List-Decoding with Double Samplers *

Irit Dinur[†]   Prahladh Harsha[‡]   Tali Kaufman[§]   Inbal Livni Navon[¶]

Amnon Ta-Shma[‖]

May 6, 2021

Abstract

We strengthen the notion of *double samplers*, first introduced by Dinur and Kaufman [*Proc. 58th FOCS*, 2017], which are samplers with additional combinatorial properties, and whose existence we prove using high dimensional expanders.

The ABNNR code construction [*IEEE Trans. Inform. Theory*, 38(2):509–516] achieves large distance by starting with a base code C with moderate distance, and then amplifying the distance using a sampler. We show that if the sampler is part of a larger double sampler then the construction has an efficient list-decoding algorithm. Our algorithm works even if the ABNNR construction is not applied to a base code C but to any string. In this case the resulting code is *approximate*-list-decodable, i.e. the output list contains an approximation to the original input.

Our list-decoding algorithm works as follows: it uses a local voting scheme from which it constructs a unique games constraint graph. The constraint graph is an expander, so we can solve unique games efficiently. These solutions are the output of the list-decoder. This is a novel use of a unique games algorithm as a subroutine in a decoding procedure, as opposed to the more common situation in which unique games are used for demonstrating hardness results.

Double samplers and high dimensional expanders are akin to pseudorandom objects in their utility, but they greatly exceed random objects in their combinatorial properties. We believe that these objects hold significant potential for coding theoretic constructions and view this work as demonstrating the power of double samplers in this context.

1 Introduction

We develop the notion of a *double sampler*, which is an enhanced sampler. An (α, β) sampler is a bipartite graph $G = (U, V, E)$ such that for every function $f : V \to [0, 1]$ with expectation $\mu = \mathbb{E}_{v \in V}[f(v)]$, one has $|\mu_u - \mu| \leq \alpha$ for all but a β fraction of the vertices u, where $\mu_u = \mathbb{E}_{v \sim u}[f(v)]$ (see [Zuc97] for more details).

Towards defining double samplers we observe that in every given sampler $G = (U, V, E)$, every $u \in U$ can be identified with the set of its neighbors $\{v \in V \mid v \sim u\}$. In this way U is a collection of subsets of V. In the other direction, given a ground set V and a collection of subsets $\{S \subset V\}$, the graph G pops out as the bipartite inclusion graph with an edge from $v \in V$ to S iff $v \in S$.

A double sampler (see Figure 1 for an illustration) consists of a triple (V_2, V_1, V_0), where V_0 is the ground set, V_1 is a collection of m_1-subsets of V_0 and V_2 is a collection of m_2-subsets of V_0, where $m_2 > m_1$. We say that (V_2, V_1, V_0) is an $(\alpha, \beta), (\alpha_0, \beta_0)$ *double sampler* if

- The inclusion graphs on (V_2, V_1) and (V_2, V_0) are (α, β) samplers, the inclusion graph on (V_1, V_0) is an $(\alpha + \alpha_0, \beta + \beta_0)$ sampler. An inclusion graph is a graph where we connect two subsets by an edge if one contains the other; here a single vertex is also considered to be a singleton subset.

- For every $T \in V_2$, let $V_1(T) = \{S \in V_1 : S \subset T\}$ be the sets in V_1 that are contained in T. Let $G_{|T}$ be the bipartite inclusion graph connecting elements in T (viewed as elements in the ground set V_0) to subsets in $V_1(T)$. We require that for every $T \in V_2$, the graph $G_{|T}$ is an (α_0, β_0) sampler. We call this property the *locality property* of the double sampler.

Our definition of double samplers is stronger than the previous definition due to Dinur and Kaufman [DK17], that was missing the locality property[1]. Whereas the definition of Dinur and Kaufman [DK17] can be obtained e.g. by concatenating two samplers, our definition herein is much stronger and carries properties not known to be obtained by any random construction. It is quite remarkable that high dimensional expanders [LSV05, KO18] give rise to an infinite family of double samplers for which $|V_1|, |V_2| = O(|V_0|)$:

Theorem 1.1 (Informal, see formal version in Theorem 2.11)**.** *For every $\alpha, \beta, \alpha_0, \beta_0 > 0$ there are integers $m_1, m_2 = \mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}), D = \exp(\mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}))$, such that there is an explicit polynomial time construction of an $(\alpha, \beta), (\alpha_0, \beta_0)$ double sampler on n vertices, for infinitely many $n \in \mathbb{N}$, such that the subsets in V_i are of size m_i and the bipartite inclusion graphs on V_0, V_1 and on V_0, V_2 have degree at most D.*

On random double samplers. To appreciate the remarkableness of double samplers, think of concrete parameters such as $m_1 = 2, m_2 = 3$. A random construction amounts to placing n vertices in V_0, a linear (in n) number of edges in V_1 and a linear number of triples in V_2. Consider for example the $G(n, p)$-like model, where triangles are chosen independently. In

---

[1]The main result of Dinur and Kaufman [DK17] was proven directly from high dimensional expanders, and not from double samplers, so this locality property was used implicitly. It is possible that the result of Dinur and Kaufman [DK17] can be proven directly from our revised definition of double samplers.
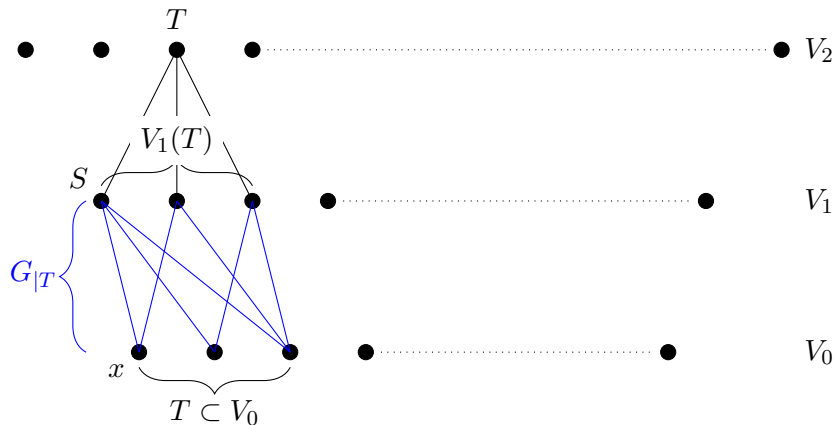
Figure 1: Double sampler. Each vertex $S \in V_1$ is a set containing m_1 elements from V_0, and each vertex $T \in V_2$ is a set containing m_2 elements from V_0. The edges in the graph denote inclusion, for example $x \in S \subset T$.

this case two triangles will almost never share an edge. In either case the inclusion graph on V_1, V_2 is highly disconnected, and certainly not a sampler[2].

We elaborate more on the construction of double samplers towards the end of the introduction.

Samplers and distance amplification. Alon, Bruck, Naor, Naor and Roth [ABNNR92] showed how to amplify the distance of any code, simply by pushing the symbols along edges of a sampler graph. Let us describe their encoding in a notation consistent with the above. We think of the graph as a sampler $G = (V_1, V_0 = [n])$, where V_1 is a collection of m-sets of $[n]$. Given an n-bit string $x \in \{0, 1\}^n$, we place x_i on the i-th vertex and then each subset $S \in V_1$ "collects" all of the symbols of its elements and gets a short string $x|_S : S \to \{0, 1\}$. The resulting codeword is the sequence $E_G(x) := (x|_S)_{S \in V_1}$ which can be viewed as a string of length $|V_1|$ over the alphabet $\Sigma = \{0, 1\}^m$. We refer to the mapping $x \mapsto E_G(x)$ as the ABNNR encoding.

If the string x happens to come from an initial code $C \subset \{0, 1\}^n$ with minimum distance 2α, then, altogether we get $E_G(C) := \{E_G(x) \mid x \in C\}$. Assuming G is an (α, β) sampler, the minimum distance of $E_G(C)$ is at least $1 - \beta$. Of course the length of the words in $E_G(C)$ depends on the size of $|V_1|$, so the shorter the better.

Approximate error correcting code. The ABNNR encoding itself has some interesting decoding properties. As an encoding, E_G does not have a noticeable minimal distance, but still, it is an *approximate*-list-decodable error correcting code. Namely, for every $z \in \Sigma^{|V_1|}$ there is a short list $L_z \subset \{0, 1\}^n$, such that L_z contains an approximation to every string x whose encoding $E_G(x)$ is close to z (in Hamming distance).

The elegant encoding of ABNNR is very local and easy to compute in the forward direction (from x to $E_G(x)$), and indeed it has been found useful in several coding theory constructions,

---

[2]Observe that for the chosen parameters of $m_1 = 2$ and $m_2 = 3$, there are obvious limits on the (α, δ) parameters of the sampler, since each triple is connected to at most 3 edges.

e.g. [GI05, KMRS17]. In this work we study the inverse question, also known as decoding: given a noisy version of $E_G(x)$, find an approximation to x (or x itself, in case $x \in C$). Moreover, we wish to be able to recover from as many errors as possible.

Decoding and list-decoding A decoding algorithm for E_G gets as input a string $(z_S)_{S \in V_1}$, and needs to find a word x such that $x|_S = z_S$ for as many $S \in V_1$ as possible. A natural approach is the "maximum likelihood decoding" algorithm: assign each $i \in [n]$ the most likely symbol, by looking at the "vote" of each of the subsets $S \ni i$,

$$x_i' := \text{majority}_{S:S \ni i}[z_S(i)].$$

The resulting x' is an approximation for x, and in the case where $x \in C$ we can run the decoder of C and retrieve x. Assuming G is a good sampler, this approach gives an approximate-decoding algorithm for E_G that recovers x from error rates almost up to $1/2$.

Going beyond the unique-decoding radius, we show using the Johnson bound that the ABNNR encoding on a sampler graph is (combinatorially) approximate-list-decodable, up to an error rate approaching one as the sampler parameters go to zero (see Section 3). However, the maximum likelihood decoder stops working in this regime: one cannot rule out the situation where for each vertex i, both 0 and 1 symbols occur with equal likelihood, and it is not known, in general,[3] how to recover x.

Thus, it is natural to ask for an algorithm that approximate-list-decodes the ABNNR encoding up to the Johnson bound radius. Our main result is a list-decoding algorithm that goes beyond the unique-decoding barrier of $1/2$ and works for error rates approaching 1. The algorithm works whenever the underlying graph $G = (V_1, [n])$ is part of a double sampler, namely where there is a collection V_2 of sets of size $m_2 > m_1 = m$ so that the triple $(V_2, V_1, [n])$ is a double sampler.

Theorem 1.2 (Main - informal, see Theorem 4.1 and Corollary 4.2). *For every $\gamma, \varepsilon > 0$ there exist $\alpha, \beta, \alpha_0, \beta_0 > 0$, integers m_1, D and an $(\alpha, \beta), (\alpha_0, \beta_0)$-double sampler $(X = (V_2, V_1, [n]))$ such that $V_1 \subseteq \binom{[n]}{m_1}$ and $|V_1| \leq D \cdot n$ and such that the following holds. Let G be the restriction of X to layers V_1 and $[n]$, and let $E_G : \{0,1\}^n \to (\{0,1\}^{m_1})^{V_1}$ be the ABNNR encoding defined above. Then there is a polynomial time algorithm which receives an input $z \in (\{0,1\}^{m_1})^{V_1}$, and outputs a list L_z of size $O(\frac{1}{\gamma^2})$ which includes an ε-approximation for every $x \in \{0,1\}^n$ such that $\text{dist}(E_G(x), z) \leq 1 - \gamma$.*

We omitted here the conditions on the constants and the parameter requirements on X, which appear in Theorem 4.1. We remark that the dependence of m_1 and c on γ and ε is quite far from optimal, and is discussed in Section 4.

Combining our main theorem with a unique-decodable base code C, we get a code $E_G(C)$ that is list-decodable, whenever $G = ([n], V_1)$ is the first two layers of a double sampler,

Corollary 1.3 (Informal, see Corollary 4.4). *For every $\gamma, \varepsilon > 0$, let $(X = (V_2, V_1, [n]))$ be a double sampler as above. Suppose $C \subset \{0,1\}^n$ is an error correcting code with a polynomial time unique-decoding algorithm from an ε-fraction of errors, then the following holds. Let G be the restriction of X to layers V_1 and $[n]$. Then the code $E_G(C) = \{E_G(x) \mid x \in C\}$ has a polynomial time list-decoding algorithm from a $(1 - \gamma)$-fraction of errors, with list size $O(\frac{1}{\gamma})$.*

---

[3]We remark that when the ABNNR encoding is applied over a base code C with additional special properties it is possible that more can be done (see e.g [GI05]), but our focus is on a generic decoding mechanism.

Notice that the list-decoding algorithm in the corollary outputs a shorter list than the approximate-list-decoding algorithm in our main theorem (length $1/\gamma$ versus $1/\gamma^2$). This is because in the case of error correcting codes we can prune the list more efficiently, as explained in Section 3.

At this point the reader may be wondering how the double sampler property helps facilitate list-decoding. Roughly speaking, a double sampler is a collection of (small) subsets that have both large overlaps as well as strong expansion properties. The expansion properties are key for distance amplification, and the large overlaps, again with good sampling properties, are key for the list-decoding algorithm.

1.1 Related work

There are several known list-decodable codes with efficient decoding algorithms, with varying parameters. This includes codes which use algebraic structure, such as Reed Solomon codes [Sud97, GS99], folded Reed Solomon codes [PV05, GR08], multiplicity codes [Kop15], algebraic-geometric codes [GS99, GX14] and constructions using a more combinatorial approach such as [GI01, GI05, GI03]. Most of these constructions get parameters better than our constructions, and some of them get very close to optimal rate and alphabet size. Some of these constructions have other advantages, such as having linear time encoding and decoding algorithms.

Our construction starts with a basic code with constant distance and amplifies it to a code that can be list-decoded from a distance approaching 1, and does it in a black box way. Among all the codes listed above only the construction by Guruswami and Indyk [GI03] does something similar. However, their reduction is recursive and has no underlying double sampler.

It is interesting to compare our construction to the work of Trevisan [Tre03] who showed that derandomized direct product theorems can be used to transform a code from unique-decodable into list-decodable. The work of Impagliazzo, Jaiswal, Kabanets and Wigderson [IJKW10] also uses derandomized direct product to create list-decodable codes. The encoding we use in this work can also be viewed as a derandomized direct product encoding. The rate of the resulting code depends on the quality of the derandomization. Whereas previous derandomizations had a sub constant rate, double samplers give a much stronger derandomization that results in constant rate.

The ABNNR construction starts with a binary code and constructs a code over a larger alphabet by having each vertex $S \in V_1$ collect the bits from all indices $i \in S$. It is natural to consider the *direct sum* variant, where each $S \in V_1$ XORs these bits together into a single bit. It would not be hard to adapt our algorithm to that setting, though we haven't explicitly done so. A subsequent work [AJQST20] went quite a step further and showed an algorithm for list-decoding the direct sum code for a strictly broader family of samplers. Their algorithm uses the Sum-of-Squares (SOS) semi-definite programming hierarchy to list-decode the direct sum for all samplers that satisfy a "splittability" condition which they introduce. These include not only samplers that come from double samplers, but also samplers that come from random walks on expanders, which are not covered by our work.

Even more recently, [JST21] presented an algorithm for list-decoding the direct sum code in nearly linear time. Furthermore, their work gives an efficient decoding algorithm for the error correcting code of [TaS17]. The code in [TaS17] is the first explicit construction of binary

codes with distance close to half and nearly optimal rate. Prior to [JST21], it was not known whether the binary code of [TaS17] can be efficiently decoded.

1.2 The list-decoding algorithm

On input $(z_S)_{S \in V_1}$, our algorithm starts out with a voting step, similar to the maximum likelihood decoder. Here we vote not on the value of each bit $i \in [n]$ but rather on the value of x restricted to an entire set $T \in V_2$. Since the graph $X_{|T}$ between $V_1(T)$ and T is a sampler (this is the locality property), we can come up with a short list of popular candidates for $x|_T$. This is done by looking at z_S for all subsets $S \in V_1$, $S \subset T$. We define

$$\forall T \in V_2, \quad list(T) := \{\sigma \in \{0,1\}^T \ : \ \Pr_{S \subset T, S \in V_1}[z_S = \sigma|_S] > \varepsilon/2 \}.$$

Note that since T has constant size, we are able to search exhaustively over all $\sigma \in \{0,1\}^T$ in constant time.

Given a list for each T, we now need to stitch these lists together, and here we again use the fact that (V_2, V_1) is a good sampler. Whenever $T_1 \cap T_2$ is significantly large, we will match $\sigma_1 \in list(T_1)$ with $\sigma_2 \in list(T_2)$ iff $\sigma_1|_{T_1 \cap T_2} = \sigma_2|_{T_1 \cap T_2}$. Moreover, the double sampler property allows us to come up with an *expander* graph whose vertex set is V_2, and whose edges connect T_1 to T_2 when they have significant overlap. This guarantees that for almost all edges (T_1, T_2) there is a matching between the list of T_1 and the list of T_2.

At this point what we are looking at is a **unique games instance**, where the said expander is the constraint graph, and the said matchings are the unique constraints.[4] We now make two important observations. First, a word with noticeable correlation with the received word, corresponds to a solution for the unique games instance with very high value (i.e., satisfying a large fraction of the constraints). Second, we have an efficient algorithm for finding a high-value solution, because the underlying unique games constraint graph is an expander! This is originally due to Arora, Khot, Kolla, Steurer, Tulsiani Vishnoi [AKKSTV08] but we actually use the variant of Makarychev and Makarychev [MM10], because it has better parameters. A more naive greedy belief propagation algorithm would fail miserably because it takes about $\log n$ steps to reach a typical point in an expander graph, and this accumulates an intolerable $\varepsilon \cdot \log n \gg 1$ amount of error.

If we want an approximate error correcting code we are done. Otherwise, it remains to run the unique-decoding algorithm of C on each of the solutions of the unique games instance, to remove any small errors, and this completes the list-decoding.

The above high level description gives the rough idea for our algorithm, but the implementation brings up some subtle difficulties, which we explain below.

Every set T induces a constant size local view $E_G|_T$ on the code E_G, which has no reason to be an error correcting code, and in particular has no distance. Thus, there could be several valid candidates $\sigma \in \{0,1\}^T$ that are very close in Hamming distance. Suppose $\sigma, \sigma' \in list(T_1)$ differ only in a single bit, then for most $T_2 \cap T_1$, we don't know which element in $list(T_2)$ should be matched to σ and which to σ'. Saying it differently, what we really need to do is to approximately list-decode the local view. Equivalently, for each T we prune $list(T)$ and enforce minimal distance r between each two list items, while holding a "covering" property

---

[4]For definitions, please see the Preliminary section.

- that if σ was in the initial list $List(T)$, then there exists some σ' in the final list that is r-close to σ.

However, for reasons that become clear in the proof we need the following stronger property: We require that the pruned list covers all the elements in $List(T)$ with radius r, while elements in the pruned list are at least $R \gg r$ away from each other (think of R as being $2r$). We show that there is a small set of possible radii r, such that for every T at least one radius from the set is good, in a sense that pruning $list(T)$ with r results in a pruned list such that its elements are $R = 5r$ far from each other. Thus, the pruning algorithm chooses r dynamically over T, see Section 3.3).

Given T_1, T_2 with $list(T_1), list(T_2)$ and the same radius r, we match $\sigma_1 \in list(T_1)$ to $\sigma_2 \in list(T_2)$ if they are close (with respect to r) on $T_1 \cap T_2$. If however T_1, T_2 have different radii, we don't know how to match these lists correctly. Therefore, our unique games instance is created on a subgraph containing only those vertices T that share the same radius r. We show that there exists such a subgraph which is itself an expander.

1.3   Double samplers and high dimensional expanders

Let us briefly explain how double samplers are constructed from high dimensional expanders (proving Theorem 1.1). A high dimensional expander is a d-dimensional simplicial complex X, which is just a hypergraph with hyperedges of size $\leq d+1$ and a closure property: for every hyperedge in the hypergraph, all of its subsets are also hyperedges in the hypergraph. The hyperedges with $i+1$ elements are denoted $X(i)$, and the complex is said to be an expander if certain spectral conditions are obeyed, see Section 6.

Dinur and Kaufman [DK17] prove that a two-sided spectral high dimensional expander gives rise to a multi-partite graph with interesting spectral expansion properties. Kaufman and Oppenheim [KO20] proved a stronger bound which allows using one-sided spectral expander. The multi-graph has vertices $X(d) \cup X(d-1) \cup \ldots \cup X(0)$, and we place edges for inclusion. Namely, $S \in X(m_1)$ is connected by an edge to $T \in X(m_2)$ if $S \subset T$. It is shown that the graph induced by focusing on layers i and j has $\lambda(G(X(i), X(j))) \leq \frac{i+1}{j+1} + o(1)$. We show, in Section 6, that by narrowing our focus to three layers in this graph (namely, $X(m_2 - 1), X(m_1 - 1)$ and $X(0)$) we get a double sampler. This is proven by observing that the spectral properties are strong enough to yield a sampler (an expander mixing lemma argument suffices since we are only seeking relatively weak sampling properties).

Better double samplers?   Double samplers with super-linear (polynomial and even exponential) size have appeared implicitly (or somewhat similarly as "intersection codes") in the works of impagliazzo, Jaiswal, Kabanets and Wigderson [IKW12, IJKW10]. Two concrete constructions were studied,

- The first where $V_i = \binom{V}{m_i}$, so $|V_i| \approx n^{m_i}$, for $n = |V_0|$.

- The second where V is identified with a vector space over some finite field and then V_i consists of all d_i-dimensional subspaces of V. Here $|V_i| \approx n^{d_i}$.

These constructions could fit our encoding scheme but the polynomial size of the sampler means that the code rate would approach zero. In addition, our algorithm is only efficient when

the sets in V_2, V_1 are very small (of constant or at most logarithmic size), so constructions with larger sets, such as restricting multivariate polynomials to lines also don't fit our algorithm.

The current work is the first to construct double samplers with linear size. This raises the question of finding the best possible parameters for these objects. In particular, for given sampler parameters α and δ, how small can $|V_1|/|V_0|$ be?

Our current construction is based on Ramanujan complexes of Lubotzky, Samuels and Vishne [LSV05] that are optimal with respect to the spectrum of certain Laplacian operators, and not necessarily with respect to obtaining best possible double samplers. It is an interesting challenge to meet and possibly improve upon these parameters through other constructions.

Unlike other pseudorandom objects, there is no known random construction of a double sampler. In particular, we cannot use it as a yardstick for the quality of our parameters. It remains to explore what possible parametric limitations there are for these objects.

We believe that double samplers capture a powerful feature of high dimensional expanders whose potential merit more study. Previously, Dinur and Kaufman [DK17] showed that high dimensional expanders give rise to a very efficient derandomization of the direct product code that is nevertheless still testable. Part of the contribution of the current work is a demonstration of the utility of these objects in a new context, namely of list-decoding.

1.4   Derandomized direct product and approximate-list-decoding

Our list-decoding algorithm can also be viewed in the context of decoding derandomized direct products. The *direct product* encoding takes a string $g \in \{0,1\}^N$ and encodes it into $Enc(g) = (g|_S)_{S \in \mathcal{S}}$ where $\mathcal{S} = \binom{[N]}{k}$ contains all possible k-subsets of $[N]$. An encoding with $|\mathcal{S}| \ll \binom{N}{k}$, as in this paper, is called a *derandomized* direct product encoding.

Direct products and derandomized direct products are important in several contexts, primarily for **hardness amplification**. This type of amplification goes back to Yao's XOR lemma [Yao82, Lev87] (which concerns direct sum, not direct product). In hardness amplification using direct product one begins with a string $g \in \{0,1\}^N$ that is viewed as a truth table of a function $g : \{0,1\}^n \to \{0,1\}$ (here $N = 2^n$), and analyzes the hardness of the new function defined by $Enc(g)$. A typical hardness amplification argument proceeds by showing that if no algorithm (in a certain complexity class) computes g on more than $1 - \varepsilon_0$ of its inputs, then no algorithm computes $Enc(g)$ on more than ε of its inputs. Namely, $Enc(g)$ is much harder than g.

Such a statement is proven, as first described in [Tre05, Imp03], through a (list-) decoding argument: given a hypothetical algorithm that computes $Enc(g)$ successfully on at least an ε-fraction of the inputs, the approximate-list-decoder computes g on $(1 - \varepsilon_0)$ of its inputs.

Our list-decoding result falls short of being useful for hardness amplification, because it is not local, and hardness amplification requires an additional feature called local list-decoding which we discuss in the next subsection.

1.5   Future directions

Both this work and the work of Alex, Jeronimo, Quintana, Shashank and Tulsiani [AJQST20] have a global decoding algorithm, which reads the entire codeword before decoding. An interesting direction is to achieve local list-decoding. In local-list-decoding the algorithm

receives an index i and has query-access to a noisy version of $E_G(x)$. The algorithm queries the codeword in a few locations and should output x_i with high probability (or a list including x_i).

There is a significant technical hurdle that one faces, related to the diameter of the bipartite graph corresponding to (V_1, V_0). In the local-list-decoding constructions analyzed in [IKW12, IJKW10] (both derandomized and non-derandomized) the diameter is $O(1)$, and this is crucially used in the list-decoding algorithm. The reason is that decoding occurs through querying vertices whose distance from a given v is bounded. In the above small-diameter situations nearly all of the vertices in the graph are in a constant distance from v, and they can't all be corrupted by an adversary.

When we move to a linear-size derandomized direct product encoding, as we do in this work, we have a sparse graph and a super-constant diameter. Clearly balls of bounded radii remain very small in this case, and can easily be corrupted. This is what makes the approximate-list-decoding algorithm performed in our work much more challenging (even in the non-local setting), and the algorithm more complicated than the analogous task performed by [IKW12, IJKW10].

In a preliminary version of this manuscript, we asked if there can be an efficient decoding algorithm for the error correcting codes constructed in [TaS17] by the last author. These codes are binary error correcting codes with distance close to half, that achieve nearly optimal rate, and whose encoding algorithm is similar to the one presented here. In a recent work, Jeronimo, Srivastava, and Tulsiani [JST21] resolved this question by presenting a nearly linear decoding algorithm for variants of the error correcting codes in [TaS17].

2 Preliminaries and Notations

For $\sigma, \sigma' \in \Sigma^n$ and $S \subseteq n$ we define

$$\text{dist}_S(\sigma, \sigma') = \Pr_{i \in S}[\sigma_i \neq \sigma'_i].$$

For $L \subseteq \Sigma^n$ we define

$$\text{dist}_S(\sigma, L) = \min_{\sigma' \in L} \text{dist}_S(\sigma, \sigma').$$

When $S = [n]$ we omit the subscript S.

An error correcting code C is a function $C : \Sigma_0^n \to \Sigma_1^m$. It has distance r if for every $x \neq y \in \Sigma_0^n$, $dist(C(x), C(y)) \geq r$. Sometimes we identify the error correcting code C with its image, i.e. $C \subset \Sigma_1^m$. Furthermore, C is (η, ℓ) list-decodable if for every $y \in \Sigma_1^m$,

$$|\{x \in C \mid \text{dist}(x, y) \leq \eta\}| \leq \ell.$$

An algorithm is said to uniquely decode an error correcting code $C \subset \Sigma_1^m$ from an ε-fraction of errors, if for every $x \in \Sigma_1^m$ such that $\text{dist}(x, C) \leq \varepsilon$, it outputs $y \in C$ such that $\text{dist}(x, y) \leq \varepsilon$. An algorithm is said to (η, ℓ) list-decode an error correcting code $C : \Sigma_0^n \to \Sigma_1^m$, if for every $z \in \Sigma_1^m$, it outputs a list $L = \{x \in \Sigma_0^n \mid \text{dist}(C(x), z) \leq \eta\}$, $|L| \leq \ell$.

We use the following version of a Chernoff tail bound [CL06]. Let $Y_1, \ldots Y_n$ be independent random variables, with $\Pr[Y_i = 1] = p_i, \Pr[Y_i = 0] = 1 - p_i$, and let $Y = \sum_{i=1}^{n} w_i Y_i$ for $w_i > 0$. We define $\nu = \sum_{i=1}^{n} w_i^2 p_i$, then

$$\Pr[Y \leq (1 - \delta) \, \mathbb{E}[Y]] \leq e^{-\frac{\delta^2 (\mathbb{E}[y])^2}{2\nu}},$$

$$\Pr[Y \geq \mathbb{E}[Y] + \eta] \leq e^{-\frac{\eta^2}{2(\nu + w\frac{\eta}{3})}},$$

for $w = \max_{i \in [n]} \{w_i\}$.

2.1 Weighted graphs

We say (G, W) is a weighted graph if $G = (V, E)$ is an undirected graph, and $W : E \to \mathbb{R}_{\geq 0}$ is a weight function that associates with each edge e a non-negative weight w_e. We have the convention that non-edges have zero weight. Given the edge weights w_e the weight of a vertex is defined as

$$w_v \quad := \quad \sum_{e : v \in e} w_e.$$

The edge weights induce a distribution on edges (and vertices) which we denote by μ_G where $\mu_G(e) = \frac{w_e}{\sum_{e' \in E} w_{e'}}$ and similarly for vertices. We overload μ_G to denote both the distribution on edges and on vertices. When the graph G is clear, we omit the subscript G from the distribution μ_G. We denote by $v \sim V$ a random vertex in the graph according to the distribution μ_G, and by $e \sim E$ a random edge. For a vertex $v \in V$, we denote by $u \sim v$ a random neighbor of v, according to the edge weights.

Definition 2.1. *We say a distribution Π over V has* irregularity *at most D, for an integer $D \in \mathbb{N}$, if there is some $p \in (0, 1]$ such that for every $v \in V$, $\Pi(v) \in \{p, 2p, \ldots, Dp\}$. The* irregularity *of $v \in V$ is defined to be $\frac{\Pi(v)}{p}$.*

Clearly the uniform distribution has irregularity 1.

2.2 Expanders

Let $(G = (V, E), W)$ be a weighted graph.

- The *edge expansion* of G is

$$h_G \quad := \quad \min_{V' \subset V, \mu(V') \leq \frac{1}{2}} \frac{\mu(E(V', V \setminus V'))}{\mu(V')},$$

  where $E(A, B)$ denotes the set of edges between A and B.

- The normalized adjacency matrix A of G is defined by

$$A_{u,v} \quad = \quad \frac{w_{u,v}}{\sqrt{w_u w_v}}.$$

- $\lambda_2(G)$ denotes the second largest eigenvalue (in absolute value) of A.

In the case of bipartite graph $G = (U, V, E)$, let A^{bip} be the normalized adjacency matrix of G defined by $A_{u,v}^{\text{bip}} = \frac{w_{u,v}}{\sqrt{w_u w_v}}$ for every $u \in U, v \in V$ (A^{bip} is not symmetric). Let $\lambda_2^{\text{bip}}(G)$ be the second largest singular value of A^{bip}.

2.3 Samplers

Definition 2.2. *(Sampler) A weighted bipartite graph $(G = (V_2, V_1, E), W)$ is an (α, β) sampler if for every $f : V_1 \to [0, 1]$,*

$$\Pr_{v_2 \sim V_2}\left[\left|\underset{v_1 \sim v_2}{\mathbb{E}}[f(v_1)] - \underset{v_1 \sim V_1}{\mathbb{E}}[f(v_1)]\right| \geq \alpha\right] \leq \beta.$$

When the distribution is uniform, we say G is an (α, β) sampler.

It is possible to convert a weighted sampler into an unweighted one, if the weights satisfy certain conditions.

Definition 2.3 (Flattening). *Let $(G = (V_2, V_1, E), W)$ be an (α, β) sampler, such that W is uniform on V_1, has irregularity at most D on V_2, and for every $v \in V_2$, all the edges touching v have the same weight. Then the flattening of G is the unweighted bipartite graph $G' = (V_2', V_1, E')$, with vertex set V_2' containing every vertex $v \in V_2$ repeated $M(v)$ times, where $M(v) \in [D]$ is the irregularity of v in G. The edge set E' contains (v', w) if v' is a duplicate of $v \in V_2$ and $(v, w) \in E$.*

Claim 2.4. *Let $(G = (V_2, V_1, E), W)$ be an (α, β) sampler with weights that satisfy the conditions of Definition 2.3, then $G' = (V_2', V_1, E')$ the flattening of (G, W) is also an (α, β) sampler.*

Proof. Fix an arbitrary function $f : V_1 \to [0, 1]$, let $B \subset V_2$ be

$$B = \left\{v \in V_2 \ \middle| \ |\underset{u \sim v}{\mathbb{E}}[f(u)] - \underset{u \sim V_1}{\mathbb{E}}[f(u)]| \geq \alpha\right\}.$$

The graph (G, W) is an (α, β) sampler, so $\Pr_{v \sim V_2}[v \in B] \leq \beta$.

Let $B' \subset V_2'$ be

$$B' = \left\{v \in V_2' \ \middle| \ |\underset{u \sim v}{\mathbb{E}}[f(u)] - \underset{u \sim V_1}{\mathbb{E}}[f(u)]| \geq \alpha\right\}.$$

The distribution W is uniform over V_1, so $\mathbb{E}_{u \sim V_1}[f(u)]$ is the same in G and in G'. For every $v' \in V_2'$ a copy of $v \in V_2$, the distribution over its neighbors is uniform, so $\mathbb{E}_{u \sim v'}[f(u)] = \mathbb{E}_{u \sim v}[f(u)]$. Therefore, the set B' contains exactly all $v' \in V_2'$ which are copies of $v \in B$, and

$$\Pr_{v' \sim V_2'}[v' \in B'] = \Pr_{v \sim V_2}[v \in B] \leq \beta.$$

$\square$

2.4 Every sampler contains an induced expander

The two-step walk of a weighted bipartite graph $(G' = (V_2, V_1, E'), W')$, is the weighted graph

$$(G = (V_2, E), W).$$

For every $T_1, T_2 \in V_2$ and $S \in V_1$ such that S is a common neighbor of T_1, T_2, we connect T_1, T_2 by an edge and label the edge $(T_1, T_2)_S$.

The weight of the edge $(T_1, T_2)_S$ corresponds to the probability of picking $S \sim V_1$ and then independent neighbors T_1, T_2 of S in G' according to the edge weights W'. More explicitly, $W((T_1, T_2)_S) = \mu_{G'}(S) \Pr_{T_1, T_2 \sim S}[T_1 = u_1 \wedge T_2 = u_2]$.

Notice that the graph G contains parallel edges and self loops.

Claim 2.5. *Choosing a random edge $e \in G$ according to the weights W and a random vertex $T \in e$, has the same distribution as picking $T \sim V_2$ according to the vertex weights W'.*

Proof. Fix a vertex $T \in V_2$, the probability to choose T in G equals:

$$\Pr_{(T_1, T_2) \sim E}[T_1 = T] = \sum_{S \in V_1} \Pr_{S_1 \sim V_1}[S_1 = S] \Pr_{T_1 \sim S}[T_1 = T].$$

We omit the second endpoint of the edge because it's independent.

By Bayes' rule,

$$\Pr_{(T_1, T_2) \sim E}[T_1 = T] = \sum_{S \in V_1} \Pr_{S_1 \sim V_1}[S_1 = S] \Pr_{(S_1, T_1) \sim E'}[T_1 = T | S_1 = S]$$

$$= \sum_{S \in V_1} \Pr_{S_1 \sim V_1}[S_1 = S] \frac{\Pr_{T_1 \sim V_2}[T_1 = T] \Pr_{(S_1, T_1) \sim E'}[S_1 = S | T_1 = T]}{\Pr_{S_1 \sim V_1}[S_1 = S]}$$

$$= \Pr_{T_1 \sim V_2}[T_1 = T] \sum_{S \in V_1} \Pr_{(S_1, T_1) \sim E'}[S_1 = S | T_1 = T]$$

$$= \Pr_{T_1 \sim V_2}[T_1 = T].$$

$\square$

In Section 7 we show that the two-step graph of a sampler always contains an expander.

Theorem 2.6 (Every sampler contains an induced expander). *Let*

$$(G' = (V_2, V_1, E'), W')$$

be an (α, β) sampler for some $\alpha, \beta \in (0, 1)$. Let $(G = (V_2, E), W)$ be the two-step walk graph of G'. Fix $\eta \in (0, 1)$ such that $\eta > 10\sqrt{\max\{\alpha, \beta\}}$.

Then for every $A \subseteq V_2$ with $\mu_G(A) = \eta$ there exists a set $B \subseteq A$ such that:

- $\mu_G(B) \geq \frac{\eta}{4}$.

- $\lambda_2(G_B) \leq \frac{99}{100}$, *where G_B is the induced graph of G on B with the same edge weights.*

Furthermore, given A, such a set B can be found in time polynomial in $|V|$.

In order to prove the above theorem, we use a variant of the expander mixing lemma on the two-step walk graph G. Even though G is not an expander, we show that large sets in it expand.

Claim 2.7. *Let $(G' = (V_2, V_1, E'), W')$ be an (α, β) sampler and let $(G = (V_2, E), W)$ be the two-step walk on G'.*

Then for every $A, B \subset V_2$ satisfying $\mu_G(A) > \alpha, \mu_G(B) > \beta$,

$$\Pr_{(T_1, T_2) \sim E}[T_1 \in A, T_2 \in B] \geq (\mu_G(A) - \alpha)(\mu_G(B) - \beta)$$

$$\Pr_{(T_1, T_2) \sim E}[T_1 \in A, T_2 \in B] \leq \mu_G(A)(\mu_G(B) + \beta) + \alpha.$$

12

Proof. Fix sets $A, B \subset V_2$ that satisfy the conditions of the claim. We define the function $f : V_1 \to [0, 1]$ by,

$$\forall v \in V_1, \quad f(v) = \Pr_{T \sim v}[T \in A],$$

In words, $f(v)$ is the probability of a random neighbor (according to the edge weights) of v in G' to be in A. From Claim 2.5, $\mathbb{E}_{v \sim V_1}[f(v)] = \Pr_{T \sim V_2}[T \in A] = \mu_G(A)$.

For every $T \in V_2$, let p_T be the probability of a random neighbour of T in G to be in A. Using f, $p_T = \Pr_{(T_1,T_2) \sim E}[T_2 \in A | T_1 = T] = \mathbb{E}_{v \sim T}[f(v)]$. The last equality is because G is a two-step random walk of G'.

From the sampling properties of G',

$$\Pr_{T \sim V_2}\left[\left|\mathbb{E}_{v \sim T}[f(v)] - \mathbb{E}_{v \sim V_1}[f(v)]\right| > \alpha\right] \le \beta,$$

Substituting $\mathbb{E}_{v \sim T}[f(v)]$ by p_T, we get that $\Pr_{T \sim V_2}[|p_T - \mu_G(A)| > \alpha] \le \beta$.

Let $R \subset V_2$ be the set

$$R = \{T \in V_2 \mid |p_T - \mu_G(A)| > \alpha\}.$$

From above, $\mu_G(R) \le \beta$. For every $T_1 \notin R$, $p_T \in [\mu_G(A) - \alpha, \mu_G(A) + \alpha]$.

Therefore,

$$\Pr_{(T_1,T_2) \sim E}[T_1 \in A, T_2 \in B] \ge \Pr_{T_2 \sim V_2}[T_2 \in B \setminus R] \Pr_{(T_1,T_2) \sim E}[T_1 \in A | T_2 \in B \setminus R]$$
$$\ge (\mu_G(B) - \beta)(\mu_G(A) - \alpha).$$

$$\Pr_{(T_1,T_2) \sim E}[T_1 \in A, T_2 \in B] \le \Pr_{T_2 \sim V_2}[T_2 \in R]$$
$$+ \Pr_{T_2 \sim V_2}[T_2 \in B \setminus R] \Pr_{(T_1,T_2) \sim E}[T_1 \in A | T_2 \in B \setminus R]$$
$$\le \beta + \mu_G(B)(\mu_G(A) + \alpha).$$

$\square$

2.5 Double samplers

A double sampler is a two-layered graph with some additional properties. It is convenient to view it as an inclusion graph, defined as follows

Definition 2.8. *(Inclusion graph) An* inclusion *graph* $X = (V_2, V_1, V_0)$ *with cardinalities* $m_2 > m_1 > 0$ *is a tri-partite graph with vertices* $V = V_2 \cup V_1 \cup V_0$, *where* $V_i \subseteq \binom{V_0}{m_i}$ *for* $i = 1, 2$ *and* $(a, b) \in E$ *iff* $a \subseteq b$.

Given a distribution W_2 on V_2, define a distribution Π on $V_2 \times V_1 \times V_0$ by sampling $v_2 \in V_2$ according to W_2, then choosing a random neighbor $v_1 \in V_1$ of v_2 and then a random neighbor $v_0 \in V_0$ of v_1.

We denote by Π_i the i-th coordinate of Π, and by $\Pi_{i,j}$ the distribution Π restricted to layers i, j. Notice that $\Pi_2 = W$. For every $j \ne i \in \{0, 1, 2\}$ and every $v_i \in V_i$, we denote by

$(\Pi_j | \Pi_i = v_i)$ the distribution of Π_j conditioned on the i-th layer vertex equals v_i, explicitly $\Pr_{u_j \sim (\Pi_j | \Pi_i = v_i)}[v_j = T] = \Pr_{(u_2, u_1, u_0) \sim \Pi}[u_j = T | u_i = v_i]$.

By the way we constructed Π, it satisfies the following property: for every $T \in V_2$, the distribution $(\Pi_1 | \Pi_2 = T)$ is uniform, and for every $S \in V_1$, the distribution $(\Pi_0 | \Pi_1 = S)$ is also uniform. Note that other conditional distributions may not be uniform, for example, $(\Pi_2 | \Pi_1 = S)$ might not be uniform.

Given an inclusion graph $X = (V_2, V_1, V_0)$ and a distribution Π, we denote by $(X(V_{i+1}, V_i), \Pi_{i+1,i})$ the weighted bipartite graph between V_i, V_{i+1}. For every $T \in V_2$, we define the weighted bipartite graph

$$(X_{|T} = (U, T, E) \, , \, W_T)$$

where:

- $U = \{S \in V_1 \mid S \subseteq T\}$, and recall T is a set of elements from V_0,

- $(S, i) \in E$ for $S \in U$ and $i \in T$ iff $i \in S$, and,

- $W_T = (\Pi_{1,0} | \Pi_2 = T)$.

The graph $X_{|T}$ is the subgraph of X that contains all the subsets of T, see Figure 1 for a graphic representation.

With this notation we define double samplers.

Definition 2.9 (Double Sampler). *Let $X = (V_2, V_1, V_0)$ be an inclusion graph with distribution Π on $V_2 \times V_1 \times V_0$ defined from W on V_2 as above. We say (X, W) is a $((\alpha, \beta), (\alpha_0, \beta_0))$ double sampler, if*

1. *$(X(V_2, V_1), \Pi_{2,1})$ is an (α, β) sampler.*

2. *For every $T \in V_2$, $(X_{|T}, W_T)$ is an (α_0, β_0) sampler.*

We say that X has irregularity at most D if

- *Π_2, Π_0 are uniform, and for each $T \in V_2$, the graph $X_{|T}$ is bi-regular and W_T is the uniform distribution.*

- *The distribution on Π_1 has irregularity at most D.*

We say that X is perfectly regular if it has irregularity $D = 1$.

Note that by the definition of inclusion graph over a ground set, the bipartite graphs $X(V_2, V_0)$, $X(V_1, V_0)$ and $X_{|T}$ are always left-regular. A vertex $S \in V_1$ has exactly m_1 neighbors in V_0, which are the m_1 elements the set S contains, and the same for $T \in V_2$.

The definition of a double sampler implies that the graph X has more sampling properties, proven in the claim below.

Claim 2.10. *Let X be a double sampler as in Definition 2.9, then:*

1. *$(X(V_2, V_0), \Pi_{2,0})$ is an (α, β) sampler, and,*

2. *$(X(V_1, V_0), \Pi_{1,0})$ is an $(\alpha + \alpha_0, \beta + \beta_0)$ sampler.*

Proof.

- For Item 1, fix an arbitrary function $f : V_0 \to [0,1]$, let $f' : V_1 \to [0,1]$ be $f'(S) = \mathbb{E}_{x \sim (\Pi_0 | \Pi_1 = S)}[f(x)]$. From the definition of Π, the expectation of f over Π_0 and f' over Π_1 is the same $\mathbb{E}_{S \sim \Pi_1}[f'(S)] = \mathbb{E}_{x \sim \Pi_0}[f(x)]$.

  The graph (V_2, V_1) is a (α, β) sampler,

  $$\Pr_{T \sim \Pi_2} \left[\left| \mathbb{E}_{S \sim (\Pi_1 | \Pi_2 = T)}[f'(S)] - \mathbb{E}_{S \sim \Pi_1}[f'(S)] \right| \geq \alpha \right] \leq \beta. \tag{1}$$

  The joint distribution $\Pi = \Pi_0, \Pi_1, \Pi_2$ is defined so that for every $T \in V_2$, if we first choose $S \sim \Pi_1 | \Pi_2 = T$ and then $x \sim \Pi_0 | \Pi_1 = S$, it is the same as when we choose $x \sim \Pi_0 | \Pi_2 = T$. Therefore, for every $T \in V_2$, $\mathbb{E}_{x \sim (\Pi_0 | \Pi_2 = T)}[f(x)] = \mathbb{E}_{S \sim (\Pi_1 | \Pi_2 = T)}[f'(S)]$.

  Substituting the expectations in (1) we get

  $$\Pr_{T \sim \Pi_2} \left[\left| \mathbb{E}_{x \sim (\Pi_0 | \Pi_2 = T)}[f(x)] - \mathbb{E}_{x \sim \Pi_0}[f(x)] \right| \geq \alpha \right] \leq \beta, \tag{2}$$

  which finished the proof.

- For Item 2, fix an arbitrary $f : V_0 \to [0,1]$, and let $f' : V_1 \to [0,1]$ be as in the first item, $f'(S) = \mathbb{E}_{x \sim \Pi_0 | \Pi_1 = S}[f(x)]$. Again, $\mathbb{E}_{S \sim \Pi_1}[f'(S)] = \mathbb{E}_{x \sim \Pi_0}[f(x)]$ and equations (1),(2) holds here as well.

  For every $T \in V_2$, let μ_T be the expected value of f on T, i.e. $\mu_T = \mathbb{E}_{x \sim (\Pi_0 | \Pi_2 = T)}[f(x)]$. Let A be the set of vertices $T \in V_2$ in which μ_T is close to the expectation of f on V_0,

  $$A = \left\{ T \in V_2 \ \middle| \ \left| \mu_T - \mathbb{E}_{x \sim \Pi_0}[f(x)] \right| \leq \alpha \right\}.$$

  From (2), $\Pr_{T \sim \Pi_2}[T \in A] \geq 1 - \beta$.

  For every T, $\mu_T = \mathbb{E}_{S \sim (\Pi_1 | \Pi_2 = T)}[f'(S)]$, see explanation in the first item. The bipartite graph $X_{|T}$ is an (α_0, β_0) sampler, so

  $$\Pr_{S \sim (\Pi_1 | \Pi_2 = T)} \left[\left| f'(S) - \mu_T \right| \geq \alpha_0 \right] \leq \beta_0.$$

  By the triangle inequality,

  $$\left| f'(S) - \mathbb{E}_{x \sim \Pi_0}[f(x)] \right| \leq \left| f'(S) - \mu_T \right| + \left| \mu_T - \mathbb{E}_{x \sim \Pi_0}[f(x)] \right|.$$

  For $T \in A$, $|\mu_T - \mathbb{E}_{x \sim \Pi_0}[f(x)]| \leq \alpha$. Therefore for $T \in A$

  $$\Pr_{S \sim (\Pi_1 | \Pi_2 = T)} \left[\left| f'(S) - \mathbb{E}_{x \sim \Pi_0}[f(x)] \right| \geq \alpha + \alpha_0 \right] \leq \beta_0. \tag{3}$$

We finish the proof by using the fact that the probability of $T \sim \Pi_2$ to be in A is at least $1 - \beta$. Let I be the event in which $|f'(S) - \mathbb{E}_{v \sim \Pi_0}[f(v)]| \geq \alpha + \alpha_0$. Using this notation,

$$\Pr_{S \sim \Pi_1}[I] \leq \Pr_{T \sim \Pi_2}[T \notin A] + \Pr_{T \sim \Pi_2}[I|T \in A]$$
$$\leq \beta + \beta_0. \qquad\qquad \text{(by (2) and (3))}$$

$\square$

In Section 6 we prove that bounded-degree double samplers can be constructed from high dimensional expanders. While the distributions in the explicit construction are not uniform, they are not too far off. Π_1 has bounded irregularity and Π_2, Π_0 are uniform. The irregularity stems from the irregularity of the high dimensional expanders. Very recent work [FI20] has managed to construct *regular* high dimensional expanders, which leads to perfectly regular double samplers. These were not available when an earlier version of this manuscript came out.

Theorem 2.11. *For every* $\alpha, \beta, \alpha_0, \beta_0 > 0$ *there exist constants* $m_1, m_2, D \in \mathbb{N}$, $m_1, m_2 = \text{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}), D = \exp(\text{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}))$, *such that there is a family of explicitly constructible double samplers* (X_n, W_n) *for infinitely many* $n \in \mathbb{N}$ *satisfying*

- $X_n = (V_2, V_1, V_0)$ *is an inclusion graph, where* $|V_0| = n$, $V_i \subseteq \binom{V_0}{m_i}$ *for* $i = 1, 2$.

- X_n *is an* $((\alpha, \beta), (\alpha_0, \beta_0))$ *double sampler.*

- $|V_1|, |V_2| \leq D \cdot n$.

- *The distributions* Π_0, Π_2 *are uniform and the distribution* Π_1 *has irregularity at most* D.

- *For each* $m \in \mathbb{N}$ *there is some* $n \in [m, Dm]$ *such that the complex* X_n *on* n *vertices is constructible in time* $\text{poly}(n)$.

2.6 UG constraint graphs

Definition 2.12. *(UG constraint graph) Let* $(G = (V, E), W)$ *be a weighted graph. We say* $((G, W), \{\pi_e\}_{e \in E})$ *is a UG constraint graph with* ℓ *labels if* $\pi_e : [\ell] \to [\ell]$ *is a permutation. An assignment for* G *is a function* $a : V \to [\ell]$. *We say the assignment* a *satisfies an edge* $e = (u, v)$ *if* $\pi_e(a(u)) = a(v)$. *The* value *of an assignment is the fraction of satisfied edges. We say* $((G, W), \{\pi_e\}_{e \in E})$ *is* p-satisfiable *if there exists an assignment with value at least* p.

The graph G is undirected, and since the constraints are unique it doesn't matter if π_e is represented from u to v or vice versa. Arora, Khot, Kolla, Steurer, Tulsiani and Vishnoi [AKKSTV08] showed how to solve unique games instances on expander graphs in polynomial time. This result was improved by Makarychev and Makarychev [MM10], who proved,

Theorem 2.13 ([MM10, Theorem 10]). *Let* G *be a regular graph with edge expansion* h_G. *There exist positive absolute constants* c *and* C *and a polynomial time approximation algorithm that given a* $1 - \beta$ *satisfiable instance of UG on* G *with* $\frac{\beta}{1 - \lambda_2(G)} \leq c$, *the algorithm finds a solution of value* $1 - C\frac{\beta}{h_G}$.

We need a version of this theorem with two modifications:

- The theorem, as stated, refers to unweighted regular graphs. We need the same results for non-regular weighted graphs.

- The theorem finds one assignment with high value. However, we need to get an approximation to *all* assignments with high value.

In Appendix A we go over the algorithm in [AKKSTV08, MM10] and show that the same result holds for weighted non-regular graphs. In Section 8 we show how to output a list that contains an approximation to all assignments with high value. To do that we run the algorithm several times, each time peeling off the solution that is found. We prove:

Theorem 2.14. *Let $(G = (V, E), W)$ be a weighted undirected graph with $\lambda_2(G) \leq \frac{99}{100}$. Let $\{\pi_e\}_{e \in E}$ be unique constraints over the edges of G, with ℓ labels.*

Then there is an absolute constant $c > 1$ and a polynomial time algorithm that outputs a list of assignments $L = \{a^{(1)}, \ldots, a^{(t)}\}$ with $a^{(i)} : V \to [\ell]$. The list satisfies that for every assignment $a : V \to [\ell]$ that satisfies $1 - \eta$ of the constraints for $\eta < c^{-\ell-1}$, there exists $a^{(i)} \in L$ that satisfies $\Pr_{v \sim W}[a(v) = a^{(i)}(v)] \geq 1 - \eta c^\ell$.

2.7 The Johnson bound

Johnson's bound shows that any code with a good distance is also a good list-decodable code. The bound has several versions, and the quantitative bounds are different over large and small alphabets. The bounds are usually stated for codes, but, essentially address the following geometric problem: How many vectors can be all close to the same vector, and far away from each other. As such, the bounds are also useful for approximate error correcting codes, see Section 3. In this section we repeat the Johnson bound (with this geometric interpretation in mind) and give the proofs for completeness. We follow the exposition in [Sud01, GRS].

2.7.1 Large alphabet

Theorem 2.15. *Let Σ be a finite set of cardinality q, let $z \in \Sigma^n$ and $L \subseteq \Sigma^n$. If:*

- *For every $x \in L$, $\text{dist}(x, z) \leq 1 - \gamma$, and,*

- *For every distinct $x, x' \in L$, $\text{dist}(x, x') \geq 1 - \beta$,*

where $\gamma^2 > \beta$, then, $|L| \leq \frac{\gamma - \beta}{\gamma^2 - \beta}$.

Proof. Let $z \in \Sigma^n$ and let $L = x_1, \ldots, x_\ell$ be a list satisfying the theorem requirements, $|L| = \ell$. We prove the Johnson bound by double counting. The expression we bound is the probability that two different list elements agree on a random coordinate,

$$p = \Pr_{x \neq x' \in L, i \in [n]}[x_i = x'_i].$$

By the list requirement, $x, x' \in L$ satisfies $\text{dist}(x, x') \geq 1 - \beta$, therefore $p \leq \beta$. Also,

$$p = \Pr_{x \neq x' \in L, i \in [n]}[x_i = x'_i] = \mathbb{E}_{i \in [n]}[\Pr_{x \neq x' \in L}[x_i = x'_i]]$$
$$\geq \mathbb{E}_{i \in [n]}[\Pr_{x \neq x' \in L}[x_i = x'_i = z_i]]$$

For every coordinate $i \in [n]$, let ℓ_i be, $\ell_i = |\{x \in L \mid x_i = z_i\}|$. Using this notation,

$$\Pr_{x \neq x' \in L}[x_i = x'_i = z_i] = \frac{\ell_i}{\ell} \cdot \frac{\ell_i - 1}{\ell - 1}.$$

The list requirements imply that $\mathbb{E}_{i \in [n]}[\ell_i] \geq \gamma \ell$, which gives the bound on p:

$$p \geq \mathbb{E}_{i \in [n]}\left[\frac{\ell_i(\ell_i - 1)}{\ell(\ell - 1)}\right] \geq \frac{1}{\ell(\ell - 1)}\left(\mathbb{E}_{i \in [n]}[\ell_i^2] - \mathbb{E}_{i \in [n]}[\ell_i]\right) \geq \frac{\gamma^2 \ell - \gamma}{\ell - 1}.$$

Hence $\beta(\ell - 1) \geq \gamma^2 \ell - \gamma$ and $\ell \leq \frac{\gamma - \beta}{\gamma^2 - \beta}$. $\qquad\square$

2.7.2 Binary alphabet

For Binary alphabet and distance $\gamma = \frac{1 - \alpha}{2}$ where $\alpha \approx 0$, the condition $\alpha^2 > \beta$ is restrictive, as it forces β to be about $\frac{1}{4}$ or below, which corresponds to the distance $1 - \beta$ being $\frac{3}{4}$ which is vacuous for Binary codes. We now state a version useful for Binary alphabets.

Lemma 2.16. *(See [Sud01, Lem 7.1]) Fix $\Delta, \Gamma > 0$. Suppose $x_1, \ldots, x_m \in \mathbb{R}^n$ are such that*

- $\|x_i\|^2 \leq \Gamma$, *and,*

- $\langle x_i, x_j \rangle \leq -\Delta$ *for every $i \neq j$.*

Then $m \leq 1 + \frac{\Gamma}{\Delta}$.

Proof. Set $z = \sum x_i$. Then $\langle z, z \rangle \geq 0$ and

$$\langle z, z \rangle = \sum_i \langle x_i, x_i \rangle + \sum_{i \neq j} \langle x_i, x_j \rangle \leq m\Gamma + m(m - 1)(-\Delta).$$

Together this implies $\Gamma - (m - 1)\Delta \geq 0$ and the bound. $\qquad\square$

Theorem 2.17. *(Following [Sud01]) Let $\Sigma = \{0, 1\}$, $z \in \Sigma^n$, $L \subseteq \Sigma^n$. If:*

- *For every $w \in L$, $dist(w, z) \leq \frac{1 - \alpha}{2}$, and,*

- *For every different $x, x' \in L$, $dist(x, x') \geq \frac{1 - \beta}{2}$,*

where $\alpha^2 > \beta$. Then, $|L| \leq 1 + \frac{4}{\alpha^2 - \beta}$.

Proof. We first define an embedding $E : \{0, 1\}^n \to \mathbb{R}^n$ by letting $E(b_1, \ldots, b_n) = \frac{1}{\sqrt{n}}((-1)^{b_1}, \ldots, (-1)^{b_n})$. Notice that if $\sigma_1, \sigma_2 \in \{0, 1\}^n$ then $\langle E(\sigma_1), E(\sigma_2) \rangle = 1 - 2dist(\sigma_1, \sigma_2)$ and in particular $\|E(\sigma)\| = 1$. Suppose $L = \{\sigma_1, \ldots, \sigma_m\}$. Denote $x_i = E(\sigma_i)$, $y = E(z)$ and $z_i = x_i - \alpha y$. Then:

$$\begin{aligned}
\langle z_i, z_i \rangle &= \langle x_i - \alpha y, x_i - \alpha y \rangle = 1 - 2\alpha \langle x_i, y \rangle + \alpha^2 \leq 1 + 2\alpha + \alpha^2 = (1+\alpha)^2 \leq 4, \text{ and,} \\
\langle z_i, z_j \rangle &= \langle x_i - \alpha y, x_j - \alpha y \rangle = \langle x_i, x_j \rangle - \alpha[\langle x_i, y \rangle + \langle x_j, y \rangle] + \alpha^2 \\
&= 1 - 2\text{dist}(x_i, x_j) - \alpha(1 - 2\text{dist}(x_i, y) + 1 - 2\text{dist}(x_j, y)] + \alpha^2 \\
&\leq \beta - 2\alpha^2 + \alpha^2 = \beta - \alpha^2.
\end{aligned}$$

It follows by Lemma 2.16 that $m \leq 1 + \frac{4}{\alpha^2 - \beta}$. □

3 Approximate Error Correcting Codes

In this section we revisit the definition of approximate error correcting codes (ECCs), and the ABNNR encoding [ABNNR92] in both direct product and direct sum form. We also relate approximate ECC to distance amplification.

Definition 3.1. *(Approximate ECC) [IJKW10, Def 1.5] $E : \Sigma_0^n \to \Sigma_1^m$ is an (r, η, ℓ) approximate-list-decodable error correcting code if for every $z \in \Sigma_1^m$ there is a list $L \subseteq \Sigma_0^n$ of cardinality at most ℓ such that for every codeword $E(x)$ with $\text{dist}(E(x), z) \leq \eta$ we have $\text{dist}(x, L) \leq r$.*

There are a few important differences between an approximate-list-decodable ECC and a list-decodable ECC. A code $C : \Sigma_0^n \to \Sigma_1^m$ is (η, ℓ) *list-decodable*, if for every $z \in \Sigma_1^m$, $|L| = |\{x \in \Sigma_0^m \mid \text{dist}(C(x), z) \leq \eta\}| \leq \ell$. In an approximate-list-decodable ECC, we relax the condition and only require the list L to contain an *approximation* for x such that $\text{dist}(E(x), z) \leq \eta$. This requirement is much weaker, and an approximate-list-decodable ECC might not have a noticeable minimal distance.

Another difference between the codes is the uniqueness of the list. In list-decoding, for every code C and input z there is a *unique* list L which satisfies the requirements, $L = \{x \in \Sigma_0^m \mid \text{dist}(C(x), z) \leq \eta\}$. In addition, given an efficient encoding algorithm for C it is easy to check if x is in L, by checking if $\text{dist}(C(x), z) \leq \eta$. In approximate-list-decoding, for every encoding E and input z there can be many possible lists which satisfy the requirements, as each x satisfying $\text{dist}(E(x), z) \leq \eta$ has many strings that r-approximate it. Furthermore, given a string $x' \in \Sigma_0^n$, it is not clear how to check efficiently if x' is an r-approximation to some x such that $\text{dist}(E(x), z) \leq \eta$, as going over all strings which are r-close to x' takes possibly $\exp(n)$ time.

From the above paragraph we understand why the list size in Theorem 4.1, which is an approximate-list-decoding algorithm, is larger than the list size in Corollary 1.3. In the corollary there is a list-decoding algorithm, where it is possible to check and prune the list, see a proof of the corollary in Section 4.4.

3.1 The ABNNR direct-product construction

We restate the ABNNR encoding [ABNNR92]. In the original paper, it is defined as an amplification step applied on an underlying code. Here we state it as an approximate error correcting code.

Definition 3.2. *(ABNNR encoding[ABNNR92]) Let $G = (V_1, V_0, E)$ be a d left-regular bipartite graph, and let Σ_0 be a constant size alphabet. The ABBNR encoding*

$$E_{G,\Sigma_0} : \Sigma_0^{|V_0|} \to \Sigma_1^{|V_1|}$$

for $\Sigma_1 = \Sigma_0^d$ *is defined as follows. For every* $v \in V_1$ *let* $S_v \subset V_0$ *be the neighbours of* v. *For every* $x \in \Sigma_0^{|V_0|}$ *the encoding* $E_G(x)$ *is defined by*

$$\forall v \in V_1, \quad E_G(x)_v = x_{|S_v}.$$

In order to simplify the notations, when the alphabet is clear we omit Σ_0 and use E_G. To clarify, d left-regular means that each $v \in V_1$ has exactly d neighbors.

The key property of this construction is distance amplification.

Claim 3.3. *For every* $x, x' \in \Sigma^{|V_0|}$ *such that* $dist(x, x') > \alpha$,

$$dist(E_G(x), E_G(x')) \geq 1 - \beta.$$

Proof. Let $f : V_0 \to \{0, 1\}$ equal 1 for all $u \in V_1$ such that $x_{|u} \neq x'_{|u}$, and 0 else. As $dist(x, x') > \alpha$ we have $\mathbb{E}_{u \in V_0}[f(u)] > \alpha$. By the sampler property of G, all except a β fraction of the vertices in V_1 satisfy

$$\mathbb{E}_{u \sim v}[f(u)] \geq \mathbb{E}_{u \in V_0}[f(u)] - \alpha > 0.$$

Thus, for all these vertices v, $E_G(x)_v \neq E_G(x')_v$ which implies $dist(E_G(x), E_G(x')) \geq 1 - \beta$. $\square$

Using this, we show that E_G is an approximate-list-decodable code.

Lemma 3.4. *Let* $G = (V_1, V_0, E)$ *be a* d *left-regular* (α, β) *sampler. Let* $\gamma > \sqrt{\beta}$. *Then* E_G *is* $(\alpha, 1 - \gamma, \ell = \frac{\gamma - \beta}{\gamma^2 - \beta})$ *approximate-list-decodable ECC.*

Proof. Fix an arbitrary $z \in \Sigma_1^{|V_1|}$, and define

$$L_0 = \{x \in \Sigma_0^{|V_0|} \mid dist(E_G(x), z) \leq 1 - \gamma\}.$$

The list L_0 contains all x such that $E_G(x)$ has γ agreement with z, but it is possibly very large. We reduce the size of L_0 by removing elements which are too close to each other. Let $L_\alpha \subset L_0$ be a maximal subset of L_0, such that different $x, x' \in L_\alpha$ are at distance more than α. By definition, for each $x \in L_0$ there exists $x' \in L_\alpha$ such that $dist(x, x') \leq \alpha$, or else it is possible to add x into L_α which contradicts L_α being maximal. It remains to bound $\ell = |L_\alpha|$. We bound the cardinality of L_α by noting that:

- For different $x, x' \in L_\alpha$, $dist(E_G(x), E_G(x')) \geq 1 - \beta$ (by Claim 3.3), and,

- For every $x \in L_\alpha$, $dist(E_G(x), z) \leq 1 - \gamma$ (by definition of L_0).

Hence we get ℓ vectors that are all close to one vector z, but are pairwise far apart. By a variant of the Johnson bound, Theorem 2.15, we have $\ell \leq \frac{\gamma - \beta}{\gamma^2 - \beta}$. We remark that for this proof it is enough to take a disperser instead of a sampler. $\square$

3.2 A direct sum construction over the binary alphabet

In this section we take a small detour to analyse the direct sum encoding, which is very related to the direct product discussed in the rest of this paper. We show that it is list-decodable.

We focus on binary alphabet, i.e. $\sigma \in \{0,1\}^n$, we define $\mathrm{bias}(\sigma) = |\sum_{i=1}^n (-1)^{\sigma_i}|$. In this case $\mathrm{dist}(\sigma, \sigma') = \delta$ implies $\mathrm{bias}(\sigma + \sigma') = |1 - 2\delta|$.

Definition 3.5. *(A direct sum encoding) Let $G = (V_1, V_0, E)$ be a d left-regular bipartite graph. The direct sum encoding*

$$E^\oplus : \{0,1\}^{|V_0|} \to \{0,1\}^{|V_1|}$$

is defined as follows: For every $v \in V_1$ let $S_v \subset V_0$ be the neighbours of v. For every $x \in \{0,1\}^{|V_0|}$, the encoding $E^\oplus(x)$ is defined by

$$\forall v \in V_1, \quad E^\oplus(x)_v = \sum_{u \in S_v} x_u \bmod 2.$$

Definition 3.6. *(Parity sampler) Let $G = (V_1, V_0, E)$ be a bipartite graph. Given $f : V_0 \to \{0,1\}$ we define a function $G(f) : V_1 \to \{0,1\}$ by*

$$\forall v_1 \in V_1, \quad G(f)(v_1) = \sum_{v \in S_{v_1}} f(v) \bmod 2.$$

We say G is an (α, β) parity sampler if for every function $f : V_0 \to \{0,1\}$ with $\mathrm{bias}(f) \le \alpha$, it holds that $\mathrm{bias}(G(f)) \le \beta$.

Lemma 3.7. *Let $G = (V_1, V_0, E)$ be a d left-regular (α, β) parity sampler. Fix $\gamma > \sqrt{\beta}$. Then the direct sum encoding on G is an $(\alpha, \frac{1-\gamma}{2}, \ell = 1 + \frac{4}{\gamma^2 - \beta})$ approximate-list-decodable ECC.*

Proof. Let $E^\oplus$ denote the direct sum encoding on G. Fix any $z \in \{0,1\}^{|V_1|}$, and define

$$L_0 = \{x \in \{0,1\}^{|V_0|} \mid \mathrm{dist}(E^\oplus(x), z) \le \frac{1-\gamma}{2}\}.$$

The list L_0 contains all x such that $E^\oplus(x)$ has $\frac{1+\gamma}{2}$ agreement with z, but it might be too large a set. We reduce the size of L_0 by removing elements which are too close to each other. Let $L_\alpha \subset L_0$ be a maximal subset of L_0, such that different $x, x' \in L_\alpha$ have $\mathrm{bias}(x + x') \le \alpha$. By definition, for each $x \in L_0$ there exists $x' \in L_\alpha$ such that $\mathrm{bias}(x, x') > \alpha$, or else it is possible to add x into L_α which contradicts L_α being maximal. It remains to bound $\ell = |L_\alpha|$. We bound the cardinality of L_α by noting that:

- For different $x, x' \in L_\alpha$, we have $\mathrm{bias}(x + x') \le \alpha$, hence by the parity sampler property $\mathrm{bias}(E^\oplus(x + x')) = \mathrm{bias}(E^\oplus(x) + E^\oplus(x')) \le \beta$. In particular $\mathrm{dist}(E^\oplus(x), E^\oplus(x')) \ge \frac{1-\beta}{2}$. Also,

- By definition of L_0, for every $x \in L_\alpha$, $\mathrm{dist}(E^\oplus(x), z) \le \frac{1-\gamma}{2}$.

Hence if $\ell = |L_\alpha|$, we get ℓ vectors that are all close to one vector z, but are all far apart from each other. By the Johnson bound, Theorem 2.17, we have $\ell \le 1 + \frac{4}{\gamma^2 \beta}$. $\qquad \square$

21

3.3 Well-separated approximate ECC

In some cases when list-decoding, we also want the elements in the output list L to have at least some distance between them, we define such a list as r-separated.

Definition 3.8. *Let $L \subseteq \Sigma^n$ and $r \in [0,1]$. We say L is r-separated if for every $\sigma_1, \sigma_2 \in L$, $dist(\sigma_1, \sigma_2) \geq r$.*

Let $G = (V_1, V_0, E)$ be a d left-regular (α, β)-sampler, and let $E_G : \Sigma_0^{|V_0|} \to \Sigma_1^{|V_1|}$ be as in Definition 3.2. Fix $\gamma > \sqrt{\beta}$, we show an (inefficient) algorithm which $(r, 1 - \gamma, \ell)$ approximate-list-decodes E_G and outputs a $5r$-separated list, by dynamically adjusting r for every input.

Algorithm 1 (Well-Separated List Decoding Algorithm)**.** The algorithm has a decoding parameter $1 - \gamma$. The input is a word $z \in \Sigma_1^{|V_1|}$, the output is a list $L \subset \Sigma_0^{|V_0|}$ and a radius r.

- Set $i = 0, \tau_0 = \alpha, L_0 = \left\{ x \in \Sigma_0^{|V_0|} \;\middle|\; \text{dist}(E_G(x), z) \leq 1 - \gamma \right\}$.

- At stage i, if L_i is $10\tau_i$-separated, output $L_i, r = 2\tau_i$.

  Otherwise, say $v_1, v_2 \in L$ are independent if $\text{dist}(v_1, v_2) \geq 10\tau_i$. Set L_{i+1} to be a maximal independent set in L_i. Set $\tau_{i+1} = 10\tau_i$, $i = i + 1$ and repeat the loop.

Lemma 3.9. *For every $z \in \Sigma_1^{|V_1|}$ the algorithm above outputs a list L of size at most $\ell = \lfloor \frac{\gamma - \beta}{\gamma^2 - \beta} \rfloor$ and a radius $r = 2\alpha 10^i$ for some $i \in \{0, \dots \ell\}$ such that*

- *For every $x \in \Sigma_0^{|V_0|}$ such that $dist(E_G(x), z) \leq 1 - \gamma$, $dist(x, L) \leq r$.*

- *L is $5r$-separated.*

Proof. We start by bounding the size of the lists L_i produced by the algorithm.

Claim 3.10. *For every $\tau > \alpha$, a list L_i which is τ-separated satisfies $|L_i| \leq \ell$.*

Proof. Every $x, x' \in L_i$ satisfy $\text{dist}(x, x') \geq \tau$. Using the sampler properties of G, Claim 3.3, $\text{dist}(E_G(x), E_G(x')) \geq 1 - \beta$. From the definition of L_0, every $x \in L_i \subset L_0$ also satisfies $\text{dist}(E_G(x), z) \leq 1 - \gamma$. By the Johnson bound, Theorem 2.15, $|L_i| \leq \ell$. $\qquad\square$

The algorithm outputs a list L which is at least 10α-separated, so the claim above proves $|L| \leq \ell$.

We now bound the number of steps the algorithm performs. Denote by t the index i with which we quit (if we quit), we prove that $t \leq \ell$. By the algorithm definition, for every i, $L_{i+1} \subsetneq L_i$, as in the case where $L_{i+1} = L_i$ the algorithm stops. The list L_1 is 10α-separated, so from the claim above $|L_1| \leq \ell$, together we get that $t \leq \ell$.

Next we prove the covering property, let x be such that $\text{dist}(E_G(x), z) \leq 1 - \gamma$, we show that $\text{dist}(x, L) \leq r$. By the definition of L_0, $x \in L_0$. For every $i < t$ and every $x' \in L_i$, $\text{dist}(x', L_{i+1}) \leq 10\tau_i = \tau_{i+1}$, for otherwise x' can be added to the independent set L_{i+1} contradicting its maximality. Hence, for $x \in L_0$:

$$\text{dist}(x, L_t) \;\leq\; \sum_{i=0}^{t} \tau_i = \alpha \sum_{i=0}^{t} 10^i \;=\; \frac{10^{t+1} - 1}{10 - 1}\alpha \;\leq\; \frac{10}{9}\tau_t < r.$$

Where the last inequality holds since $r = 2\tau_t$. The algorithm always outputs a list that is $10\tau_t = 5r$-separated, which finishes the proof. $\qquad\square$

22

The algorithm runs in time $\exp(|V_0| \log |\Sigma_0|) \operatorname{poly}(\ell)$, because it goes over all $x \in \Sigma_0^{|V_0|}$ to create the initial list L_0, and the loop runs at most ℓ times.

3.4 Approximate-list-decoding implies distance amplification

In this section we show that composing an approximate-list-decodable code with a an error correcting code results in a list-decodable error correcting code.

Claim 3.11. *Let $E : \Sigma_0^n \to \Sigma_1^m$ be an (r, η, ℓ) approximate-list-decodable code, and let $C \subset \Sigma_0^n$ be an error correcting code with distance $2r$. Then the code:*

$$E(C) = \{E(x) \mid x \in C\},$$

is (η, ℓ) list-decodable.

Proof. Let $z \in \Sigma_1^m$ and let $L = List_E(z)$ be the cardinality ℓ list guaranteed by the approximate-list-decoding property of E. Let L' be the list

$$L' = \{x \in C \mid \exists y \in L, \operatorname{dist}(x, y) \leq r\}.$$

We claim that L' is small and contains all codewords close to z.

Suppose $x \in C$ is such that $\operatorname{dist}(E(x), z) \leq \eta$. Then by the definition of approximate-list-decoding, $\exists y \in L$ such that $\operatorname{dist}(x, y) \leq r$, so by definition $x \in L'$.

For every $y \in L$, there exist at most a single $x \in C$ which is r-close to y, because the distance of C is $2r$. Therefore, $|L'| \leq \ell$. $\qquad\square$

It is easy to see that if E has an approximate-list-decoding algorithm and C has a unique-decoding algorithm, then $E(C)$ has a list-decoding algorithm. To decode $E(C)$ we simply run the decoding algorithm of E getting L, then run the decoding algorithm of C on all elements in L and output the result.

4 The Code and its Approximate List-Decoding Algorithm

In this section we describe our code and an approximate-list-decoding algorithm for it. The code is the ABNNR encoding over the first two layers of a double sampler, defined as follows.

Let $(X = (V_2, V_1, V_0 = [n]), \Pi)$ be a double sampler. It is recommended to first focus on the case where the X is perfectly regular, namely every pair of layers give rise to a bi-regular graph. In the slightly more general case, X has irregularity at most D, i.e. Π_0, Π_2 are uniform and Π_1 has irregularity at most D. Furthermore, we assume that for every $T \in V_2$, the bipartite subgraph $X_{|T}$ containing all subsets of T in X is a bi-regular unweighted graph.

We define the encoding E_X as follows. Consider the graph obtained by restricting X to layers V_0, V_1, and let G be its flattening on vertex sets V_0 and V_1', see Definition 2.3 (in the perfectly regular case, there's no need for flattening). Next, we define $E_X = E_G$ to be the ABNNR encoding as given in Definition 3.2. Namely, as G is a bipartite graph on vertex sets $[n]$ and V_1', we have $E_X : \Sigma_0^n \to \Sigma_1^{|V_1'|}$. The encoding of a string $z \in \Sigma_0^n$ is given by

$$\forall S \in V_1' \quad (E_X(z))_S := z_{|S}.$$

The code is defined over alphabet $\Sigma_1 = \Sigma_0^{m_1}$, where m_1 is the size of subsets in V_1'. The blocklength of the code is $|V_1'|$, which is bounded by $D|V_1|$, where D is the irregularity of X.

Our main theorem below is an approximate-list-decoding algorithm for E. Recall that an algorithm $(\varepsilon, 1 - \gamma, \frac{70}{\gamma^2})$ approximates-list-decodes E if for any input $z \in \Sigma_1^{|V_1'|}$ it outputs a list $L_{out}, |L_{out}| \leq \frac{70}{\gamma^2}$ such that for every x that satisfies $\mathrm{dist}(E(x), z) \leq 1 - \gamma$, it holds that $\mathrm{dist}(x, L_{out}) \leq \varepsilon$.

Theorem 4.1 (Main Theorem (formal version of Theorem 1.2))**.** *There exists $c > 0$ such that the following holds. For every $\gamma, \varepsilon > 0$, let D be some constant and let $\alpha, \beta, \alpha_0, \beta_0 > 0$ satisfy*

$$\alpha_0 \leq \frac{\varepsilon}{4} 10^{-\frac{\gamma}{8}}, \qquad\qquad \beta_0 \leq \frac{\varepsilon\gamma}{1000(c^{\frac{8}{\gamma}} + 1)},$$

$$\alpha \leq \frac{\varepsilon\sqrt{\gamma}}{1000}, \qquad\qquad \beta \leq \frac{\varepsilon}{1000(c^{\frac{8}{\gamma}} + 1)}.$$

There exists $c' > 0$ such that if $(X = (V_2, V_1, V_0), W)$ is an $((\alpha, \beta), (\alpha_0, \beta_0))$ double sampler, with irregularity at most D, the encoding $E = E_X$ has a randomized approximate-list-decoding algorithm that runs in time polynomial in n with success probability $1 - e^{-c'n}$ and parameters $(\varepsilon, 1 - \gamma, \frac{70}{\gamma^2})$.

Here $n = |V_0|$ and we assume that $|V_2|, |V_1| = \Theta(n)$ and the subsets in V_1, V_2 have bounded size.

Theorem 2.11 proves the existence of double samplers with the required parameters. Combining it with the above theorem results in the following corollary.

Corollary 4.2. *For every $\gamma, \varepsilon > 0$ there exist an integer D, a constant $c' > 0$ and an infinite family of bounded-degree bipartite graphs $G_n = (A_n, B_n, E_n)$ such that $E_{G_n} : \Sigma_0^{A_n} \to \Sigma_1^{B_n}$ has rate $\exp(-\mathrm{poly}(\frac{1}{\varepsilon}\exp(-\frac{1}{\gamma})))$ and a randomized polynomial time approximate-list-decoding algorithm with parameters $(\varepsilon, 1 - \gamma, \frac{70}{\gamma^2})$ and success probability $1 - e^{-c'|G_n|}$. Moreover, for every large enough m, there is some G_n with $m \leq |A_n| \leq Dm$.*

Proof. Fix $\gamma, \varepsilon > 0$, and let $\alpha_0 = \frac{\varepsilon}{4} 10^{-\frac{\gamma}{8}}, \beta_0 = \frac{\varepsilon\gamma}{1000(c^{\frac{8}{\gamma}}+1)}, \alpha = \frac{\varepsilon\sqrt{\gamma}}{1000}, \beta = \frac{\varepsilon}{1000(c^{\frac{8}{\gamma}}+1)}$.

Let X_n be an infinite family of $((\alpha, \beta), (\alpha_0, \beta_0))$ double samplers, promised from Theorem 2.11. The family is dense, such that for every integer $m \in \mathbb{N}$ there exists $X_n = (V_1, V_1, V_0)$ such that $m \leq |V_0| \leq Dm$. Furthermore, for every n the double sampler $X_n = (V_2, V_1, V_0)$ has irregularity at most $D = \exp(\mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}))$ and $|V_2|, |V_1| \leq D|V_0|$. The vertices in V_1 are m_1-sets, for $m_1 = \mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0})$ and the vertices in V_2 are m_2-sets for $m_2 = \mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0})$.

For every $X_n = (V_2, V_1, V_0)$, let $G_n = (A_n, B_n, E_n)$ be the flattening of the bipartite graph $X_n(V_1, V_0)$. From Claim 2.4, G_n is an unweighted bipartite graph with $|A_n| = |V_0|$ and $|B_n| \leq D|V_1| \leq D^2|V_0|$. In addition, the degree of each vertex $b \in B_n$ is m_1.

Let c' be the constant from Theorem 4.1. From the theorem, the ABNNR encoding E_{G_n} has a polynomial time approximate-list-decoding algorithm with parameters $(\varepsilon, 1 - \gamma, \frac{70}{\gamma^2})$ and success probability $1 - e^{-c'|A_n|}$. The rate of the code is $\frac{|A_n|}{|B_n|}\frac{1}{m_1} \geq \frac{1}{D^2 m_1} = \exp(-\mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0})) = \exp(-\mathrm{poly}(\frac{1}{\varepsilon}\exp(-\frac{1}{\gamma})))$. $\qquad\square$

In this section X is always an $((\alpha, \beta), (\alpha_0, \beta_0))$ double sampler and E is the encoding defined above. We show that E is combinatorially approximate-list-decodable in Section 4.1, in Section 4.2 we present the list-decoding algorithm and in Section 4.3 we discuss its parameters . We prove Corollary 1.3 in Section 4.4.

4.1 Combinatorial approximate-list-decoding

We briefly show that the encoding E is combinatorially approximate-list-decodable. Let G be the flattening of $X(V_1, V_0)$, as in the previous section. From Claim 2.10, Item 2, $X(V_0, V_1)$ is an $(\alpha + \alpha_0, \beta + \beta_0)$ sampler. It follows from Claim 2.4 that G is also an $(\alpha + \alpha_0, \beta + \beta_0)$ sampler.

By Lemma 3.4 our encoding E is $(\alpha + \alpha_0, 1 - \gamma, l = \frac{\gamma - \beta - \beta_0}{\gamma^2 - \beta - \beta_0})$ approximate-list-decodable code, for every $\gamma > \sqrt{\beta + \beta_0}$. Choosing $\gamma = \sqrt{2\beta + \beta_0}$, we get,

Corollary 4.3. E is an $(\alpha + \alpha_0, 1 - \sqrt{2(\beta + \beta_0)}, l = \frac{2}{\beta + \beta_0})$ approximate-list-decodable code.

The above corollary is about combinatorial list-decoding, and does not imply that there is an efficient algorithm that approximate-list-decodes E with these parameters.

4.2 The list-decoding algorithm

In this section we describe a polynomial time list-decoding algorithm for E. Denote the input $z' \in \Sigma_1^{|V_1'|}$. We interpret it as $\{z_S'\}_{S \in V_1'}$. As a preprocessing step, we create $z \in \Sigma_1^{|V_1|}$ as follows: for every $S \in V_1$, the multi-set V_1' has at most D copies of S. The algorithm picks at random one such copy, $S' \in V_1'$, and sets $z_S = z_{S'}'$ (this preprocessing step is not needed in the perfectly regular setup).

1. **Approximate-list-decoding of Local Views**

   For every $T \in V_2$, the graph $X_{|T}$ is an unweighted (α_0, β_0) sampler. The restriction of E to $X_{|T}$ is the ABNNR encoding over the sampler $X_{|T}$. We apply the well-separated list-decoding algorithm, Algorithm 1, on $X_{|T}$ with input $\{z_S\}_{S \in V_1, S \subset T}$, and decoding parameter $(1 - \frac{\gamma}{2})$. The algorithm outputs a list $L_T \subset \Sigma_0^T$ and a radius r_T such that $|L_T| \leq \ell$ for $\ell \leq \frac{8}{\gamma}$ and $r_T = 2 \cdot 10^i \alpha_0$ for some $i \in \{0, \ldots \ell\}$. The list L_T satisfies

   - Every $\sigma \in \Sigma_0^T$ such that $\Pr_{S \sim (\Pi_1 | \Pi_2 = T)}[\sigma_{|S} = z_S] \geq \frac{\gamma}{2}$ is r_T-close to one of the elements in L_T, and,
   - L_T itself is $R_T = 5r_T$-separated, i.e. $\forall \sigma \neq \sigma' \in L_T, \text{dist}(\sigma, \sigma') > R_T$.

   W.l.o.g. the list L_T has size exactly ℓ, otherwise we add dummy strings that obey the distance requirements.

2. **Creating a UG constraint graph**

   - We define the constraint graph $(G_C = (V_2, E_C), W = \{w_e\}_{e \in E_C})$ as follows: The vertices are V_2 and for every triple T_1, S, T_2 such that $S \subset T_1 \cap T_2, S \in V_1$, we have an edge (T_1, T_2) labeled by S, denoted by $(T_1, T_2)_S$. The weight of $(T_1, T_2)_S$ corresponds to choosing a random $S \sim \Pi_1$ and then T_1, T_2 independently from the distribution $(\Pi_2 | \Pi_1 = S)$. Thus, the graph contains parallel edges and self loops. This is the two-step walk graph obtained from $X(V_2, V_1)$, see Section 2.4.

25

- For every T, the label set is L_T (note that $|L_T| = \ell$).
- Given an edge $(T_1, T_2) \in E_C$ with label $S \subset T_1 \cap T_2$, set the constraint π of (T_1, T_2) with label S as follows:

  (a) For every $\sigma \in L_{T_1}$ if there is an unmatched $\sigma' \in L_{T_2}$ such that
  $$\text{dist}_S(\sigma, \sigma') \leq 2(r_{T_1} + \alpha_0),$$
  then set $\pi(\sigma) = \sigma'$.

  (b) For every unmatched $\sigma \in L_T$, set $\pi(\sigma)$ to an arbitrary unmatched label.

  Observe that we always output unique constraints, because we only ever set $\pi(\sigma)$ to an unmatched label.

3. **Finding a large expanding UG constraint subgraph**

   For every $i \in \{0, \ldots \ell\}$, let $V^{(i)} \subset V_2$ be all T such that $r_T = 2\alpha_0 10^i$. For every i such that $\mu_{G_C}(V^{(i)}) \geq \frac{1}{2(\ell+1)}$, we run Algorithm 2 on the graph G_C with the subset $V^{(i)}$. The algorithm finds a set $U^{(i)} \subset V^{(i)}$ such that,

   - $\Pr_{T \sim \Pi_2}[T \in U^{(i)}] \geq \frac{1}{4} \Pr_{T \sim \Pi_2}[T \in V^{(i)}]$.
   - Denote by $G^{(i)}$ the induced subgraph of G_C on $U^{(i)}$. Then, $\lambda_2(G^{(i)}) \leq \frac{99}{100}$.

4. **Solving the Unique Constraints**

   For every i as above, we run the unique games algorithm, Algorithm 3, on $G^{(i)}$ and get a list $\mathcal{L}^{(i)}$ of assignments. For each assignment $b : U^{(i)} \to [\ell]$ in $\mathcal{L}^{(i)}$ we define $x \in \Sigma_0^{V_0}$ as follows. For every $j \in V_0$ we pick a random $T \in U^{(i)}$ such that $j \in T$, according to the vertex weights of $G^{(i)}$. Let $\sigma \in \Sigma_0^T$ be the $b(T)$'th element in L_T, then we set $x_j = \sigma_j$. We add x to L_{out}. The output is a list $L_{out} \subset \Sigma_0^n$.

4.3 Algorithm Parameters

In this section we analyze the parameters of the decoding algorithm, the size of the output list, the runtime and the randomness.

Output List Size In the first step of the algorithm (see Item 1) the algorithm creates a list L_T and a radius $r_T = 2\alpha_0 10^i$ for every $T \in V_2$. The size of L_T is at most ℓ and the radius has one of $\ell + 1$ possible values. In the second step, Item 2, the algorithm creates a constraint graph G_C with constraints $\pi_e : [\ell] \to [\ell]$.

In the third step, Item 3, the algorithm creates a subgraph $G^{(i)}$ for every $i \in \{0, \ldots, \ell\}$. There are at most $(\ell + 1)$ such graphs $G^{(i)}$, one for every possible value of r_T. For every graph $G^{(i)}$, the algorithm solves a unique games instance and outputs a list containing at most ℓ assignments (see Item 4). For each such assignment the algorithm adds a string to L_{out}, therefore $|L_{out}| \leq (\ell + 1)\ell \leq \frac{70}{\gamma^2}$.

The same encoding E is combinatorially approximate-list-decodable with list size $O(\frac{1}{\gamma})$, see Section 4.1. Our algorithm outputs a list of size $O(\frac{1}{\gamma^2})$, and we don't know how to shorten the list. If we had a way to check if x should be in L_{out}, i.e. if it is an ε-approximation to x' s.t. $\text{dist}(E(x'), z') \leq 1 - \gamma$, we could have reduced the output list size to $O(\frac{1}{\gamma})$. If the encoding E is used to amplify the distance of a uniquely-decodable error correcting code C, then $E(C)$ has a decoding algorithm with output list of $O(\frac{1}{\gamma})$, see Section 4.4.

Runtime We bound the runtime of the decoding algorithm by going over each step it performs and calculating its runtime. The preprocessing step takes linear time in $|V_0|$. In the first step the algorithm performs Algorithm 1 on $X_{|T}$ for every $T \in V_2$. Preforming Algorithm 1 on $X_{|T}$ takes $\ell 2^{|T|}$. Since $|T|$ and ℓ are constants, the total runtime of this step is linear in $|V_2|$.

In the second step the algorithm creates the constraint graph G_C by performing a two-step walk over the bipartite graph $X(V_2, V_1)$. The size of G_C can be seen to be linear in $|V_2|$. It takes polynomial time in $|X|$ to construct it. Creating the constraints for each edge is a local operation which takes constant time for each edge.

In the third step the algorithm goes over all $i \in \{0, \ldots, \ell\}$ and finds an expander subgraph $G^{(i)}$ of G_C by running Algorithm 2. Algorithm 2 runs in time polynomial in the original graph size, which means that the entire step takes polynomial time in $|X|$.

In the last step, the algorithm performs Algorithm 3 on all of the graphs $G^{(i)}$ (there are at most $\ell + 1$ such graphs). By Theorem 2.14, Algorithm 3 runs in time $\mathrm{poly}(|G^{(i)}|)$, so the entire step also takes polynomial time in $|X|$.

Randomness The algorithm uses randomness twice directly, and one more time when running the unique games algorithm of Makarychev and Makarychev [MM10]. The unique games algorithm of [MM10] can be derandomized, as they explain in their paper. The randomness directly inside our algorithm occurs in the following places.

- Preprocessing step. This step is avoided when X is a perfectly regular double sampler. In the more general irregular case, the preprocessing step chooses for each $S \in V_1$ a copy of S in V_1'. To derandomize, one can reuse the same randomness for every $S \in V_1$. This way, the number of random bits needed is $\log D$, and we can cycle through all of these easily. We can enumerate over all possible random strings in $\{0,1\}^{\log D}$ and generate an output list for each random string. The output of the derandomized algorithm is the union of the output lists of the random strings. Our randomized algorithm succeeds with high probability, so there must be a random string which succeeds.

- Final step (step 4) of the algorithm. In this step for each assignment $b : U^{(i)} \to [\ell]$, and each $j \in [n]$ the algorithm picks a random $T \in U^{(i)}$ which contains j and uses it to define $x(j)$. There are $k = O(1)$ possible choices [5] for a set T such that $j \in T$ and we can instead proceed as follows. First define $x(j)$ according to the first choice for all j. This leads to a list of possible codewords. Next, define $x(j)$ according to the second choice for all j, and so forth. Finally combine the lists from all possible choices.

The derandomized algorithm may output a somewhat larger list than the randomized one, as it outputs a list for every choice used. In approximate-list-decoding, we are not able to prune the output list and shorten it (see discussion in the beginning of Section 3). Luckily, if the encoding E is used to amplify the distance of an ECC C, the list size does not increase, see the next section for more details.

---

[5]By our assumption, the distribution Π_0 is uniform, meaning that each j participates in the same number of sets $T \in V_2$, which must be a constant number.

4.4 Proof of Corollary 1.3

In this section we show how our main theorem implies a list-decoding algorithm for the code $E(C) = \{E(x) \mid x \in C\}$.

Corollary 4.4. *For all $\gamma, \varepsilon > 0$, let $(X = (V_2, V_1, V_0 = [n]), W)$ be a double sampler with parameters $\alpha, \beta, \alpha_0, \beta_0$ as in Theorem 4.1, and irregularity at most D. Let $C \subset \Sigma_0^n$ be an error correcting code with a polynomial time unique-decoding algorithm from an ε-fraction of errors. Then the error correcting code $E(C)$ has a randomized polynomial time $(1 - \gamma, l = \frac{\gamma - \beta - \beta_0}{\gamma^2 - \beta - \beta_0})$ list-decoding algorithm.*

Plugging in the double samplers from Theorem 2.11, as done in Corollary 4.2, results in an error correcting code with rate $\exp(-\operatorname{poly}(\frac{1}{\varepsilon}\exp(-\frac{1}{\gamma}))) \cdot \operatorname{Rate}(C)$.

Proof. Fix an input $z \in \Sigma_1^{|V_1'|}$, the list-decoding algorithm of $E(C)$ on z proceeds as follows.

- Run the approximate-list-decoding algorithm for E on input z, receive a list L_{out} of size at most $\frac{70}{\gamma^2}$.

- For each $x \in L_{out}$, run the decoding algorithm of C on x.

  - If failed, do nothing.
  - If it outputs $x' \in C$, insert x' into L' only if $\operatorname{dist}(E(x'), z) \leq 1 - \gamma$.

The above algorithm is polynomial time since the decoding algorithms of E, C are polynomial, and $|L_{out}|$ is constant.

We prove the correctness of the algorithm. Fix $z' \in \Sigma_1^{|V_1'|}$, and let $y \in C$ be a string satisfying $\operatorname{dist}(z, E(y)) \leq 1 - \gamma$. From Theorem 4.1, there is $x \in L_{out}$ such that $\operatorname{dist}(x, y) \leq \varepsilon$. The unique-decoding algorithm of C on input x should return y, and the algorithm inserts y into L' as $\operatorname{dist}(E(y), z) \leq 1 - \gamma$.

We are left with bounding the list size. The encoding E is $(1 - \gamma, \alpha + \alpha_0, l = \frac{\gamma - \beta - \beta_0}{\gamma^2 - \beta - \beta_0})$ combinatorially approximate-list-decodable (see Section 4.1). The error correcting code C has distance at least 2ε, and $\varepsilon > \alpha + \alpha_0$ (by the conditions of Theorem 4.1). Claim 3.11 proves that $E(C)$ is $(\alpha + \alpha_0, l = \frac{\gamma - \beta - \beta_0}{\gamma^2 - \beta - \beta_0})$ combinatorially list-decodable, that is, that $L = \{x \in C \mid \operatorname{dist}(E(x), z)\}$ has size at most l. The output list of the algorithm L' satisfies $L' \subseteq L$, because the algorithm inserts $x' \in C$ into L' only if $\operatorname{dist}(E(x'), z) \leq 1 - \gamma$, so $|L'| \leq |L| \leq l$. $\square$

5 Proof of Correctness

Fix $g : V_0 \to \Sigma_1$ such that $\operatorname{Pr}_{S \in V_1'}[g_{|S} = z'_S] \geq \gamma$. It suffices to show that with high probability, $\operatorname{dist}(g, L_{out}) \leq \varepsilon$. Whenever we say "correct" in this section we always mean correct with respect to the fixed function g.

Let $a : V_2 \to [\ell]$ be an assignment. For $T \in V_2$ we overload notation and denote by $a(T)$ the $a(T)$'th element in the list L_T (formally this is $L_T(a(T))$). Similarly we treat a constraint $\pi_{(T_1, T_2)_S} : [\ell] \to [\ell]$ as $\pi_{(T_1, T_2)_S} : L_{T_1} \to L_{T_2}$.

Preprocessing Step   We first prove that $z \in \Sigma_1^{V_1}$ constructed from the input $z' \in \Sigma_1^{V_1'}$ is noticeably correlated with g.

Claim 5.1. *With probability at least $1 - e^{-c'n}$, for some constant c',*

$$\Pr_{S \sim \Pi_1}[z_S = g_{|S}] \geq \frac{3}{4}\gamma.$$

Proof. Each $S \in V_1$ has possibly several copies in the multiset V_1'. The algorithm chooses S' to be a random copy, and sets $z_S = z'_{S'}$. It remains to use a tail bound to show that it is highly unlikely that the correlation drops below $3\gamma/4$ after moving to $\{z_S\}$.

For every $S \in V_1$, let I_S be the random variable that indicates the event $z_S = g_{|S}$. The input z' is γ-close to g:

$$\mathbb{E}_{S \sim \Pi_1}[I_S] = \Pr_{S' \in V_1'}[z'_{S'} = g_{|S'}] \geq \gamma.$$

We use a Chernoff tail bound (see Section 2) on the independent random variables $\{I_S\}_{S \in V_1}$. We define the random variable $I = \sum_{S \in V_1} \Pi_1(S)I_S$, then $\mathbb{E}[I] = \mathbb{E}_{S \sim \Pi_1}[I_S]$ and define $\nu = \sum_{S \in V_1} \Pi_1^2(S)\,\mathbb{E}[I_S]$. The distribution Π_1 has irregularity at most D, which lets us bound ν: $\nu \leq \frac{D}{|V_1|} \sum_{S \in V_1} \Pi_1(S)\,\mathbb{E}[I_S] \leq \frac{D}{|V_1|}\mathbb{E}[I]$.

$$\Pr\left[\mathbb{E}[I] < \frac{3}{4}\gamma\right] \leq e^{-\frac{1}{32D}|V_1|\gamma}.$$

Since D, γ are constants and $|V_1| = \Theta(n)$, we pick $c' = \frac{1}{32D}\gamma\frac{n}{|V_1|}$ and finish the proof. $\square$

In the rest of the proof, we assume that z is such that $\Pr_{S \sim \Pi_1}[z_S = g_{|S}] \geq \frac{3}{4}\gamma$.

5.1   Approximate-list-decoding of local views

The decoding algorithm takes each $T \in V_2$, and applies the well-separated list-decoding algorithm, Algorithm 1, to the graph $X_{|T}$ on input $\{z_S\}_{S \in V_1, S \subset T}$ and closeness parameter $(1 - \frac{\gamma}{2})$. The graph $X_{|T}$ is an (α_0, β_0) unweighted sampler, and $\gamma > 10\sqrt{\beta_0}$. Algorithm 1 outputs a list L_T and radius a r_T. From Lemma 3.9, the list and radius satisfy

$$|L_T| = \ell \leq \frac{\frac{\gamma}{2} - \beta_0}{\frac{\gamma^2}{4} - \beta_0} \leq \frac{8}{\gamma},$$

and $r_T = 2\alpha_0 10^i$ for some $i \in \{0, \dots \ell\}$. The list L_T is $R_T = 5r_T$ separated.

Definition 5.2. *(Correct vertex) A vertex $T \in V_2$ is* correct *if there exists $\sigma \in L_T$ such that $dist_T(g, \sigma) \leq r_T$.*

We are left with showing that almost every $T \in V_2$ is correct.

Claim 5.3. $\Pr_{T \sim \Pi_2}[T \text{ is correct}] \geq 1 - \beta$.

Proof. We look at the bipartite sampler $(X(V_2, V_1), \Pi_{2,1})$. Let $f : V_1 \to [0,1]$ equal 1 if $z_S = g_{|S}$ and 0 otherwise. We know that $\mathbb{E}_{S \sim \Pi_1}[f(S)] \geq \frac{3}{4}\gamma > 3\alpha$. Let

$$Bad = \left\{T \in V_2 \;\middle|\; \mathbb{E}_{S \sim T}[f(S)] < \frac{3}{4}\gamma - \alpha\right\}.$$

29

As $(X(V_2, V_1), \Pi_{2,1})$ is an (α, β) sampler, $\Pr_{T \sim \Pi_2}[T \in Bad] \leq \beta$.

For every $T \notin Bad$,

$$\mathbb{E}_{S \sim T}[z_S = g_{|S}] \geq \frac{3}{4}\gamma - \alpha \geq \frac{\gamma}{2}.$$

From Lemma 3.9, for every T such that the above equation holds, the list L_T contains σ such that $dist_T(\sigma, g) \leq r_T$. $\qquad\square$

5.2   Creating a UG constraint graph

The algorithm creates a constraint graph $(G_C = (V_2, E_C), W)$ which is a two-step random walk on the weighted bipartite sampler graph $(X(V_2, V_1), \Pi_{2,1})$. For each edge $(T_1, T_2)_S$ in G_C, the algorithm creates a matching between the lists L_{T_1}, L_{T_2}. In this section we prove that for every radius $r = 2\alpha_0 10^i$, and for most edges $(T_1, T_2)_S$ for which $r_{T_1} = r_{T_2}$, the algorithm matches the list element closest to g in L_{T_1} to the list element closest to g in L_{T_2}.

Definition 5.4 (Correct assignment and constraint)**.** *Let $a : V_2 \to [\ell]$ be the assignment that for every $T \in V_2$ assigns the element in L_T closest to $g|_T$. Ties are broken arbitrarily. We say π is* correct *on $e = (T_1, T_2)_S \in E$ if $dist_{T_1}(a(T_1), g) \leq r_{T_1}$, $dist_{T_2}(a(T_2), g) \leq r_{T_2}$ and $\pi_e(a(T_1)) = a(T_2)$.*

Definition 5.5. *Let T be a correct vertex. For $S \in V_1$, $S \subseteq T$, we say that the pair (S, T) is* correct *if*

- $dist_S(a(T), g) \leq r_T + \alpha_0$, *and,*

- *For every $\sigma \in L_T, \sigma \neq a(T)$ we have $dist_S(\sigma, g) > R_T - r_T - \alpha_0$.*

We prove that if T is a correct vertex, then for almost all of $S \subset T, S \in V_1$, the pair (S, T) is correct.

Claim 5.6. *Suppose $T \in V_2$ is correct. Then $\Pr_{S \sim (\Pi_1 | \Pi_2 = T)}[(S, T)$ is correct$] \geq 1 - \ell\beta_0$.*

Proof. Fix a correct $T \in V_2$, and denote $X_{|T} = (U_T, T, E)$ (i.e. U_T is all $S \in V_1$ such that $S \subset T$).

For every $\sigma \in L_T$, let

$$Bad_\sigma = \{S \in U_T \mid |dist_S(\sigma, g) - dist_T(\sigma, g)| > \alpha_0\}.$$

As X is double sampler, $X_{|T}$ is an (α_0, β_0) sampler and for every $\sigma \in \Sigma_0^T$, $\Pr_{S \sim (\Pi_1 | \Pi_2 = T)}[S \in Bad_\sigma] \leq \beta_0$.

For a correct T, $dist_T(a(T), g) \leq r_T$ and for every other $\sigma' \in L_T, \sigma' \neq a(T)$, we have

$$dist_T(\sigma', g) \geq dist_T(\sigma', a(T)) - dist_T(a(T), g) \geq R_T - r_T.$$

This implies that for $S \notin \cup_{\sigma \in L_T} Bad_\sigma$,

- $dist_S(a(T), g) \leq dist_T(a(T), g) + \alpha_0 \leq r_T + \alpha_0.$

- For every $\sigma' \in L_T, \sigma' \neq a(T)$, we have $dist_S(\sigma', g) \geq dist_T(\sigma', g) - \alpha_0 \geq R_T - r_T - \alpha_0.$

So the pair (S,T) is correct for every $S \notin \cup_{\sigma \in L_T} Bad_\sigma$, taking a union bound over the probability of $S \in Bad_\sigma$ for every $\sigma \in L_T$ finishes the proof. $\qquad \square$

Lemma 5.7. *For $T_1, T_2 \in V_2$ and $S \in V_1, S \subset T_1 \cap T_2$ let $\pi_{(T_1,T_2)_S}$ denote the constraint for the edge $(T_1, T_2)_S$. If $T_1, T_2, (S, T_1), (S, T_2)$ are correct and $r_{T_1} = r_{T_2} = r$, then $\pi_{(T_1,T_2)_S}$ is correct.*

Proof. First notice that $\pi_{(T_1,T_2)_S}$ can match $a(T_1)$ to $a(T_2)$, because

$$\operatorname{dist}_S(a(T_1), a(T_2)) \leq \operatorname{dist}_S(a(T_1), g) + \operatorname{dist}_S(a(T_2), g) \leq r_{T_1} + r_{T_2} + 2\alpha_0 = 2(r + \alpha_0).$$

Next, observe that no other list element can be matched to either $a(T_1)$ or $a(T_2)$. To see that consider a pair $(\sigma, a(T_2))$ for some $a(T_1) \neq \sigma \in L_{T_1}$. Then,

$$
\begin{aligned}
\operatorname{dist}_S(\sigma, a(T_2)) &\geq \operatorname{dist}_S(\sigma, g) - \operatorname{dist}_S(a(T_2), g) \\
&\geq R_{T_1} - r_{T_1} - \alpha_0 - (r_{T_2} + \alpha_0) \\
&\geq R_{T_1} - 2(r + \alpha_0) > 2(r + \alpha_0),
\end{aligned}
$$

which holds because $R_T = 5r, R_T \geq 10\alpha_0$, then $R_T > 4(r + \alpha_0)$. Hence $\pi_{(T_1,T_2)_S}$ matches $a(T_1)$ to $a(T_2)$ and only to $a(T_2)$ and vice versa. $\qquad \square$

5.3 Finding a large expanding UG constraint subgraph

The starting point of this section is the constraint graph G_C which has unique constraints. In this section we show that the algorithm finds an induced subgraph of G_C which is an expander, and that almost all of its constraints are correct.

Recall that $V^{(i)}$ is the set of all $T \in V_2$ with $r_T = 2\alpha_0 10^i$. The decoding algorithm goes over all i such that $V^{(i)}$ is not too small, and finds an expander subgraph $G^{(i)}$ of G_C in which all vertices are in $V^{(i)}$. In this section we show that there is at least one i in which the expander graph $G^{(i)}$ is correct.

Denote $\mu(V^{(i)}) = \Pr_{T \sim \Pi_2}[T \in V^{(i)}]$. Let $E_C^{(i)} \subset E_C$ be all edges in G_C such that both endpoints have radius $r = 2\alpha_0 10^i$.

Lemma 5.8. *Denote $\eta = \beta + \ell\beta_0$. There exists $i \in \{0, \ldots \ell\}$, such that:*

- $\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)}$, *and,*

- $\Pr_{(T_1,T_2)_S \in E_C^{(i)}}[\pi_{(T_1,T_2)_S}$ *is correct*$] \geq 1 - 4\eta$.

Proof. For every i let

$$
\begin{aligned}
p &= \Pr_{(S,T) \sim (\Pi_1, \Pi_2)}[T, (S,T) \text{ are correct}] \\
p^{(i)} &= \Pr_{(S,T) \sim (\Pi_1, \Pi_2)}[T, (S,T) \text{ are correct} | T \in V_2^{(i)}].
\end{aligned}
$$

Then by Claim 5.3 and Claim 5.6

$$
\begin{aligned}
p &= \Pr[T, (S,T) \text{ are correct}] = \Pr[T \text{ is correct}] \cdot \Pr[(S,T) \text{ is correct} \mid T \text{ is correct}] \\
&\geq (1 - \beta) \cdot (1 - \ell\beta_0) \geq 1 - \beta - \ell\beta_0 = 1 - \eta.
\end{aligned}
$$

We claim:

Claim 5.9. *There exists $i \in \{0, \dots \ell\}$ with $\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)}$ and $p^{(i)} \geq 1 - 2\eta$.*

Proof. Assume towards contradiction that no such i exists. Let

$$q = \sum_{i:\mu(V^{(i)}) < \frac{1}{2(\ell+1)}} \mu(V^{(i)}).$$

Therefore $q < \frac{1}{2}$ and

$$
\begin{aligned}
p &= \sum_{i \in [\ell]} \mu(V^{(i)})p^{(i)} = \sum_{i:\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)}} \mu(V^{(i)})p^{(i)} + \sum_{i:\mu(V^{(i)}) < \frac{1}{2(\ell+1)}} \mu(V^{(i)})p^{(i)} \\
&< (1-2\eta)(1-q) + q \\
&= 1 - 2\eta(1-q) < 1 - \eta,
\end{aligned}
$$

which contradicts $p \geq 1 - \eta$. $\qquad\square$

Now, fix i such that $\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)}$ and $p^{(i)} \geq 1 - 2\eta$. Let $(V^{(i)}E_C^{(i)}), W^{(i)}$ be the induced graph on $V^{(i)}$. We prove the second item,

$$
\begin{aligned}
\Pr_{(T_1,T_2)_S \sim E_C^{(i)}} &[\text{ one of } T_1, T_2, (T_1, S)(T_2, S) \text{ is incorrect }] \\
&\leq \Pr_{(S,T_1) \sim (\Pi_1, \Pi_2)} [T_1 \text{ or } (T_1, S) \text{ is incorrect} \mid T_1 \in V_2^{(i)}] \\
&\quad + \Pr_{(S,T_2) \sim (\Pi_1, \Pi_2)} [T_2 \text{ or } (T_2, S) \text{ is incorrect} \mid T_2 \in V_2^{(i)}] \\
&\leq 2(1 - p^{(i)}) \leq 4\eta.
\end{aligned}
$$

The inequality holds because the weight of an edge $(T_1, T_2)_S \sim E_C^{(i)}$ is the probability of picking $S \sim \Pi_1$, and $T_1, T_2 \sim (\Pi_2|\Pi_1 \mathcal{S})$ independently, given that $r_{T_1} = r_{T_2} = 2\alpha_0 10^i$.

The second item follows by Lemma 5.7, because whenever all of $T_1, T_2, (T_1, S)$ and (T_2, S) are correct, we have that $\pi_{(T_1,T_2)_S}$ is correct, i.e.

$$
\Pr_{(S,T_1,T_2) \sim E_C^{(i)}} [\pi_{(T_1,T_2)} \text{ is incorrect }] \leq 4\eta.
$$

$\qquad\square$

We have $\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)} > 10\sqrt{\max\{\alpha, \beta\}}$. By Theorem 2.6, Algorithm 2 returns a subset $U^{(i)} \subset V^{(i)}$ such that

- $\mu_{G_C}(U^{(i)}) \geq \frac{\mu(V^{(i)})}{4} \geq \frac{1}{8(\ell+1)}$, and,

- $\lambda_2(G^{(i)}) \leq \frac{99}{100}$.

Furthermore, since $\mu(U^{(i)}) \geq \frac{1}{4}\mu(V^{(i)})$, from Claim 2.7,

$$
\Pr_{(T_1,T_2) \sim E_C} [T_1, T_2 \in U^{(i)} \mid T_1, T_2 \in V^{(i)}] \geq \frac{1}{20}.
$$

We bound the probability of an incorrect edge:

$$\Pr[\pi_{(T_1,T_2)} \text{ is incorrect}|T_1, T_2 \in U^{(i)}] \leq \frac{\Pr[\pi_{(T_1,T_2)} \text{ is incorrect}|T_1, T_2 \in V^{(i)}]}{\Pr[T_1, T_2 \in U^{(i)} \mid T_1, T_2 \in V^{(i)}]}$$
$$\leq 20 \cdot 4\eta, \tag{4}$$

where correct means correct with respect to g, as in the entire proof.

At this point we have found an expanding subgraph $G^{(i)}$ of G_C, such that almost all of its edges are correct with respect to g.

5.4 Solving the unique constraints

For every graph $G^{(i)}$, the decoding algorithm runs Algorithm 3 on $G^{(i)}$ and outputs a list of assignments $\mathcal{L}^{(i)}$. Then, the decoding algorithm takes every assignment in $b \in \mathcal{L}^{(i)}$ and transforms it into a string $x \in \Sigma_0^n$. In this section we prove that there exists one such x which approximates g.

Let $i \in \{0, \ldots \ell\}$ be the index promised from Lemma 5.8, and let $L_{out}^{(i)} \subset L_{out}$ be the subset of the output list created by the algorithm when running on $G^{(i)}$.

Claim 5.10. *With high probability, there exist $x \in L_{out}^{(i)}$ such that*

$$dist(g,x) \leq r_T + 80\eta(c^\ell + 1) + \alpha + 8\beta(\ell + 2).$$

Proof. Recall $a : V_2 \to [\ell]$ is the assignment which assigns each vertex $T \in V_2$ the list element closest to g. We prove the claim by showing there exists $b \in \mathcal{L}^{(i)}$ which is close to a on $U^{(i)}$. Then we prove that with high probability, the algorithm generates from b a string $x \in \Sigma_0^n$ which approximates g, details follows.

By (4), with probability $1 - 80\eta$ a random edge $(T_1, T_2)_S \in G^{(i)}$ is correct, that is $\pi_{(T_1,T_2)_S}(a(T_1)) = a(T_2)$. This means that the assignment a has value at least $1 - 80\eta$ on the unique games instance of $G^{(i)}$. The parameters of the double sampler promise that $\eta = \beta + \ell\beta_0$ is small enough to satisfy $80\eta c^{\ell+1} < 1$, so Theorem 2.14 guarantees that Algorithm 3 outputs an assignment $b : U^{(i)} \to [\ell]$ in $\mathcal{L}^{(i)}$ such that

$$\Pr_{T \sim U^{(i)}}[a(T) \neq b(T)] \leq c^\ell 80\eta, \tag{5}$$

where $T \sim U^{(i)}$ is the probability to pick a vertex T according to the weights of $G^{(i)}$.

We show that with high probability, the decoding algorithm on b outputs x which approximates g. For every $j \in [n]$, the decoding algorithm picks a random $T \sim U^{(i)}$ which contains j, and sets $x_j = b(T)_j$. For each j, let $err(j)$ be the probability that x_j is decoded incorrectly,

$$err(j) = \Pr_{T \sim U^{(i)}|j \in T}[b(T)_j \neq g_j].$$

If there is some $j \in [n]$ such that no $T \in U^{(i)}$ contains it, we define $err(j) = 1$. $\mathbb{E}_{j \sim \Pi_0}[err(j)] = \mathbb{E}_{j \in [n]}[err(j)]$ is the probability of a random coordinate to be decoded incorrectly (Π_0 is uniform over $[n]$) and our goal is to bound it.

Choosing a random $T \sim U^{(i)}$ then a random $j \in T$ results in a weighted distribution over V_0, denote this distribution by $\Pi_0^{(i)}$. We show two things:

- $\mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] \leq r_T + 80\eta(c^\ell + 1)$ on Claim 5.11, and,

- $\mathbb{E}_{j\in[n]}[err(j)] \leq \mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] + \alpha + 8\beta(\ell + 1)$ on Claim 5.12,

together this implies $\mathbb{E}_{j\in[n]}[err(j)] \leq r_T + 80\eta(c^\ell + 1) + \alpha + 8\beta(\ell + 1)$.

To finish the proof, we need to show that with high probability $\text{dist}(x, g)$ is small. The algorithm chooses for each $j \in [n]$ a random set $T \ni j$ independently, so we can apply a Chernoff tail bound,

$$\Pr\left[\Pr_{j\in[n]}[w_j \neq g_j] > (r_T + 80\eta(c^\ell + 1) + \alpha + 8\beta(\ell + 1)) + \beta\right] \leq e^{-\beta^2 n},$$

and get that with probability $\exp(-n)$ the string x satisfies $\text{dist}(x, g) \leq r_T + 80\eta(c^\ell + 1) + \alpha + 8\beta(\ell + 2)$.

$\square$

The parameters satisfy $r_T \leq \alpha_0 10^{\frac{8}{\gamma}}$, $\ell = \frac{8}{\gamma}$ and $\eta = \beta + \ell\beta_0$, which results in $\varepsilon = 10^{\frac{8}{\gamma}} \cdot \alpha_0 + 80(\beta + \frac{8}{\gamma}\beta_0) \cdot (c^{\frac{8}{\gamma}} + 1) + \alpha + 8\beta(\frac{8}{\gamma} + 2)$. In the proof of Lemma 5.8 we use the fact that $\mu(V^{(i)}) \geq \frac{1}{2(\ell+1)} > 10\sqrt{\max\{\alpha,\beta\}}$, which holds for $\gamma > \max\{200\sqrt{\alpha}, 200\sqrt{\beta}\}$. In order for the well-separated list-decoding algorithm to succeed, we require that that $\gamma > 10\sqrt{\beta_0}$. The unique games algorithm from Claim 5.10 requires $80(\beta + \frac{8}{\gamma}\beta_0) \cdot c^{\frac{8}{\gamma}+1} < 1$. The $\alpha, \beta, \alpha_0, \beta_0$ which satisfy the conditions of Theorem 4.1 satisfy all these requirements.

The algorithm uses randomness in the preprocessing step and in the final step above. In each time the success probability is $1 - \exp(-n)$, so the total success probability is also $1 - \exp(-n)$.

We remark that our algorithm uses as black box the unique games algorithm of [MM10]. Their algorithm is randomized, and can be derandomized without changing the algorithm parameters, as they explain in their paper.

To finish the proof, we are left with proving the two claims.

Claim 5.11. $\mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] \leq r_T + 80\eta(c^\ell + 1)$.

Proof. $\mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)]$ is the probability the following experiment fails: pick $j \sim \Pi_0^{(i)}$ and $T \sim U^{(i)}$ which contains j, and check whether $b(T)_j = g_j$. This is the same distribution as picking $T \sim U^{(i)}$ and then a uniform $j \in T$. Hence,

$$\begin{aligned}
\mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] &= \mathbb{E}_{T\sim U^{(i)}, j\in T}[b(T)_j \neq g_j] \\
&\leq \Pr_{T\sim U^{(i)}}[\text{dist}_T(b(T), g) \leq r_T] \cdot r_T + \Pr_{T\sim U^{(i)}}[\text{dist}_T(b(T), g) > r_T] \\
&\leq r_T + (c^\ell + 1)80\eta.
\end{aligned}$$

The last inequality is true because

$$\begin{aligned}
\Pr_{T\sim U^{(i)}}[\text{dist}_T(b(T), g) > r_T] &\leq \Pr_{T\sim U^{(i)}}[\text{dist}_T(a(T), g) > r_T] + \Pr_{T\sim U^{(i)}}[a(T) \neq b(T)] \\
&\leq 80\eta + c^\ell 80\eta. &&\text{(by (4) and (5))}
\end{aligned}$$

$\square$

Next we prove:

Claim 5.12. $\mathbb{E}_{j\in[n]}[err(j)] \le \mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] + \alpha + 8\beta(\ell+1)$.

Proof. Let $Bad \subseteq V_2$ be the set of T for which

$$\left| \mathbb{E}_{j\sim(\Pi_0|\Pi_2=T)}[err(j)] - \mathbb{E}_{j\in[n]}[err(j)] \right| \ge \alpha.$$

By Claim 2.10, the graph $X(V_2, V_0)$ is an (α, β) sampler and $\Pr_{T\sim\Pi_2}[T \in BAD] \le \beta$ (recall that Π_0 is uniform, so $\mathbb{E}_{j\in[n]}[err(j)] = \mathbb{E}_{j\sim\Pi_0}[err(j)]$). Since $\Pr_{T\sim\Pi_2}[T \in U^{(i)}] \ge \frac{1}{8(\ell+1)}$, $\Pr_{T\sim U^{(i)}}[T \in BAD] \le 8(\ell+1)\beta$.

We have

$$
\begin{aligned}
\mathbb{E}_{j\sim\Pi_0^{(i)}}[err(j)] &= \mathbb{E}_{T\sim U^{(i)}}\left[\mathbb{E}_{j\in T}[err(j)] \right] \\
&\ge \Pr_{T\sim U^{(i)}}[T \notin BAD] \, \mathbb{E}_{T\sim U^{(i)}\setminus BAD}\left[\mathbb{E}_{j\in T}[err(j)] \right] \\
&\ge (1 - 8\beta(\ell+1))(\mathbb{E}_{j\in[n]}[err(j)] - \alpha),
\end{aligned}
$$

where the last inequality is because the term is a convex combination of elements that are within α of the common number $\mathbb{E}_{j\in[n]}[err(j)]$. $\square$

6 High Dimensional Expanders yield Double Samplers

In this section we describe how to construct double samplers from high dimensional expanders, proving Theorem 2.11. In a nutshell, we take the high dimensional expanders constructed by Lubotzky, Samuels and Vishne [LSV05], and let the layers of the double sampler be $V_0 = X(0), V_1 = X(a), V_2 = X(b)$ for appropriately chosen $0 < a < b$, and put edges for inclusion of subsets. Below we give some minimal background on high dimensional expanders and prove that this construction is indeed a double sampler.

A d-dimensional complex X is given by a collection of $(d+1)$-subsets of a ground set $[n]$, called d-faces. For each $i < d$ we define a distribution $\mathcal{D}_i$ over $(i+1)$-subsets as follows: choose a d-dimensional face uniformly and then remove $d - i$ elements from this set at random. A set of size $i+1$ that has positive probability is called an i-face of the complex, and we denote the collection of i-faces by $X(i)$. We let $\mathcal{D}_d$ denote the uniform distribution on the top faces and remark that even though $\mathcal{D}_d$ is uniform, $\mathcal{D}_i$ need not be uniform, since some i-faces can be contained in more d-faces than others[6]. We also set $X(0) = [n]$ to be the ground set.

For every $s \in X(i)$ for $0 \le i \le d - 2$, we define the graph X_s:

- The vertices are $x \in X(0)$ such that $s \cup \{x\} \in X(i+1)$ is a face (clearly $x \notin s$).

- Two vertices x, y are connected by an edge if $s \cup \{x, y\} \in X(i+2)$.

---

[6]A very recent work [FI20] shows how to construct regular high dimensional expanders, giving rise to uniform $\mathcal{D}_i$ for all i, and with parameters very similar to those of [LSV05]. This was not available when this manuscript was completed yet it gives rise to perfectly regular double samplers which simplify some of the work done here.

- The weight of the edge $\{x, y\}$ is

$$w_s(x, y) := \Pr_{t \sim \mathcal{D}_{i+2}} [t = (s \cup \{x, y\}) \mid t \supset s].$$

The edge weights w_s define a marginal weight distribution on the vertices in X_s. Explicitly, for $x \in X_s$,

$$w_s(x) = \sum_{y \sim x} w_s(x, y) = \Pr_{t \sim \mathcal{D}_{i+1}} [t \supset s \cup \{x\} \mid t \supset s] = \Pr_{(u,v) \sim (\mathcal{D}_{i+1}, \mathcal{D}_i)} [u = s \cup \{x\} \mid v = s].$$

The graph X_s is called the 1-skeleton of the link of s in literature on high dimensional expanders.

There are several different definitions of high dimensional expansion. For our purposes, the most relevant is the one-sided spectral expansion,

Definition 6.1 (Spectral high dimensional expander (HDX)). *A d-dimensional complex X is said to be a γ-spectral high dimensional expander if for every $i \leq d - 2$, and every face $s \in X(i)$ the graph X_s is an expander with $\lambda(X_s) \leq \gamma$.*

Where $\lambda(G)$ is the second largest eigenvalue in G, *not* in absolute value. In a previous version of this paper we relied on a slightly stronger definition of *two-sided* high dimensional expansion, but it turns out that the above (one-sided) definition suffices, and in fact gives us a slightly cleaner result.

Lubotzky, Samuels and Vishne [LSV05] constructed an explicit family of Ramanujan complexes.

Theorem 6.2 (LSV). *For every prime $q \in \mathbb{N}$ and dimension $d \in \mathbb{N}$ there is a sequence of d-dimensional complexes $\{X_n\}_n$ which are $\frac{1}{\sqrt{q}}$-spectral high dimensional expanders. The vertex set of X_n has size q^{cn} for all large enough $n \in \mathbb{N}$, for some constant $1 < c \leq d^2$. X_n is constructible in time $\mathrm{poly}(n)$ and satisfies the following,*

- **Bounded degree:** *For each $i < d$ each face $s \in X_n(i)$ is contained in at most $D = q^{d^2}$ d-faces.*

- **Uniform top and bottom:** *The distribution $\mathcal{D}_d$ on the top d-faces is uniform, and the distribution $\mathcal{D}_0$ on the vertices is uniform as well (but $\mathcal{D}_i$ isn't uniform for $0 < i < d$).*

Let X be a d-dimensional simplicial complex. Fix $d = m_2 - 1 > m_1 - 1 \geq 0$ and define a graph as follows. Let $V_2 = X(m_2 - 1)$, $V_1 = X(m_1 - 1)$, and $V_0 = X(0)$ and look at the inclusion graph (V_2, V_1, V_0) together with the distribution $\Pi_1 = \mathcal{D}_{m_1 - 1}$ on V_1 and uniform distributions on V_2, V_0.

We first prove that the above construction is a spectral version of double sampler, and then use this lemma to prove Theorem 2.11.

Lemma 6.3 (Spectral version of double sampler). *Let X be a d-dimensional complex which is a γ-spectral high dimensional expander, and let $d = m_2 - 1 > m_1 - 1 \geq 0$. Let $V_2 = X(m_2 - 1)$, $V_1 = X(m_1 - 1)$ and $V_0 = X(0)$. Let $G_{2,1} = X(V_2, V_1)$ and $G_{1,0} = X(V_1, V_0)$ be the weighted bipartite inclusion graphs between (V_2, V_1) and (V_1, V_0) respectively, where the distributions over V_2, V_1, V_0 are the distributions $\mathcal{D}_2, \mathcal{D}_1, \mathcal{D}_0$ of X. The following spectral bounds hold,*

- $\lambda_2^{\mathrm{bip}}(G_{1,0})^2 \leq 1/m_1 + O(m_1\gamma)$, and

- $\lambda_2^{\mathrm{bip}}(G_{2,1})^2 \leq m_1/m_2 + O(m_1 m_2 \gamma)$.

Proof. The lemma follow essentially by combining the analysis of Dinur and Kaufman [DK17] of random walks on high dimensional expanders together with [KO20] that shows that the bounds in [DK17] hold also for the more general one-sided expansion case (which is what we defined above), we elaborate next.

For every $i \in [d]$, let A_i be the normalized adjacency matrix of the weighted bipartite graph between $X(i)$ and $X(i+1)$ (see Section 2.2). The matrix A_i can also be viewed as the random-walk operator, moving from an i-face to a random $(i+1)$-face containing it. In [KO20, Theorem 5.4] the authors defined the operator $M_i^+ \in \mathbb{R}^{X(i) \times X(i)}$, corresponding to a two-step random walk called the "non-lazy upper walk". That is, starting at an i-face, go to a random $(i+1)$-face r containing it, and then to an i-face contained in r that *isn't* the face you started with.

The operator A_i satisfies the following identity with M_i^+,

$$A_i^t A_i = \frac{1}{i+2} Id + \frac{i+1}{i+2} M_i^+ .$$

One can take this to be an explicit definition of $M_i^+ := \frac{i+2}{i+1} A_i^t A_i - \frac{1}{i+1} Id$. The bound proven in [KO20, Theorem 5.4] is that the second eigenvalue of M_i^+ is at most $(\frac{i}{i+1} + O(i \cdot \gamma))^{1/2}$, which means that $\lambda_2(A_i A_i^t) = \frac{i+1}{i+2} + O(i \cdot \gamma)$. We refer the reader to [DK17] to see more details on M_i^+ and how the relation between operators is derived.

The normalized adjacency matrix of the graph $G_{2,1}$ can also be described by the operator $A_{m_1} A_{m_1+1} \cdots A_{m_2}$, i.e. by moving in a random-walk fashion from m_1-face to (m_1+1)-face to (m_1+2)-face and so on all the way to m_2-face. Composing the corresponding linear operators, we get that the second eigenvalue is

$$\lambda_2^{\mathrm{bip}}(G_{2,1}) \leq \left(\frac{m_1+1}{m_1+2} \cdot \frac{m_1+2}{m_1+3} \cdots \frac{m_2-1}{m_2} \right)^{1/2} + O(\gamma m_1 m_2)$$

$$= \left(\frac{m_1+1}{m_2} \right)^{\frac{1}{2}} O(\gamma m_1 m_2),$$

where the error term comes from taking all error terms for every $m_1 \leq i \leq m_2$.

The calculation in the case of $G_{1,0}$ is identical, only we go over all $0 \leq i \leq m_1$. $\qquad \square$

We are now ready to prove Theorem 2.11, which we restate for convenience.

Theorem 2.11. *(restated) For every $\alpha, \beta, \alpha_0, \beta_0 > 0$ there exist constants $m_1, m_2, D \in \mathbb{N}$ such that $m_1, m_2 = \mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}), D = \exp(\mathrm{poly}(\frac{1}{\alpha\beta\alpha_0\beta_0}))$, such that there is a family of explicitly constructible double samplers (X_n, W_n) for infinitely many $n \in \mathbb{N}$ satisfying*

- $X_n = (V_2, V_1, V_0)$ *is an inclusion graph, where* $|V_0| = n$, $V_i \subseteq \binom{V_0}{m_i}$ *for* $i = 1, 2$.

- X_n *is an* $((\alpha, \beta), (\alpha_0, \beta_0))$ *double sampler.*

- $|V_1|, |V_2| \leq D \cdot n$.

- *The distributions Π_0, Π_2 are uniform and the distribution Π_1 has irregularity at most D.*

- *For each $m \in \mathbb{N}$ there is some $n \in [m, Dm]$ such that the complex X_n on n vertices is constructible in time $poly(n)$.*

Proof. We construct the double sampler from the LSV high dimensional expander promised by Theorem 6.2.

We start by choosing the parameters of the high dimensional expander, we choose $\gamma < 1/(m_1 m_2)^2$ small enough so that the term $O(m_1 m_2 \gamma)$ is negligible with respect to m_1/m_2 and $1/m_1$. We choose m_1, m_2 so that $2/m_1 < \min(\alpha^2\beta, \alpha_0^2\beta_0,)$ and $2m_1/m_2 < \alpha^2\beta$. We set the dimension $d = m_2 - 1$ and define $D = \gamma^{-2d^2}$. Summarizing:

- $m_1 = \max(1/2\alpha^2\beta, 1/2\alpha_0^2\beta_0)$

- $m_2 = \frac{1}{2}m_1/\alpha^2\beta$

- $D \le \exp(poly(m_2)) \le \exp(poly(1/\alpha, 1/\beta, 1/\alpha_0, 1/\beta_0))$.

Let X' be the d-dimensional γ-spectral high dimensional expander promised by Theorem 6.2, with $|X'(0)| = n \in [n', Dn']$. The theorem states it can be constructed in $poly(n)$ time. Let $X = (V_2, V_1, V_0)$ be the double sampler defined by $V_2 = X'(d), V_2 = X'(m_1)$ and $V_0 = X'(0)$, with the distribution $\Pi_1 = \mathcal{D}_{m-1}$ and uniform distributions on V_2, V_0.

From Theorem 6.2, each $s \in X'(0)$ is in at most D d-faces, which means that the size of V_2, V_1 is bounded by Dn. Moreover, the distributions $\mathcal{D}_d, \mathcal{D}_0$ are uniform, and $\mathcal{D}_{m_1} = \Pi_1$ has irregularity at most D (as each $S \in X'(V_1)$ is contained in at most D d-faces).

Lemma 6.3 proves that the bipartite graph $G_{2,1}$ between V_2 and V_1 is a spectral expander $\lambda_2^{bip}(G_{2,1})^2 \le m_1/m_2 + O(m_1 m_2 \gamma)$. Our choice of m_1, m_2, γ promises that $\lambda_2^{bip}(G_{2,1})^2 \le \frac{1}{2}\alpha\beta^2$. According to Claim 7.2, $G_{2,1}$ is an (α, β)-sampler.

We are left with proving the local sampling property, i.e. showing that for each $T \in V_2$, the graph $X_{|T}$ is an (α_0, β_0) sampler. $X_{|T}$ is the bipartite graph whose one side is all subsets of T of size m_1 and whose other side is all of the elements of T. Each subset is connected to its m_1 elements, so it is bi-regular and has uniform distribution. The claim on the eigenvalue follows either by invoking again the HDX machinery, or by a more direct argument, from considering the two-step walk and noticing it is a convex combination of the identity matrix with probability $1/m_1 \cdot (m_2 - m_1)/(m_2 - 1)$ and the all-ones matrix (normalized) with remaining probability. Our choice of m_1, m_2 promises that $\lambda(X_{|T}) \le \frac{1}{2}\alpha_0^2\beta_0$, so by Claim 7.2 is it an (α_0, β_0) sampler. $\qquad\square$

7 Relation Between Sampler and Expansion

In this section we look at sampler and expander graphs, show when a bipartite expander is also a sampler, and how can sampler derive an expander graph. The main difference between a sampler graph and an expander is that expanders are a *worst case* definition, whereas samplers allow exceptions from the expansion requirements.

7.1 From spectral gap to a sampler

We show that every bipartite expander graph is also a sampler graph, and calculate the relation between the expansion and the sampler parameters.

We use a variant of the expander mixing lemma from [DK17], Claim 7.1 below, to deduce the sampler property from the spectral gaps. The proof of this claim is very similar to the proof of the expander mixing lemma.

Claim 7.1. *[DK17, Proposition 2.8] Let $(G = (U, V, E), W)$ be a weighted bipartite graph with edge weights $W = \{w_e\}_{e \in E}$, let $f : V \to [0,1]$ and $g : U \to [0,1]$. Then*

$$\left| \underset{(u,v) \sim E}{\mathbb{E}}[f(v)g(u)] - \mathbb{E}[f]\,\mathbb{E}[g] \right| \le \lambda_2^{\mathrm{bip}}(G)\sqrt{\mathbb{E}[f]\,\mathbb{E}[g]}.$$

Where the expectations are over the weights of the edges and vertices.

Claim 7.2. *A weighted bipartite graph $(G = (U, V, E), W)$ with $\lambda = \lambda_2^{\mathrm{bip}}(G)$ is an $(\alpha, \frac{2\lambda^2}{\alpha^2})$ sampler. In other words, to get an (α, β) sampler, it suffices to take a graph with $\lambda < \frac{1}{2}\alpha\sqrt{\beta}$.*

Proof. Let $f : V \to [0,1]$ have $\mathbb{E}[f] = \eta$. Let A be the set of vertices that see too little of f

$$A = \left\{ u \in U \,\Big|\, \underset{v \sim u}{\mathbb{E}}[f(v)] < \eta - \alpha \right\},$$

recall that $v \sim u$ is a random neighbor of u. Similarly, let B be the set of vertices that see too much of f,

$$B = \left\{ u \in U \,\Big|\, \underset{v \sim u}{\mathbb{E}}[f(v)] > \eta + \alpha \right\}.$$

We will show $\Pr[A] + \Pr[B] \le 2\lambda^2\eta/\alpha^2$. Write

$$(\eta + \alpha)\Pr[B] \le \underset{(u,v) \sim W}{\mathbb{E}}[f(v)1_B(u)] \le \mathbb{E}[f]\Pr[B] + \lambda\sqrt{\mathbb{E}[f]\Pr[B]}$$

where the first inequality is by definition of B and the second inequality is relying on Claim 7.1. Dividing both sides by $\sqrt{\mathbb{E}[f]\Pr[B]}$ and rearranging, we get $\frac{\Pr[B]}{\mathbb{E}[f]} \le \lambda^2/\alpha^2$ so $\Pr[B] \le \lambda^2\eta/\alpha^2$.

Similarly for A, by Claim 7.1

$$|\mathbb{E}[f]\Pr[A] - \underset{(u,v) \sim W}{\mathbb{E}}[f(v)1_A(u)]| \le \lambda\sqrt{\mathbb{E}[f]\Pr[A]}$$

so

$$\mathbb{E}[f]\Pr[A] - \lambda\sqrt{\mathbb{E}[f]\Pr[A]} \le \underset{(u,v) \sim W}{\mathbb{E}}[f(v)1_A(u)] \le (\eta - \alpha)\Pr[A]$$

and again we get $\Pr[A] \le \lambda^2\eta/\alpha^2$.

The function $f : V \to [0,1]$ has a maximum value 1, so $\mathbb{E}[f] = \eta \le 1$ and $\Pr[A], \Pr[B] \le \frac{\alpha^2\eta}{\lambda^2} \le \frac{\alpha^2}{\lambda^2}$. If we want an (α, β) sampler, we choose $\lambda < \frac{1}{2}\alpha\sqrt{\beta}$ to get $2\frac{\alpha^2}{\lambda^2} < \beta$. $\square$

7.2 From sampler to spectral gap

A sampler graph is not necessarily an expander, an expander graph doesn't have even a single disconnected vertex, whereas a sampler graph can tolerate a small number of less connected vertices. Nevertheless, we prove in Theorem 2.6 that the two-step random walk over a sampler graph contains a large expander. In fact, we prove that if G is the two-step random walk over a sampler graph (see Section 2.4 for the definition), then every large set of vertices in G contains an expanding subgraph.

We restate the theorem for convenience.

Theorem 7.3 (Theorem 2.6 restated)**.** *Let $\alpha, \eta, \beta \in (0, 1)$ be constants such that $\alpha, \beta < \frac{\eta^2}{100}$. Let $(G_{samp} = (V_2, V_1, E_s), W_S)$ be an (α, β) sampler. Let $(G = (V_2, E), W)$ be the two-step walk of G_{samp}. Then for every set $A \subseteq V_2$ with $\mu_G(A) = \eta$, there exists a set $B \subseteq A$ such that:*

- *$\mu_G(B) \geq \frac{\eta}{4}$.*

- *Let G_B be the induced graph of G on B with the same edge weights, then $\lambda_2(G_B) \leq \frac{99}{100}$.*

Furthermore, given A Algorithm 2 finds such set B in polynomial time in $|V|$.

We first present the algorithm, then prove its correctness. The idea of Algorithm 2 is based on [DG18]. In the algorithm, we gradually remove sparse cuts form G_B until reaching an expanding subgraph. We find the sparse cuts using the proof of Cheeger inequality, which is constructive. The proof of the theorem uses the fact the large sets in G expand, see Claim 2.7.

Algorithm 2 (Finding an expanding subgraph)**.** The algorithm receives a graph $(G = (V, E), W)$, and a subset $A \subset V$. The output is a subset $B \subset A$.

Initialization : Set $i = 0$, $A_0 = A$ and let $(G_0 = (A_0, V_0), W_0)$ be the subgraph induced by A_0 with the same edge weights W.

Graph Improvement : While $\lambda_2(G_i) \geq \frac{99}{100}$:

    1. Find a cut $(U_i, A_i \backslash U_i)$ in G_i such that $\mu_{G_i}(E(U_i, A_i \backslash U_i)) \leq \sqrt{2(1 - \lambda(G_i))} \mu_{G_i}(U_i)$. Let U_i be the smaller part of the cut, i.e. $\mu_{G_i}(U_i) \leq \frac{1}{2}$. See [Chu05] for the algorithm.

    2. Set $A_{i+1} = A_i \setminus U_i$ and let $(G_{i+1} = (A_{i+1}, E_{i+1}), W_{i+1})$ be the subgraph induced by A_{i+1} with the same edge weights W.

    3. Increase i to $i + 1$.

Output: $B = A_i$.

Proof. We start with the runtime of the algorithm. Each iteration in the loop runs an algorithm for finding a sparse cut, which takes polynomial time. The number of iterations in the loop is bounded by $|A|$, because for each iteration i, $A_{i+1} \subsetneq A_i$. Therefore, the total runtime of the algorithm is polynomial in $|G|$.

Let l be the number of steps performed by the algorithm. The output of the algorithm is always expanding, i.e. G_l satisfies $\lambda_2(G_l) \leq \frac{99}{100}$. It remains to show that $\mu_G(A_l) \geq \frac{\eta}{4}$.

Assume towards contradiction that $\mu_G(A_l) < \frac{\eta}{4}$, this implies that $\mu_G(U_1 \cup \cdots \cup U_{l-1}) \geq \frac{3}{4}\eta$. The sets U_i are always set to be the smaller part in the partition $(U_i, A_i \setminus U_i)$, so $\mu_{G_i}(U_i) \leq \frac{1}{2}$. In the graph G, this implies that

$$\mu_G(U_i) = \Pr_{u \sim W}[u \in U_i] \leq \Pr_{u \sim W}[u \in A_0] \Pr_{u \sim W}[u \in U_i | u \in A_0] \leq \eta \frac{1}{2}.$$

Therefore, there must be $j \in [l-1]$ such that $\mu_G(U_1 \cup \cdots \cup U_j) \in [\frac{\eta}{4}, \frac{3\eta}{4}]$. Denote $U = U_1 \cup \cdots \cup U_j$.

We show a contradiction by upper bounding and lower bounding the fraction of edges between U and $A_0 \setminus U$ in G.

Lower Bound: From the variant of the expander mixing lemma on G, Claim 2.7

$$\mu_G(E(A_0 \setminus U, U)) \geq (\mu_G(A_0 \setminus U) - \beta)(\mu_G(U) - \alpha) \geq (\frac{\eta}{4} - \beta)(\frac{\eta}{4} - \alpha),$$

where we used the fact that $\mu_G(U), \mu_G(A_0 \setminus U) \in \left[\frac{\eta}{4}, \frac{3\eta}{4}\right]$.

Upper Bound: Let $h = \sqrt{2(1 - \frac{99}{100})}$, in particular $h \leq \frac{1}{20}$. We upper bound the cut $(U, A_0 \setminus U)$ in G by showing that it is contained in the union of all of the cuts used by the algorithm. More explicitly, we show that $E(U, A_0 \setminus U) \subset \bigcup_{i=1}^{j} E(U_i, A_i \setminus U_i)$.

Let $e = (u, v)$ be an edge in $E(U, A_0 \setminus U)$. Then there exists some $i \leq j$ such that $u \in U_i$. By definition, $A_i = A_0 \setminus (\cup_{t<i} U_t)$, so $A_0 \setminus U \subset A_i \setminus U_i$, and $v \in A_i \setminus U_i$.

The cuts $E(U_i, A_i \setminus U_i)$ are disjoint, so the inclusion implies

$$\mu_G(E(U, A_0 \setminus U)) \leq \sum_{i=1}^{j} \mu_G(E(U_i, A_i \setminus U_i)). \tag{6}$$

Each cut $(U_i, A_i \setminus U_i)$ is a sparse cut in G_i, and satisfies $\mu_{G_i}(E(U_i, A_i \setminus U_i)) \leq h\mu_{G_i}(U_i)$. We can translate this inequality into an inequality with μ_G instead of μ_{G_i}, by using the fact that G_i is an induced subgraph of G. Using Claim 7.4 we get,

$$\mu_G(E(U_i, A_i \setminus U_i)) \leq h\mu_{G_i}(U_i) \leq h\mu_G(E(U_i, A_i)).$$

Using this bound in (6), we get

$$\mu_G(E(U, A_0 \setminus U)) \leq \sum_{i=1}^{j} h\mu_G(E(U_i, A_i)) \leq h\mu_G(E(U, A_0)).$$

The last inequality holds since $A_i \subset A_0$ for every i, and U is partitioned into $U_1, \ldots, U_j$. This implies that $\bigcup_{i=1}^{j} E(U_i, A_i) \subset E(U, A_0)$, and that the sets in the union are disjoint.

We use again the variant of the expander mixing lemma on the graph G, Claim 2.7,

$$\mu_G(E(U, A_0)) \leq \mu_G(U) \left(\mu_G(A_0) + \alpha\right) + \beta \leq \frac{3\eta}{4}(\eta + \alpha) + \beta.$$

Which means that

$$\mu_G(E(U, A_0 \setminus U)) \leq h\left(\frac{3\eta}{4}(\eta + \alpha) + \beta\right).$$

Since $h < \frac{1}{20}$ and $\alpha, \beta \leq \frac{\eta^2}{100}$, we reach a contradiction. $\qquad\square$

We are left with proving the "translation" between μ_G and μ_{G_i}.

Claim 7.4. *Let $(G = (V, E), W)$ be a weighted graph, and let $(G' = (V', E'), W')$ be an induced subgraph inheriting the weights of G. Then for every $V'' \subset V', E'' \subset E'$:*

$$\mu_{G'}(E'') = \frac{\mu_G(E'')}{\mu_G(E')}, \quad \mu_{G'}(V'') = \frac{\mu_G(E(V'', V'))}{\mu_G(E')}.$$

Proof. By definition (recall Section 2.1), $\mu_{G'}(E'')$ is the probability to pick a random edge from E'' when picking a random edge in G'. The weights in G' are the same as in G.

$$\mu_{G'}(E'') = \Pr_{e \sim W'}[e \in E''] = \Pr_{e \sim W}[e \in E'' | e \in E'] = \frac{\Pr_{e \sim W}[e \in E'']}{\Pr_{e \sim W}[e \in E']} = \frac{\mu_G(E'')}{\mu_G(E')}.$$

For vertex weights, the weight of every vertex is the sum of its adjacent edges. The weight of a vertex $v \in V''$ in G' is the sum of its adjacent edges in G'.

$$\begin{aligned}
\mu_{G'}(V'') &= \Pr_{(u,v) \sim W'}[u \in V''] \\
&= \Pr_{(u,v) \sim W}[u \in V'' | (u, v) \in E'] \\
&= \Pr_{(u,v) \sim W}[(u, v) \in E(V'', V') | (u, v) \in E'] \\
&= \frac{\mu_G(E(V'', V'))}{\mu_G(E')}.
\end{aligned}$$

$\square$

8 List-Decoding of Unique Games over Expanders

The unique games algorithm takes a solvable unique games instance and outputs a single solution. We want a list-decoding algorithm, so we need a list of all possible solutions. We do so by running the unique games algorithm of [MM10] multiple times, and removing the solution after each time. We restate the theorem for convenience.

Theorem 8.1 (Theorem 2.14 restated). *Let $(G = (V, E), W)$ be a weighted undirected graph with $\lambda_2(G) \leq \frac{99}{100}$. Let $\{\pi_e\}_{e \in E}$ be unique constraints over the edges of G, with ℓ labels.*

Then, there is an absolute constant $c > 1$ and a polynomial time algorithm, Algorithm 3, that on input $(G = (V, E), W), \{\pi_e\}_{e \in E}$ outputs a list of assignments $L = \{a^{(1)}, \ldots, a^{(t)}\}$, with $a^{(i)} : V \to [\ell]$. The list satisfies that for every assignment $a : V \to [\ell]$ that satisfies $1 - \eta$ of the constraints for $\eta < c^{-\ell-1}$, there exists $a^{(i)} \in L$ that satisfies $\Pr_{v \sim W}[a(v) = a^{(i)}(v)] \geq 1 - \eta c^\ell$.

The constant c is derived from the constant of the unique games algorithm from [MM10], see Theorem A.2. For c', C' the constants in Theorem A.2, we set $c = \max\{\frac{100}{c'}, 101(1+50C')\}$.

Algorithm 3 (List decoding unique games)**.** The algorithm receives a weighted constraint graph $G = (V, E), W = \{w_e\}_{e \in E}, \{\pi_e\}_{e \in E}$ and returns a list of assignments $L = \{a^{(1)}, \ldots a^{(t)}\}$, $a^{(i)} : V \to [\ell]$.

Initialization : Set $i = 1$, and set $\pi_e^{(1)} = \pi_e$ for every $e \in E$.

Solving unique constraints : Repeat

1. Use the unique games algorithm from [MM10] (see Theorem A.2) on the graph G with constraints $\{\pi^{(i)}\}_{e \in E}$.

2. If the algorithm didn't return a solution, quit the loop.

3. Otherwise, let $a^{(i)} : V \to [\ell - i + 1]$ to be the solution.

4. Let $\{\pi^{(i+1)}\}_{e \in E}$ be the constraints after removing $a^{(i)}$ (see details after the algorithm). For every edge e, $\pi^{i+1}(e) : [\ell - i + 1] \to [\ell - i + 1]$.

5. Set $i = i + 1$ and repeat.

Output : Output $L = a^{(1)}, \ldots, a^{(i-1)}$, written as assignments from V to $[\ell]$.

Removing the assignment a from $\pi : [j] \to [j]$, getting $\pi' : [j - 1] \to [j - 1]$ is done as follows:

- For every vertex v, reorder the elements such that $a(v) = j$.

- If a satisfies π, i.e. $\pi(j) = j$, then π' is equal to π restricted to $[j - 1]$.

- Otherwise, there exist $i, l \neq j$ such that $\pi(l) = j$ and $\pi(j) = i$. Set $\pi'(l) = i$, and the rest is identical to π.

Transforming $a^{(j)} : V \to [\ell - j + 1]$ to $a^{(j)} : V \to [\ell]$ is done by reversing the permutations done in the iterations of the loop. On each iteration of the loop, the algorithm removes the previous solution from all of the constraints by reordering the elements in $[\ell]$. In the output, the algorithm translates back each solution to be $a^{(j)} : V \to [\ell]$ by reversing the order changes.

Before proving Theorem 2.14, we prove the following simple claim. If a, a' are two assignments satisfying almost all of the constraints in an expander graphs, then they must be either almost identical or completely different.

Claim 8.2. *Let G be a graph with $\lambda_2(G) \leq \frac{99}{100}$, and let $a, a' : V \to [\ell]$ be two assignments satisfying $1 - \eta$ and $1 - \eta'$ of the constraints in G. Then, either $\Pr_{v \in V}[a(v) = a'(v)] \geq 1 - 50(\eta' + \eta)$ or $\Pr_{v \in V}[a(v) = a'(v)] \leq 50(\eta' + \eta)$.*

Proof. Let $D \subset E$ be the set of disagreeing vertices,

$$D = \left\{ v \in V \mid a(v) \neq a'(v) \right\}.$$

The constraints in G are unique, so for every edge $(v_1, v_2) \in E$ if both a, a' satisfies the edge constraint, and $a(v_1) = a'(v_1)$, then it must be that $a(v_2) = a'(v_2) = \pi_{(1,2)}(a(v_1))$.

Therefore, if an edge (v_1, v_2) has $v_1 \notin D, v_2 \in D$, it is not possible that both a, a' satisfy it. This gives a bound on the cut $D, V \setminus D$,

$$\mu(E(D, V \setminus D)) \leq \eta + \eta'.$$

The second largest eigenvalue of G is at most $\frac{99}{100}$, so its edge expansion is at least $\frac{1}{50}$. By Cheeger inequality.

$$\mu(E(D, V \setminus D)) \geq \frac{1}{50} \min\{\mu(D), \mu(V \setminus D)\}.$$

This means that $\min\{\mu(D), \mu(V \setminus D)\} \leq 50(\eta + \eta')$, which finishes the proof. $\qquad\square$

Proof of Theorem 2.14. Let $a : V \to [\ell]$ be an assignment satisfying $1 - \eta$ of the constraints of (G, W). For c', C' the constants in Theorem A.2, we set $c = \max\{\frac{100}{c'}, 101(1 + 50C')\}$.

Denote by t the number of solutions in L, when running on $(G, W), \{\pi_e\}_{e \in E}$. For every $i \in [t]$, recall $\{\pi_e^{(i)}\}_{e \in E}$ are the constraints used in the ith step of the algorithm runtime. Let η_i be the fraction of the constraints in the ith round unsatisfied by the assignment a:

$$\eta_i = \Pr_{(u,v) \sim E}[a(u) \neq \pi_{u,v}^{(i)}(a(v))].$$

In the following claim we show that if $a^{(i)}$ is very different than a, then after removing $a^{(i)}$ the assignment a still satisfies a large fraction of the new constraints $\pi^{(i+1)}$.

Claim 8.3. *If* $\eta_i \leq \frac{100}{c}$ *and* $\Pr_u[a(u) = a^{(i)}(u)] \leq \frac{1}{2}$, *then* $\eta_{i+1} \leq c\eta_i$.

Proof. The assignment a on (G, W) with constraints $\pi^{(i)}$, satisfies $1 - \eta_i$ of the constraints. If $\eta_i \leq \frac{100}{c}$, then the unique games algorithm, Theorem A.2, outputs an assignment $a^{(i)}$ satisfying at least $1 - 50C'\eta_i$ of the constraints.

The constraints $\pi^{(i+1)}$ are created from $\pi^{(i)}$ by removing the labels of $a^{(i)}$. For every edge (u, v), if $a, a^{(i)}$ differs on *both* endpoints of the edge, then removing $a^{(i)}$ won't "ruin" the constrain for a.

This gives us a bound on η_{i+1},

$$\begin{aligned}
\eta_{i+1} &= \Pr_{(u,v) \sim E}[a(u) \neq \pi_{u,v}^{(i+1)}(a(v))] \\
&\leq \Pr_{(u,v) \sim E}[a(u) \neq \pi_{u,v}^{(i)}(a(v))] + \Pr_{(u,v) \sim E}[a(u) = a^{(i)}(u) \vee a(v) = a^{(i)}(v)] \\
&\leq \eta_i + 100(\eta_i + 50C'\eta_i) \leq c\eta_i,
\end{aligned}$$

where the last inequality is by applying Claim 8.2 on the assignments $a, a^{(i)}$. The constant c such that $c > 101(1 + 50C')$. $\square$

We now show that $L = \{a^{(i)}, \ldots a^{(t)}\}$ contains an assignment close to a. Assume towards contradiction that for all $i \in [t]$, $\Pr_u[a(u) = a^{(i)}(u)] \leq \frac{1}{2}$, then by the above claim

$$\eta_t = \Pr_{(u,v) \sim E}[a(u) \neq \pi_{u,v}^{(t)}(a(v))] \leq c\eta_{t-1} \leq c^2\eta_{t-2} \leq \cdots \leq c^t\eta.$$

Since $\eta < c^{-\ell-1}$, $\eta c^\ell \leq \frac{1}{c} \leq \frac{c'}{100}$ the unique games algorithm should have outputted a solution and not stopped at t, reaching a contradiction.

Therefore, there must be $i \in [t]$ such that $\Pr_u[a(u) = a^{(i)}(u)] > \frac{1}{2}$ and for each $i' < i$, $\Pr_u[a(u) = a^{(i')}(u)] < \frac{1}{2}$. The claim above implies that $\eta_i \leq c^{i-1}\eta$. The unique games algorithm outputs $a^{(i)}$ which satisfies at least $1 - 50C'\eta_i \geq 1 - 50C'\eta_i$ of the constraints, so by Claim 8.2

$$\Pr_u[a(u) \neq a^{(j)}(u)] \leq 50(\eta_i + 50C\eta_i) \leq c^i\eta_i \leq c^\ell\eta.$$

which finishes the proof. $\square$

References

[ABNNR92] Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ron M. Roth. *Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs*. IEEE Trans. Inform. Theory, 38(2):509–516, 1992. 3, 19

[AJQST20] Vedat Levi Alev, Fernando Granha Jeronimo, Dylan Quintana, Shashank Srivastava, and Madhur Tulsiani. *List decoding of direct sum codes*. In *Proc. 31st Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1412–1425. 2020. `arXiv:2011.05467`. 5, 8

[AKKSTV08] Sanjeev Arora, Subhash Khot, Alexandra Kolla, David Steurer, Madhur Tulsiani, and Nisheeth K. Vishnoi. *Unique games on expanding constraint graphs are easy: extended abstract*. In *Proc. 40th ACM Symp. on Theory of Computing (STOC)*, pages 21–28. 2008. 6, 16, 17, 49

[Chu05] Fan R. K. Chung. *Laplacians and the Cheeger inequality for directed graphs*. Ann. Comb., 9(1):1–19, 2005. 40

[CL06] Fan R. K. Chung and Lincoln Lu. *Concentration inequalities and martingale inequalities: A survey*. Internet Math., 3(1):79–127, 2006. 10

[CMM06] Eden Chlamtac, Konstantin Makarychev, and Yury Makarychev. *How to play unique games using embeddings*. In *Proc. 47th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 687–696. 2006. 50

[DG18] Irit Dinur and Yoav Gelberg. *A robustness result for expanders*, 2018. (unpublished). 40

[DHKNT19] Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon TaShma. *List decoding with double samplers*. In *Proc. 30th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 2134–2153. 2019. `arXiv:1808.00425`, `eccc:2018/TR18-198`. 1

[DK17] Irit Dinur and Tali Kaufman. *High dimensional expanders imply agreement expanders*. In *Proc. 58th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 974–985. 2017. `eccc:2017/TR17-089`. 2, 7, 8, 37, 39

[FI20] Ehud Friedgut and Yonatan Iluz. *Hyper-regular graphs and high dimensional expanders*, 2020. (manuscript). `arXiv:2010.03829`. 16, 35

[GI01] Venkatesan Guruswami and Piotr Indyk. *Expander-based constructions of efficiently decodable codes*. In *Proc. 42nd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 658–667. 2001. 5

[GI03] ———. *Linear time encodable and list decodable codes*. In *Proc. 35th ACM Symp. on Theory of Computing (STOC)*, pages 126–135. 2003. 5

[GI05]   ———. *Linear-time encodable/decodable codes with near-optimal rate*. IEEE Trans. Inform. Theory, 51(10):3393–3400, 2005. (Preliminary version in *34th STOC*, 2002). 4, 5

[GR08]   VENKATESAN GURUSWAMI and ATRI RUDRA. *Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy*. IEEE Trans. Inform. Theory, 54(1):135–150, 2008. (Preliminary version in *38th STOC*, 2006). arXiv:cs/0511072, eccc:2005/TR05-133. 5

[GRS]   VENKATESAN GURUSWAMI, ATRI RUDRA, and MADHU SUDAN. *Essential coding theory*. (draft of book). 17

[GS99]   VENKATESAN GURUSWAMI and MADHU SUDAN. *Improved decoding of Reed-Solomon and algebraic-geometry codes*. IEEE Trans. Inform. Theory, 45(6):1757–1767, 1999. (Preliminary version in *39th FOCS*, 1998). eccc:1998/TR98-043. 5

[GX14]   VENKATESAN GURUSWAMI and CHAOPING XING. *Optimal rate list decoding of folded algebraic-geometric codes over constant-sized alphabets*. In *Proc. 25th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1858–1866. 2014. eccc:2013/TR13-046. 5

[IJKW10]   RUSSELL IMPAGLIAZZO, RAGESH JAISWAL, VALENTINE KABANETS, and AVI WIGDERSON. *Uniform direct product theorems: Simplified, optimized, and derandomized*. SIAM J. Comput., 39(4):1637–1665, 2010. (Preliminary version in *40th STOC*, 2008). eccc:2008/TR08-079. 5, 7, 9, 19

[IKW12]   RUSSELL IMPAGLIAZZO, VALENTINE KABANETS, and AVI WIGDERSON. *New direct-product testers and 2-query PCPs*. SIAM J. Comput., 41(6):1722–1768, 2012. (Preliminary version in *41st STOC*, 2009). eccc:2009/TR09-090. 7, 9

[Imp03]   RUSSELL IMPAGLIAZZO. *Hardness as randomness: A survey of universal derandomization*. In *Proc. ICM 2002, Beijing*, volume 3, pages 659–672. 2003. arXiv:cs/0304040. 8

[JST21]   FERNANDO GRANHA JERONIMO, SHASHANK SRIVASTAVA, and MADHUR TULSIANI. *Near-linear time decoding of ta-shma's codes via splittable regularity*. In *Proc. 53rd ACM Symp. on Theory of Computing (STOC)*. 2021. To appear. 5, 6, 9

[KMRS17]   SWASTIK KOPPARTY, OR MEIR, NOGA RON-ZEWI, and SHUBHANGI SARAF. *High-rate locally correctable and locally testable codes with sub-polynomial query complexity*. J. ACM, 64(2):11:1–11:42, 2017. (Preliminary version in *48th STOC*, 2016). arXiv:1504.05653, eccc:2015/TR15-068. 4

[KO18]   TALI KAUFMAN and IZHAR OPPENHEIM. *Construction of new local spectral high dimensional expanders*. In *Proc. 50th ACM Symp. on Theory of Computing (STOC)*, pages 773–786. 2018. arXiv:1710.05304. 2

[KO20]   ———. *High order random walks: Beyond spectral gap*. Combinatorica, 40(1):245–281, 2020. (Preliminary version in *20th RANDOM*, 2018). `arXiv: 1707.02799`. 7, 37

[Kop15]   Swastik Kopparty. *List-decoding multiplicity codes*. Theory of Computing, 11:149–182, 2015. `eccc:2012/TR12-044`. 5

[Lev87]   Leonid A. Levin. *One-way functions and pseudorandom generators*. Combinatorica, 7(4):357–363, 1987. (Preliminary version in *17th STOC*, 1985). 8

[LSV05]   Alexander Lubotzky, Beth Samuels, and Uzi Vishne. *Explicit constructions of Ramanujan complexes of type $\tilde{A}_d$*. European J. Combin., 26(6):965––993, 2005. `arXiv:math/0406217`. 2, 8, 35, 36

[MM10]   Konstantin Makarychev and Yury Makarychev. *How to play unique games on expanders*. In *Proc. 8th International Workshop Approx. & Online Algorithms (WAOA)*, pages 190–200. 2010. `arXiv:0903.0367`, `eccc:2009/ TR09-021`. 6, 16, 17, 27, 34, 42, 43, 48, 49, 50, 51, 52, 53, 54

[PV05]   Farzad Parvaresh and Alexander Vardy. *Correcting errors beyond the Guruswami-Sudan radius in polynomial time*. In *Proc. 46th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 285–294. 2005. 5

[Sud97]   Madhu Sudan. *Decoding of Reed-Solomon codes beyond the error-correction bound*. J. Complexity, 13(1):180–193, 1997. (Preliminary version in *37th FOCS*, 1996). 5

[Sud01]   ———. *6.897: Algorithmic introduction to coding theory*, 2001. (A course on coding theory at MIT, Fall 2001). 17, 18

[TaS17]   Amnon TaShma. *Explicit, almost optimal, epsilon-balanced codes*. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 238–251. 2017. `eccc:2017/TR17-041`. 5, 6, 9

[Tre03]   Luca Trevisan. *List-decoding using the XOR lemma*. In *Proc. 44th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 126–135. 2003. `eccc: 2003/TR03-042`. 5

[Tre05]   ———. *On uniform amplification of hardness in NP*. In *Proc. 37th ACM Symp. on Theory of Computing (STOC)*, pages 31–38. 2005. 8

[Yao82]   Andrew Chi-Chih Yao. *Theory and applications of trapdoor functions (extended abstract)*. In *Proc. 23rd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 80–91. 1982. 8

[Zuc97]   David Zuckerman. *Randomness-optimal oblivious sampling*. Random Structures Algorithms, 11(4):345–367, 1997. (Preliminary version in *27th STOC*, 1996). 2

A   A Unique Games Algorithm Over Weighted Graphs

Our starting point is the following theorem from [MM10].

Theorem A.1 (Theorem 10, [MM10]). *There exists a polynomial time approximation algorithm that given a $1 - \delta$ satisfiable instance of unique games on a d-regular expander graph G with $\frac{\delta}{\lambda_G} \leq c$, the algorithm finds a solution of value*

$$1 - C\frac{\delta}{h_G},$$

where c and C are some positive absolute constants, λ_G is the laplacian second smallest eigenvalue and h_G is the edge expansion.

In this section we show that the theorem holds also for non-regular weighted graph.

Theorem A.2 (Weighted unique games). *There exists a polynomial time approximation algorithm that given a $1 - \delta$ satisfiable instance of unique games on a weighted expander graph (G, W) such that $\frac{\delta}{\lambda_G} \leq c$, the algorithm finds a solution of value*

$$1 - C\frac{\delta}{h_G},$$

where c and C are some positive absolute constants, λ_G is the laplacian second smallest eigenvalue and h_G is the edge expansion.

The laplacian smallest non-zero eigenvalue of a graph G is the eigenvalue gap $(1 - \lambda_2(G))$, where λ_2 is defined in Section 2.2.

We prove Theorem A.2 by following the algorithm and proof in [MM10] and modifying the parts which are different in the case of a non-regular weighted graph.

The algorithm in [MM10] starts from defining an SDP relaxation of the unique games instance. We follow their algorithm and do the same, only in our SDP the target function uses the edge weighted $w_{u,v}$.

For each vertex $u \in V$ and label $i \in [\ell]$ we define a vector u_i of length t.

Definition A.3 (SDP relaxation). *Minimize:*

$$\frac{1}{\omega} \sum_{(u,v) \in E} w_{u,v} \sum_{i \in [\ell]} \|u_i - v_{\pi_{u,v}(i)}\|^2$$

Subject to

$$\forall u \in V, i \neq j \in [\ell], \quad \langle u_i, u_j \rangle = 0 \tag{7}$$

$$\forall u \in V, \quad \sum_{i \in [\ell]} \|u_i\|^2 = 1 \tag{8}$$

$$\forall u, v, x \in V, i, j, l \in [\ell] \quad \|u_i - x_l\|^2 \leq \|u_i - v_j\|^2 + \|v_j - x_l\|^2 \tag{9}$$

$$\forall u, v \in V, i, j \in [\ell] \quad \|u_i - v_j\|^2 \leq \|u_i\|^2 + \|v_j\|^2 \tag{10}$$

$$\forall u, v \in V, i, j \in [\ell] \quad \|u_i\|^2 \leq \|u_i - v_j\|^2 + \|v_j\|^2 \tag{11}$$

Where $\omega = \sum_u w_u$.

An integral solution sets for each $u \in V$ a label $i \in [\ell]$. It translates into vectors by setting $u_i = \mathbf{1}\frac{1}{\sqrt{t}}$, and for each $j \neq i, u_j = \mathbf{0}$, where $\mathbf{1}, \mathbf{0}$ are the all 1 and all 0 vectors, respectively. Each integral solution satisfies all of the constraints, so the SDP value is at least the value of the unique games instance. The algorithm of Makarychev and Makarychev solves the above SDP, and then rounds the SDP solution to get an integral solution with high value. To prove the correctness of the algorithm, we prove that the output of the rounding algorithm has value at least $1 - \frac{C}{h_G}\delta$.

Before presenting the rounding algorithm, we define the the earthmover distance, similarly to [AKKSTV08, MM10].

Definition A.4. *For every two sets of orthogonal vectors $\{u_i\}_{i \in [\ell]}, \{v_i\}_{i \in [\ell]}$ let*

$$\Delta(\{u_i\}_{i \in [\ell]}, \{v_i\}_{i \in [\ell]}) = \min_{\tau \in \mathcal{S}_\ell} \left\{ \sum_{i \in [\ell]} \|u_i - v_{\tau(i)}\|^2 \right\},$$

where $\mathcal{S}_\ell$ are all permutation over ℓ elements.

For two vertices u, v, with the sets of orthogonal vectors $\{u_i\}_{i \in [\ell]}, \{v_i\}_{i \in [\ell]}$, a small distance $\Delta(\{u_i\}_{i \in [\ell]}, \{v_i\}_{i \in [\ell]})$ means that the sets of vectors are correlated (using a permutation τ), i.e. for every vector u_i there is a vector v_j which is close to it.

For an SDP solution $\{u_i\}_{u \in V, i \in [\ell]}$, we denote by $\Delta(u, v)$ the earthmover distance between the vectors of u and the vectors of v.

Arora et al. [AKKSTV08] showed that on d-regular unweighted graphs, the SDP solution has a small average earthmover distance. More explicitly, that for the SDP solution $\{u_i\}_{u \in V, i \in [\ell]}$, the expression $\mathbb{E}_{u,v \in V}[\Delta(\{u_i\}_{i \in [\ell]}, \{v_i\}_{i \in [\ell]})]$ is small. The proof in [AKKSTV08] has a lemma and a corollary, the lemma is general for any SDP solution and is not related to the graph. The corollary uses the graph regularity but can be easily modified to hold for weighted graphs as well.

Lemma A.5 (Lemma 2.2 in [AKKSTV08]). *For every positive even integer q and every SDP solution $\{u_i\}_{u \in V, i \in [\ell]}$, there exists a set of vectors $\{\boldsymbol{V}_u\}_{u \in V}$ that for every pair $u, v \in V$,*

$$\frac{1}{q}\|\boldsymbol{V}_u - \boldsymbol{V}_v\|^2 \leq \frac{1}{\ell}\Delta(u, v) \leq 2\|\boldsymbol{V}_u - \boldsymbol{V}_v\|^2 + O\left(2^{-\frac{q}{2}}\right).$$

We prove the following corollary, it is the same as the corollary in [AKKSTV08] only for weighted graph. The proof of the corollary is also almost the same.

Corollary A.6. *For every constant $R \in (0, 1)$, there exists a positive $c > 0$ such that for any $1 - \delta$ satisfiable instance of unique games on G, if $\frac{\delta}{\lambda_G} < c$, then*

$$\mathbb{E}_{u,v \in V}[\Delta(u, v)] \leq R,$$

For u, v distributed according to their weight, and $\lambda_G = 1 - \lambda(G)$ the second smallest eigenvalue of the normalized laplacian.

Proof. By Claim A.15, the second smallest eigenvalue of the laplacian of G can also be represented by

$$\lambda_G = \min_{\{z_u\}_{u \in V}} \frac{\mathbb{E}_{(u,v) \in E}[\|z_u - z_v\|^2]}{\mathbb{E}_{u,v \in V}[\|z_u - z_v\|^2]} \ , \tag{12}$$

where $\{z_u\}_{u \in V}$ is a set of vectors, one for every vertex, and the expectation is done according to the edge and vertex weights in G.

$$
\begin{aligned}
\mathbb{E}_{u,v \in V}[\Delta(u,v)] &\leq 2\ell \mathbb{E}_{u,v \in V}\left[\|\boldsymbol{V}_u - \boldsymbol{V}_v\|^2\right] + \ell O\left(2^{-\frac{q}{2}}\right) && \text{(by Lemma A.5)} \\
&\leq \frac{2\ell}{\lambda_G} \mathbb{E}_{(u,v) \in E}\left[\|\boldsymbol{V}_u - \boldsymbol{V}_v\|^2\right] + \ell O\left(2^{-\frac{q}{2}}\right) && \text{(by (12))} \\
&\leq \frac{2q\ell}{\lambda_G} \mathbb{E}_{(u,v) \in E}[\Delta(u,v)] + \ell O\left(2^{-\frac{q}{2}}\right) && \text{(by Lemma A.5)} \\
&\leq \frac{2q\ell}{\lambda_G}\varepsilon + \ell O\left(2^{-\frac{q}{2}}\right). && \text{(by the SDP solution)}
\end{aligned}
$$

Taking large enough q such that $\ell O\left(2^{-\frac{q}{2}}\right) < \frac{R}{2}$ and $c < \frac{R}{4q\ell}$ (R, ℓ, q are all constants), we finish the proof. $\square$

We present the rounding algorithm. The only difference between our rounding and the rounding in [MM10] is that in our case the initial vertex u is picked according to its weight. The proof of correctness is also very similar.

The input is an SDP solution $\{u_i\}_{u \in V, i \in [\ell]}$, the output is an assignment $a : V \to [\ell]$.

Initialization :

1. Pick a random vertex $u \in V$ according to the vertex weights w_u.
2. Pick a random label $i \in [\ell]$, each with probability $\|u_i\|^2$.
3. Pick a random number $t \in [0, \|u_i\|^2]$.
4. Pick a random $r \in [R, 2R]$.
5. Obtain vectors $\{\tilde{u}_i\}_{u \in V, i \in [\ell]}$.

Labels Assignment : For every $v \in V$:

1. Let $S_v = \left\{p \in [\ell] \,\middle|\, \|v_p\|^2 \geq t, \|\tilde{u}_i - \tilde{v}_p\|^2 \leq r\right\}$.
2. If $S_v = \{p\}$, assign the label p to v. Else, assign an arbitrary one.

The vectors $\{\tilde{u}_i\}_{u \in V, i \in [\ell]}$ are a normalized version of the vectors $\{u_i\}_{u \in V, i \in [\ell]}$ that are promised from Lemma A.7. This lemma appears as lemma 1 in [MM10], and is actually proven in [CMM06]. It is a general claim about vectors normalization, and is not related to any graph, therefore it holds for the solution of the SDP on the weighted graph as well.

Lemma A.7 (Lemma 1 from [MM10], proven in [CMM06].)**.** *For every SDP solution* $\{u_i\}_{u \in V, i \in [\ell]}$, *there exists a set of vectors* $\{\tilde{u}_i\}_{u \in V, i \in [\ell]}$ *satisfying the following properties:*

1. *Triangle inequalities: for every $u, v, w \in V$ and labels $i, j, l \in [\ell]$:*

$$\|\tilde{u}_i - \tilde{v}_j\| + \|\tilde{v}_j - \tilde{w}_l\| \leq \|\tilde{u}_i - \tilde{w}_l\|.$$

2. *For every $u, v \in V, i.j \in [\ell]$,*

$$\langle \tilde{u}_i, \tilde{v}_j \rangle = \frac{\langle u_i, v_j \rangle}{\max\{\|u_i\|^2, \|v_j\|^2\}}.$$

3. *For all non-zero vectors u_i, $\|\tilde{u}_i\| = 1$.*

4. *For every $u \in V, i \neq j \in [\ell]$, $\langle \tilde{u}_i, \tilde{u}_j \rangle = 0$.*

5. *For every $u, v \in V, i, j \in [\ell]$,*

$$\|\tilde{v}_j - \tilde{u}_i\| \leq \frac{2\|v_j - u_i\|}{\max\{\|u_i\|^2, \|v_j\|^2\}}.$$

The set of vectors $\{\tilde{u}_i\}_{u \in V, i \in [\ell]}$ can be obtained in polynomial time.

We prove the correctness of the algorithm. Let $(G = (V, E), W = \{w_{u,v}\}_{(u,v) \in E})$ and $\{\pi_{u,v}\}_{(u,v) \in E}$, s.t. $\pi_{u,v} : [\ell] \to [\ell]$ be a unique games instance on a weighted graph, that has a solution satisfying $1 - \delta$ fraction of the constraints (where the fraction is weighted). Then the SDP solution $\{u_i\}_{u \in V, i \in [\ell]}$ also has value at most δ.

We start from a few definitions.

Definition A.8. *Let $\tau_{x,v}$ be the partial mapping from $[\ell]$ to $[\ell]$ which maps p into q if $\|\tilde{v}_p - \tilde{x}_q\| \leq 4R$.*

The function is well defined, because $\{\tilde{u}_i\}_{u \in V, i \in [\ell]}$ are orthogonal and satisfy the triangle inequality. It is not possible that $\|\tilde{v}_p - \tilde{x}_q\| \leq 4R$ and $\|\tilde{v}_p - \tilde{x}_{q'}\| \leq 4R$, as it implies that $\|\tilde{x}_q - \tilde{x}_{q'}\| \leq 8R$, but $\tilde{x}_q, \tilde{x}_{q'}$ are orthogonal.

Definition A.9. *Let $X = \{x \in V \mid |S_x| = 1\}$.*

In the proof we use the following claims from [MM10]. The claims are unrelated to the graph structure, and holds for weighted graphs as well.

1. If $p \in S_v$ and $q \in S_x$ with non-zero probability for the same initial vertex and label, then $\tau_{v,x}(p) = q$.

2. $|S_v| \leq 1$.

3. If $S_v = \{p\}$, then $S_x = \{\tau_{x,v}(p)\}$ or $S_w = \emptyset$.

4. For every choice of initial vertex u, every $v \in V, p \in [\ell]$, $\Pr_{t,r}[S_v = \{p\}] \leq \|v_p\|^2$.

We reprove a weighted variant of the following lemmas from [MM10]. The Lemmas and proofs are very similar to those in [MM10], the main difference is that in the weighted case the distribution is over the weights of the edges and vertices.

Lemma A.10 (a variant of Lemma 5 in [MM10]). *If $\frac{\varepsilon}{\lambda_G} < c$, then $\mathbb{E}[\mu(X)] \geq \frac{1}{4}$.*

Proof. Suppose u is the initial vertex, then for every $v \in V$ we express the probability of $v \in X$ using $\Delta(u, v)$.

For every label $p \in [\ell]$, if $\exists q \in [\ell]$ such that $\|u_q - v_p\|^2 \leq \frac{R}{2}\|v_p\|^2$, then

$$\|\tilde{v}_p - \tilde{u}_q\| \leq \frac{2\|u_q - v_p\|}{\max\{\|u_q\|^2, \|v_p\|^2\}} \leq R \leq r,$$

which implies $\tau_{u,v}(p) = q$. In this case, if q is the initial label and $t \leq \|v_p\|^2$, then $S_v = \{p\}$ which implies $v \in X$. Therefore,

$$\Pr_{i,t}[S_v = \{p\}] \geq \Pr_{i,t}[i = q \wedge t \leq \|v_p\|^2] \tag{13}$$

$$= \|u_q\|^2 \min\{1, \frac{\|v_p\|^2}{\|u_q\|^2}\} = \min\{\|u_q\|^2, \|v_p\|^2\} \geq \frac{1}{2}\|v_p\|^2. \tag{14}$$

The last inequality is by the triangle inequality, using the fact that $\|u_q - v_p\|^2 \leq \frac{R}{2}\|v_p\|^2$.

Going over all possible labels p for v:

$$\Pr[v \in X] = \Pr_{i,t,r}[\exists p \text{ s.t. } S_v = \{p\}]$$

$$= \sum_p \Pr_{i,t,r}[S_v = \{p\}]$$

$$\geq \sum_{p \text{ s.t. } \exists q, \|u_q - v_p\|^2 \leq \frac{R}{2}\|v_p\|^2} \frac{1}{2}\|v_p\|^2$$

$$\geq \sum_p \frac{1}{2}\|v_p\|^2 - \sum_{p \text{ s.t. } \forall q, \|u_q - v_p\|^2 > \frac{R}{2}\|v_p\|^2} \frac{1}{2}\|v_p\|^2$$

$$\geq \frac{1}{2} - \frac{1}{2}\sum_p \frac{2}{R}\min_q\{\|v_p - u_q\|^2\} \qquad \text{(since for all } q, \|v_p\|^2 < \frac{2}{R}\|v_p - u_q\|^2)$$

$$= \frac{1}{2} - \frac{1}{R}\Delta(u, v).$$

By the earthmover distance lemma, $\mathbb{E}_{u,v \in V}[\Delta(u, v)] < R$, when u, v are distributed according to their weight in the graph, so

$$\mathbb{E}[\mu(X)] = \sum_{u,v \in V} \mu(u)\mu(v)\Pr[v \in X | u \text{ initial vertex}]$$

$$\geq \sum_{u,v \in V} \mu(u)\mu(v)\left(\frac{1}{2} - \frac{1}{R}\Delta(u, v)\right)$$

$$\geq \frac{1}{2} - \frac{1}{R}\mathbb{E}_{u,v \in V}[\Delta(u, v)]$$

$$\geq \frac{1}{2} - \frac{1}{4}.$$

$\square$

Corollary A.11.

$$\Pr[\mu(X) \geq \frac{1}{8}] \geq \frac{1}{8}.$$

Lemma A.12 (a variant of Lemma 7 in [MM10])**.**

$$\mathbb{E}[\mu(X, V \setminus X)] \leq \frac{6\delta}{R}.$$

Proof. Fix $u \in V$ the initial vertex, we bound the probability of $v \in X, x \notin X$ by $\frac{6}{R} \sum_p \|v_p - x_{\pi_{x,v}(p)}\|^2$.

If $v \in X, x \notin X$, then $S_v = \{p\}, S_x = \emptyset$. Let $q = \pi_{x,v}(p)$. Since $S_v = \{p\}$, then $\|v_p\|^2 \geq t, \|\tilde{u}_i - \tilde{v}_p\|^2 \leq r, i = \tau_{u,v}(p)$. One of the two cases must happen

1. $\|x_q\|^2 < t$.

2. $\|x_q\|^2 \geq t, \|\tilde{x}_q - \tilde{u}_i\|^2 > r$.

We sum over all p the probability that these events occur (each p has a $q = \pi_{x,v}(p)$).

$$\Pr_{i,t,r}[1] \leq \sum_p \Pr[i = \sigma_{v,u}(p)] \Pr[\|x_q\|^2 < t \leq \|v_p\|^2 | i = \sigma_{v,u}(p)]$$

$$\leq \sum_p \|u_{\sigma_{v,u}(p)}\|^2 \frac{\|v_p\|^2 - \|x_q\|^2}{\|u_{\sigma_{v,u}(p)}\|^2}$$

$$\leq \sum_p \left(\|v_p\|^2 - \|x_q\|^2\right).$$

$$\Pr_{i,t,r}[2] = \sum_p \Pr[i = \sigma_{v,u}(p)] \Pr[t \leq \|v_p\|^2] \Pr[\|\tilde{u}_i - \tilde{v}_p\|^2 \leq r < \|\tilde{x}_q - \tilde{u}_i\|^2 | i = \sigma_{v,u}(p)]$$

$$\leq \sum_p \|u_{\sigma_{v,u}(p)}\|^2 \frac{\|v_p\|^2}{\|u_{\sigma_{v,u}(p)}\|^2} \frac{\|\tilde{x}_q - \tilde{u}_i\|^2 - \|\tilde{u}_i - \tilde{v}_p\|^2}{R} \qquad \text{(triangle inequality)}$$

$$\leq \sum_p \|v_p\|^2 \frac{1}{R} \|\tilde{v}_p - \tilde{x}_q\|^2$$

$$\leq \sum_p \|v_p\|^2 \frac{1}{R} \frac{2\|v_p - x_q\|^2}{\max\{\|v_p\|^2, \|x_q\|^2\}}$$

$$\leq \sum_p \frac{2}{R} \|v_p - x_q\|^2.$$

Therefore, for every edge (v, x), $\Pr[(v, x) \in E(X, V \setminus X)] \leq \sum_p (1 + \frac{2}{R})\|v_p - x_q\|^2$.

The expected value of the cut:

$$\mathbb{E}[\mu(E(X, V \setminus X))] = \frac{2}{\omega} \sum_{(v,x) \in E} w_{v,w} \Pr[(v, x) \in E(X, V \setminus X)]$$

$$\leq \frac{2}{\omega} \sum_{(v,x) \in E} w_{v,w} \frac{3}{R} \|v_p - x_q\|^2 \qquad \text{(SDP value} \leq \delta)$$

$$\leq \frac{6\delta}{R}.$$

$\square$

Lemma A.13 (a variant of Lemma 8 in [MM10])**.** *If $\delta < \min\{c_R\lambda_G, \frac{h_G R}{1000}\}$ then with probability at least $\frac{1}{16}$, $\mu(X) \geq 1 - \frac{100\delta}{h_G R}$.*

Proof. By the definition of h_G, $\mu(E(X, V \setminus X)) \geq h_G \min\{\mu(X), \mu(V \setminus X)\}$, which implies

$$\frac{6\delta}{R} \geq \mathbb{E}[\mu(E(X, V \setminus X))]$$
$$\geq h_G \, \mathbb{E}[\min\{\mu(X), \mu(V \setminus X)\}].$$

We get that $\mathbb{E}[\min\{\mu(X), \mu(V \setminus X)\}] \leq \frac{6\delta}{h_G R}$, by Markov inequality

$$\Pr[\min\{\mu(X), \mu(V \setminus X)\} \leq \frac{100\delta}{h_G R}] \geq 1 - \frac{1}{16}.$$

We also know that $\Pr[\mu(X) \geq \frac{1}{8}] \geq \frac{1}{16}$, so with probability at most $1/16$, the set $V \setminus X$ is large $\mu(V \setminus X) \leq \frac{100\delta}{h_G R}$. $\qquad\square$

The following lemma is independent of the graph, so the proof in [MM10] holds here as well.

Lemma A.14 (Lemma 9 in [MM10])**.** *For every edge $(v, x) \in E$,*

$$\Pr[v, x \in X, (v, x) \textit{isn't satisfied}] \leq 4\delta_{v,x},$$

for $\delta_{v,x} = \frac{1}{2} \sum_{i \in [\ell]} \|v_i - x_{\pi_{x,v}(i)}\|^2$.

And we are ready to prove the theorem, the proof is almost identical to the proof in [MM10].

proof of Theorem A.2. We show that the randomized algorithm described above solves the UG instance with constant probability. It can then easily be derandomized.

The algorithm solves the SDP, then runs the rounding algorithm. If $\mu(X) \geq 1 - \frac{100\delta}{h_G R}$, it outputs the labelling, else it fails.

Suppose the algorithm doesn't fail, then by definition

$$\mu(E(X, X)) \geq 1 - \frac{100\delta}{h_G R},$$

as $\mu(V \setminus X) \leq \frac{100\delta}{h_G R}$).

The expected fraction of violated constraints inside X is at most,

$$\frac{2}{\omega} \sum_{(v,x) \in E} w_{v,x} 4\delta_{v,x} \leq \frac{2}{\omega} \sum_{(v,x) \in E} w_{v,x} 4 \sum_{p \in [\ell]} \|v_p - x_{\pi_{x,v}(p)}\|^2 \leq 64\delta.$$

Therefore with constant probability the algorithm outputs a solution satisfying $1 - 64\delta - \frac{100\delta}{h_G R}$ of the constraints. $\qquad\square$

A.1 Eigenvalue proof

Claim A.15. *Let $G = (V, E)$ be a weighted graph with weights $\{w_{u,v}\}_{(u,v)\in E}$, and let $\mathcal{L}$ be the normalized laplacian matrix of G,*

$$\mathcal{L}_{v,u} = \begin{cases} 1 & \text{if } u = v \\ -\frac{w_{u,v}}{\sqrt{w_u w_v}} & \text{if } (u, e) \in E \\ 0 & \text{else} \end{cases},$$

where $w_u = \sum_{v \text{ s.t. } (u,v)\in E} w_{u,v}$. The second smallest eigenvalue of the laplacian corresponds to

$$\lambda_2 = \min_{\{z_u\}_{u\in V}} \frac{\mathbb{E}_{(u,v)\sim w}[\|z_u - z_v\|^2]}{\mathbb{E}_{u,v\sim V}[\|z_u - z_v\|^2]} \; .$$

Where $\{z_u\}_{u\in V}$ is a set of vectors, $\forall u, z_u \in \mathbb{R}^t$.

Proof. We define a new matrix $\mathcal{L}' \in \mathbb{R}^{|V|t \times |V|t}$, which is composed of $t \times t$ scalar matrix blocks, i.e. for every $u, v \in V$, the matrix $\mathcal{L}'_{u,v}$ is a $t \times t$ scalar matrix, $\mathcal{L}'_{u,v} = I^{t\times t}\mathcal{L}_{u,v}$. Formally, we denote each row and column by two indices $u \in V, i \in [t]$ and

$$\mathcal{L}'_{(u,i),(v,j)} = \begin{cases} \mathcal{L}_{u,v} & \text{if } i = j \\ 0 & \text{else} \end{cases}.$$

$\mathcal{L}$ has a single eigen value 0, the new matrix $\mathcal{L}'$ has t eigenvalues 0. One eigenvectors basis for the nullspace is $y^1, \ldots y^t \in \mathbb{R}^{|V|t}$, $y^l_{u,j} = \begin{cases} y_u & l = j \\ 0 & \text{else} \end{cases}$, for y the eigenvector of $\mathcal{L}$.

The spectrum of $\mathcal{L}'$ is identical to the spectrum of $\mathcal{L}$, only each eigenvalue repeats t times. Therefore the second largest eigenvalue of $\mathcal{L}$ is equal to the $t + 1$ eigenvalue of $\mathcal{L}'$, and is equal

$$\lambda_2 = \min_{x\in\mathbb{R}^{t|V|}} \left\{ \frac{\langle x, \mathcal{L}'x \rangle}{\langle x, x - y^1\langle x, y^1\rangle - \cdots - y^t\langle x, y^t\rangle \rangle} \right\}. \tag{15}$$

The numerator equals:

$$\langle x, \mathcal{L}'x \rangle = \sum_{u,v\in V, i,j\in[t]} x_{u,i}\mathcal{L}'_{(u,i),(v,j)}x_{v,j}$$

$$= \sum_{u,v\in V, i\in[t]} x_{u,i}\mathcal{L}_{u,v}x_{v,i}$$

$$= \sum_{u\in V, i\in[t]} x^2_{u,i} - 2 \sum_{(u,v)\in E, i\in[t]} \frac{w_{u,v}}{\sqrt{w_u w_v}} x_{u,i}x_{v,i}$$

$$= \sum_{u\in V} \|x_u\|^2 - 2 \sum_{(u,v)\in E} \frac{w_{u,v}}{\sqrt{w_u w_v}} \langle x_u, x_v \rangle.$$

Where x_u is the length t vector containing $x_{u,i}$ for $i \in [t]$.

55

The denominator:

$$
\begin{aligned}
\langle x, x - y^1\langle x, y^1\rangle - \cdots - y^t\langle x, y^t\rangle\rangle =& \langle x, x\rangle - \left(\langle x, y^1\rangle\right)^2 - \cdots \left(\langle x, y^t\rangle\right)^2 \\
=& \sum_{u\in V, i\in[t]} x_{u,i}^2 - \sum_{l\in[t]} \left(\langle x, y^l\rangle\right)^2 \\
=& \sum_{u\in V, i\in[t]} x_{u,i}^2 - \sum_{l\in[t]} \sum_{u,v\in V, i,j\in t} x_{u,i}y_{u,i}^l x_{v,j}y_{v,j}^l \\
=& \sum_{u\in V, i\in[t]} x_{u,i}^2 - \sum_{l\in[t]} \sum_{u,v\in V} x_{u,l}y_u x_{v,l}y_v \\
=& \sum_{u\in V} \|x_u\|^2 - \sum_{u,v\in V} y_u y_v \langle x_u, x_v\rangle \\
=& \sum_{u\in V} \|x_u\|^2 - \sum_{u,v\in V} \frac{\sqrt{w_u w_v}}{\omega} \langle x_u, x_v\rangle.
\end{aligned}
$$

We write the expectations explicitly:

$$
\begin{aligned}
\mathop{\mathbb{E}}_{(u,v)\sim w}[\|z_u - z_v\|^2] =& \frac{2}{\omega} \sum_{(u,v)\in E} w_{u,v}\langle z_u - z_v, z_u - z_v\rangle \\
=& \frac{2}{\omega} \sum_{(u,v)\in E} w_{u,v}(\|z_u\|^2 + \|z_v\|^2 - 2\langle z_u, z_v\rangle) \\
=& \frac{2}{\omega} \sum_{u\in V} w_u \|z_u\|^2 - \frac{4}{\omega} \sum_{(u,v)\in E} w_{u,v}\langle z_u, z_v\rangle.
\end{aligned}
$$

$$
\begin{aligned}
\mathop{\mathbb{E}}_{u,v\sim V}[[\|z_u - z_v\|^2] =& \frac{1}{\omega^2} \sum_{u,v\in V} w_u w_v \langle z_u - z_v, z_u - z_v\rangle \\
=& \frac{1}{\omega^2} \sum_{u,v\in V} w_u w_v(\|z_u\|^2 + \|z_v\|^2 - 2\langle z_u, z_v\rangle) \\
=& \frac{1}{\omega^2} \sum_{u\in V} 2\omega w_u \|z_u\|^2 - \frac{2}{\omega^2} \sum_{u,v\in V} w_u w_v \langle z_u, z_v\rangle.
\end{aligned}
$$

For every $u\in V, i\in[t]$ let $x_{u,i} = \sqrt{w_u} z_{u,i}$,

$$
\langle x, \mathcal{L}'x\rangle = \frac{\omega}{2} \mathop{\mathbb{E}}_{(u,v)\sim w}[\|z_u - z_v\|^2],
$$

$$
\langle x, x - y^1\langle x, y^1\rangle - \cdots - y^t\langle x, y^t\rangle\rangle = \frac{\omega}{2} \mathop{\mathbb{E}}_{u,v\sim V}[[\|z_u - z_v\|^2].
$$

The factor of $\frac{\omega}{2}$ cancels out, and the minimum value is not affected by the multiplication in $\sqrt{w_u}$, as it is taken over all vectors in $\mathbb{R}$. $\qquad\square$