensemble $\mathbf{Z}$ more concretely. Suppose $S' \subseteq V(G) \times V(H)$ is a subset of size at most p, where p is the locality of the ensemble, and define $T = \{v \mid (v, h) \in S'\}$. The distribution $\mu_{S'}$ of $\mathbf{Y}_{S'}$ is defined based on the distribution μ_T of $\mathbf{Z}_T$ by $\mu_{S'}(\alpha) = \mu_T(\alpha|_T)$, where $\alpha \in \mathbb{F}_2^{S'}$ is an assignment to S' whose value on each vertex (v, h) only depends on v.

Observe that the introduction of the ensemble $\mathbf{Y}$ is only necessary on the first level of the Ta-Shma code cascade between the codes $\mathcal{C}_0$ and $\mathcal{C}_1$, which takes place on the s-wide replacement product. Higher levels of the cascade use walks on graphs whose vertices are the walks from the level below. The association of the bits of a codeword to the vertices of this graph has no consistency requirement, so we simply use a single local ensemble $\mathbf{Z}$ with a variable for each vertex.

7.4 Splittability Implies Tensoriality

The connection between splittability and tensoriality will be made with the help of a version of the triangle inequality.

Claim 7.21 (Triangle inequality, adapted from [AJQ+20]). *Let $s \in \mathbb{N}^+$ and $\mathcal{T}$ be an s-interval splitting tree. Then*

$$\mathop{\mathbb{E}}_{w \in W[0,s-1]} \left\| \{\mathbf{Z}_w\} - \prod_{i=0}^{s-1} \left\{\mathbf{Z}_{w(i)}\right\} \right\|_1 \le \sum_{(k_1,k_2,k_3) \in \mathcal{T}} \mathop{\mathbb{E}}_{w \in W[k_1,k_3]} \left\| \{\mathbf{Z}_w\} - \left\{\mathbf{Z}_{w(k_1,k_2)}\right\}\left\{\mathbf{Z}_{w(k_2+1,k_3)}\right\} \right\|_1,$$

where the sum is taken over the labels of the internal nodes of $\mathcal{T}$.

To prove tensoriality, we will use the method of [BRS11] and [AJT19] to show that we can break correlations over expanding collections of tuples arising in the s-wide replacement product of the form

$$\mathop{\mathbb{E}}_{\substack{ww' \in W[k_1,k_3] \\ w \in W[k_1,k_2], w' \in W[k_2+1,k_3]}} \|\{\mathbf{Z}_{ww'}\} - \{\mathbf{Z}_w\}\{\mathbf{Z}_{w'}\}\|_1$$

appearing on the right-hand side of the triangle inequality.

7.4.1 The First Level of the Cascade

We now check the technical details to obtain tensoriality for the first lifting in the code cascade between the codes $\mathcal{C}_0$ and $\mathcal{C}_1$, which corresponds to taking s steps in Ta-Shma's construction. Recall that in order to obtain an assignment $z' \in \mathbb{F}_2^n$ whose lifting is consistent on vertices with the same G-component, we need to prove tensoriality for the ensemble $\mathbf{Y}$ with a variable for each vertex in $V(G) \times V(H)$.

The proof of tensoriality will make use of a specific entropic potential function. For an arbitrary random variable $\mathbf{X}$ taking values in a finite set $[q]$, define the function $\mathcal{H}(\mathbf{X})$ as

$$\mathcal{H}(\mathbf{X}) \ := \ \frac{1}{q} \sum_{a \in [q]} \mathrm{H}(\mathbb{1}_{[\mathbf{X}=a]}) \ = \ \mathbb{E}_{a \in [q]} \mathrm{H}(\mathbb{1}_{[\mathbf{X}=a]}),$$

where H is the binary entropy function. Using this, we define a potential function for a weighted undirected graph G.

Definition 7.22 (Graph Potential). *Let $G = (V, E)$ be a weighted graph with edge distribution Π_E. Let Π_V be the marginal distribution on V. Suppose that $\{\mathbf{Y}_i\}_{i \in V}$ is a p-local PSD ensemble for some $p \geq 1$. We define Φ^G to be*

$$\Phi^G \ := \ \mathop{\mathbb{E}}_{i \sim \Pi_V} \left[\mathcal{H}(\mathbf{Y}_i) \right].$$

Let $\mathcal{T}$ be an s-interval splitting tree associated with the s-wide replacement product of graphs G and H. We define

$$\Phi^{\mathcal{T}} := \sum_{(k_1,k_2,k_3) \in \mathcal{T}} \Phi^{\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})},$$

where $\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})$ is the associated bipartite undirected graph of the operator S_{k_1,k_2,k_3}.

Lemma 7.23 (Splittability Implies Tensoriality). *Let $W[0, s-1]$ be the walk collection of the s-wide replacement product of two graphs G and H. If $L \geq 128 \cdot (s^4 \cdot 2^{4s} / \mu^4)$ and $W[0, s-1]$ is τ-splittable with $\tau \leq \mu/(4s \cdot 2^{4s})$, then $W[0, s-1]$ is (μ, L)-tensorial.*

Proof. We need to show that

$$\mathop{\mathbb{E}}_{w \in W[0,s-1]} \left\| \{\mathbf{Y}'_w\} - \prod_{i=0}^{s-1} \left\{ \mathbf{Y}'_{w(i)} \right\} \right\|_1 \leq \mu,$$

which can be proven by adapting a potential argument technique from [BRS11]. First, set the potential

$$\Phi_m = \mathop{\mathbb{E}}_{S \sim \Pi_m} \mathop{\mathbb{E}}_{\sigma \sim \{\mathbf{Z}_S\}} \Phi^{\mathcal{T}}_{|\mathbf{Z}_S = \sigma}, \tag{5}$$

where the distribution Π_m on $S \subseteq V(G)$ is obtained from the process of choosing S in propagation rounding (Algorithm 7.16) once m has been fixed. Consider the error term

$$\mu_m := \mathop{\mathbb{E}}_{S \sim \Pi_m} \mathop{\mathbb{E}}_{\sigma \sim \{\mathbf{Z}_S\}} D(S, \sigma), \tag{6}$$

where $D(S,\sigma) := \mathbb{E}_{w \in W[0,s-1]} \left\| \{\mathbf{Y}_w \mid \mathbf{Z}_S = \sigma\} - \prod_{i=0}^{s-1} \left\{\mathbf{Y}_{w(i)} \mid \mathbf{Z}_S = \sigma\right\} \right\|_1$. If $\mu_m \geq \mu/2$, then

$$\underset{S \sim \Pi_m, \sigma \sim \{\mathbf{Z}_S\}}{\mathbb{P}} [D(S,\sigma) \geq \mu_m/2] \geq \frac{\mu}{4}.$$

For each choice of S and σ such that $D(S,\sigma) \geq \mu/2$, applying the triangle inequality from Claim 7.21 to the conditioned variables gives us

$$\begin{aligned} \frac{\mu}{2} &\leq \underset{w \in W[0,s-1]}{\mathbb{E}} \left\| \{\mathbf{Y}_w \mid \mathbf{Z}_S = \sigma\} - \prod_{i=0}^{s-1} \left\{\mathbf{Y}_{w(i)} \mid \mathbf{Z}_S = \sigma\right\} \right\|_1 \\ &\leq \sum_{(k_1,k_2,k_3) \in \mathcal{T}} \underset{w \in W[k_1,k_3]}{\mathbb{E}} \left\| \{\mathbf{Y}_w \mid \mathbf{Z}_S = \sigma\} - \left\{\mathbf{Y}_{w(k_1,k_2)} \mid \mathbf{Z}_S = \sigma\right\} \left\{\mathbf{Y}_{w(k_2+1,k_3)} \mid \mathbf{Z}_S = \sigma\right\} \right\|_1. \end{aligned}$$

Hence, there exists (k_1, k_2, k_3) such that

$$\frac{\mu}{2s} \leq \underset{w \in W[k_1,k_3]}{\mathbb{E}} \left\| \{\mathbf{Y}_w \mid \mathbf{Z}_S = \sigma\} - \left\{\mathbf{Y}_{w(k_1,k_2)} \mid \mathbf{Z}_S = \sigma\right\} \left\{\mathbf{Y}_{w(k_2+1,k_3)} \mid \mathbf{Z}_S = \sigma\right\} \right\|_1.$$

Note that choosing $w \in W[0, s-1]$ uniformly and restricting to $w(k_1, k_3)$ gives a uniformly random element of $W[k_1, k_3]$. If we choose $w(k_1, k_2)$ or $w(k_2+1, k_3)$ with equal probability, then the final walk is distributed according to the stationary measure of $\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})$. Let w' denote the chosen walk. Observe that $\mathbf{Y}_{w'}$ is a deterministic function of $\mathbf{Z}_{w'} \mid \mathbf{Z}_S = \sigma$. Now, we sample $\mathbf{Z}_{w'} \mid \mathbf{Z}_S = \sigma$, which gives us a sample of $\mathbf{Y}_{w'}$. Applying Lemma A.1, we have

$$\Phi^{\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})}_{|\{\mathbf{Y}_{w'} | \mathbf{Z}_S = \sigma\}} \leq \Phi^{\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})}_{\mathbf{Z}_S = \sigma} - \frac{\mu^2}{16 s^2 \cdot 2^{4s}}.$$

This conditioning on an assignment to $\mathbf{Z}_{\mathrm{set}(w')} \mid \mathbf{Z}_S = \sigma$ does not increase the other terms of $\Phi^{\mathcal{T}}$ associated to split operators other than $\mathcal{U}(\mathsf{S}_{k_1,k_2,k_3})$ since entropy is non-increasing under conditioning. Similarly, conditioning on the remaining variables that are part of w but not w' does not increase $\Phi^{\mathcal{T}}$. Then, we obtain

$$\Phi_m - \Phi_{m+1} \geq \underset{S \sim \Pi_m, \sigma \sim \{\mathbf{Z}_S\}}{\mathbb{P}} [D(S,\sigma) \geq \mu_m/2] \cdot \frac{\mu^2}{16 s^2 \cdot 2^{4s}}.$$

Since $s \geq \Phi_1 \geq \cdots \geq \Phi_{L/(s+1)} \geq 0$, there can be at most $32 s^3 \cdot 2^{4s}/\mu^3$ indices $m \in [L/s]$ such that $\mu_m \geq \mu/2$. In particular, since the total number of indices is L/s, we have

$$\underset{m \in [L/s]}{\mathbb{E}} [\mu_m] \leq \frac{\mu}{2} + \frac{s}{L} \cdot \frac{32 s^3 \cdot 2^{4s}}{\mu^3}.$$

Our choice of L is more than enough to ensure $\mathbb{E}_{m \in [L/s]}[\mu_m] \leq \mu$. ■

Applying the list decoding framework will require the stronger property of two-step tensoriality, which we can obtain under the same assumptions.

Lemma 7.24 (Splittability Implies Two-step Tensoriality)**.** *Let $W[0, s-1]$ be the walk collection of the s-wide replacement product of two graphs G and H. If $L \geq 128 \cdot (s^4 \cdot 2^{4s}/\mu^4)$ and $W[0, s-1]$ is τ-splittable with $\tau \leq \mu/(4s \cdot 2^{4s})$, then $W[0, s-1]$ is (μ, L)-two-step tensorial.*

Proof. Under our assumptions the (μ, L)-tensorial property follows from Lemma 7.23 (which is the only place where the assumption on τ is used), so we only need to show

$$\mathop{\mathbb{E}}_{w,w' \in W[0,s-1]} \left\| \{\mathbf{Y}'_w \mathbf{Y}'_{w'}\} - \{\mathbf{Y}'_w\}\{\mathbf{Y}'_{w'}\} \right\|_1 \le \mu,$$

which can be proven by adapting a potential argument technique from [BRS11]. First, set the potential

$$\Phi_m = \mathop{\mathbb{E}}_{S \sim \Pi_m} \mathop{\mathbb{E}}_{\sigma \sim \{\mathbf{Z}_S\}} \mathop{\mathbb{E}}_{w \in W[0,s-1]} \mathcal{H}(\mathbf{Y}_w \mid \mathbf{Z}_S = \sigma), \tag{7}$$

where the distribution Π_m on $S \subseteq V(G)$ is obtained from the process of choosing S in propagation rounding (Algorithm 7.16) once m has been fixed. Consider the error term

$$\mu_m := \mathop{\mathbb{E}}_{S \sim \Pi_m} \mathop{\mathbb{E}}_{\sigma \sim \{\mathbf{Z}_S\}} D(S, \sigma), \tag{8}$$

where $D(S,\sigma) := \mathbb{E}_{w,w' \in W[0,s-1]}[\|\{\mathbf{Y}_w \mathbf{Y}_{w'} \mid \mathbf{Z}_S = \sigma\} - \{\mathbf{Y}_w | \mathbf{Z}_S = \sigma\}\{\mathbf{Y}_{w'} | \mathbf{Z}_S = \sigma\}\|_1]$. If $\mu_m \ge \mu/2$, then

$$\mathop{\mathbb{P}}_{S \sim \Pi_m, \sigma \sim \{\mathbf{Z}_S\}} [D(S,\sigma) \ge \mu_m/2] \ge \frac{\mu}{4}.$$

Let $G' = (V = W[0,s-1], E)$ be the graph with edges $E = \{\{w, w'\} \mid w, w' \in W[0, s-1]\}$. Local correlation (expectation over the edges) on this graph G' is the same as global correlation (expectation over two independent copies of vertices). Then, we obtain [8]

$$\Phi_m - \Phi_{m+1} \ge \mathop{\mathbb{P}}_{S \sim \Pi_m, \sigma \sim \{\mathbf{Z}_S\}} [D(S,\sigma) \ge \mu_m/2] \cdot \frac{\mu^2}{2 \cdot 2^{2s}}.$$

Since $1 \ge \Phi_1 \ge \cdots \ge \Phi_{L/(s+1)} \ge 0$, there can be at most $8 \cdot 2^{2s}/\mu^3$ indices $m \in [L/s]$ such that $\mu_m \ge \mu/2$. In particular, since the total number of indices is L/s, we have

$$\mathop{\mathbb{E}}_{m \in [L/s]} \mu_m \le \frac{\mu}{2} + \frac{k}{L} \cdot \frac{8 \cdot 2^{2s}}{\mu^3}.$$

Our choice of L is more than enough to ensure $\mathbb{E}_{m \in [L/s]}[\mu_m] \le \mu$. ■

We have already established that $W[0,s-1]$ is τ-splittable with $\tau = \sigma_2(G) + 2\sigma_2(H) + \sigma_2(H)^2$ in Corollary 7.13, so we can obtain (μ, L)-two-step tensoriality for any μ if this quantity is small enough.

7.4.2 Higher Levels of the Cascade

We now discuss tensoriality of the other levels of the cascade between $\mathcal{C}_{i-1}$ and $\mathcal{C}_i$ for $i \ge 2$. Tensorial properties are simpler to establish here than on the first level of the cascade. The relevant split operators are special cases of S_{k_1,k_2,k_3} where $k_1 \equiv 0 \pmod{s}$ and $k_2, k_3 \equiv -1 \pmod{s}$. The main difference now is that we can associate the parity bits of $\mathcal{C}_{i-1}$ with the vertices of $\mathcal{U}(\mathsf{S}^{\triangle}_{r,r})$, which themselves represent walks. As this association of parity bits doesn't need to satisfy a consistency condition, we only need to work with a single ensemble $\mathbf{Z}$ instead of working with two different ensembles as in the previous case. The proofs of Lemma 7.23 and Lemma 7.24 with these slight modifications give us two-step tensoriality.

[8] See [AJT19] or [BRS11] for the details.

Lemma 7.25 (Two-step Tensoriality for Higher Levels). *Let $W(k)$ be the set of walks defined using $(k-1)$ steps of the operator $\mathsf{S}^{\triangle}_{r,r}$. If $L \geq 128 \cdot (k^4 \cdot 2^{4k}/\mu^4)$ and $W(k)$ is τ-splittable with $\tau \leq \mu/(4k \cdot 2^{4k})$, then $W(k)$ is (μ, L)-two-step tensorial.*

We know that the collection of walks obtained from $\sigma_2(\mathsf{S}^{\triangle}_{r,r})$ is $(\sigma_2(G) + 2 \cdot \sigma_2(H) + \sigma_2(H)^2)$-splittable, so the parameters necessary to obtain two-step tensoriality are the same as in the first level of the cascade.

8 Choosing Parameters for Ta-Shma's Construction

We explore how some choices of parameters for Ta-Shma's construction interact with the requirements of our decoding algorithm. The analysis is divided into rounds of increasingly stronger decoding guarantees with later rounds relying on the codes obtained in previous rounds. Naturally, the stronger guarantees come with more delicate and technical considerations. We briefly summarize the goals of each round and some key parameters.

1. Round I: For any constant $\beta > 0$, we obtain *efficient unique decodable* codes $\mathcal{C}_\ell$ with distance at least $1/2 - \varepsilon$ and rate $\Omega(\varepsilon^{2+\beta})$ for infinitely many *discrete* values of $\varepsilon > 0$ (with ε as close to 0 as desired). In this regime, it suffices for the expansion of H to be constant. This round leads to Theorem 6.6.
2. Round II: Similar to Round I, but now ε can be any value in an interval $(0, b)$ with $b < 1/2$ being a function of β. Again the expansion of H can be constant. This round leads to Theorem 6.7.
3. Round III: We want β to vanish as ε vanishes (this is qualitatively similar to Ta-Shma's result). In this regime, we make the expansion of H be a function of ε, and we rely on the uniquely decodable codes of Round II. This round leads to Theorem 1.1.
4. Round IV: For any constant $\beta_0 > 0$, we obtain *efficient list decodable* codes $\mathcal{C}_\ell$ with list decoding radius $1/2 - \beta_0$ and rate $\Omega(\varepsilon^{2+\beta})$ with $\beta \to 0$ as $\varepsilon \to 0$. In this regime, we make the expansion of H be a function of ε, and we rely on the uniquely decodable codes of Round III. This round leads to Theorem 1.2.

The way we choose parameters for Ta-Shma's construction borrows heavily from Ta-Shma's arguments in [TS17]. We fix some notation common to all rounds. A graph is said to be an (n, d, λ)-graph provided it has n vertices, is d-regular, and has second largest singular value of its normalized adjacency matrix at most λ.

Notation 8.1. *We use the following notation for the graphs G and H used in the s-wide replacement product.*

- *The outer graph G will be an (n', d_1, λ_1)-graph.*
- *The inner graph H will be a (d_1^s, d_2, λ_2)-graph.*

The parameters $n', d_1, d_2, \lambda_1, \lambda_2$ and s will be chosen in the subsequent sections.

8.1 Round I: Initial Analysis

We are given the dimension D of the desired code and $\varepsilon \in (0, 1/2)$. We set a parameter $\alpha \leq 1/128$ such that (for convenience) $1/\alpha$ is a power of 2 and

$$\frac{\alpha^5}{4\log_2(1/\alpha)} \geq \frac{1}{\log_2(1/\varepsilon)}. \tag{9}$$

We can assume that $\alpha \leq 1/128$ satisfy Eq. (9) since otherwise ε is a constant and we can use the list decodable codes from [AJQ$^+$20]. The use of Eq. (9) will be clear shortly. It becomes a necessity from round III onward. For rounds I and II, the parameter α will be a constant, but it will be useful to establish the analysis in more generality now so that subsequent rounds can reuse it.

The inner graph H. The choice of H is similar to Ta-Shma's choice. More precisely, we set $s = 1/\alpha$ and $d_2 = s^{4s^2}$ (Ta-Shma took $d_2 = s^{4s}$). We obtain a Cayley graph $H = \mathrm{Cay}(\mathbb{F}_2^{4s\log_2(d_2)}, A)$ such that H is an $(n_2 = d_2^{4s}, d_2, \lambda_2)$ graph where $\lambda_2 = b_2/\sqrt{d_2}$ and $b_2 = 4s\log_2(d_2)$. (The set of generators, A, comes from a small bias code derived from a construction of Alon et al. [AGHP92], but we will rely on Ta-Shma's analysis embodied in Lemma B.2 and not discuss it further.)

The base code $\mathcal{C}_0$. Set $\varepsilon_0 = 1/d_2^2 = \lambda_2^4/b_2^4 \leq \lambda_2^4/3$ (this choice differs from Ta-Shma's and it appears because we are essentially working with H^2 rather than H). We will choose a base code $\mathcal{C}_0$ such that the desired code will be obtained as a direct sum lifting of $\mathcal{C}_0$, and because this lifting preserves the dimension, the dimension of $\mathcal{C}_0$ should be D. We choose $\mathcal{C}_0$ to be an ε_0-balanced code with dimension D and block length $n = O_{\varepsilon_0}(D)$. For instance, we can start with any good (constant rate and relative distance) linear base code $\mathcal{C}_0$ that has an efficient unique decoding algorithm and obtain a ε_0-balanced lifted code that can be efficiently unique decoded (as long as ε_0 is constant) using the framework in [AJQ$^+$20].

The outer graph G. Set $d_1 = d_2^4$ so that $n_2 = d_1^s$ as required by the s-wide replacement product. We apply Ta-Shma's explicit Ramanujan graph Lemma B.1 with parameters n, d_1 and θ to obtain an (n', d_1, λ_1) Ramanujan graph G with $\lambda_1 \leq 2\sqrt{2}/\sqrt{d_1}$ and $n' \in [(1-\theta)n, n]$ or $n' \in [(1-\theta)2n, 2n]$. Here, θ is an error parameter that we set as $\theta = \lambda_2^4/6$ (this choice of θ differs from Ta-Shma). Because we can construct words with block length $2n$ (if needed) by duplicating each codeword, we may assume w.l.o.g. that n' is close to n and $(n - n') \leq \theta n \leq 2\theta n'$. See Appendix B for a more formal description of this graph.

Note that $\lambda_1 \leq \lambda_2^4/6$ since $\lambda_1 \leq 3/\sqrt{d_1} = 3/d_2^2 = 3 \cdot \lambda_2^4/b_2^4 \leq \lambda_2^4/6$. Hence, $\varepsilon_0 + 2\theta + 2\lambda_1 \leq \lambda_2^4$.

The walk length. Set the walk length $t-1$ to be the smallest integer such that

$$(\lambda_2^2)^{(1-5\alpha)(1-\alpha)(t-1)} \leq \varepsilon.$$

This will imply using Ta-Shma's analysis that the bias of the final code is at most ε as shown later.

$s = 1/\alpha$, such that $\frac{\alpha^4}{4\log_2(1/\alpha)} \geq \frac{1}{\log_2(1/\varepsilon)}$

$H : (n_2, d_2, \lambda_2), \quad n_2 = d_1^s, \quad d_2 = s^{4s^2}, \quad \lambda_2 = \frac{b_2}{\sqrt{d_2}}, \quad b_2 = 4s \log d_2$

$G : (n', d_1, \lambda_1), \quad n' \approx n = O(D/\beta_0^c), \quad d_1 = d_2^4, \quad \lambda_1 \leq \frac{2\sqrt{2}}{d_1}$

t : smallest integer such that $(\lambda_2^2)^{(1-5\alpha)(1-\alpha)(t-1)} \leq \varepsilon$

Claim 8.2. *We have $t - 1 \geq s/\alpha = s^2$.*

Proof. Using $d_2 = s^{4s^2}$ and Eq. (9), we have

$$\left(\frac{1}{\lambda_2^2}\right)^{(1-5\alpha)(1-\alpha)s/\alpha} \leq \left(\frac{1}{\lambda_2^2}\right)^{s/\alpha} = \left(\frac{d_2}{b_2^2}\right)^{s/\alpha} \leq (d_2)^{s/\alpha} = s^{4s^3/\alpha}$$
$$= 2^{4s^3 \log_2(s)/\alpha} = 2^{4\log_2(1/\alpha)/\alpha^4} \leq 2^{\log_2(1/\varepsilon)} = \frac{1}{\varepsilon}.$$

Hence, $\varepsilon \geq (\lambda_2^2)^{(1-5\alpha)(1-\alpha)s/\alpha}$ and thus $t-1$ must be at least s/α. ■

Remark 8.3. *By our choice of t, we have $(\lambda_2^2)^{(1-5\alpha)(1-\alpha)(t-2)} \geq \varepsilon$. Since $1/(t-1) \leq \alpha$, we get $(\lambda_2^2)^{(1-5\alpha)(1-\alpha)^2(t-1)} \geq \varepsilon$.*

Final Bias. We denote by $\mathcal{C}_\ell$ the final code obtained by t steps of the s-wide replacement product. The bias of $\mathcal{C}_\ell$ is given by Corollary 4.10 (which in turn is a simple corollary of Ta-Shma's Fact 4.9) as shown next.

Corollary 8.4. *The code $\mathcal{C}_\ell$ is ε-balanced.*

Proof. Using Corollary 4.10, we have that the final bias

$$b := \left(\sigma_2(H^2)^{s-1} + (s-1) \cdot \sigma_2(H^2)^{s-2} + (s-1)^2 \cdot \sigma_2(H^2)^{s-4}\right)^{\lfloor (t-1)/s \rfloor}$$

is bounded by

$$\begin{aligned} b &\leq (3(s-1)^2 \sigma_2(H^2)^{s-4})^{((t-1)/s)-1} && \text{(Using } \sigma_2(H^2) \leq 1/3s^2\text{)} \\ &\leq ((\sigma_2(H^2)^{s-5})^{(t-1-s)/s} \\ &= \sigma_2(H^2)^{(1-5/s)(1-s/(t-1))(t-1)} \\ &\leq \sigma_2(H^2)^{(1-5\alpha)(1-\alpha)(t-1)} \\ &= \left(\lambda_2^2\right)^{(1-5\alpha)(1-\alpha)(t-1)} \leq \varepsilon, \end{aligned}$$

where the last inequality follows from $s = 1/\alpha$ and $t - 1 \geq s/\alpha$, the latter from Claim 8.2. ■

Rate. The proof of the rate follows a similar structure of Ta-Shma's original argument except that we take s to be a constant independent of ε so that ε_0, λ_1, and λ_2 are also constants independent of ε. Note that we previously said $\alpha = 1/s$ needs to satisfy Equation 9, but that implies only an upper bound for s, and smaller (even constant) values for s are still permissible.

Claim 8.5. *$\mathcal{C}_\ell$ has rate $\Omega(\varepsilon^{2+26\cdot\alpha})$ provided $\varepsilon_0 > 0$ is constant.*

Proof. The support size is the number of walks of length t on the s-wide replacement product of G and H (each step of the walk has d_2^2 options), which is

$$\begin{aligned}|V(G)||V(H)|d_2^{2(t-1)} = n' \cdot d_1^s \cdot d_2^{2(t-1)} &= n' \cdot d_2^{2(t-1)+4s} \le n \cdot d_2^{2(t-1)+4s} \\ &= \Theta_{\varepsilon_0}\left(D \cdot d_2^{2(t-1)+4s}\right) \\ &= \Theta\left(D \cdot (d_2^2)^{t-1+2s}\right) \\ &= O\left(D \cdot (d_2^2)^{(1+2\alpha)(t-1)}\right),\end{aligned}$$

where the penultimate equality follows from the assumption that ε_0 is a constant.

Note that $d_2^\alpha = d_2^{1/s} = s^{4s} \ge b_2$ since $b_2 = 4s\log_2(d_2) = 16s^3\log_2(s) \le s^4$ (recall that $s = 1/\alpha \ge 128$). Thus,

$$d_2^{1-2\alpha} = \frac{d_2}{d_2^{2\alpha}} \le \frac{d_2}{b_2^2} = \frac{1}{\sigma_2(H^2)}.$$

We obtain

$$\begin{aligned}(d_2^2)^{(t-1)} &\le \left(\frac{1}{\sigma_2(H^2)}\right)^{\frac{2(t-1)}{1-2\alpha}} \\ &\le \left(\frac{1}{\varepsilon}\right)^{\frac{2}{(1-2\alpha)(1-5\alpha)(1-\alpha)^2}} && \text{(Using Remark 8.3)} \\ &\le \left(\frac{1}{\varepsilon}\right)^{2(1+10\alpha)},\end{aligned}$$

which implies a block length of

$$O\left(D \cdot (d_2^2)^{(1+2\alpha)(t-1)}\right) = O\left(D\left(\frac{1}{\varepsilon}\right)^{2(1+10\alpha)(1+2\alpha)}\right) = O\left(D\left(\frac{1}{\varepsilon}\right)^{2(1+13\alpha)}\right).$$

■

Lemma 8.6 (Codes Near the GV bound I)**.** *For every constant $\beta > 0$, there exists a sufficiently large constant s in the above analysis so that for any dimension value $D \in \mathbb{N}^+$ (sufficiently large) and $\varepsilon > 0$ (sufficiently small) the final code $\mathcal{C}_{N,\varepsilon,\beta}$, where N is the block length, satisfies*

- *$\mathcal{C}_{N,\varepsilon,\beta}$ is ε-balanced,*
- *$\mathcal{C}_{N,\varepsilon,\beta}$ has rate $\Omega(\varepsilon^{2+\beta})$, and*
- *$\mathcal{C}_{N,\varepsilon,\beta}$ is a linear code of dimension D.*

Remark 8.7. *As a consequence of code cascading, the final attainable walk lengths have the form $s^\ell - 1$ where ℓ is a positive integer. Given $\beta > 0$, we have infinitely many values of ε attainable by such walk lengths which gives infinitely many codes $\mathcal{C}_{N,\varepsilon,\beta}$. This means that although the bias ε cannot be arbitrary, we have an infinite sequence of values of ε for which the rates of the codes $\mathcal{C}_{N,\varepsilon,\beta}$ are near the GV bound. In Section 8.2, we show how to bypass this artificial limitation. These codes are used in the proof of Theorem 6.6.*

We can view the above analysis as defining a function Γ that receives

- the dimension $D \in \mathbb{N}^+$,
- the final bias $\varepsilon > 0$,
- the approximating error $\alpha \in (0, 1/128]$ with $s := 1/\alpha$ being a power of two, and
- a multiplying factor $Q \in \mathbb{N}^+$ such that $d_2 = s^{4s^2 \cdot Q}$ (in the above Q was 1).

and outputs a tuple of parameters $(t, \varepsilon_0, \theta, d_1, \lambda_1, n')$, graphs G and H (as above) where, in particular, the number of steps $t \in \mathbb{N}^+$ is such that the final code $\mathcal{C}_\ell$ has bias at most ε and rate $\Omega(\varepsilon^{2+26\cdot\alpha})$.

In future rounds, Γ may be called with $Q = s$ instead of $Q = 1$. This will cause d_2 to increase from s^{4s^2} to $s^{4s^2 \cdot Q}$, and so in the proof of Claim 8.2, $2^{4\log_2(1/\alpha)/\alpha^4}$ will be replaced by $2^{4\log_2(1/\alpha)/\alpha^5}$. This explains why Eq. (9) has a stricter requirement than needed in the $Q = 1$ case above.

8.2 Round II: A More Careful Analysis

We are given the dimension of the code D and $\varepsilon \in (0, 1/2)$. As before, we set a parameter $\alpha \leq 1/128$ such that (for convenience) $1/\alpha$ is a power of 2. Set $s = 1/\alpha$ and $Q = s$.

Apply Γ to $(D, \varepsilon, \alpha, Q)$ to obtain all parameters except t. Choose t to be the smallest integer satisfying

$$(\lambda_2^2)^{(1-5\alpha)(1-2\alpha)(1-\alpha)(t-1)} \leq \varepsilon,$$

where observe that an extra $(1 - 2\alpha)$ factor appears in the exponent. This change in t will worsen the rate but by losing a factor of $\frac{1}{1-2\alpha}$ in the exponent, we can lower bound the rate. That is, $(d_2^2)^{-(t-1)} = \Omega(\varepsilon^{\frac{2+26\cdot\alpha}{1-2\alpha}})$.

Set $\ell \in \mathbb{N}^+$ to be the smallest value such that $s^\ell \geq t$ (here we are implicitly assuming that $t > s$). If $s^\ell = t$, we are done since we can use all the parameters returned by Γ for the construction of $\mathcal{C}_\ell$. Now assume $s^\ell > t$ and let $\zeta = t/s^{\ell-1}$. Note that $\zeta \in (1, s)$. Choose P to be the integer in the interval $[Q, s \cdot Q]$ such that

$$0 \leq \frac{P}{Q} - \zeta \leq \frac{1}{Q}.$$

Because $s^\ell > t$, and only powers of s may be chosen for walk length, we might overshoot in walk length by a multiplicative factor of s. This will cause a corresponding decay in rate computation that we cannot afford. To overcome this, in the last level of the cascade between codes $\mathcal{C}_{\ell-1}$ and $\mathcal{C}_\ell$, perform the direct sum over walks of length $(P-1)$ instead of length $(s-1)$. The new total number of vertices is $t' = Ps^{\ell-1}$. Note that P can be as large as s^2, so our splittability guarantee of $W(P)$ (the walk collection from the lift between $\mathcal{C}_{\ell-1}$ and $\mathcal{C}_\ell$) has to be strong enough to accommodate this larger arity and not only arity s.

Claim 8.8. *We have $t - 1 \leq \frac{t'-1}{Q} \leq (1 + 2\alpha)(t-1)$.*

Proof. By construction, we have the sequence of implications

$$\begin{aligned}
&0 \le \frac{P}{Q}s^{\ell-1} - \zeta s^{\ell-1} \le \frac{s^{\ell-1}}{Q}\\
\Rightarrow &0 \le \frac{t'}{Q} - t \le \frac{s^{\ell-1}}{Q} \le \frac{t}{Q}\\
\Rightarrow &t - \frac{1}{Q} \le \frac{t'-1}{Q} \le (t-1)\left(1+\frac{1}{Q}\right) + 1,
\end{aligned}$$

from which we obtain

$$t - 1 \le t - \frac{1}{Q} \le \frac{t'-1}{Q}$$

and

$$\frac{t'-1}{Q} \le (t-1)\left(1+\frac{1}{Q}\right) + 1 = (1+\alpha)(t-1) + 1 < (1+2\alpha)(t-1),$$

the latter using $Q = s = 1/\alpha$. ■

We apply Γ again but this time to $(D, \varepsilon, \alpha, 1)$ to obtain new parameters $(t'', \varepsilon_0', \theta', d_1', \lambda_1', n'')$, and graphs G' and H'.

Claim 8.9. *The code $\mathcal{C}_\ell'$ obtained by t' walk steps on the s-wide replacement product of G' and H' from the second application of Γ has bias at most ε and rate $\Omega(\varepsilon^{2+40\alpha})$.*

Proof. Let $d_2 = s^{4s^2 \cdot Q}$, $b_2 = 4s\log_2(d_2)$ and $\lambda_2 = b_2/\sqrt{d_2}$ be the parameters of the first invocation of Γ. Recall that t was chosen to be the smallest integer satisfying

$$(\lambda_2^2)^{(1-5\alpha)^2(1-\alpha)(t-1)} \le \varepsilon.$$

Let $d_2' = s^{4s^2}$, $b_2' = 4s\log_2(d_2')$ and $\lambda_2' = b_2'/\sqrt{d_2'}$ be the parameters of the second invocation of Γ. Observe that

$$\begin{aligned}
(\lambda_2')^Q &= \frac{(b_2')^Q}{\sqrt{(d_2')^Q}} = \frac{(b_2'^Q)}{\sqrt{d_2}} = \frac{(16s^3\log_2(s))^Q}{s^{2s^2\cdot Q}}\\
&\le \frac{s^{4Q}}{s^{2s^2\cdot Q}} = \frac{1}{s^{2s^2\cdot Q(1-\frac{2}{s^2})}} = \left(\frac{1}{s^{2s^2\cdot Q}}\right)^{1-2\alpha} \le \left(\frac{b_2}{\sqrt{d_2}}\right)^{1-2\alpha} = \lambda_2^{1-2\alpha}.
\end{aligned}$$

Then the bias of $\mathcal{C}_\ell'$ is at most

$$\begin{aligned}
(((\lambda_2')^Q)^2)^{(1-5\alpha)(1-\alpha)(t'-1)/Q} &\le (\lambda_2^2)^{(1-5\alpha)(1-2\alpha)(1-\alpha)(t'-1)/Q}\\
&\le (\lambda_2^2)^{(1-5\alpha)(1-2\alpha)(1-\alpha)(t-1)} \le \varepsilon.
\end{aligned}$$

For the rate computation of $\mathcal{C}_\ell'$, we will lower bound the term $((d_2')^2)^{-(t'-1)}$. Since $d_2 = (d_2')^Q$, $(d_2^2)^{-(t-1)} = \Omega(\varepsilon^{\frac{2+26\cdot\alpha}{1-2\alpha}})$ and $\frac{t'-1}{Q} \le (1+2\alpha)(t-1)$ (the latter by Claim 8.8), the rate of $\mathcal{C}_\ell'$ is

$$\Omega(((d_2')^2)^{-(t'-1)}) = \Omega((d_2^2)^{-(t'-1)/Q}) = \Omega((d_2^2)^{-(1+2\alpha)(t-1)}) = \Omega((\varepsilon^{2+26\cdot\alpha})^{\frac{1+2\alpha}{1-2\alpha}}) = \Omega(\varepsilon^{2+40\cdot\alpha}).$$

■

8.3 Round III: Vanishing β as ε Vanishes

We are given the dimension of the code D and $\varepsilon \in (0, 1/2)$. As before, we set a parameter $\alpha \leq 1/128$ such that (for convenience) $1/\alpha$ is a power of 2. Set $s := 1/\alpha$.

We will consider the regime where s is a function of ε. As a consequence, the parameters $d_2, \lambda_2, d_1, \lambda_1, \varepsilon_0$ will also depend on ε. Since $x \leq 1/\log_2(1/x)$ for $x \leq 1/2$ (and $\alpha \leq 1/2$), if α satisfies $\alpha^6/4 \geq 1/\log_2(1/\beta)$, it also satisfies Eq. (9) (we lose a log factor by replacing $1/\log_2(1/\alpha)$ by α, but we will favor simplicity of parameters). In particular, we can set α so that s is

$$s = \Theta((\log_2(1/\varepsilon))^{1/6}),$$

and satisfy Eq. (9).

We follow the same choices as in Round II except for the base code $\mathcal{C}_0$.

The base code $\mathcal{C}_0$. Set $\varepsilon_0 = 1/d_2^2 = \lambda_2^4/b_2^4 \leq \lambda_2^4/3$. We choose an ε_0-balanced code $\mathcal{C}_0$ with support size $n = O(D/\varepsilon_0^c)$ where $c = 2.001$ (this choice of c is arbitrary, it is enough to have c as a fixed small constant) using the construction from Round II. It is crucial that we can unique decode $\mathcal{C}_0$ (using our algorithm), since this is required in order to apply the list decoding framework.

Note that ε_0 is no longer a constant. For this reason, we need to consider the rate computation of the final code $\mathcal{C}_\ell$ more carefully. The proof will follow an argument similar to Ta-Shma's.

Claim 8.10. *$\mathcal{C}_\ell$ has rate $\Omega(\varepsilon^{2+26\cdot\alpha})$ where $\alpha = \Theta(1/(\log_2(1/\varepsilon))^{1/6})$.*

Proof. The support size is the number of walks of length $t-1$ on the s-wide replacement product of G and H (each step of the walk has d_2^2 options), which is

$$\begin{aligned}
|V(G)||V(H)|d_2^{2(t-1)} = n' \cdot d_1^s \cdot d_2^{2(t-1)} &= n' \cdot d_2^{2(t-1)+4s} \leq n \cdot d_2^{2(t-1)+4s} \\
&= \Theta\left(\frac{D}{\varepsilon_0^c} \cdot d_2^{2(t-1)+4s}\right) \\
&= \Theta\left(D \cdot (d_2^2)^{(t-1)+2s+2.001}\right) \\
&= O\left(D \cdot (d_2^2)^{(1+2\alpha)(t-1)}\right).
\end{aligned}$$

From this point the proof continues exactly as the proof of Claim 8.5. ■

8.4 Round IV: Arbitrary Gentle List Decoding

In round III, when we take

$$s = \Theta((\log_2(1/\varepsilon))^{1/6}),$$

we will have $\lambda_2 = 4s\log(s^{4s^2})/s^{2s^2} \leq s^{-s^2}$ provided s is large enough. This non-constant λ_2 will allow us perform "gentle" list decoding with radius arbitrarily close to $1/2$. More precisely, we have the following.

Theorem 8.11 (Gentle List Decoding (restatement of Theorem 1.2))**.** *For every $\varepsilon > 0$ sufficiently small, there are explicit binary linear Ta-Shma codes $\mathcal{C}_{N,\varepsilon,\beta} \subseteq \mathbb{F}_2^N$ for infinitely many values $N \in \mathbb{N}$ with* *(i) distance at least* $1/2 - \varepsilon/2$ *(actually* ε*-balanced),*

(ii) rate $\Omega(\varepsilon^{2+\beta})$ *where* $\beta = O(1/(\log_2(1/\varepsilon))^{1/6})$*, and*

(iii) a list decoding algorithm that decodes within radius $1/2 - 2^{-\Theta((\log_2(1/\varepsilon))^{1/6})}$ *in time* $N^{O_{\varepsilon,\beta}(1)}$*.*

Proof. We consider some parameter requirements in order to apply the list decoding framework Theorem 9.1 between $\mathcal{C}_{\ell-1}$ and $\mathcal{C}_\ell$. Suppose we want to list decode within radius $1/2 - \sqrt{\eta}$. For parity sampling, we need

$$s \geq \Theta(\log_2(1/\eta)).$$

Since the number of vertices in a walk can be at most s^2, for splittability we need

$$\eta^8/(s^2 \cdot 2^{2s^2}) \geq 2 \cdot s^{-s^2}.$$

In particular, we can take $\eta = 2^{-\Theta(s)}$ and satisfy both conditions above. ■

9 Instantiating the List Decoding Framework

We established the tensoriality (actually two-step tensoriality) and parity sampling properties of every lifting between consecutive codes $\mathcal{C}_{i-1}$ and $\mathcal{C}_i$ in Ta-Shma's cascade. Using these properties, we will be able to invoke the list decoding framework from [AJQ+20] to obtain the following list decoding result.

Theorem 9.1 (Restatement of Theorem 6.1)**.** *Let* $\eta_0 \in (0, 1/4)$ *be a constant,* $\eta \in (0, \eta_0)$*, and*

$$k \geq k_0(\eta) := \Theta(\log(1/\eta)).$$

Suppose $\mathcal{C} \subseteq \mathbb{F}_2^n$ *is an* η_0*-balanced linear code and* $\mathcal{C}' = \mathrm{dsum}_{W(k)}(\mathcal{C})$ *is the direct sum lifting of* $\mathcal{C}$ *on a* τ*-splittable collection of walks* $W(k)$*, where* $W(k)$ *is either the set of walks* $W[0, s]$ *on an* s*-wide replacement product graph or a set of walks using the random walk operator* $\mathsf{S}^{\triangle}_{r,r}$*. There exists an absolute constant* $K > 0$ *such that if*

$$\tau \leq \tau_0(\eta, k) := \frac{\eta^8}{K \cdot k \cdot 2^{4k}},$$

then the code $\mathcal{C}'$ *is* η*-balanced and can be efficiently list decoded in the following sense:*

If $\tilde{y}$ *is* $(1/2 - \sqrt{\eta})$*-close to* $\mathcal{C}'$*, then we can compute the list*

$$\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') := \left\{ (z, \mathrm{dsum}_{W(k)}(z)) \mid z \in \mathcal{C}, \Delta\Big(\mathrm{dsum}_{W(k)}(z), \tilde{y}\Big) \leq \frac{1}{2} - \sqrt{\eta} \right\}$$

in time

$$n^{O(1/\tau_0(\eta,k)^4)} \cdot f(n),$$

where $f(n)$ *is the running time of a unique decoding algorithm for* $\mathcal{C}$*. Otherwise, we return* $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') = \emptyset$ *with the same running time of the preceding case* [9]*.*

[9] In the case $\tilde{y}$ is not $(1/2 - \sqrt{\eta})$-close to $\mathcal{C}'$, but the SOS program turns out to be feasible, some of the calls to the unique decoding algorithm of $\mathcal{C}$ (issued by the list decoding framework) might be outside all unique decoding balls. Such cases may be handled by returning failure if the algorithm does not terminate by the time $f(n)$. Even if a codeword in $\mathcal{C}$ is found, the pruning step of list decoding [AJQ+20] will return an empty list for $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}')$ since $\tilde{y}$ is not $(1/2 - \sqrt{\eta})$-close to $\mathcal{C}$.

9.1 List Decoding Framework

We recall the precise statement of the list decoding framework tailored to direct sum lifting.

Theorem 9.2 (List Decoding Theorem (Adapted from [AJQ+20])). *Suppose* $\mathrm{dsum}_{W(k)}$ *is an* $(\eta^8/2^{30}, L)$*-two-step tensorial direct sum lifting from an* η_0*-balanced code* $\mathcal{C} \subseteq \mathbb{F}_2^n$ *to* $\mathcal{C}'$ *on a multiset* $W(k) \subseteq [n]^k$ *which is a* $(1/2 + \eta_0/2, \eta)$*-parity sampler.*

Let $\tilde{y} \in \mathbb{F}_2^{W(k)}$ *be* $(1/2 - \sqrt{\eta})$*-close to* $\mathcal{C}'$*. Then the List Decoding algorithm returns the coupled code list* $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}')$*. Furthermore, the running time is* $n^{O(L+k)}\,(\mathrm{polylog}(1/\eta) + f(n))$ *where* $f(n)$ *is the running time of an unique decoding algorithm of* $\mathcal{C}$*.*

We apply the list decoding framework of Theorem 9.2 to the liftings arising in the Ta-Shma cascade to obtain Theorem 9.1. This requires choosing parameters so that both the parity sampling and tensoriality requirements are met at every level of the cascade, which we do by appealing to our results from Section 7.

Proof of Theorem 9.1. We want to define parameters for τ-splittability so that $W(k)$ satisfies strong enough parity sampling and tensoriality assumptions to apply Theorem 9.2.

For parity sampling, we require $W(k)$ to be an $(1/2 + \eta_0/2, \eta)$-parity sampler. Suppose $W(k)$ is τ-splittable with $\tau < 1/16$. By Corollary 7.4 or Corollary 7.7 and splittability, the collection of walks $W(k)$ is an (η_0', η')-parity sampler, where $\eta' \leq (\eta_0' + 2\tau)^{\lfloor (k-1)/2 \rfloor}$. To achieve the desired parity sampling, we take $\eta_0' = 1/2 + \eta_0/2$ and choose a value of k large enough so that $\eta' \leq \eta$. Using the assumption $\eta_0 < 1/4$, we compute

$$\eta' = (\eta_0' + 2\tau)^{\lfloor (k-1)/2 \rfloor} \leq (1/2 + \eta_0/2 + 2\tau)^{k/2-1} < (3/4)^{k/2-1},$$

which will be smaller than η as long as k is at least

$$k_0(\eta) = 2\left(1 + \frac{\log(1/\eta)}{\log(4/3)}\right) = \Theta(\log(1/\eta)).$$

Achieving this level of parity sampling also ensures that the lifted code $\mathcal{C}'$ is η-balanced.

The list decoding theorem also requires $(\eta^8/2^{30}, L)$-two-step tensoriality. Lemma 7.24 (with $s = k$) and Lemma 7.25 each provide (μ, L)-two-step tensoriality for τ-splittable walk collections on the s-wide replacement product and using $\mathrm{S}^{\triangle}_{r,r}$, respectively, with

$$L \geq \frac{128k^4 \cdot 2^{4k}}{\mu^4} \quad \text{and} \quad \tau \leq \frac{\mu}{4k \cdot 2^{4k}}.$$

To get $\mu = \eta^8/2^{30}$, we require

$$L \geq \frac{K' \cdot k^4 \cdot 2^{4k}}{\eta^{32}} \quad \text{and} \quad \tau \leq \tau_0(\eta, k) = \frac{\eta^8}{K \cdot k \cdot 2^{4k}},$$

where K and K' are (very large) constants. This ensures that τ is small enough for the parity sampling requirement as well. With these parameters, the running time for the list decoding algorithm in Theorem 9.2 becomes

$$n^{O(L+k)}(\mathrm{polylog}(1/\eta) + f(n)) = n^{O(L)} \cdot f(n) = n^{O(1/\tau_0(\eta,k)^4)} \cdot f(n).$$

■

For decoding in fixed polynomial time, we also need a variation of list decoding where we don't run the unique decoding algorithm of the base code and only obtain an approximate list of solutions. The proof is very similar to the proof of Theorem 9.1 above.

Theorem 9.3 (Restatement of Theorem 6.12)**.** *Let $\eta_0 \in (0, 1/4)$ be a constant, $\eta \in (0, \eta_0)$, $\zeta = 1/8 - \eta_0/8$, and*

$$k \geq k_0'(\eta) := \Theta(\log(1/\eta)).$$

Suppose $\mathcal{C} \subseteq \mathbb{F}_2^n$ is an η_0-balanced linear code and $\mathcal{C}' = \mathrm{dsum}_{W(k)}(\mathcal{C})$ is the direct sum lifting of $\mathcal{C}$ on a τ-splittable collection of walks $W(k)$, where $W(k)$ is either the set of walks $W[0, s]$ on an s-wide replacement product graph or a set of walks using the random walk operator $\mathsf{S}^{\triangle}_{r,r}$. There exists an absolute constant $K > 0$ such that if

$$\tau \leq \tau_0(\eta, k) := \frac{\eta^8}{K \cdot k \cdot 2^{4k}},$$

then the code $\mathcal{C}'$ is η-balanced, $W(k)$ is a $(1 - 2\zeta, \eta)$-parity sampler, and we have the following:

If $\tilde{y}$ is $(1/2 - \sqrt{\eta})$-close to $\mathcal{C}'$, then we can compute a ζ-cover $\mathcal{L}'$ of the list

$$\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}') := \left\{ (z, \mathrm{dsum}_{W(k)}(z)) \mid z \in \mathcal{C}, \Delta\Big(\mathrm{dsum}_{W(k)}(z), \tilde{y}\Big) \leq \frac{1}{2} - \sqrt{\eta} \right\}$$

in which $\Delta(y', \tilde{y}) \leq 1/2 - \sqrt{\eta}$ for every $(z', y') \in \mathcal{L}'$ [10]*, in time*

$$n^{O(1/\tau_0(\eta,k)^4)}.$$

Otherwise, we return $\mathcal{L}' = \emptyset$ with the same running time of the preceding case.

Proof. The list decoding framework produces a cover $\mathcal{L}'$ of the list $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}')$, and, as its final step, corrects the cover to obtain the actual list $\mathcal{L}(\tilde{y}, \mathcal{C}, \mathcal{C}')$ by running the unique decoding algorithm of $\mathcal{C}$ on each entry of $\mathcal{L}'$ (see [AJQ+20] for details). Using Theorem 9.2 with a $(1 - 2\zeta, \eta)$-parity sampler and omitting this final step of the algorithm, we can obtain the ζ-cover $\mathcal{L}'$ in time $n^{O(L+k)}\mathrm{polylog}(1/\eta)$.

The tensoriality part of the proof of Theorem 9.1 applies here unchanged, so we need only make sure k is large enough to yield the stronger parity sampling necessary for this theorem. As in that proof, we have that $W(k)$ is an (η_0', η')-parity sampler with $\eta' \leq (\eta_0' + 2\tau)^{\lfloor (k-1)/2 \rfloor}$. Take $\eta_0' = 1 - 2\zeta = 3/4 + \eta_0/4$. Using $\eta_0 < 1/4$ and assuming $\tau < 1/16$, we have

$$\eta' \leq (\eta_0' + 2\tau)^{\lfloor (k-1)/2 \rfloor} \leq (3/4 + \eta_0/4 + 2\tau)^{k/2-1} < (15/16)^{k/2-1},$$

which will be smaller than η as long as k is at least

$$k_0'(\eta) = 2\left(1 + \frac{\log(1/\eta)}{\log(16/15)}\right) = \Theta(\log(1/\eta)).$$

■

[10]A randomized rounding will ensure this, but see Appendix D for obtaining this property deterministically.

Acknowledgement

We thank Amnon Ta-Shma for suggesting the problem of decoding in fixed polynomial running time (with the exponent of N independent of ε) which led us to think about Theorem 6.9. Part of this work was done when some of the authors were visiting the Simons Institute in Berkeley, and we thank them for their kind hospitality.

References

[ABN+92] N. Alon, J. Bruck, J. Naor, M. Naor, and R. Roth. Construction of asymptotically good, low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 28:509–516, 1992. 2

[AGHP92] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992. 1, 42, 58

[AJQ+20] Vedat Levi Alev, Fernando Granha Jeronimo, Dylan Quintana, Shashank Srivastava, and Madhur Tulsiani. List decoding of direct sum codes. In *Proceedings of the 31st ACM-SIAM Symposium on Discrete Algorithms*, pages 1412–1425. SIAM, 2020. 3, 6, 7, 9, 10, 17, 18, 20, 27, 35, 37, 42, 48, 49, 50, 60

[AJT19] Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high-dimensional expanders. In *Proceedings of the 60th IEEE Symposium on Foundations of Computer Science*, pages 180–201, 2019. 7, 37, 40

[Ari09] E. Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, July 2009. 4

[Aro02] Sanjeev Arora. How NP got a new definition: a survey of probabilistically checkable proofs. In *Proceedings of the International Congress of Mathematicians*, pages 637–648, 2002. Volume 3. 2

[Bog12] Andrej Bogdanov. A different way to improve the bias via expanders. Lecture notes, April 2012. URL: `http://www.cse.cuhk.edu.hk/~andrejb/csc5060/notes/12L12.pdf`. 2, 6, 18

[BRS11] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science*, pages 472–481, 2011. 8, 33, 37, 38, 40, 55

[Cha16] Siu On Chan. Approximation resistance from pairwise-independent subgroups. *J. ACM*, 63(3), August 2016. 2

[Chu97] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997. 55

[CT06] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006. 56

[DDG$^+$15] Roee David, Irit Dinur, Elazar Goldenberg, Guy Kindler, and Igor Shinkar. Direct sum testing. ITCS '15, pages 327–336, New York, NY, USA, 2015. ACM. 2

[Del75] P. Delsarte. The association schemes of coding theory. In *Combinatorics*, pages 143–161. Springer Netherlands, 1975. 1

[DHK$^+$19] Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon Ta-Shma. List decoding with double samplers. In *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms*, pages 2134–2153, 2019. 2

[DK17] Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science*, pages 974–985, 2017. 2

[DS14] Irit Dinur and David Steurer. Direct product testing. In *Proceedings of the 29th IEEE Conference on Computational Complexity*, CCC '14, pages 188–196, 2014. 2

[Gal62] R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1):21–28, 1962. 4

[GI01] Venkatesan Guruswami and Piotr Indyk. Expander-based constructions of efficiently decodable codes. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, pages 658–667, 2001. 2

[GI03] Venkatesan Guruswami and Piotr Indyk. Linear time encodable and list decodable codes. In *Proceedings of the 35th ACM Symposium on Theory of Computing*, 2003. 3

[GI04] Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting Gilbert-Varshamov bound for low rates. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 756–757, 2004. 3, 4

[Gil52] E.N. Gilbert. A comparison of signalling alphabets. *Bell System Technical Journal*, 31:504–522, 1952. 1, 4

[GKO$^+$17] Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the Gilbert-Varshamov bound. In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 2073–2091, 2017. 4

[GR06] Venkatesan Guruswami and Atri Rudra. Explicit capacity-achieving list-decodable codes. In *Proceedings of the 38th ACM Symposium on Theory of Computing*, pages 1–10, 2006. 1, 3, 4

[GR08] Venkatesan Guruswami and Atri Rudra. Concatenated codes can achieve list-decoding capacity. In *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 258–267, 2008. 4

[Gri01] Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001. 7

[GRS19] Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory. Available at `https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/index.html`, 2019. 1

[GRY19] Venkatesan Guruswami, Andrii Riazanov, and Min Ye. Arikan meets Shannon: Polar codes with near-optimal convergence to channel capacity. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:154, 2019. 4

[GS11] Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with psd objectives. In *FOCS*, pages 482–491, 2011. 8

[Gur05] Venkatesan Guruswami. Algebraic-geometric generalizations of the Parvaresh-Vardy codes. *Electronic Colloquium on Computational Complexity (ECCC)*, (132), 2005. 4

[Gur09] Venkatesan Guruswami. List decoding of binary codes–a brief survey of some recent results. In *Coding and Cryptology*, pages 97–106. Springer Berlin Heidelberg, 2009. 1, 2, 3, 4

[Gur10] Venkatesan Guruswami. Bridging Shannon and Hamming: List error-correction with optimal rate. In *ICM*, 2010. 1, 4

[Ham50] Richard Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29:147–160, 1950. 4

[Hås97] J. Håstad. Some optimal inapproximability results. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 1–10, 1997. 3

[Hås01] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001. 7

[HRW17] B. Hemenway, N. Ron-Zewi, and M. Wootters. Local list recovery of high-rate tensor codes applications. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science*, pages 204–215, Oct 2017. 4

[IKW09] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. New direct-product testers and 2-query PCPs. In *Proceedings of the 41st ACM Symposium on Theory of Computing*, STOC '09, pages 131–140, 2009. 2

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ unless E has sub-exponential circuits. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 220–229, 1997. 2

[LPS88] Alexander Lubotzky, R. Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8:261–277, 1988. 57

[MMT11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In *Advances in Cryptology – ASIACRYPT 2011*, pages 107–124, 2011. 3

[MRRW77] R. McEliece, E. Rodemich, H. Rumsey, and L. Welch. New upper bounds on the rate of a code via the Delsarte-MacWilliams inequalities. *IEEE Transactions on Information Theory*, 23(2):157–166, 1977. 1

[MRRZ$^+$19] Jonathan Mosheiff, Nicolas Resch, Noga Ron-Zewi, Shashwat Silas, and Mary Wootters. LDPC codes achieve list decoding capacity, 2019. arXiv:1909.06430. 4

[NN90] J. Naor and M. Naor. Small-bias probability spaces: efficient constructions and applications. In *Proceedings of the 22nd ACM Symposium on Theory of Computing*, pages 213–223, 1990. 1

[NS09] Michael Navon and Alex Samorodnitsky. Linear programming bounds for codes via a covering argument. *Discrete Comput. Geom.*, 41(2):199–207, March 2009. 1

[RT12] Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *Proceedings of the 23rd ACM-SIAM Symposium on Discrete Algorithms*, SODA 2012, pages 373–387, 2012. 56

[RVW00] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, 2000. 2, 10, 11, 13

[Sha48] Claude Shannon. A mathematical theory of communications. *Bell System Technical Journal*, 27:379–423, 623–656, 1948. 4

[SKS19] Ronen Shaltiel, Swastik Kopparty, and Jad Silbak. Quasilinear time list-decodable codes for space bounded channels. 2019. 4

[Sti08] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Springer Publishing Company, Incorporated, 2nd edition, 2008. 3

[Tho83] C. Thommesen. The existence of binary linear concatenated codes with Reed-Solomon outer codes which asymptotically meet the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 29(6):850–853, November 1983. 3, 4

[TS17] Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th ACM Symposium on Theory of Computing*, STOC 2017, pages 238–251, New York, NY, USA, 2017. ACM. 1, 2, 4, 5, 6, 10, 15, 16, 18, 41, 57, 58

[Var57] R.R. Varshamov. Estimate of the number of signals in error correcting codes. *Doklady Akademii Nauk SSSR*, 117:739–741, 1957. 1, 4

[vL99] Jacobus H. van Lint. *Introduction to Coding Theory*. Springer-Verlag, 1999. 3

A Auxiliary Results to Obtain Tensoriality

A key result used in the SOS rounding analysis is embodied in Lemma A.1 below. Roughly speaking, it quantifies the decrease in the potential Φ^G, under conditioning on a random

$\mathbf{Y}_i$ for $i \sim V$, when the ensemble $\{\mathbf{Y}_i\}$ has non-trivial correlation over the edges and G is a strong enough expander graph. A generalization of this result to low threshold rank graphs was present in [BRS11]. To derive sharper parameters in the simpler expander case and to make the presentation self-contained, we give (essentially) a full proof of this result.

Lemma A.1 (Progress Lemma)**.** *Suppose G satisfy $\lambda_2(G) \leq \beta^2/q^4$. If*

$$\mathop{\mathbb{E}}_{i\sim j}\left[\left\|\{\mathbf{Y}_i\mathbf{Y}_j\} - \{\mathbf{Y}_i\}\{\mathbf{Y}_j\}\right\|_1\right] \geq \beta,$$

then

$$\mathop{\mathbb{E}}_{j\sim V}\left[\Phi^G_{|\mathbf{Y}_j}\right] \leq \Phi^G - \frac{\beta^2}{4\cdot q^4}.$$

A.1 Expander Case

We will need the following characterization of the spectral gap of regular graph G. We denote by A_G its adjacency operator and by L_G its Laplacian operator [Chu97].

Fact A.2 (Spectral Gap [Chu97])**.**

$$\lambda_2(\mathsf{L}_G) \;=\; \min_{v_1,\ldots,v_n\in\mathbb{R}^n} \frac{\mathbb{E}_{i\sim j}\left\|v_i - v_j\right\|^2}{\mathbb{E}_{i,j\sim V}\left\|v_i - v_j\right\|^2}.$$

Using the above characterization, we derive the following local-to-global result.

Lemma A.3 (Local-to-Global)**.** *Let $v_1,\ldots,v_n \in \mathbb{R}^n$ be vectors in the unit ball. Suppose $\lambda_2(\mathsf{L}_G) \geq 1-\beta/2$ (equivalently $\lambda_2(\mathsf{A}_G) \leq \beta/2$). If $\mathbb{E}_{i\sim j}\langle v_i, v_j\rangle \geq \beta$, then*

$$\mathbb{E}_{i,j\sim V}\langle v_i, v_j\rangle \;\geq\; \frac{\beta}{2}.$$

Proof. Using Fact A.2, we have

$$\lambda_2(\mathsf{L}_G) \;\leq\; \frac{\mathbb{E}_{i\sim V}\|v_i\|^2 - \mathbb{E}_{i\sim j}\langle v_i, v_j\rangle}{\mathbb{E}_{i\sim V}\|v_i\|^2 - \mathbb{E}_{i,j\sim V}\langle v_i, v_j\rangle}.$$

Set $\lambda_2 = \lambda_2(\mathsf{L}_G)$. We consider two cases: $\lambda_2 \leq 1$ and $\lambda_2 > 1$. First, suppose $\lambda_2 \leq 1$. Then

$$\begin{aligned}\mathbb{E}_{i,j\sim V}\langle v_i, v_j\rangle \;&\geq\; \frac{1}{\lambda_2}\mathbb{E}_{i\sim j}\langle v_i, v_j\rangle - \left(\frac{1-\lambda_2}{\lambda_2}\right)\mathbb{E}_{i\sim V}\|v_i\|^2\\ &\geq\; \frac{1}{\lambda_2}\left(\beta - (1-\lambda_2)\right)\\ &\geq\; \frac{1}{\lambda_2}\left(\beta - \left(\frac{\beta}{2}\right)\right) \geq \frac{\beta}{2}.\end{aligned}$$

Now suppose $\lambda_2 > 1$. Then

$$\begin{aligned}\mathbb{E}_{i,j\sim V}\langle v_i, v_j\rangle \;&\geq\; \frac{1}{\lambda_2}\mathbb{E}_{i\sim j}\langle v_i, v_j\rangle - \left(\frac{1-\lambda_2}{\lambda_2}\right)\mathbb{E}_{i\sim V}\|v_i\|^2\\ &\geq\; \frac{1}{\lambda_2}\mathbb{E}_{i\sim j}\langle v_i, v_j\rangle \geq \frac{1}{\lambda_2}\cdot\beta \geq \frac{\beta}{2},\end{aligned}$$

where the last inequality follows from $\lambda_2 \leq 2$ for any graph G. ∎

More Preliminaries

We will need some standard notions in information theory [CT06].

Definition A.4 (Relative Entropy/Kullback-Leibler Divergence)**.** *The relative entropy of two distributions D_1 and D_2 with support contained in $\mathcal{Q}$ is*

$$\mathrm{KL}(D_1, D_2) \;\coloneqq\; \sum_{a \in \mathcal{Q}} D_1(a) \log\left(\frac{D_1(a)}{D_2(a)}\right).$$

Notation A.5. *Let $\mathbf{X}$ be a random variable. We denote by $\{\mathbf{X}\}$ the distribution of $\mathbf{X}$.*

Definition A.6 (Mutual Information)**.** *Let $\mathbf{X}, \mathbf{Y}$ be two random variables. The mutual information $\mathrm{I}(\mathbf{X}, \mathbf{Y})$ is*

$$\mathrm{I}(\mathbf{X}, \mathbf{Y}) \;\coloneqq\; \mathrm{KL}(\{\mathbf{X}, \mathbf{Y}\}, \{\mathbf{X}\}\{\mathbf{Y}\}).$$

Fact A.7.

$$\mathrm{I}(\mathbf{X}, \mathbf{Y}) \;=\; \mathrm{H}(\mathbf{X}) - \mathrm{H}(\mathbf{X}|\mathbf{Y}).$$

Fact A.8 (Fact B.5 of Raghavendra and Tan [RT12])**.** *Let $\mathbf{X}_a$ and $\mathbf{X}_b$ be indicator random variables. Then*

$$\mathrm{Cov}(\mathbf{X}_a, \mathbf{X}_b)^2 \;\le\; 2 \cdot \mathrm{I}(\mathbf{X}_a, \mathbf{X}_b).$$

Progress Lemma

We are ready to prove Lemma A.1 which we restate below for convenience.

Lemma A.9 (Progress Lemma (restatement of Lemma A.1))**.** *Suppose G satisfy $\lambda_2(G) \le \beta^2/q^4$. If*

$$\mathop{\mathbb{E}}_{i \sim j}\left[\left\|\{\mathbf{Y}_i\mathbf{Y}_j\} - \{\mathbf{Y}_i\}\{\mathbf{Y}_j\}\right\|_1\right] \ge \beta,$$

then

$$\mathop{\mathbb{E}}_{j \sim V}\left[\Phi^G_{|\mathbf{Y}_j}\right] \le \Phi^G - \frac{\beta^2}{4 \cdot q^4}.$$

Proof. Firstly, we show how to relate the distances $\left\|\{\mathbf{Y}_i\mathbf{Y}_j\} - \{\mathbf{Y}_i\}\{\mathbf{Y}_j\}\right\|_1$ over the edges $i \sim j$ to certain covariances. Let $a, b \in [q]^2$. Observe that

$$\left|\mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)\right| \;=\; \left|\Pr[\mathbf{Y}_i = a \wedge \mathbf{Y}_j = b] - \Pr[\mathbf{Y}_i = a]\Pr[\mathbf{Y}_j = b]\right|.$$

We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{i \sim j}\left[\frac{1}{q^2}\sum_{a,b \in [q]^2} \mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)^2\right] \;&\ge\; \left(\mathop{\mathbb{E}}_{i \sim j}\left[\frac{1}{q^2}\sum_{a,b \in [q]^2}\left|\mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)\right|\right]\right)^2 \\
&\ge\; \frac{1}{q^4}\left(\mathop{\mathbb{E}}_{i \sim j}\left[\left\|\{\mathbf{Y}_i\mathbf{Y}_j\} - \{\mathbf{Y}_i\}\{\mathbf{Y}_j\}\right\|_1\right]\right)^2 \ge \frac{\beta^2}{q^4}.
\end{aligned}$$

Note that the graph $\mathcal{F} \coloneqq G \otimes J/q$ is an expander with $\lambda_2(G \otimes J/q) = \lambda_2(G)$. Moreover, the matrix $\mathsf{C} \coloneqq \{\mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)\}_{i,j \in V; a,b \in [q]^2}$ is PSD since the vectorization $\{v_{i,a} -$

$\mathbb{E}[\mathbf{Y}_{i,a}] \cdot v_\varnothing\}_{i \in V; a \in [q]}$ gives a Gram matrix decomposition of C. Thus, the covariance matrix $\{\mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)^2\}_{i,j \in V; a,b \in [q]^2}$ is also PSD since it is the Schur product (i.e., entrywise product) of two PSD matrices, namely, $\mathsf{C} \circ \mathsf{C}$. Therefore, we are in position of applying the local-to-global Lemma A.3 with the expander $\mathcal{F}$ and a vectorization for $\mathsf{C} \circ \mathsf{C}$. We have

$$\begin{aligned}
\frac{\beta^2}{q^4} &\le \mathop{\mathbb{E}}_{i \sim j}\left[\frac{1}{q^2}\sum_{a,b \in [q]^2} \mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)^2\right] \\
&\le 2 \mathop{\mathbb{E}}_{i,j \sim V^{\otimes 2}}\left[\frac{1}{q^2}\sum_{a,b \in [q]^2} \mathrm{Cov}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)^2\right] && \text{(local-to-global Lemma A.3)} \\
&\le \frac{4}{q^2} \mathop{\mathbb{E}}_{i,j \sim V^{\otimes 2}}\left[\sum_{a,b \in [q]^2} \mathrm{I}\left(\mathbf{Y}_{i,a}, \mathbf{Y}_{j,b}\right)\right] && \text{(Fact A.8)} \\
&\le \frac{4}{q^2} \mathop{\mathbb{E}}_{i,j \sim V^{\otimes 2}}\left[\sum_{a,b \in [q]^2} \mathrm{H}\left(\mathbf{Y}_{i,a}\right) - \mathrm{H}\left(\mathbf{Y}_{i,a} | \mathbf{Y}_{j,b}\right)\right] \\
&\le \frac{4}{q}\left[\mathop{\mathbb{E}}_{i \sim V}\left[\sum_{a \in [q]} \mathrm{H}\left(\mathbf{Y}_{i,a}\right)\right] - \mathop{\mathbb{E}}_{i,j \sim V^{\otimes 2}}\left[\sum_{a \in [q]} \mathrm{H}\left(\mathbf{Y}_{i,a} | \mathbf{Y}_{j}\right)\right]\right] \\
&= 4\left[\mathop{\mathbb{E}}_{i \sim V}\left[\mathcal{H}\left(\mathbf{Y}_i\right)\right] - \mathop{\mathbb{E}}_{i,j \sim V^{\otimes 2}}\left[\mathcal{H}\left(\mathbf{Y}_i | \mathbf{Y}_j\right)\right]\right] \\
&= 4\left[\Phi^G - \mathop{\mathbb{E}}_{j \sim V}\left[\Phi^G_{|\mathbf{Y}_j}\right]\right].
\end{aligned}$$

Therefore, we have $\mathbb{E}_{j \sim V}[\Phi^G_{|\mathbf{Y}_j}] \le \Phi^G - \beta^2/(4 \cdot q^4)$, as claimed. ■

B Explicit Structures

We recall some explicit structures used in Ta-Shma's construction.

B.1 Explicit Ramanujan Graphs

The outer graph G in the s-wide replacement product was chosen to be a Ramanujan graph. Ta-Shma provides a convenient lemma to efficiently obtain explicit Ramanujan graphs given the intended number of vertices n (which might end up being nearly twice this much), the expansion λ and an error parameter $\theta > 0$. These Ramanujan graphs are based on the LPS construction [LPS88]. Due to number theoretic reasons, we might be forced to work with slightly different parameters, but this is not an issue.

Lemma B.1 (Lemma 2.10 [TS17]). *For every $\theta > 0$, there exists an algorithm that given n and $\lambda \in (0,1)$ runs in time* $\mathrm{poly}(n)$ *and outputs a Ramanujan graph G such that*

- *G has degree $d \le 8/\lambda$,*
- *$\sigma_2(G) \le \lambda$, and*

- *$|V(G)|$ is either in the range $[(1-\theta)n, n]$ or in the range $[(1-\theta)2n, 2n]$.*

Moreover, the algorithm outputs a locally invertible function $\varphi\colon [d] \to [d]$ computable in polynomial time in its input length.

B.2 Explicit Biased Distribution

The inner graph H in the s-wide replacement product is chosen to be a Cayley graph on $\mathbb{Z}_2^m$ for some positive integer m. Ta-Shma uses the construction of Alon et al. [AGHP92] (AGHP) to deduce a result similar to Lemma B.2 below. To compute the refined parameter version of our main result Theorem 1.1, we will need the specifics of the AGHP construction.

Lemma B.2 (Based on Lemma 6 [TS17])**.** *For every $\beta = \beta(m)$, there exists a fully explicit set $A \subseteq \mathbb{Z}_2^m$ such that*

- *$|A| \leq 4 \cdot m^2/\beta^2$, and*
- *for every $S \subseteq [m]$, we have $|\mathbb{E}_{z \in A} \chi_S(z)| \leq \beta$.*

Furthermore, if m/β is a power of 2, then $|A| = m^2/\beta^2$. In particular, the graph $\mathsf{Cay}(\mathbb{Z}_2^m, A)$ is a $(n = 2^m, d = |A|, \lambda = \beta)$ expander graph.

Remark B.3. *Given $d, m \in \mathbb{N}^+$ such that d is the square of a power of 2 with $d \leq 2^m$, by setting $\beta = m/\sqrt{d}$ we can use Lemma B.2 with β and m (note that m/β is a power of 2) to obtain a Cayley graph $\mathsf{Cay}(\mathbb{Z}_2^m, A)$ with parameters $(n = 2^m, d = |A|, \lambda = \beta)$.*

C Zig-Zag Spectral Bound

We prove the zig-zag spectral bound Fact 4.4.

Claim C.1. *Let G be an outer graph and H be an inner graph used in the s-wide replacement product. For any integer $0 \leq i \leq s-1$,*

$$\sigma_2((I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)) \leq \sigma_2(G) + 2 \cdot \sigma_2(H) + \sigma_2(H)^2.$$

Proof. Let v be a unit vector such that $v \perp \mathbf{1}$, and decompose it into $v = u + w$ such that $u \in \mathcal{W}^{\parallel} = \mathrm{span}\{a \otimes b \in \mathbb{R}^{V(G)} \otimes \mathbb{R}^{V(H)} \mid b = \mathbf{1}\}$ and $w \in \mathcal{W}^{\perp} = (\mathcal{W}^{\parallel})^{\perp}$.

$$\begin{aligned}
|\langle v, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)v\rangle| &\leq |\langle u, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)u\rangle| + |\langle u, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)w\rangle| + \\
&\quad |\langle w, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)u\rangle| + |\langle w, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)w\rangle| \\
&\leq |\langle u, G_i u\rangle| + |(I \otimes \mathsf{A}_H)w| + \\
&\quad |(I \otimes \mathsf{A}_H)w| + |(I \otimes \mathsf{A}_H)w|^2 \\
&\leq |\langle u, G_i u\rangle| + 2\sigma_2(H) + \sigma_2^2(H)
\end{aligned}$$

To bound $|\langle u, (I \otimes \mathsf{A}_H)G_i(I \otimes \mathsf{A}_H)u\rangle|$, observe that $u = x \otimes \mathbf{1}$ for some $x \in \mathbb{R}^{V(G)}$. Then,

$$0 = \langle v, \mathbf{1}\rangle = \langle u, \mathbf{1}\rangle + \langle w, \mathbf{1}\rangle = \langle u, \mathbf{1}\rangle = \langle x, \mathbf{1}_G\rangle$$

so that $x \perp \mathbf{1}_G$. Because u is uniform over H-component, $|\langle u, G_i u\rangle| = |\langle x, Gx\rangle| \leq \sigma_2(G)$, which completes the proof. ■

We also derive a (simple) tighter bound for the expansion of the zig-zag product in a particular parameter regime.

Claim C.2. *Let G be a λ_1-two-sided expander and H be a λ_2-two-sided expander such that both are regular graphs. If $\lambda_1 \leq \lambda_2$, then*

$$\sigma_2(G \textcircled{z} H) \leq 2 \cdot \lambda_2.$$

Proof. Let $v = a \cdot v^{\parallel} + b \cdot v^{\perp}$ with $a^2 + b^2 = 1$ be such that $v \perp 1$. In particular, if $v^{\parallel} = v^G \otimes 1^H$, then $v^G \perp 1^G$ since otherwise $\langle v, 1 \rangle = \langle v^G, 1^G \rangle \neq 0$. We have

$$\begin{aligned}\max_{a,b \in \mathbb{R}\colon a^2+b^2=1} a^2 \cdot \lambda_1 + 2ab \cdot \lambda_2 + b^2 \cdot \lambda_2^2 &\leq \max_{a,b \in \mathbb{R}\colon a^2+b^2=1} a^2 \cdot \lambda_2 + 2ab \cdot \lambda_2 + b^2 \cdot \lambda_2 \\ &= \max_{a,b \in \mathbb{R}\colon a^2+b^2=1} \lambda_2 + 2ab \cdot \lambda_2,\end{aligned}$$

where the inequality follows from the assumption $\lambda_1 \leq \lambda_2$ (and trivially $\lambda_2^2 \leq \lambda_2$) and the equality follows from $a^2 + b^2 = 1$. Since we also have $2ab = (a+b)^2 - (a^2 + b^2) \leq 1$, the result follows. ■

D Derandomization

To unique decode in fixed polynomial time (i.e., poly(n/ε)) we need to prune the list of coupled words $\mathcal{L}$ covering the list $\mathcal{L}^*(\tilde{y}) = \{(z, y = \mathrm{dsum}(z)) \mid z \in \mathcal{C}, \Delta(\tilde{y}, y) \leq 1/2 - \sqrt{n}\}$ of codewords we want to retrieve. To do so, given $(z^*, y^* = \mathrm{dsum}(z^*)) \in \mathcal{L}^*(\tilde{y})$, we need to have $(z, y = \mathrm{dsum}(z)) \in \mathcal{L}$ such that

1. $|\langle y^*, \mathrm{dsum}(z) \rangle|$ is not too small, and
2. $\langle \tilde{y}, \mathrm{dsum}(z) \rangle$ is not too small (in order to apply Lemma 6.11).

The slice (S, σ) of the SOS solution from which y^* is recoverable satisfies in expectation

$$\mathbb{E}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle y^*, \mathrm{dsum}(z) \rangle^2 \right] \geq 3\eta^2,$$

and

$$\mathbb{E}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle \tilde{y}, \mathrm{dsum}(z) \rangle \right] \geq 3\sqrt{\eta}/2.$$

Moreover, since $z \mapsto \langle y^*, \mathrm{dsum}(z) \rangle^2$ and $z \mapsto \langle \tilde{y}, \mathrm{dsum}(z) \rangle$ are $O(1/n)$-Lipschitz [11] with respect to the ℓ_1-norm, Hoeffding's inequality gives

$$\mathop{\mathbb{P}}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle y^*, \mathrm{dsum}(z) \rangle^2 < \eta^2 \right] \leq \exp\left(-\Theta(n)\right),$$

and

$$\mathop{\mathbb{P}}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle \tilde{y}, \mathrm{dsum}(z) \rangle < \sqrt{\eta} \right] \leq \exp\left(-\Theta(n)\right).$$

[11] In this fixed polynomial time regime, the parameters $s, d_1, d_2, \varepsilon_0, \eta$ are constant independent of the final bias ε.

At least randomly, such a z can be easily found. In [AJQ+20], alternatively to satisfying Item 1 it was shown that by choosing $z' \in \{\pm 1\}^n$ by majority vote, i.e.

$$z'_i = \operatorname*{argmax}_{b \in \{\pm 1\}} \mathbb{P}[\mathbf{Z}_i = b]$$

for $i \in [n]$, one has that $|\langle z^*, z' \rangle|$ is large which is enough to address the first item. More precisely implicit in [AJQ+20], for any constant $\beta \in (0,1)$ as long as parity sampling is sufficiently strong we have

$$\mathbb{E}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle z', z \rangle^2 \right] \geq 1 - \beta.$$

Similarly $z \mapsto \langle z', z \rangle^2$ is $O(1/n)$-Lipschitz with respect to the ℓ_1-norm, so Hoeffding's inequality yields

$$\mathbb{P}_{z \sim \{\mathbf{Z}^{\otimes}|_{(S,\sigma)}\}} \left[\langle z', z \rangle^2 < 1 - \beta/2 \right] \leq \exp\left(-\Theta(n)\right).$$

However, we want to efficiently and deterministically find a z satisfying $\langle z', z \rangle^2 \geq 1 - \beta/2$ as well as satisfying Item 2. Note that at this stage in the decoding process y^* is not known (without issuing a recursive unique decoding call), so running expectation maximization to satisfy item Item 1 would not be possible. Fortunately, the majority z' can be cheaply computed without a recursive call to an unique decoder. On the other hand z satisfying only Item 2 can be found by expectation maximization. The challenge is to satisfy both conditions at the same time. For this reason, we design a simultaneous expectation maximization derandomization procedure tailored to our setting.

D.1 Abstract Derandomization: Simultaneous Expectation Maximization

Suppose that Ω is a probability space where two random variables $\mathbf{A}$ and $\mathbf{B}$ are defined satisfying the following first moment conditions

$$\mathbb{E}[\mathbf{A}] \geq a \quad \text{and} \quad \mathbb{E}[\mathbf{B}] \geq 1 - \beta.$$

We provided a sufficient conditions so that $\omega \in \Omega$ satisfying

$$\mathbf{A}(\omega) \geq a' \quad \text{and} \quad \mathbf{B}(\omega) \geq 1 - \beta'$$

can be efficiently deterministically found with the aid of an oracle, where $a \approx a'$ and $\beta \approx \beta'$. More precisely, we have the following lemma.

Lemma D.1. *Let $\Omega = (\{-1,1\}^n, \nu_1 \times \cdots \times \nu_n)$ be a probability space with a product distribution. Suppose $\mathbf{A} \in [-1,1]$ is a random variable on Ω satisfying, for $a > 0$ and for some function $e_A \colon \mathbb{N} \to \mathbb{R}^+$,*

$$\mathbb{P}[\mathbf{A} < a] \leq e_A(n).$$

Suppose $\mathbf{B} \in [-1,1]$ is a random variable on Ω satisfying, for some function $e_B \colon \mathbb{N} \times \mathbb{R}^+ \to \mathbb{R}^+$,

$$\mathbb{P}[\mathbf{B} < 1 - \beta] \leq e_B(n, \beta).$$

Suppose that there is an oracle to evaluate $\mathbb{E}\left[\mathbf{A}\mathbf{B}^{2k}\right]$ under any product distribution $\mu'_1 \times \cdots \times \mu'_n$ for $k \in \mathbb{N}$. Given $\delta, \beta \in (0,1)$, if

$$e_A(n) + e_B(n, \beta/(4\lceil -\ln(a(1-\beta)+1)/\delta \rceil)) \leq a\frac{\beta}{2}, \tag{10}$$

then it is possible to find $\omega \in \{\pm 1\}^n$ using $2n$ invocations to the oracle and satisfying

$$\mathbf{A}(\omega) \geq a(1-\beta) \quad \textit{and} \quad |\mathbf{B}(\omega)| \geq 1-\delta.$$

Proof. Set $k = \lceil -\ln(a(1-\beta)+1)/\delta \rceil$. Set $\beta' = \beta/(4k)$. Note that

$$\mathbb{E}\left[\mathbf{A}\mathbf{B}^{2k}\right] \geq a\left(1-\frac{\beta}{4k}\right)^{2k} - e_A(n) - e_B(n,\beta') \geq a\left(1-\beta\right),$$

where we use Eq. (10) in the last inequality. Do expectation maximization to deterministically find $\omega \in \{\pm 1\}^n$, with $2 \cdot n$ invocations to the oracle of $\mathbb{E}\left[\mathbf{A}\mathbf{B}^{2k}\right]$, such that

$$\mathbf{A}(\omega)\mathbf{B}(\omega)^{2k} \geq a\left(1-\beta\right).$$

Since $\mathbf{B}(\omega)^{2k} \leq 1$, we have $\mathbf{A}(\omega) \geq a\left(1-\beta\right)$. Towards a contradiction suppose $|\mathbf{B}(\omega)| \leq 1-\delta$. Using that $\mathbf{A}(\omega) \leq 1$, we have

$$e^{-2k\cdot\delta} \geq (1-\delta)^{2k} \geq \mathbf{A}(\omega)\mathbf{B}(\omega)^{2k} \geq a(1-\beta). \tag{11}$$

By our choice of k, we get

$$e^{-2k\cdot\delta} < a(1-\beta),$$

contradicting Eq. (11). ■

D.2 Implementing the Oracle

Now, we provide an efficient deterministic oracle for our setting. We take

$$\mathbf{A} := \langle \tilde{y}, \operatorname{dsum}(z) \rangle \quad \text{and} \quad \mathbf{B} := \langle z', z \rangle^2,$$

where $z'_i = \operatorname{argmax}_{b \in \{\pm 1\}} \mathbb{P}[\mathbf{Z}_i = b]$. Note that

$$\mathbf{A}\mathbf{B}^{2k} = \sum_{T \subset [n] \colon |T| = O(1)} \alpha_T \prod_{i \in T} z_i.$$

To compute $\mathbb{E}\left[\mathbf{A}\mathbf{B}^{2k}\right]$ under any product distribution $\mu'_1 \times \cdots \times \mu'_n$, use linearity of expectation and sum at most $n^{O(1)}$ terms $\alpha_T \mathbb{E}\left[\prod_{i \in T} z_i\right]$ where each can be computed in $O(1)$ since restricted to T we have a product distribution taking values in $\{\pm 1\}^T$.