Improved Decoding of Tanner Codes

Zhaienhe Zhou* Zeyu Guo†

Abstract

In this paper, we present improved decoding algorithms for expander-based Tanner codes.

We begin by developing a randomized linear-time decoding algorithm that, under the condition that $\delta d_0 > 2$, corrects up to αn errors for a Tanner code $T(G, C_0)$, where G is a (c, d, α, δ)-bipartite expander with n left vertices, and $C_0 \subseteq \mathbb{F}_2^d$ is a linear inner code with minimum distance d_0. This result improves upon the previous work of Cheng, Ouyang, Shangguan, and Shen (RANDOM 2024), which required $\delta d_0 > 3$.

We further derandomize the algorithm to obtain a deterministic linear-time decoding algorithm with the same decoding radius. Our algorithm improves upon the previous deterministic algorithm of Cheng et al. by achieving a decoding radius of αn, compared with the previous radius of $\frac{2\alpha}{d_0(1+0.5c\delta)}n$.

Additionally, we investigate the size-expansion trade-off introduced by the recent work of Chen, Cheng, Li, and Ouyang (IEEE TIT 2023), and use it to provide new bounds on the minimum distance of Tanner codes. Specifically, we prove that the minimum distance of a Tanner code $T(G, C_0)$ is approximately $f_\delta^{-1}\left(\frac{1}{d_0}\right)\alpha n$, where $f_\delta(\cdot)$ is the Size-Expansion Function. As another application, we improve the decoding radius of our decoding algorithms from αn to approximately $f_\delta^{-1}\left(\frac{2}{d_0}\right)\alpha n$.

1 Introduction

Low-density parity-check (LDPC) codes are an important class of error-correcting codes known for their near-optimal error-correction performance and low decoding complexity. They are characterized by sparse parity-check matrices, where most entries are zero, which makes them efficient in both encoding and decoding. LDPC codes were first introduced by Gallager [Gal62] in the 1960s but gained renewed attention due to their suitability for modern communication systems, including wireless and satellite communications, as well as for storage applications.

A key feature of LDPC codes is their representation as a bipartite graph, with bits on the left side and parity checks on the right. This leads to a parity-check graph with a constant degree, where the sparsity of the graph enables the development of efficient decoding algorithms. In particular, the belief-propagation algorithm, introduced by Gallager [Gal62], was later proven by Zyablov and Pinsker [ZP75] to correct a constant fraction of errors on random LDPC codes.

Later, Tanner [Tan81] extended LDPC codes by developing a recursive approach to construct long error-correcting codes from shorter inner codes. *Tanner codes* are constructed by assigning a linear inner code C_0 of length d and minimum distance d_0 to the vertices of a sparse bipartite graph. Specifically, bits are placed on the left side of the bipartite graph, and each vertex on the right side is assigned an inner code that imposes constraints on the connected bits. Note that when the inner

*zhaienhezhou@gmail.com, School of the Gifted Young & College of Computer Science, University of Science and Technology of China, Hefei 230026, China.

†zguotcs@gmail.com, Department of Computer Science and Engineering, The Ohio State University

code is a parity-check code, the Tanner code simplifies to an LDPC code. This construction offers greater flexibility in code design, enabling codes with both large minimum distances and efficient decoding properties.

To analyze the decoding algorithms of LDPC and Tanner codes, Sipser and Spielman [SS96] introduced the concept of vertex expansion. Expander codes are a special class of Tanner codes constructed from (c, d, α, δ)-bipartite expanders, where c, d, α, and δ are constants. Specifically, the graph $G = (L \cup R, E)$ is left-regular of degree c and right-regular of degree d, and for any set $S \subseteq L$ with $|S| \leq \alpha n$, the size of the neighborhood of S is at most $\delta c|S|$. Expander codes are known for their efficient decoding algorithms, which can correct $\Omega(n)$ errors in linear time. Research on expander codes in coding theory and theoretical computer science [SS96, SR03, RU01, FWK05, FMS+07, Vid13a, Vid13b, CCLO23] has focused on optimizing the decoding radius and other parameters while keeping the decoding algorithm linear-time.

Consider the special case where the inner code is a parity-check code. In this case, the flip algorithm introduced by Sipser and Spielman [SS96] can decode up to $(2\delta - 1)\alpha n$ errors in linear time for any expander code with $\delta > \frac{3}{4}$. Later, Viderman [Vid13a] proposed a new decoding method, which corrects up to $\frac{3\delta-2}{2\delta-1}\alpha n$ errors in linear time when $\delta \geq \frac{2}{3}$.

More recently, Chen, Cheng, Li, and Ouyang [CCLO23] presented an improved decoding algorithm by combining previous approaches and introducing a method they term "expansion guessing." They also discovered a size-expansion trade-off, which enables expansion of larger sets to be inferred from smaller sets. Their work showed that expander codes achieve a minimum distance of $\frac{1}{2(1-\delta)}\alpha n$, and their decoding algorithm achieves a decoding radius of $\frac{3}{16(1-\delta)}\alpha n$, which is nearly half of the code's distance. However, their algorithm still requires $\delta > \frac{3}{4}$ to use the flip algorithm. This raises an open question: What is the minimum δ required to decode a linear number of errors in linear time? It was shown in [Vid13a] that $\delta > \frac{1}{2}$ is necessary.

The above studies focus on the special case of expander codes where the inner code C_0 is a parity-check code with minimum distance $d_0 = 2$. Progress has also been made on the general case [CNVM10, DG18, COSS24]. Notably, Dowling and Gao [DG18] proved that the condition $d_0\delta^2 = \Omega(c)$ is sufficient for error correction using a flip-based decoding algorithm. More recently, Cheng, Ouyang, Shangguan, and Shen [COSS24] improved this result by showing that $\delta d_0 > 3$ is sufficient for error correction. They also proved that $\delta d_0 > 1$ is necessary, thereby generalizing an earlier result of Viderman [Vid13a] for expander codes with parity-check inner codes.

However, many questions remain open about the optimal parameters. In particular, there is still a gap between the sufficient and necessary conditions for δd_0 to enable a linear-time decoding. In this paper, we narrow this gap by proving that $\delta d_0 > 2$ is sufficient for expander-based Tanner codes.

1.1 Main Results

Let $T(G, C_0)$ be a Tanner code based on a bipartite expander G and an inner code C_0 (see Definition 2.2). Our first main result is a deterministic linear-time decoding algorithm for $T(G, C_0)$.

Theorem 1.1 (Informal version of Theorem 4.7)**.** *Suppose $\delta d_0 > 2$. There exists a deterministic $O(n)$-time algorithm that corrects up to αn errors for any Tanner code $T(G, C_0) \subseteq \mathbb{F}_2^n$, where G is a (c, d, α, δ)-bipartite expander and C_0 is an inner code with minimum distance d_0.*

Previously, under the condition $\delta d_0 > 3$, Cheng, Ouyang, Shangguan, and Shen [COSS24] gave a randomized linear-time decoding algorithm for $T(G, C_0)$ that corrects up to αn errors, as well as a deterministic linear-time decoding algorithm with a slightly smaller decoding radius

$\frac{2\alpha}{d_0(1+0.5c\delta)}n$. Theorem 1.1 improves on their results by (1) relaxing the condition to $\delta d_0 > 2$, and (2) derandomizing the randomized decoding algorithm without reducing the decoding radius αn.

To prove Theorem 1.1, we first establish a weaker result: Assuming $\delta d_0 > 2$, there exists a randomized linear-time algorithm correcting αn errors. This result is stated as Theorem 3.6, and its proof closely follows the approach in [COSS24]. Then, in Section 4, we derandomize the algorithm while preserving the decoding radius.

We also investigate the size-expansion trade-off introduced by [CCLO23]. Specifically, we define the Size-Expansion Function $f_\delta(k)$ (see Definition 5.2), which satisfies the following property: For any (c, d, α, δ)-bipartite expander, the graph remains an expander with parameters $(c, d, k\alpha, f_\delta(k))$, where $k > 1$ is a constant. Consequently, our decoding algorithm achieves a decoding radius of approximately $f_\delta^{-1}\left(\frac{2}{d_0}\right)\alpha n$, which is strictly larger than αn.

Theorem 1.2 (Informal version of Theorem 5.7)**.** *Theorem 1.1 still holds with the decoding radius increased to approximately* $f_\delta^{-1}\left(\frac{2}{d_0}\right)\alpha n$.

Finally, we establish the following tight bound on the minimum distance of $T(G, C_0)$:

Theorem 1.3 (Informal version of Theorems 5.6 and 5.8)**.** *Suppose* $\delta d_0 > 1$. *The minimum distance of the Tanner code* $T(G, C_0)$ *is at least approximately* $f_\delta^{-1}\left(\frac{1}{d_0}\right)\alpha n$. *Furthermore, this lower bound is tight in the sense that it is achieved by infinitely many examples.*

The proof of Theorem 1.3 has two parts. First, we show that the minimum distance is bounded from below by $f_\delta^{-1}\left(\frac{1}{d_0}\right)\alpha n$. Second, we present a construction showing that this bound is achievable when α is sufficiently small but still constant. This construction builds on the approach in [CCLO23]. For details, see Section 5.

2 Preliminaries

2.1 Notation and Definitions

Let $\mathbb{N} = \{0, 1, 2, \dots\}$ and $\mathbb{N}^+ = \{1, 2, \dots\}$. For $n \in \mathbb{N}$, denote by $[n]$ the set $\{1, 2, \dots, n\}$. The functions $\log(\cdot)$ and $\exp(\cdot)$ use base e. The time complexity of algorithms is analyzed using the RAM model, in which random access of memory and arithmetic operations are assumed to take constant time. Graphs are represented using adjacency lists within the algorithms.

Codes. In this paper, all codes are assumed to be *Boolean linear codes*. That is, a code is defined as a subspace $C \subseteq \mathbb{F}_2^n$ over the finite field $\mathbb{F}_2$. The parameter n is called the *length* of C.

The Hamming weight of a vector $x \in \mathbb{F}_2^n$, denoted $\mathrm{wt}(x)$, is defined as the number of nonzero coordinates in x. The Hamming distance between two vectors $x, y \in \mathbb{F}_2^n$ is defined as $d_H(x, y) \coloneqq \mathrm{wt}(x - y)$. The *minimum distance* of a code C is $d_H(C) \coloneqq \min\{d_H(x, y) : x, y \in C, x \neq y\}$.

Bipartite graphs and expanders. A bipartite graph $G = (L \cup R, E)$ is called (c, d)-*regular* if $\deg(u) = c$ for all $u \in L$ and $\deg(v) = d$ for all $v \in R$.

For any subset of vertices $S \subseteq L \cup R$, let $N(S)$ denote the set of all neighbors of S. Define $N_i(S)$ as the set of vertices adjacent to exactly i vertices in S. Additionally, we use the following shorthand notations for convenience:

$$N_{\geq i}(S) \coloneqq \bigcup_{j \geq i} N_j(S), \quad N_{\leq i}(S) \coloneqq \bigcup_{j \leq i} N_j(S).$$

We define $E(S,T)$ as the set of edges connecting the two vertex sets S and T.

Definition 2.1 (Bipartite expander)**.** *A (c,d,α,δ)-bipartite expander is a (c,d)-regular bipartite graph $G=(L\cup R,E)$ such that for any subset of vertices $S\subseteq L$ with $|S|\le\alpha|L|$, it holds that $|N(S)|\ge\delta c|S|$.*

For non-empty $S\subseteq L$, we call $\frac{N(S)}{c|S|}$ the expansion factor *of S. The above expansion property is equivalent to that the expansion factor of every non-empty $S\subseteq L$ of size at most $\alpha|L|$ is at least δ.*

We now proceed to define Tanner codes, the central object studied in this paper.

Definition 2.2 (Tanner code)**.** *Let C_0 be a code of length d. Let $G=(L\cup R,E)$ be a (c,d,α,δ)-bipartite expander, where $L=[n]$ for some positive integer n. For each $v\in R$, fix a total ordering on $N(v)$, and let $N(v,i)$ denote its i-th element for $i\in[d]$. For $x\in\mathbb{F}_2^n$ and $v\in R$, define*

$$x_{N(v)}\coloneqq(x_{N(v,1)},\dots,x_{N(v,d)})\in\mathbb{F}_2^d.$$

In other words, if x is viewed as an assignment $L\to\mathbb{F}_2$, then $x_{N(v)}$ is its restriction to $N(v)$.

The Tanner code $T(G,C_0)$ is a code of length n, defined as

$$T(G,C_0)\coloneqq\{x\in\mathbb{F}_2^n : x_{N(v)}\in C_0 \text{ for all } v\in R\}\subseteq\mathbb{F}_2^n.$$

In other words, a vector $x\in\mathbb{F}_2^n$ is a codeword of $T(G,C_0)$ if for every $v\in R$, the "local view" $x_{N(v)}$ is a codeword of C_0.

Throughout this paper, we fix positive integers c,d and real numbers $\alpha,\delta\in(0,1]$. The parameters c,d,α,δ are viewed as constants independent of the growing parameter n. Also, let $G=(L\cup R,E)$ be a (c,d,α,δ)-bipartite expander with $L=[n]$, and let C_0 be a code of length d with minimum distance d_0. All lemmas and theorems are stated under the assumption that G and C_0 are given, without explicitly mentioning this.

For convenience, we introduce the following definition.

Definition 2.3 (Corrupt bits and unsatisfied checks)**.** *For $x,y\in\mathbb{F}_2^n$, define $F(x,y)=\{i\in[n] : x_i\neq y_i\}$. Define $F(x)=F(x,y)$, where y is the closest codeword to x in $T(G,C_0)$ with respect to the Hamming distance. (If there are multiple closest codewords, y is chosen to be the lexicographically smallest one.)*

Let $U(x)\subseteq R$ denote the set of unsatisfied checks, defined as $U(x)=\{v\in R : x_{N(v)}\notin C_0\}$.

2.2 Probabilistic Tools

We use standard tools from probability theory, such as Hoeffding's inequality and Azuma's inequality, which are detailed in textbooks like [MU17].

Lemma 2.4 (Hoeffding's inequality [MU17])**.** *Let $X_1,X_2,\dots,X_n$ be independent random variables. Let $X=\sum_{i=1}^n X_i$. If $X_i\in[\ell,r]$ for all i, then for any $a>0$,*

$$\Pr[X\ge\mathbb{E}[X]+a]\le\exp\left(-\frac{2a^2}{n(r-\ell)^2}\right)\quad\text{and}\quad\Pr[X\le\mathbb{E}[X]-a]\le\exp\left(-\frac{2a^2}{n(r-\ell)^2}\right).$$

Recall that a sequence of random variables $X_0,X_1,\dots,X_n$ is called a *martingale* if for $i=0,1,\dots,n-1$,

$$\mathbb{E}[X_{i+1}|X_0,X_1,\dots,X_{i-1}]=\mathbb{E}[X_i].$$

Azuma's inequality provides a concentration bound for martingales.

Lemma 2.5 (Azuma's inequality [MU17])**.** *Let $X_0, X_1, \ldots, X_n$ be a martingale such that $|X_k - X_{k-1}| \le c_k$ for all k. Then, for any $t \ge 1$ and $\lambda > 0$,*

$$\Pr(|X_t - X_0| \ge \lambda) \le 2\exp\left(-\frac{\lambda^2}{2\sum_{k=1}^{t} c_k^2}\right).$$

2.3 Auxiliary Lemmas

We present some useful auxiliary lemmas.

Lemma 2.6. *For any $S \subseteq L$ with $|S| \le \alpha n$ and integer $t \ge 0$, it holds that*

$$|N_{\le t}(S)| \ge \frac{\delta(t+1)-1}{t} \cdot c|S|.$$

Proof. The claim follows from a double-counting argument. Since G is left-regular of degree c, we have $c|S| = |E(S, N(S))|$. On the other hand,

$$\begin{aligned}
|E(S, N(S))| &= \sum_{i=1}^{d} i|N_i(S)| \\
&\ge |N_{\le t}(S)| + (t+1)(|N(S)| - |N_{\le t}(S)|) \\
&= (t+1)|N(S)| - t|N_{\le t}(S)| \\
&\ge (t+1)\delta c|S| - t|N_{\le t}(S)|,
\end{aligned}$$

where the last inequality follows from the expansion property of G.

Therefore, we have $c|S| \ge (t+1)\delta c|S| - t|N_{\le t}(S)|$. Rearranging this establishes the lemma. □

Lemma 2.7. *Let $x \in \mathbb{F}_q^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \le \alpha n$. Let $F = F(x, y)$. Then*

$$c|F| \ge |U(x)| \ge |N_{\le d_0 - 1}(F)| \ge \frac{\delta d_0 - 1}{d_0 - 1} \cdot c|F|.$$

Proof. Since G is left-regular of degree c, the number of edges incident to $F \subseteq L$ is $c|F|$. Each vertex in $U(x) \subseteq R$ is incident to at least one of these edges. Therefore, $|U(x)| \le c|F|$.

Since the minimum distance of C_0 is d_0, every $v \in R$ that is a common neighbor of no more than $d_0 - 1$ vertices in F must be an unsatisfied check, i.e., $N_{\le d_0 - 1}(F) \subseteq U(x)$. It follows that $|N_{\le d_0 - 1}(F)| \le |U(x)|$.

Finally, applying Lemma 2.6 with $t = d_0 - 1$ and $S = F$ gives the inequality

$$|N_{\le d_0 - 1}(F)| \ge \frac{\delta d_0 - 1}{d_0 - 1} \cdot c|F|.$$ □

Lemma 2.8. *For any $S \subseteq L$,*

$$|N_{\ge t}(S)| \le \frac{c}{t}|S|.$$

Proof. Since G is left-regular of degree c, we have $|E(S, N(S))| = c|S|$. Moreover, by the definition of $N_{\ge t}(\cdot)$, we have $|E(S, N(S))| \ge t|N_{\ge t}(S)|$. The lemma follows. □

Lemma 2.9. *Let $T(G, C_0)$ be a Tanner code where G is a (c, d, α, δ)-bipartite expander and the inner code C_0 has distance d_0. If $\delta d_0 > 1$, then the distance of $T(G, C_0)$ is greater than αn.*

Proof. For any linear code, its distance is equal to the minimum Hamming weight among all of its codewords.

Assume, to the contrary, that $T(G, C_0)$ has a nonzero codeword x such that $\mathrm{wt}(x) \leq \alpha n$. Let $F = \{i \in [n] : x_i \neq 0\}$, whose size is $\mathrm{wt}(x)$. By Lemma 2.6, we have

$$N_{\leq d_0 - 1}(F) \geq \frac{\delta d_0 - 1}{d_0 - 1} \cdot c|F| > 0.$$

This implies that there exists $u \in R$ that has fewer than d_0 neighbors in F. Consequently, $\mathrm{wt}(x_{N(u)}) < d_0$.

As the minimum distance of C_0 is d_0, we have $x_{N(u)} \notin C_0$, which contradicts the assumption that $x \in T(G, C_0)$. Thus, the lemma is proven. □

3 Randomized Decoding

In this section, we present an improved randomized flipping algorithm for the decoding regime $\delta d_0 > 2$, and then extend it to a full randomized decoding algorithm.

Our flipping algorithm follows the approach of [COSS24], which is based on the following idea: For each unsatisfied check $v \in U(x)$, if $x_{N(v)}$ is sufficiently close to a codeword $c \in C_0$ of the inner code, it is likely that $x_{N(v)}$ should be corrected to c. We let each such check v cast a "vote" on which bits to flip. Then, each bit is flipped with a probability determined by the votes it receives. This process corrects a constant fraction of errors. By repeating it logarithmically many times, the received word can be corrected with high probability.

Our improvement is achieved by allowing each v to send a weighted vote based on $d_H(x_{N(v)}, y)$, where $y \in C_0$ is the closest codeword to $x_{N(v)}$, rather than using an unweighted vote when $d_H(x_{N(v)}, y) < d_0/3$, as was done in [COSS24].[1] This modification enables a tighter analysis.

3.1 Randomized Flipping

In the following, let $\mathsf{Decode}(x)$ denote the function that, given $x \in \mathbb{F}_2^d$, returns the codeword $y \in C_0$ closest to x in Hamming distance, with ties broken by selecting the lexicographically smallest y.

We now present the pseudocode of the randomized flipping algorithm.

[1] At a high level, this bears some similarity with the GMD decoding algorithm for concatenated codes [For66], where a large $d_H(x_{N(v)}, y)$ suggests that y is likely incorrect.

Algorithm 1 RandFlip(x)

Input: $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$, where $n = |L|$.

1: $t \leftarrow \frac{d_0}{2}$
2: $(p_1, \ldots, p_n) \leftarrow (0, \cdots, 0) \in \mathbb{R}^n$
3: **for** each $v \in R$ **do**
4: $\quad w_v \leftarrow \mathsf{Decode}(x_{N(v)})$
5: $\quad$ **if** $1 \le d_H(w_v, x_{N(v)}) < t$ **then**
6: $\quad\quad$ Choose the smallest $i \in N(v)$ where w_v and $x_{N(v)}$ differ[2]
7: $\quad\quad p_i \leftarrow p_i + \frac{t - d_H(w_v, x_{N(v)})}{ct}$
8: $\quad$ **end if**
9: **end for**
10: **for** each $i \in [n]$ **do**
11: $\quad$ Flip x_i with probability p_i
12: **end for**
13: **return** x

We first check that the values p_i are within $[0, 1]$, ensuring the validity of Line 11.

Lemma 3.1. *For $i \in [n]$, we have $p_i \in [0, 1]$ at Line 11.*

Proof. Each p_i is the sum of at most $\deg(i) = c$ terms of the form $\frac{t - d_H(w_v, x_{N(v)})}{ct}$, where each term lies between 0 and $\frac{1}{c}$. The lemma follows. □

The following theorem states that the algorithm RandFlip(x) is expected to correct at least a constant fraction of errors of x under certain conditions.

Theorem 3.2. *Assume $d_0\delta > 2$. Let $\varepsilon_0 = \frac{d_0}{2} - \frac{1}{\delta} > 0$. Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \le \alpha n$. Let x' be the output of Algorithm 1 with x as input. Then $\mathbb{E}[d_H(x', y)|] \le (1 - \frac{\varepsilon_0\delta}{t}) d_H(x, y)$.*

To prove Theorem 3.2, we need the following lemma. Recall that $F(x, y) = \{i \in [n] : x_i \ne y_i\}$.

Lemma 3.3. *Let $x \in \mathbb{F}_2^n$, $y \in T(G, C_0)$, and $F = F(x, y)$. Let $v \in N_k(F)$ for some integer k. Let w_v be as in Algorithm 1, i.e., $w_v = \mathsf{Decode}(x_{N(v)})$. If $w_v = y_{N(v)}$, then $d_H(w_v, x_{N(v)}) = k$. On the other hand, if $w_v \ne y_{N(v)}$, then $d_0 - k \le d_H(w_v, x_{N(v)}) \le k$. The latter case occurs only if $k \ge \frac{d_0}{2} = t$, i.e., $v \in N_{\ge t}(F)$.*

Proof. By the definition of F and the choice of v, we have $d_H(y_{N(v)}, x_{N(v)}) = k$. As $y \in T(G, C_0)$, we have $y_{N(v)} \in C_0$. As w_v is a vector in C_0 closest to $x_{N(v)}$, we have

$$d_H(w_v, x_{N(v)}) \le d_H(y_{N(v)}, x_{N(v)}) = k.$$

If $w_v = y_{N(x)}$, we have $d_H(w_v, x_{N(v)}) = d_H(x_{N(v)}, y_{N(v)}) = k$. On the other hand, if $w_v \ne y_{N(x)}$, then the distance between these two codewords of C_0 is at least d_0, which implies $d_H(w_v, x_{N(v)}) \ge d_H(w_v, y_{N(v)}) - d_H(x_{N(v)}, y_{N(v)}) \ge d_0 - k$. This proves the lemma. □

Now we are ready to prove Theorem 3.2.

[2]Implicitly, we are viewing w_v and $x_{N(v)}$ as elements in $\mathbb{F}_2^{N(v)} : N(v) \to \mathbb{F}_2$ by identifying $\mathbb{F}_2^d$ with $\mathbb{F}_2^{N(v)}$ through the total ordering on $N(v)$ defined in Definition 2.2.

Proof of Theorem 3.2. Let $F = \{i \in [n] : x_i \neq y_i\}$, whose size is $d_H(x, y) \leq \alpha n$. By definition, we have

$$d_H(x', y) = |F| - \{i \in F : x_i \text{ is flipped}\} + \{i \in [n] \setminus F : x_i \text{ is flipped}\}. \tag{1}$$

By (1) and linearity of expectation, we have

$$\mathbb{E}[d_H(x', y)] = |F| - \left(\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i\right). \tag{2}$$

Consider arbitrary $v \in R$. In the iteration of the first `for` loop (Lines 3–9) corresponding to v, some p_i may increase by $\frac{t - d_H(w_v, x_{N(v)})}{ct}$. We analyze how this affects the quantity $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i$. Suppose v has k neighbors in F, i.e., $v \in N_k(F)$.

Case 1: $k = 0$. In this case, $d_H(w_v, x_{N(v)}) = 0$. Due to the condition $1 \leq d_H(w_v, x_{N(v)}) < t$ at Line 5, the iteration corresponding to v does not affect $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i$.

Case 2: $1 \leq k < t$. In this case, we have $d_H(w_v, x_{N(v)}) = k \in [1, t)$ and $w_v = y_{N(v)}$ by Lemma 3.3. In the iteration corresponding to v, the index i chosen at Line 6 is in F since $w_v = y_{N(v)}$. Thus, this iteration contributes exactly

$$\frac{t - d_H(w_v, x_{N(v)})}{ct} = \frac{t - k}{ct}$$

to $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i$.

Case 3: $t \leq k < d_0$. In this case, we have $d_H(w_v, x_{N(v)}) \geq d_0 - k = 2t - k$ by Lemma 3.3. Thus, the iteration corresponding to v contributes at least

$$-\frac{t - d_H(w_v, x_{N(v)})}{ct} \geq -\frac{t - (2t - k)}{ct} = \frac{t - k}{ct}$$

to $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i$.

Case 4: $k \geq d_0$. In this case, we have $d_H(w_v, x_{N(v)}) \geq 0$. Thus, the iteration corresponding to v contributes at least

$$-\frac{t - d_H(w_v, x_{N(v)})}{ct} \geq -\frac{t}{ct} \geq \frac{t - k}{ct},$$

to $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i$, where the last inequality uses the fact that $k \geq d_0 = 2t$.

By the above discussion, We have

$$\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i \geq \sum_{k=1}^{d} \frac{t - k}{ct} |N_k(F)|. \tag{3}$$

Next, we establish a lower bound on the RHS of (3). By the definition of $N_k(\cdot)$ and the fact that G is left-regular of degree c, we have

$$\sum_{k=1}^{d} k |N_k(F)| = |E(F, N(F))| = c|F|. \tag{4}$$

As $|F| \le \alpha n$ and G is a (c, d, α, δ)-bipartite expander, we have

$$\sum_{k=1}^{d} |N_k(F)| = |N(F)| \ge \delta c |F|. \tag{5}$$

Multiplying both sides of (5) by $t = \frac{1}{\delta} + \varepsilon_0$ and subtracting both sides of (4), we obtain

$$\sum_{k=1}^{d} (t-k)|N_k(F)| \ge \varepsilon_0 \delta c |F|,$$

or equivalently,

$$\sum_{k=1}^{d} \frac{t-k}{ct} |N_k(F)| \ge \frac{\varepsilon_0 \delta}{t} |F|. \tag{6}$$

Combining (3) and (6) shows

$$\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i \ge \frac{\varepsilon_0 \delta}{t} |F|. \tag{7}$$

And (2) and (7) together yield $\mathbb{E}[d_H(x', y)] \le (1 - \frac{\varepsilon_0 \delta}{t})|F| = (1 - \frac{\varepsilon_0 \delta}{t}) d_H(x, y)$, as desired. □

While Algorithm 1 is expected to reduce the number of corrupt bits by a constant factor, the number may increase depending on the chosen randomness. Nevertheless, the following lemma shows that any such increase will not be too large. This result will be used later.

Lemma 3.4. *Let $x \in \mathbb{F}_2^n$, $y \in T(G, C_0)$, and $F = \{i \in [n] : x_i \ne y_i\}$. The number of $i \in [n] \setminus F$ such that $p_i > 0$ at the end of* RandFlip(x) *is at most $\frac{c}{t}|F|$. In particular, for $x' =$* RandFlip(x)*, we always have $d_H(x', y) \le (1 + \frac{c}{t}) d_H(x, y)$.*

Proof. Consider $v \in U(x)$ such that the corresponding iteration increases p_i from zero to nonzero for some $i \in [n] \setminus F$. By the way i is chosen at Line 6, we know w_v and $x_{N(v)}$ differ at this bit. As $i \in [n] \setminus F$, we know $x_{N(v)}$ and $y_{N(v)}$ agree at this bit. So $w_v \ne y_{N(v)}$. By Lemma 3.3, this occurs only if $v \in N_{\ge t}(F)$. Finally, by Lemma 2.8, the number of $v \in N_{\ge t}(F)$ is at most $\frac{c}{t}|F|$. □

3.2 Time Complexity of Randomized Flipping

It is straightforward to implement Algorithm 1 in $O(n)$ time. However, since the main algorithm runs it $O(\log n)$ times, this is not sufficient to achieve overall linear time. Instead, we adapt Algorithm 1 to run in time proportional to the number of corrupt bits, following analyses similar to those in [SS96,DG18,COSS24]. This adjustment ensures that the total runtime across all executions of Algorithm 1 forms a geometric series that sums to $O(n)$. The details are provided below for completeness.

In the following, let $x \in \mathbb{F}_2^n$ and y be a code of $T(G, C_0)$.

Maintaining the set $U(x)$. First, we design a data structure to store the set $U(x)$ of unsatisfied checks, ensuring the following properties:

- The elements in $U(x)$ can be enumerated in $O(|U(x)|)$ time.
- Given $v \in R$, we can determine whether $v \in U(x)$ in constant time.

- Each time a bit of x is flipped, $U(x)$ is updated in constant time.

Specifically, we store the elements of $U(x)$ via a linked list. Additionally, we maintain an array that records for each $v \in R$:

1. Whether $v \in U(x)$.
2. The pointer to v in the linked list.

The above information can be accessed in constant time for any given $v \in R$. When flipping a bit $i \in [n]$, we update $U(x)$ in constant time by iterating over each $v \in N(i)$, updating its status in the array, and adding or removing v from the linked list as needed.

Maintaining the set P. Define

$$P := \{i \in [n] : p_i \neq 0\}.$$

By similarly using an array and a linked list, we maintain the set P together with the flipping probabilities p_i such that:

- The elements in P can be enumerated in $O(|P|)$ time.
- Given $i \in [n]$, we can find p_i in constant time.
- Each time some p_i changes, the set P is updated in constant time.

Initialization and clean-up. The array and the linked list used to maintain $U(x)$ are treated as global variables. They are initialized in $O(n)$ time at the start of the main algorithm, described in Section 3.3, and are maintained throughout the algorithm.

At the start of Algorithm 1, we require $p_i = 0$ for all $i \in [n]$ and $P = \emptyset$. However, manually assigning these variables as in Line 3 of Algorithm 1 would take $\Theta(n)$ time. Instead, we treat the array and the linked list used to maintain p_i and P as global variables, which are initialized in $O(n)$ time at the start of the main algorithm. Then, at the end of each execution of Algorithm 1, we reset the array and the linked list to their initial states so that they can be reused in the next execution of Algorithm 1. The resetting takes time linear in $|P|$.

Enumerating fewer elements. Beyond the previously described measures, we further modify Algorithm 1 to enumerate only the unsatisfied checks and the bits with nonzero flipping probabilities.

Specifically, we replace $v \in R$ at Line 3 of Algorithm 1 with $v \in U(x)$. To see that this change does not affect the flipping probabilities p_i, observe that for $v \in R \setminus U(x)$, $x_{N(v)}$ is a codeword of C_0. Consequently, $w_v = \mathsf{Decode}(x_{N(v)})$ is simply $x_{N(v)}$ itself, implying that $d_H(w_v, (x_{N(v)}) = 0$. So p_i is unchanged in the iteration corresponding to v.

Moreover, we modify $i \in [n]$ at Line 11 of Algorithm 1 to $i \in P$, where $P = \{i \in [n] : p_i \neq 0\}$.

With these adjustments, the first loop executes at most $|U(x)| \leq c|F(x,y)|$ times, and the second loop at most $|P| \leq |U(x)| \leq c|F(x,y)|$ times. The function $\mathsf{Decode}(x_{N(v)})$ can be computed in constant time, e.g., by brute-force search. Thus, we have:

Lemma 3.5. *Algorithm 1 can be implemented to run in $O(|F(x,y)|)$ time, where $x \in \mathbb{F}_2^n$ is the input and y is any codeword of $T(G, C_0)$.*

3.3 Flipping Iteratively

We present the pseudocode of our randomized decoding algorithm:

Algorithm 2 $\mathsf{RandDecode}(x)$

Input: $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$

1: **while** $|U(x)| > 0$ **do**
2: $\quad x \leftarrow \mathsf{RandFlip}(x)$
3: **end while**
4: **return** x

Theorem 3.6. *Assume $d_0\delta > 2$ and let $\varepsilon_0 = \frac{d_0}{2} - \frac{1}{\delta} > 0$. Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \le \alpha n$. Then given the input x, Algorithm 2 outputs y in $O(n)$ time with probability $1 - o(1)$.*

The $o(1)$ term is with respect to the growing parameter n. We also assume $n \ge n_0$ for some large enough constant n_0; otherwise, the unique decoding of $T(G, C_0)$ can be solved in constant time, e.g., via brute-force search.

Proof of Theorem 3.6. Let $t = \frac{d_0}{2}$, $\varepsilon_1 = \frac{\varepsilon_0\delta}{t}$, and $\varepsilon_2 = \frac{c}{t}$. Let $\beta \in (0, 1)$ be a small enough constant depending only on ε_1 and ε_2. We consider two phases of the algorithm:

Phase 1: Many corrupt bits. Suppose at some point during the algorithm, calling $\mathsf{RandFlip}(x)$ changes $F(x, y)$ from a set F to another set F', and $|F| \ge n^\beta$, where $C > 0$ is a large enough constant depending only on ε_1 and ε_2. Then

$$|F'| = |F| - \left(\sum_{i \in F} X_i - \sum_{i \in [n] \setminus F} X_i \right),$$

where each X_i is the indicator random variable associated with the event that x_i is flipped in $\mathsf{RandFlip}(x)$. Each X_i takes the value one with probability p_i (see Algorithm 1) and zero with probability $1 - p_i$, and these random variables X_i are independent. Also note that by Lemma 3.4, the number of $i \in [n]$ such that $p_i \ne 0$ is at most $(1 + \varepsilon_2)|F|$. Let $\mu = \mathbb{E}[\sum_{i \in F} X_i - \sum_{i \in [n] \setminus F} X_i]$. By Theorem 3.2, we have $\mu \ge \varepsilon_1 |F|$. Choose $k := \lceil \frac{\log(n^{1-\beta})}{-\log(1 - \frac{\varepsilon_2}{2})} \rceil$ so that $n \cdot \left(1 - \frac{\varepsilon_2}{2}\right)^k \le n^\beta$. We have

$$\begin{aligned}
\Pr\left[|F'| \ge (1 - \frac{\varepsilon_1}{2})|F|\right] &\le \Pr\left[\sum_{i \in F} X_i - \sum_{i \in [n] \setminus F} X_i \le \mu - \frac{\varepsilon_1 |F|}{2}\right] \\
&\le \exp\left(-\frac{2\left(\frac{\varepsilon_1}{2}|F|\right)^2}{(1 + \varepsilon_2)|F|}\right) && \text{(Hoeffding's inequality)} \\
&\le \exp\left(-\frac{\varepsilon_1^2}{2(1 + \varepsilon_2)} n^\beta\right) && \text{(since } |F| \ge n^\beta\text{)} \\
&= o(1/k).
\end{aligned}$$

Thus, with probability at least $1 - o(1/k)$, the size of $F(x, y)$ decreases by at least a factor of $1 - \frac{\varepsilon_1}{2}$. By the union bound, with probability $1 - o(1)$, $|F(x, y)|$ decreases by at least a factor

of $1-\frac{\varepsilon_1}{2}$ in each run of $\mathsf{RandFlip}(x)$ until $|F(x,y)| \le n \cdot \left(1-\frac{\varepsilon_2}{2}\right)^k \le n^\beta$. When this happens, the total time it takes is $O\left(\sum_{i=0}^{k-1}\left(n \cdot \left(1-\frac{\varepsilon_1}{2}\right)^i\right)\right) = O(n)$ by Lemma 3.5.

Phase 2: Few corrupt bits. Now assume $|F(x,y)| \le n^\beta$. Consider running $\mathsf{RandFlip}(x)$ another $\ell := \lceil \frac{2\beta \log n}{-\log(1-\varepsilon_1)} \rceil$ times. By Lemma 3.4, the size of $F(x,y)$ may increase by at most a factor of $1+\varepsilon_2$ each time. Therefore, $|F(x,y)|$ would not exceed

$$n^\beta \cdot (1+\varepsilon_2)^\ell \le n^{0.5} < \alpha n,$$

where we use the fact that β is a small enough constant depending on ε_1 and ε_2. In particular, as $|F(x,y)| \le \alpha n$, Theorem 3.2 applies to each run of $\mathsf{RandFlip}(x)$, which shows that the expectation of $|F(x,y)|$ after running $\mathsf{RandFlip}(x)$ ℓ times is at most

$$p := n^\beta \cdot (1-\varepsilon_1)^\ell \le n^{-\beta} = o(1).$$

As the algorithm outputs y whenever $|F(x,y)| = 0$, by Markov's inequality, the probability that the algorithm does not output y after running $\mathsf{RandFlip}(x)$ ℓ times is $o(1)$. And the running time of this phase is $O(\ell n^{0.5}) = o(n)$ by Lemma 3.5.

Combining the two phases shows that with probability $1-o(1)$, Algorithm 2 correctly outputs y in $O(n)$ time. □

4 Deterministic Decoding

In this section, we begin by applying the same derandomization technique as in [COSS24] to develop a deterministic algorithm that corrects γn errors when $\delta d_0 > 2$, where $\gamma = \left(1+\frac{c}{t}\right)^{-1}\frac{\delta d_0 - 1}{d_0 - 1}\alpha$. Subsequently, we introduce an additional deterministic step before this algorithm, extending the decoding radius to αn.

The main idea in [COSS24] is as follows: The vertices in $L = [n]$ are divided into a constant number of buckets based on their flipping probability p_i. It can be shown that at least one of these buckets contains a significantly higher proportion of corrupt bits than uncorrupt bits. By flipping the bits in this bucket, a small but constant fraction of the errors can be corrected.

However, the specific bucket containing the majority of corrupt bits is not known in advance. Therefore, we must recursively search through all possible choices until the number of errors is significantly reduced, as indicated by a substantial decrease in the size of $U(x)$. We prune branches where $|U(x)|$ does not decrease significantly. While this approach may appear to rely on brute force, careful analysis shows that the algorithm still runs in linear time.

The previous process requires the number of corrupt bits to be bounded by γn initially to guarantee that this number remains below αn during the search, allowing the expansion property to apply. Our key new idea is that, even if the initial number of corrupt bits exceeds γn (but remains bounded by αn), we can search through the first few steps to find a branch where the number of corrupt bits drops below γn. Although we cannot immediately verify which branch works, there is only a constant number of branches. So we can run the aforementioned decoding process on all these branches and check whether any of them produces a valid codeword.

4.1 Deterministic Flipping

We begin by modifying Algorithm 1 to obtain the following deterministic flipping algorithm.

Algorithm 3 DeterFlip(x, q)

Input: $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$ and $q \in \mathbb{R}$

$t \leftarrow \frac{d_0}{2}$
$p = (p_1, \ldots, p_n) \leftarrow (0, \cdots, 0) \in \mathbb{R}^n$
for each $v \in R$ **do**
    $w_v \leftarrow \mathsf{Decode}(x_{N(v)})$
    if $1 \le d_H(w_v, x_{N(v)}) < t$ **then**
        Choose the smallest $i \in N(v)$ where w_v and $x_{N(v)}$ differ
        $p_i \leftarrow p_i + \frac{t - d_H(w_v, x_{N(v)})}{ct}$
    end if
end for
for each $i \in [n]$ **do**
    Flip x_i if $p_i = q$
end for
return x

Algorithm 3 is derived from Algorithm 1 with the following modifications: First, it takes an additional input $q \in \mathbb{R}$ as a guess of the flipping probability. Second, instead of flipping each x_i with probability p_i, it flips x_i when p_i equals q. In particular, Algorithm 3 is deterministic.

Define the finite set

$$W \coloneqq \left\{ \frac{i}{cd_0} : i \in \mathbb{Z}, 0 \le i \le cd_0 \right\}.$$

Note that each $p_i \in [0, 1]$ is an integral multiple of $\frac{1}{2ct} = \frac{1}{cd_0}$ and, therefore, lies within W.

The following lemma shows that there exists $q \in W$ such that flipping all x_i with $p_i = q$ corrects a constant fraction of errors.

Lemma 4.1. *Assume $d_0\delta > 2$ and let $\varepsilon_0 = \frac{d_0}{2} - \frac{1}{\delta} > 0$. Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \le \alpha n$. Let $F = F(x, y)$. For $q \in W$, let P_q be the set of $i \in [n]$ such that $p_i = q$ at the end of* DeterFlip(x, q). *Then there exists $q \in W \setminus \{0\}$ such that $|P_q \cap F| - |P_q \setminus F| \ge \frac{\varepsilon_0\delta}{2ct^2}|F|$.*

Proof. Assume to the contrary that W does not contain any q satisfying the lemma. In other words, for every $q \in W \setminus \{0\}$, it holds that

$$|P_q \cap F| - |P_q \setminus F| < \frac{\varepsilon_0\delta}{2ct^2}|F| \tag{8}$$

Then we have

$$\begin{aligned}
\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i &= \sum_{q \in W} q \left(|P_q \cap F| - |P_q \setminus F| \right) \\
&\le \sum_{q \in W \setminus \{0\}} \left(|P_q \cap F| - |P_q \setminus F| \right) \\
&< (|W| - 1) \frac{\varepsilon_0\delta}{2ct^2}|F| \\
&\le \frac{\varepsilon_0\delta}{t}|F|
\end{aligned}$$

where the last two inequalities hold by (8) and the fact that $|W| - 1 = cd_0 = 2ct > 0$. On the other hand, we know $\sum_{i \in F} p_i - \sum_{i \in [n] \setminus F} p_i \ge \frac{\varepsilon_0\delta}{t}|F|$ (see (7) in the proof of Theorem 3.2). This is a contradiction. □

As $\mathsf{DeterFlip}(x, q)$ only flips the bits x_i with $i \in P_q$, we immediately derive the following corollary:

Corollary 4.2. *Under the notation and conditions in Lemma 4.1, there exists $q \in W \setminus \{0\}$ such that $\mathsf{DeterFlip}(x, q)$ corrects at least a $\frac{\varepsilon_0\delta}{2ct^2}$-fraction of corrupt bits, i.e., $|F(x', y)| \le \left(1 - \frac{\varepsilon_0\delta}{2ct^2}\right)|F(x, y)|$, where x' is the output of $\mathsf{DeterFlip}(x, q)$.*

The proofs of Lemma 3.4 and Lemma 3.5 still hold and yield the following counterparts.

Lemma 4.3. *Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$. For all $q \in W \setminus \{0\}$ and $x' = \mathsf{DeterFlip}(x, q)$, it holds that $d_H(x', y) \le (1 + \frac{c}{t})d_H(x, y)$, or equivalently, $|F(x', y)| \le (1 + \frac{c}{t})|F(x, y)|$.*

Lemma 4.4. *For all $q \in W \setminus \{0\}$, Algorithm 3 can be implemented to run in $O(|F(x, y)|)$ time, where (x, q) is the input and y is any codeword of $T(G, C_0)$.*

4.2 Search for a Sequence of q

In the following, we assume $\delta d_0 > 2$ and let $\varepsilon_0 = \frac{d_0}{2} - \frac{1}{\delta} > 0$.

The deterministic algorithm below is based on Algorithm 3. Theorem 4.6 will show that it corrects a constant fraction of corrupt bits, though it is only guaranteed to work within a decoding radius somewhat smaller than αn.

Algorithm 4 $\mathsf{DeepFlip}(x)$

Input: $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$

1: $s \leftarrow \left\lceil \frac{\log\left(\frac{\delta d_0 - 1}{2(d_0-1)}\right)}{\log(1-\varepsilon)} \right\rceil$, where $\varepsilon := \frac{\varepsilon_0\delta}{2ct^2}$ and $t = d_0/2$.
2: $k_{\min} \leftarrow |R| + 1$
3: $x_{\min} \leftarrow \perp$
4: **for** each $(q_1, \ldots, q_s) \in (W \setminus \{0\})^s$ **do**
5: $\quad x^{(0)} \leftarrow x$
6: $\quad$ **for** $i \leftarrow 1$ to s **do**
7: $\quad\quad x^{(i)} \leftarrow \mathsf{DeterFlip}(x^{(i-1)}, q_i)$
8: $\quad\quad$ **if** $|U(x^{(i)})| > c\gamma n$ **then**
9: $\quad\quad\quad$ Exit the inner loop
10: $\quad\quad$ **else if** $i = s$ and $|U(x^{(s)})| < k_{\min}$ **then**
11: $\quad\quad\quad k_{\min} \leftarrow |U(x^{(s)})|$
12: $\quad\quad\quad x_{\min} \leftarrow x^{(s)}$
13: $\quad\quad$ **end if**
14: $\quad$ **end for**
15: **end for**
16: **return** $x_{\min}$

Lemma 4.5. *Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \le \gamma n$, where $\gamma = \left(1 + \frac{c}{t}\right)^{-1} \frac{\delta d_0 - 1}{d_0 - 1}\alpha$. Suppose $\mathsf{DeepFlip}(x)$ outputs some $x' \in \mathbb{F}_2^n$. Then $|F(x', y)| \le \frac{d_0-1}{\delta d_0 - 1}\gamma n$.*

Proof. Suppose the output value $x_{\min}$ is assigned $x^{(s)}$ at Line 12 in the iteration corresponding to $(q_1, \ldots, q_s) \in (W \setminus \{0\})^s$. We have $x^{(0)} = x$, $x^{(i)} = \mathsf{DeterFlip}(x^{(i-1)}, q_i)$ for $i \in [s]$, and $x_{\min} = x^{(s)}$.

As $x^{(0)} = x$, we have $|F(x^{(0)}, y)| = |F(x, y)| \le \gamma n \le \frac{d_0-1}{\delta d_0 - 1}\gamma n$. Now, consider $i \in [s]$ and assume $|F(x^{(i-1)}, y)| \le \frac{d_0-1}{\delta d_0 - 1}\gamma n$. By Lemma 4.3,

$$|F(x^{(i)}, y)| \le \left(1 + \frac{c}{t}\right)|F(x^{(i-1)}, y)| \le \left(1 + \frac{c}{t}\right)\frac{d_0 - 1}{\delta d_0 - 1}\gamma n = \alpha n.$$

By Lemma 2.7, we have

$$|U(x^{(i)})| \geq \frac{\delta d_0 - 1}{d_0 - 1} \cdot c|F(x^{(i)}, y)|. \tag{9}$$

If $|F(x^{(i)}, y)| > \frac{d_0-1}{\delta d_0-1}\gamma n$, then by (9), we would have $|U(x^{(i)})| > c\gamma n$. In this case, the algorithm would exit the inner loop at Line 9 and modify $(q_1, \ldots, q_s)$ to a different sequence, contradicting the choice of $(q_1, \ldots, q_s)$. Therefore, $|F(x^{(i)}, y)| \leq \frac{d_0-1}{\delta d_0-1}\gamma n$.

By induction, it follows that $|F(x^{(i)}, y)| \leq \frac{d_0-1}{\delta d_0-1}\gamma n$ for $i = 0, 1, \ldots, s$. Choosing $i = s$ proves the lemma. □

Theorem 4.6. *Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \leq \gamma n$, where $\gamma = \left(1 + \frac{c}{t}\right)^{-1} \frac{\delta d_0-1}{d_0-1}\alpha$. Then $\mathsf{DeepFlip}(x)$ outputs an element $x' \in \mathbb{F}_2^n$ in $O(|F(x, y)|)$ time such that $|F(x', y)| \leq \frac{1}{2}|F(x, y)|$.*

Proof. By Corollary 4.2, there exists $(q_1, \ldots, q_s) \in (W \setminus \{0\})^s$ such that for $(x^{(0)}, \ldots, x^{(s)})$ defined by $x^{(0)} = x$ and $x^{(i)} = \mathsf{DeterFlip}(x^{(i-1)}, q_i)$, we have $|F(x^{(i)}, y)| \leq \left(1 - \frac{\varepsilon_0\delta}{2ct^2}\right)^i |F(x, y)|$ for $i \in [s]$. By Lemma 2.7, we have $|U(x^{(i)})| \leq c|F(x^{(i)}, y)| \leq c|F(x, y)| \leq c\gamma n$ for $i \in [s]$. Therefore, the iteration of the outer loop corresponding to $(q_1, \ldots, q_s)$ passes the test at Line 8 for $i \in [s]$ and reaches Line 12.

The element $x' = x_{\min}$ is chosen to minimize the number of unsatisfied checks among all branches that reach Line 12. So we have

$$|U(x')| \leq |U(x^{(s)})| \leq c|F(x^{(s)}, y)| \leq c\left(1 - \frac{\varepsilon_0\delta}{2ct^2}\right)^s |F(x, y)|. \tag{10}$$

By Lemma 4.5, we have $|F(x', y)| \leq \frac{d_0-1}{\delta d_0-1}\gamma n \leq \alpha n$. So Lemma 2.7 applies to x' and y. Finally,

$$|F(x', y)| \leq \frac{d_0 - 1}{\delta d_0 - 1} \cdot \frac{1}{c} \cdot |U(x')| \leq \frac{d_0 - 1}{\delta d_0 - 1}\left(1 - \frac{\varepsilon_0\delta}{2ct^2}\right)^s |F(x, y)| \leq \frac{1}{2}|F(x, y)|$$

where the first inequality holds by Lemma 2.7, the second one holds by (10), and the last one holds by the choices of s.

Next, we bound the time complexity. Since $|W \setminus \{0\}| = cd_0 = O(1)$ and $s = O(1)$, by Lemma 4.3 and Lemma 4.4, the running time is bounded by:

$$O\left((cd_0)^s \cdot s\left(1 + \frac{c}{d_0/2}\right)^s |F(x, y)|\right) = O(|F(x, y)|).$$ □

4.3 The Deterministic Decoding Algorithm

We now present the deterministic decoding algorithm. The idea is to enumerate all sequences $(q_1, \ldots, q_r) \in (W \setminus \{0\})^r$ of length $r = O(1)$. For each sequence, we iteratively apply $x \leftarrow \mathsf{DeterFlip}(x, q_i)$, $i = 1, \ldots, r$, aiming to reduce the number of corrupt bits.

There is guaranteed to be at least one sequence $(q_1, \ldots, q_r)$ that reduces the number of corrupt bits to below γn. While we do not know which sequence achieves this, we can enumerate all possible sequences, run $\mathsf{DeepFlip}$ repeatedly for each, and verify the final result.

In the following, let $t = d_0/2$, $\gamma = \left(1 + \frac{c}{t}\right)^{-1} \frac{\delta d_0-1}{d_0-1}\alpha$, $\varepsilon_0 = \frac{d_0}{2} - \frac{1}{\delta}$, and $\varepsilon = \frac{\varepsilon_0\delta}{2ct^2}$.

Algorithm 5 MainDecode(x)

Input: $x = (x_1, \ldots, x_n) \in \mathbb{F}_2^n$

1: $r \leftarrow \left\lceil \frac{\log \gamma}{\log(1-\varepsilon)} \right\rceil$,
2: $r' \leftarrow \lceil \log_2(\gamma n) \rceil + 1$
3: **for** each $(q_1, \ldots, q_r) \in (W \setminus \{0\})^r$ **do**
4: $\quad \hat{x} \leftarrow x$
5: $\quad$ **for** $i \leftarrow 1$ to r **do**
6: $\quad\quad \hat{x} \leftarrow \mathsf{DeterFlip}(\hat{x}, q_i)$
7: $\quad$ **end for**
8: $\quad$ **for** $i \leftarrow 1$ to r' **do**
9: $\quad\quad \hat{x} \leftarrow \mathsf{DeepFlip}(\hat{x})$
10: $\quad\quad$ **if** $|U(\hat{x})| > c2^{-i}\gamma n$ **then**
11: $\quad\quad\quad \hat{x} \leftarrow \perp$
12: $\quad\quad\quad$ Exit the inner loop
13: $\quad\quad$ **end if**
14: $\quad$ **end for**
15: $\quad$ **return** $\hat{x}$ **if** $\hat{x} \neq \perp$ and $|U(\hat{x})| = 0$ and $d_H(x, \hat{x}) \leq \alpha n$
16: **end for**

Theorem 4.7. *Assume $\delta d_0 > 2$. Then Algorithm 5 can be implemented to correct up to αn errors in $O(n)$ time for the code $T(G, C_0)$.*

Proof. We first prove the correctness of Algorithm 5. Let $x \in \mathbb{F}_2^n$ and $y \in T(G, C_0)$ such that $d_H(x, y) \leq \alpha n$.

By Corollary 4.2, there exists a sequence $(q_1, \ldots q_r) \in (W \setminus \{0\})^r$ such that after completing the first inner loop (Lines 5–7) with respect to this sequence, we have $|F(\hat{x}, y)| \leq \gamma n$. Fix this sequence $(q_1, \ldots q_r)$ and consider the corresponding iteration of the outer loop (assuming it is executed). By Theorem 4.6, $|F(\hat{x}, y)|$ is reduced by at least half each time we apply $\hat{x} \leftarrow \mathsf{DeepFlip}(\hat{x})$ (Line 9). By Lemma 2.7, for $i \in [r']$, we have $|U(\hat{x})| \leq c|F(\hat{x}, y)| \leq c2^{-i}\gamma n$ at Line 10 in the i-th iteration of the second inner loop (Lines 8–13), which guarantees that the algorithm does not exit this inner loop at line 12. Finally, after completing the second inner loop, $|F(\hat{x}, y)|$ will be reduced to at most $2^{-r'}\gamma n < 1$, which implies that $F(\hat{x}, y) = 0$ and hence $\hat{x} = y$. So y is output at Line 15.

This is assuming that the iteration corresponding to $(q_1, \ldots q_r)$ is executed. However, the algorithm may terminate at Line 15 earlier and output some $\hat{x} \in \mathbb{F}_2^n$ satisfying $|U(\hat{x})| = 0$ and $d_H(x, \hat{x}) \leq \alpha n$. The former condition means $\hat{x} \in T(G, C_0)$. In this case, the output is still $\hat{x} = y$ since αn is less than half of the minimum distance of $T(G, C_0)$ [DG18]. This proves the correctness of Algorithm 5.

Next, we bound the time complexity. Note that $r, r', |W \setminus \{0\}| = O(1)$. By Lemma 4.4, the first inner loop runs in $O(n)$ time. The condition on $|U(\hat{x})|$ at Line 10 ensures that at the end of the i-th iteration of the second inner loop (Lines 5–7), we have $|U(\hat{x})| \leq c2^{-i}\gamma n$ and, consequently, $|F(x, y)| \leq d|U(\hat{x})| \leq dc2^{-i}\gamma n$. By Theorem 4.6, the of second inner loop (Lines 5–7) also runs in $O(n + \sum_{i=1}^{r'} dc2^{-i}\gamma n) = O(n)$ time. It follows that Algorithm 5 runs in $O(n)$ time. □

5 Distance and Decoding Radius

5.1 Size-Expansion Trade-off

In this subsection, we briefly review the size-expansion trade-off introduced in [CCLO23], which will be used later in this section. We also compare these results with related work and establish relevant notation.

Recall that a (c, d, α, δ)-bipartite expander $G = (L \cup R, E)$ satisfies the condition that for any $S \subseteq L$ with $|S| \leq \alpha n$, we have $|N(S)| \geq \delta c|S|$. This raises a natural question: Given this condition, can we infer a positive expansion factor δ' for a larger set $S \subseteq L$ of size $k\alpha n$ with $k > 1$?

To begin, we establish a trivial lower bound, $\delta' \geq \frac{\delta}{k}$, which was also used in the proof of [DG18, Lemma 1(b)]. Consider $S \subseteq L$ of size $k\alpha n$, and let $S' \subseteq S$ be an arbitrary subset of size αn. Since every neighbor of S' is also a neighbor of S, it follows that $|N(S)| \geq |N(S')| \geq \delta c|S'| = \delta c\alpha n = \frac{\delta}{k} c|S|$. Thus, choosing $\delta' = \frac{\delta}{k}$ provides a valid lower bound on the expansion factor of S.

A tighter bound was established in [CCLO23] using an expectation argument. The main idea is to choose S' uniformly at random among all subsets of S of size αn. Suppose, for the sake of contradiction, that $|N(S)|$ is too small. This would result in an upper bound on $\mathbb{E}[|N(S')|]$ that is too strong, contradicting the fact that every subset S' of size αn must have an expansion factor of at least δ.

We now formalize this argument. Assume n is large enough such that $d \leq k\alpha n - \alpha n$. Consider $i \in \{0, 1 \ldots, d\}$ and let $\beta_i \coloneqq \frac{|N_i(S)|}{c\alpha n}$ (i.e., $|N_i(S)| = \beta_i c\alpha n$). For each $v \in N_i(S)$, we calculate the probability of v being a neighbor of S':

$$\Pr[v \in N(S')] = 1 - \Pr[S' \cap N(v) = \emptyset] = 1 - \frac{\binom{k\alpha n - i}{\alpha n}}{\binom{k\alpha n}{\alpha n}} = 1 - \left(1 - \frac{1}{k}\right)^i + O\left(\frac{1}{n}\right),$$

where the last equality holds by Lemma A.2.

Therefore, the expected number of neighbors of S' satisfies:

$$\mathbb{E}[|N(S')|] = \sum_{i=1}^{d} \sum_{v \in N_i(S)} \Pr[v \in N(S')] = O\left(\frac{1}{n}\right) + \sum_{i=1}^{d} \left(1 - \left(1 - \frac{1}{k}\right)^i\right) \cdot \beta_i c\alpha n.$$

On the other hand, we have $\mathbb{E}[|N(S')|] \geq \delta\alpha n$ by the expansion property. So the parameters β_i must satisfy the following constraint:

$$O\left(\frac{1}{n}\right) + \sum_{i=1}^{d} \left(1 - \left(1 - \frac{1}{k}\right)^i\right) \cdot \beta_i c\alpha n \geq \delta\alpha n. \tag{11}$$

Another constraint can be derived by double counting the number of edges in $E(S, N(S))$:

$$c \cdot k\alpha n = |E(S, N(S))| = \sum_{i=1}^{d} \beta_i c\alpha n \cdot i \tag{12}$$

Also, by definition, the number of neighbors of S is:

$$|N(S)| = \sum_{i=1}^{d} |N_i(S)| = c\alpha n \cdot \sum_{i=1}^{d} \beta_i \tag{13}$$

Thus, subject to the constraints (11) and (12) in the variables β_i, the minimum of $c\alpha n \cdot \sum_{i=1}^{d} \beta_i$ is a lower bound on $|N(S)|$. Consequently, subject to the constraints (11) and (12), the minimum of $\frac{1}{k}\sum_{i=1}^{d} \beta_i$ is a lower bound on the expansion factor $\frac{|N(S)|}{c|S|} = \frac{|N(S)|}{ck\alpha n}$ of S.

This can be formalized as a linear program (LP), as shown in [CCLO23, Equations (4) and (5)].

Definition 5.1 (Degree-Dependent Size-Expansion Function [CCLO23])**.** *For $k > 1$, define $f_{d,\delta}(k)$ as the optimal value of the following LP:*

$$\begin{aligned} \textit{minimize} \quad & \frac{1}{k}\sum_{i=1}^{d} \beta_i \\ \textit{subject to} \quad & \sum_{i=1}^{d} i \cdot \beta_i = k, \\ & \sum_{i=1}^{d} \left(1 - \left(1 - \frac{1}{k}\right)^i\right) \cdot \beta_i \geq \delta, \\ & \beta_i \geq 0, \quad \forall i \in [d]. \end{aligned}$$

We call $f_{d,\delta}$ the Degree-Dependent Size-Expansion Function.

Note that the $O\left(\frac{1}{n}\right)$ term in the constraint (11) is omitted in this definition. However, we will see in Lemma 5.3 below that this omission has a negligible effect on the LP value.

To simplify our discussion, we focus on the following LP, where the summation range is extended from $[1, d]$ to $[1, +\infty)$.

Definition 5.2 (Size-Expansion Function)**.** *For $k > 1$, define $f_\delta(k)$ as the optimal value of the following LP.*

$$\begin{aligned} \textit{minimize} \quad & \frac{1}{k}\sum_{i=1}^{\infty} \beta_i \\ \textit{subject to} \quad & \sum_{i=1}^{\infty} i \cdot \beta_i = k, \\ & \sum_{i=1}^{\infty} \left(1 - \left(1 - \frac{1}{k}\right)^i\right) \cdot \beta_i \geq \delta, \\ & \beta_i \geq 0, \quad \forall i \in \mathbb{N}^+. \end{aligned}$$

We call f_δ the Size-Expansion Function.

This modification represents a slight relaxation: The value of the new LP is less than or equal to the original, i.e., $f_\delta(k) \leq f_{d,\delta}(k)$, resulting in a formally weaker bound on the expansion factor. This is because any feasible solution to the original LP (with $\beta_i = 0$ for all $i > d$) remains feasible for the new LP, preserving the same objective function value.

Nevertheless, Lemma 5.3 below states that $f_{d,\delta}(k) = \max\left\{f_\delta(k), \frac{1}{d}\right\}$. In particular, the two are equal when $f_\delta(k) \geq 1/d$. For this reason, we focus on the simpler function f_δ, which does not depend on the parameter d.

Lemma 5.3. *The functions f_δ and $f_{d,\delta}$ satisfy the following properties:*

1. *f_δ and $f_{d,\delta}$ are non-increasing.*

2. $f_{d,\delta}(k) = \max\left\{f_\delta(k), \frac{1}{d}\right\}$.

3. $f_\delta(k) \geq \frac{\delta}{k}$.

4. *Omitting the* $O\left(\frac{1}{n}\right)$ *term in the constraint* (11) *increases the value of* $f_{d,\delta}(k)$ *by at most* $O\left(\frac{1}{n}\right)$.

5. $f_\delta(k)$ *is continuous with* $\lim_{k\to 1} f_\delta(k) = \delta$ *and* $\lim_{k\to\infty} f_\delta(k) = 0$.

Lemma 5.3 is proved in Appendix A.3. The proof also provides explicit descriptions of the functions $f_\delta(k)$ and $f_{d,\delta}(k)$ as piecewise rational functions; see (24) and (25).

Note that f_δ and $f_{d,\delta}$ are non-decreasing by Lemma 5.3. In particular, $f_\delta(k)$ is not only a lower bound for the expansion factor of sets of size $k\alpha n$, but also for smaller sets. By definition, this implies the following result.

Lemma 5.4 (Size-Expansion Trade-off [CCLO23])**.** *For* $k > 1$*, a* (c, d, α, δ)*-bipartite expander with* n *left vertices is also a* $(c, d, k\alpha, f_\delta(k) - O(\frac{1}{n}))$*-bipartite expander.*

The following result was also proved in [CCLO23].

Lemma 5.5 ([CCLO23])**.** $f_\delta(k)$ *can be efficiently computed as a piecewise rational function. Specifically,* $f_\delta(k) = 1 - (1-\delta)k$ *when* $k \leq \frac{1}{2(1-\delta)}$ *(or equivalently, when* $f_\delta(k) \geq \frac{1}{2}$*).*

Fig. 1 shows the Size-Expansion Function f_δ for various values of δ, along with the trivial lower bound $\frac{\delta}{k}$ when $\delta = 0.8$. As the figure illustrates, the bound $f_\delta(k)$ is always tighter than $\frac{\delta}{k}$ for $k > 1$.

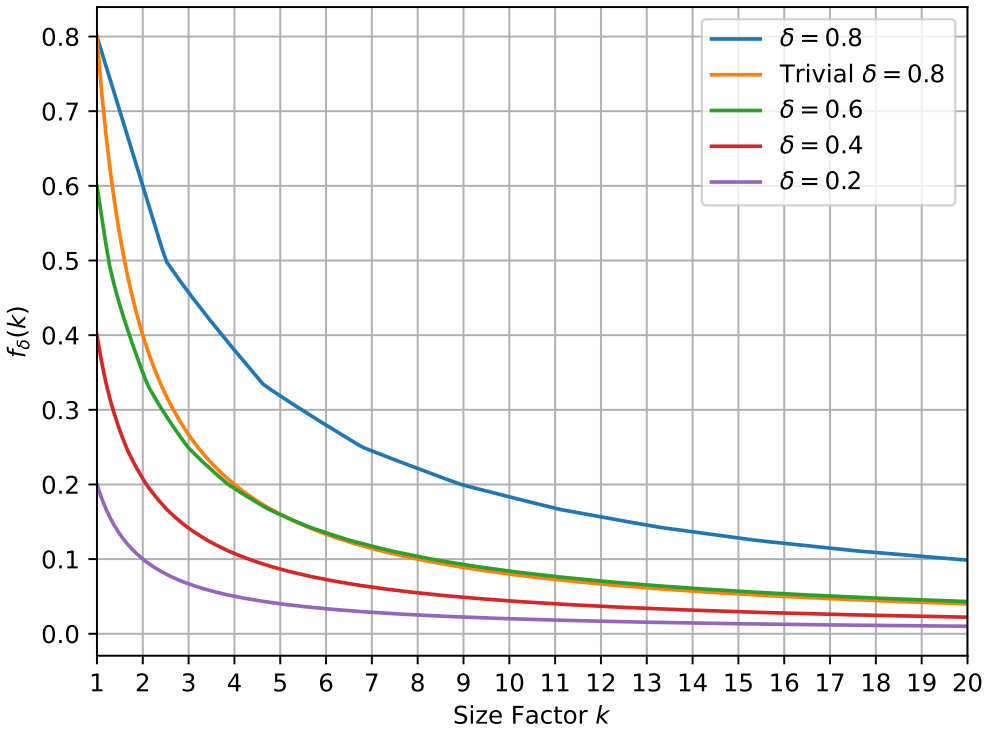

Figure 1: The Size-Expansion Function f_δ. For $\delta = 0.8$, the trivial bound $\frac{\delta}{k}$ is also shown.

5.2 Bounds on Distance and Decoding Radius

The following theorem gives a lower bound on the minimum distance of a Tanner code $T(G, C_0)$.

Theorem 5.6. *Suppose $\delta d_0 > 1$. The minimum distance of the Tanner code $T(G, C_0)$ is greater than $f_\delta^{-1}\left(\frac{1}{d_0} + \varepsilon\right)\alpha n$ for any constant*[3] *$\varepsilon \in (0, \delta - \frac{1}{d_0})$ and all sufficiently large n.*

Proof. Let $\delta' = \frac{1}{d_0} + \varepsilon$, $k = f_\delta^{-1}(\delta')$, and $\alpha' = k\alpha$. By Lemma 5.3, $f_\delta^{-1}(\delta')$ is well-defined.

Since G is a (c, d, α, δ)-bipartite expander and $k\alpha = \alpha'$, $f_\delta(k) = \delta'$, applying Lemma 5.4 implies that G is also a (c, d, α', δ')-bipartite expander. By Lemma 2.9 and the fact that $\delta' d_0 = 1 + d_0\varepsilon > 1$, we have that the distance of $T(G, C_0)$ is greater than α', thus proving this theorem. □

Remark 1. Theorem 5.6 generalizes [CCLO23, Theorem 3.1], which establishes a lower bound of $\frac{\alpha}{2(1-\delta)}n$ on the minimum distance of expander codes, where $d_0 = 2$. To see that Theorem 5.6 recovers the this result, note that when $d_0 = 2$, we have $f_\delta(k) = 1 - (1-\delta)k$ when $f_\delta(k) \geq \frac{1}{2} = \frac{1}{d_0}$ by Lemma 5.4. Therefore, $f_\delta^{-1}\left(\frac{1}{d_0}\right) = f_\delta^{-1}\left(\frac{1}{2}\right) = \frac{1}{2(1-\delta)}$. Substituting this into the bound gives $f_\delta^{-1}\left(\frac{1}{d_0}\right)\alpha n = \frac{\alpha}{2(1-\delta)}n$.

Remark 2. If the Size-Expansion Function in Theorem 5.6 is replaced with the trivial bound $\frac{\delta}{k}$, the resulting distance bound is $\delta d_0 \alpha n$, as proved in [DG18, Lemma 1(b)]. However, this trivial bound is strictly weaker than the bound provided by Theorem 5.6.

The relationship between the distance bound and the parameters δ and d_0 is illustrated in the following figure:

[3] It is likely that the statement holds for sub-constant ε as well. For simplicity, we do not discuss it further.

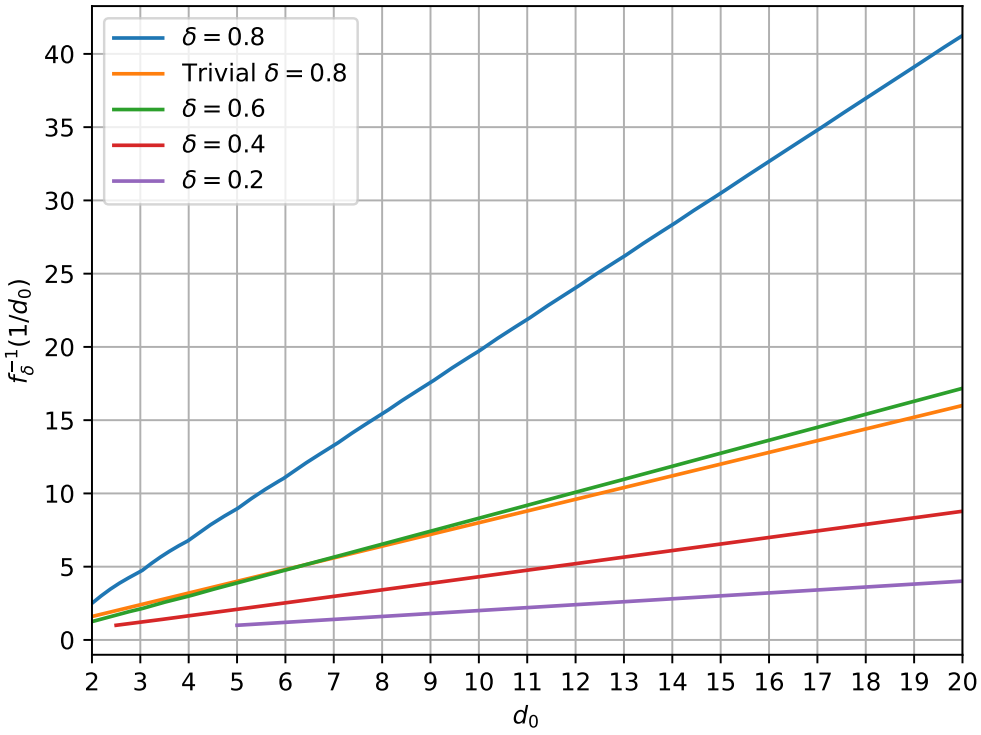

Figure 2: Plot of $f_\delta^{-1}\left(\frac{1}{d_0}\right)$. For $\delta = 0.8$, the factor δd_0 from the trivial bound $\delta d_0 \alpha n$ is also shown.

We have the following theorem on the decoding radius of our previous algorithm.

Theorem 5.7. *Suppose $\delta d_0 > 2$. Algorithm 2 and Algorithm 5 can decode up to $f_\delta^{-1}\left(\frac{2}{d_0} + \varepsilon\right)\alpha n$ errors in $O(n)$ time for any constant $\varepsilon \in (0, \delta - \frac{2}{d_0})$ and all sufficiently large n.*

Proof. Let $\delta' = \frac{2}{d_0} + \varepsilon$, $k = f_\delta^{-1}(\delta')$, and $\alpha' = k\alpha$. By Lemma 5.3, $f_\delta^{-1}(\delta')$ is well-defined.

Since G is a (c, d, α, δ)-bipartite expander and $k\alpha = \alpha'$, $f_\delta(k) = \delta'$, applying Lemma 5.4 implies that the graph G is also a (c, d, α', δ')-bipartite expander. Since $\delta' d_0 = 2 + d_0\varepsilon > 2$, it follows from Theorem 3.6 and Theorem 4.7 that Algorithm 2 and Algorithm 5 can both decode up to $\alpha' n$ errors in linear time. □

Remark 3. If the Size-Expansion Function in Theorem 5.7 is replaced with the trivial bound $\frac{\delta}{k}$, the resulting decoding radius becomes $\frac{\delta d_0}{2}\alpha n$. This bound is strictly weaker than the one derived in Theorem 5.7.

The relationship between the decoding radius and the parameters δ and d_0 is illustrated in the following figure:

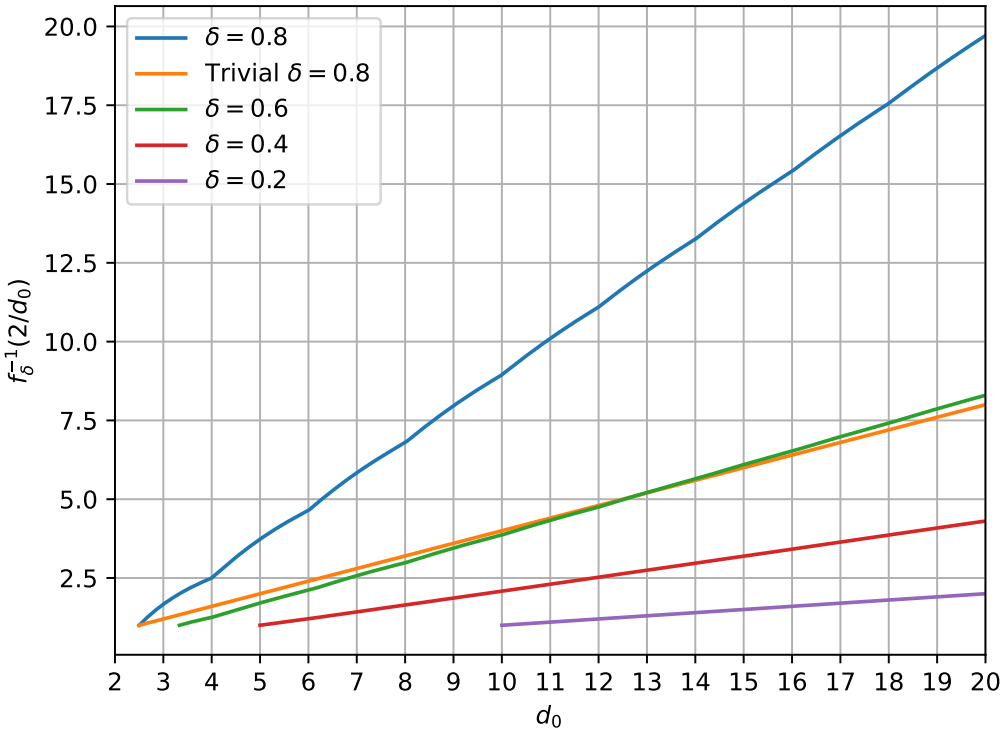

Figure 3: Plot of $f_\delta^{-1}\left(\frac{2}{d_0}\right)$. For $\delta = 0.8$, the factor $\frac{\delta d_0}{2}$ from the trivial bound $\frac{\delta d_0}{2}\alpha n$ is also shown.

5.3 Tightness of the Distance Bound

The following theorem, Theorem 5.8, demonstrates that the distance bound in Theorem 1.3 is essentially tight. This theorem follows [CCLO23, Theorem 3.2] but extends it in two significant ways. First, while [CCLO23, Theorem 3.2] establishes the existence of infinitely many expander codes ($d_0 = 2$) with a minimum distance of $\frac{\alpha}{2(1-\delta)}n$, the underlying graphs are only "almost regular" bipartite expanders. Theorem 5.8 extends the result to strictly regular bipartite expanders. It also applies to Tanner codes $T(G, C_0)$ with $d_0 > 2$.

Theorem 5.8. *Given any constants $\delta, d_0, \varepsilon > 0$ with $\delta d_0 > 1$, there exist constants c, d, and α such that for infinitely many values of n, a $(c, d, \alpha, \delta - \varepsilon)$-bipartite expander G with n left vertices exists. Moreover, for any linear code $C_0 \subseteq \mathbb{F}_2^d$ of minimum distance d_0, there exists such a graph G such that, by fixing an appropriate total ordering on $N(v)$ for each $v \in R(G)$, the minimum distance of the resulting Tanner code $T(G, C_0)$ is at most $f_\delta^{-1}\left(\frac{1}{d_0}\right)\alpha n$.*

Proof. We denote the left and right parts of a bipartite graph G by $L(G)$ and $R(G)$, respectively. Let $k = f_\delta^{-1}\left(\frac{1}{d_0}\right)$.

We construct two bipartite graphs:

- G_0 is a uniformly chosen random (c, d_0)-regular graph with $|L(G_0)| = k\alpha n$ and $|R(G_0)| = cn\frac{k\alpha}{d_0}$.

- G_1 is a uniformly chosen random graph subject to the constraints that (1) G_1 is left-regular of degree c with $|L(G_1)| = (1 - k\alpha)n$ and $|R(G_1)| = cn\frac{1}{d}$, and (2) $cn\frac{k\alpha}{d_0}$ right vertices of G_1 have degree $d - d_0$, while the remaining ones have degree d. The parameter α is chosen to satisfy $\alpha \leq \frac{d_0}{dk}$ so that this is possible.

We formalize the algorithm used to generate the regular graphs G_0 and G_1 as Algorithm 6 in Appendix A.1.

Next, we merge G_0 and G_1 into a single bipartite graph G. Specifically, for each vertex $u \in R(G_0)$, we pair it with a vertex $v \in R(G_1)$ of degree $d - d_0$, merging u and v into a single vertex. All other vertices in G_0 and G_1 remain unchanged. This process results in the final graph G.

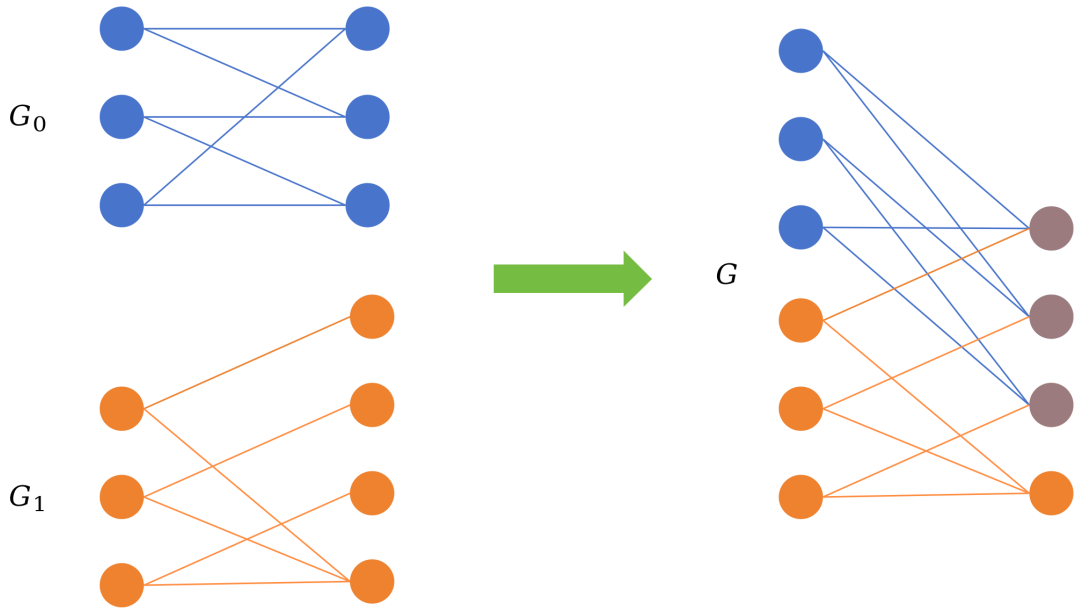

Figure 4: An illustration of merging G_0 and G_1, where $d_0 = 2$ and $d = 3$.

Assume without loss of generality that $1^{d_0}0^{d-d_0} \in C_0$. Then, for each merged vertex $v \in R(G)$, we may order its d neighbors such that the first d_0 neighbors are in $L(G_0)$. Then the unique vector in $\mathbb{F}_2^{L(G)}$ with support $L(G_0)$ is a codeword of $T(G, C_0)$, implying that the minimum distance of $T(G, C_0)$ is at most $|L_0(G)| = k\alpha n$.

It remains to prove that G is a $(c, d, \alpha, \delta - \varepsilon)$-bipartite expander with high probability. Fix a set $S \in L(G)$ with $|S| \leq \alpha n$. Let $S_0 \coloneqq S \cap L(G_0)$ and $S_1 \coloneqq S \cap L(G_1)$. Suppose $|S_0| = \alpha_0 n$ and $|S_1| = \alpha_1 n$, where $\alpha_0 + \alpha_1 =: \alpha' \leq \alpha$. Next, we bound the probability that $N(S)$ is small.

Bounding the probability when S is large. Assume $\gamma n \leq |S| \leq \alpha n$, where $\gamma = O_{\delta,\alpha,d_0,d}(1)$ is a small constant to be determined later. Denote the set of merged vertices in $R(G)$ as M. We

calculate the expected size of $N(S)$ using Azuma's inequality.

$$\begin{aligned}\mathbb{E}[N(S)] &= \sum_{u\in R(G)} \Pr[u \in N(S)] \\ &= |R(G)| - \sum_{u\in R(G), u\in M} \Pr[u \notin N(S)] - \sum_{u\in R(G), u\notin M} \Pr[u \notin N(S)] \\ &\geq cn\left(\frac{1}{d} - \frac{k\alpha}{d_0}\left(1-\frac{\alpha_0}{k\alpha}\right)^{d_0}\left(1-\frac{\alpha_1}{1-k\alpha}\right)^{d-d_0} - \left(\frac{1}{d}-\frac{k\alpha}{d_0}\right)\left(1-\frac{\alpha_1}{1-k\alpha}\right)^{d}\right), \qquad (14)\end{aligned}$$

where the last step holds since for $u \in R(G)$,

$$\Pr[u \notin N(S)] \leq \begin{cases} \left(1-\frac{\alpha_0}{k\alpha}\right)^{d_0}\left(1-\frac{\alpha_1}{1-k\alpha}\right)^{d-d_0} & u \in M, \\ \left(1-\frac{\alpha_1}{1-k\alpha}\right)^{d} & u \notin M. \end{cases}$$

We claim (14) is bounded from below by

$$cn\left(\frac{1}{d} - \frac{k\alpha}{d_0}\left(1-\delta d_0\frac{\alpha_0}{k\alpha}\right) - \left(\frac{1}{d}-\frac{k\alpha}{d_0}\right)\left(1-\alpha_1 d + O_{k,d,d_0}(\alpha'^2)\right)\right) \qquad (15)$$

$$= cn\left(\delta\alpha_0 + \alpha_1 - O_{k,d,d_0}(\alpha'^2)\right). \qquad (16)$$

To see this, we first prove the following claim.

Claim 5.9. $\left(1-\frac{1}{k}\right)^{d_0} \leq 1 - \delta d_0 \frac{1}{k}$.

Proof. Assume to the contrary that the claim is not true. Then $\left(1-\frac{1}{k}\right)^{d_0} = 1-(1-\eta)\delta d_0 \cdot \frac{1}{k}$ for some constant $\eta > 0$.

Define the function $h(x) \coloneqq \left(1-\frac{1}{k}\right)^x + (1-\eta)\delta\frac{x}{k} - 1$. Then $h(0) = 0$ and $h(d_0) = 0$. Note that $h(x)$ is a convex function. So if $h(t) < 0$ for some $t \geq d_0$, then we would have $h(d_0) < 0$, yielding a contradiction. Therefore, $h(t) \geq 0$ for all $t \geq d_0$, or equivalently,

$$1 - \left(1-\frac{1}{k}\right)^t \leq (1-\eta)\delta\frac{t}{k}, \quad \forall t \geq d_0. \qquad (17)$$

By the proof of Lemma 5.3, the optimal solution $(\hat{\beta}_i)_{i\in\mathbb{N}^+}$ of $f_\delta(k)$ is supported on at most two adjacent indices. That is, the support is either $\{j, j+1\}$ or a singleton $\{j\}$ for some integer $j \in \mathbb{N}^+$.

Case 1: $j \geq d_0$. By the constraints from Definition 5.2 and inequality (17), we have

$$\begin{aligned}\delta &\leq \sum_{i=1}^{\infty}\left(1-\left(1-\frac{1}{k}\right)^i\right)\cdot\hat{\beta}_i \\ &= \sum_{i=d_0}^{\infty}\left(1-\left(1-\frac{1}{k}\right)^i\right)\cdot\hat{\beta}_i \\ &\leq (1-\eta)\frac{\delta}{k}\sum_{i=d_0}^{\infty} i\cdot\hat{\beta}_i \\ &= (1-\eta)\frac{\delta}{k}\sum_{i=1}^{\infty} i\cdot\hat{\beta}_i \\ &= (1-\eta)\delta,\end{aligned}$$

which yields a contradiction since $(1-\eta)\delta < \delta$.

Case 2: $j < d_0$. By the constraints from Definition 5.2, either $k = j\hat{\beta}_j + (j+1)\hat{\beta}_{j+1}$ or $k = j\hat{\beta}_j$, depending or whether the support of the optimal solution is $\{j, j+1\}$ or $\{j\}$.

In the former case, we have $k < d_0(\hat{\beta}_j + \hat{\beta}_{j+1}) = kd_0 f_\delta(k)$. Equivalently, $f_\delta(k) > \frac{1}{d_0}$. However, this contradicts the definition $k = f_\delta^{-1}(\frac{1}{d_0})$.

In the latter case, we have $k < d_0\hat{\beta}_j = kd_0 f_\delta(k)$. This similarly leads to a contradiction.

Since both cases lead to a contradiction, the claim is proved. □

By the convexity of the function $g(x) \coloneqq (1-x)^{d_0}$ for $x \in [0,1]$, we know that

$$\left(1 - \frac{\alpha_0}{k\alpha}\right)^{d_0} = g\left(\frac{\alpha_0}{k\alpha}\right) \le \left(1 - \frac{\alpha_0}{\alpha}\right) g(0) + \frac{\alpha_0}{\alpha} g\left(\frac{1}{k}\right) = 1 - \frac{\alpha_0}{\alpha} + \frac{\alpha_0}{\alpha}(1 - \frac{1}{k})^{d_0} \le 1 - \delta d_0 \frac{\alpha_0}{k\alpha}, \quad (18)$$

where the last inequality holds by Claim 5.9. By (18), we see that (14), and consequently $\mathbb{E}[N(S)]$, is bounded from below by (15).

By choosing $\alpha \ge \alpha'$ to be sufficiently small (but still a constant depending on k, d, d_0, ε), we can ensure that the term $O_{k,d,d_0}(\alpha'^2)$ in (16) is at most $\alpha'\frac{\varepsilon}{2}$. Thus, as $|S| = \alpha_0 n + \alpha_1 n = \alpha' n$, we have

$$\mathbb{E}[N(S)] \ge cn\left(\delta\alpha_0 + \alpha_1 - \alpha'\frac{\varepsilon}{2}\right) \ge \left(\delta - \frac{\varepsilon}{2}\right) c|S|.$$

Now, consider the process of revealing the edges going out of S one by one, in an arbitrary order. For $i = 0, 1, \ldots, c|S|$, let

$$X_i = \mathbb{E}[N(S) \mid \text{The first } i \text{ edges going out of } S \text{ are revealed}].$$

Clearly, the random variables X_i form a martingale, and $|X_{i+1} - X_i| \le 1$. By Azuma's inequality (Lemma 2.5), for any $\lambda > 0$,

$$\Pr[|X_{c|S|} - X_0| \ge \lambda] \le 2\exp\left(-\frac{\lambda^2}{2c|S|}\right).$$

Choose $\lambda = \sqrt{\log\left(2n^2\binom{n}{|S|}\right) \cdot 2c|S|} = O\left(\sqrt{c|S|\log n + c|S|^2 \log\frac{1}{\gamma}}\right) = O_\gamma(\sqrt{c}|S|)$. Also choose sufficiently large $c = \Omega_\gamma(\frac{1}{\varepsilon^2})$ such that $\lambda \le \frac{\varepsilon}{2}c|S|$. Then we have

$$\Pr[N(S) \le (\delta - \varepsilon)c|S|] \le \Pr\left[|N(S) - \mathbb{E}[N(S)]| \ge \lambda\right] \le \left(n^2\binom{n}{|S|}\right)^{-1}. \quad (19)$$

Bounding the probability when S is small. Assume $1 \le |S| < \gamma n$. We claim that with high probability, $N(S) \ge \delta c|S|$. By the union bound,

$$\begin{aligned}
\Pr[|N(S)| \le \delta c|S|] &\le \sum_{\substack{T \subseteq R(G), \\ |T| = \delta c|S|}} \Pr[N(S) \subseteq T] \\
&\le \binom{\frac{c}{d}n}{\delta c|S|} \cdot \max\left\{\frac{d \cdot \delta c|S|}{cnk\alpha}, \frac{d \cdot \delta c|S|}{(1-k\alpha)nc}\right\}^{c|S|} \\
&= \binom{\frac{c}{d}n}{\delta c|S|} \cdot \max\left\{\frac{d \cdot \delta|S|}{nk\alpha}, \frac{d \cdot \delta|S|}{(1-k\alpha)n}\right\}^{c|S|},
\end{aligned}$$

where $\max\left\{\frac{d\cdot\delta c|S|}{cnk\alpha}, \frac{d\cdot\delta c|S|}{(1-k\alpha)nc}\right\}^{c|S|}$ is an upper bound on $\Pr[N(S)\subseteq T]$.

Using the standard estimate $\binom{n}{m} \le \left(\frac{ne}{m}\right)^m$, we obtain

$$\begin{aligned}\Pr[|N(S)| \le \delta|S|] &\le \left(\frac{ne}{d\delta|S|}\right)^{\delta c|S|} \cdot \max\left\{\frac{d\cdot\delta|S|}{nk\alpha}, \frac{d\cdot\delta|S|}{(1-k\alpha)n}\right\}^{c|S|} \\ &= \left(O_{\delta,\alpha,d_0,d}(1)\cdot\left(\frac{|S|}{n}\right)^{1-\delta}\right)^{c|S|}\end{aligned}$$

Choose γ to be a sufficiently small constant depending on δ, α, d_0, and d. And choose c to be a sufficiently large constant depending on δ.[4] We have

$$\Pr[|N(S)| \le \delta|S|] \le \left(O_{\delta,\alpha,d_0,d}(1)\cdot\gamma^{\frac{1-\delta}{2}}\left(\frac{|S|}{n}\right)^{\frac{1-\delta}{2}}\right)^{c|S|} \le \left(\frac{|S|}{en}\right)^{\frac{1-\delta}{2}c|S|} \le \left(\frac{|S|}{en}\right)^{2|S|}. \tag{20}$$

Union bound. Combining the bounds (19) and (20) and taking the union bound over all S, we see that the probability that $N(S) \ge (\delta-\varepsilon)c|S|$ for all $S \subseteq L(G)$ of size at most αn is at least

$$\begin{aligned}&1 - \Pr[\exists\, S\subseteq L(G) \text{ such that } |S|\in[\gamma n,\alpha n] \text{ and } N(S)\le(\delta-\varepsilon)cn] \\ &\quad - \Pr[\exists\, S\subseteq L(G) \text{ such that } |S|\in[1,\gamma n) \text{ and } N(S)\le(\delta-\varepsilon)cn] \\ &\ge 1 - \sum_{i=\gamma n}^{n}\binom{n}{i}\left(n^2\binom{n}{i}\right)^{-1} - \sum_{i=1}^{\gamma n-1}\binom{n}{i}\left(\frac{i}{en}\right)^{2i} \\ &\ge 1 - \sum_{i=\gamma n}^{n}\binom{n}{i}\left(n^2\binom{n}{i}\right)^{-1} - \sum_{i=1}^{\gamma n-1}\left(\frac{ne}{i}\right)^{i}\left(\frac{i}{en}\right)^{2i} \\ &\ge 1 - o(1).\end{aligned}$$

Therefore, G is a $(c,d,\alpha,\delta-\varepsilon)$-bipartite expander with high probability. □

6 Final Remarks

There remains a gap between the sufficient condition $\delta d_0 > 2$ required by our algorithms and the necessary condition $\delta d_0 > 1$ established in [COSS24]. We note that if δd_0 is not greater than 2, then there exists a set $S \subseteq [n]$ such that $|N(S)| \le \frac{2c}{d_0}|S|$. When S is the set of corrupt bits, a random vertex $v \in N(S)$ is expected to have $\frac{c|S|}{|N(S)|} \ge \frac{d_0}{2}$ neighbors that are corrupt. So we cannot expect the local word $x|_{N(v)}$ to be uniquely decodable. This suggests that $\delta d_0 > 2$ might be the best achievable condition using the approach of [DG18, COSS24], where algorithms depend solely on the unique decoding of the inner code.

However, local words might still provide useful information even when unique decoding of the inner code is not feasible. For instance, the flip algorithm of Sipser and Spielman [SS96] works in a setting where the inner code is a parity-check code. In this case, $d_0 = 2$, making unique decoding of the inner code impossible. Nevertheless, the flip algorithm proves to be effective, requiring only that $\delta \ge \frac{3}{4}$. This observation raises the question of whether insights from this special case can be generalized to achieve improved results for $d_0 > 2$.

[4] Due to the proof for large sets S, the parameter c also needs to depend on γ and ε.

Acknowledgments

The first author thanks Xue Chen for explaining their results [CCLO23]. The second author was supported by the NSF CAREER award CCF-2440926. He thanks Chong Shangguan and Yuanting Shen for helpful discussions and for explaining their results [COSS24], and Zihan Zhang for additional discussions.

References

[CCLO23] Xue Chen, Kuan Cheng, Xin Li, and Minghui Ouyang. Improved decoding of expander codes. *IEEE Transactions on Information Theory*, 69(6):3574–3589, 2023.

[CNVM10] Shashi Kiran Chilappagari, Dung Viet Nguyen, Bane Vasic, and Michael W. Marcellin. On trapping sets and guaranteed error correction capability of LDPC codes and GLDPC codes. *IEEE Transactions on Information Theory*, 56(4):1600–1611, 2010.

[COSS24] Kuan Cheng, Minghui Ouyang, Chong Shangguan, and Yuanting Shen. When can an expander code correct $\Omega(n)$ errors in $O(n)$ time? In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2024)*, pages 61:1–61:23, 2024.

[DG18] Michael Dowling and Shuhong Gao. Fast decoding of expander codes. *IEEE Transactions on Information Theory*, 64(2):972–978, 2018.

[FMS+07] Jon Feldman, Tal Malkin, Rocco A. Servedio, Cliff Stein, and Martin J. Wainwright. LP decoding corrects a constant fraction of errors. *IEEE Transactions on Information Theory*, 53(1):82–89, 2007.

[For66] G. Forney. Generalized minimum distance decoding. *IEEE Transactions on Information Theory*, 12(2):125–131, April 1966.

[FWK05] J. Feldman, M.J. Wainwright, and D.R. Karger. Using linear programming to decode binary linear codes. *IEEE Transactions on Information Theory*, 51(3):954–972, 2005.

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

[MU17] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, USA, 2nd edition, 2017.

[RU01] T.J. Richardson and R.L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Transactions on Information Theory*, 47(2):599–618, 2001.

[SR03] V. Skachek and R.M. Roth. Generalized minimum distance iterative decoding of expander codes. In *Proceedings 2003 IEEE Information Theory Workshop (Cat. No.03EX674)*, pages 245–248, 2003.

[SS96] Michael Sipser and Daniel A Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996.

[Tan81] R. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1981.

[Vid13a] Michael Viderman. Linear-time decoding of regular expander codes. *ACM Trans. Comput. Theory*, 5(3), 2013.

[Vid13b] Michael Viderman. LP decoding of codes with expansion parameter above 2/3. *Inf. Process. Lett.*, 113(7):225–228, April 2013.

[ZP75] V. V. Zyablov and M. S. Pinsker. Estimation of the error-correction complexity for Gallager low-density codes. *Probl. Peredachi Inf.*, 11(1):23–36, 1975. English translation: Problems Inform. Transmission, 1975, 11(1): 18–28.

A Appendix

A.1 Generating a Random Graph

The following algorithm generates a random graph given the degrees of the right vertices, while all left vertices have degree c.

Algorithm 6 GenerateGraph(deg)

Input: A function $\deg : R \to \mathbb{N}$ ▷ Degrees of right vertices

$E \leftarrow \{(v, j) \mid v \in R, j \in [\deg(v)]\}$
for each $u \in L$ **do**
  for each $i \in [c]$ **do**
    Choose $(v, j) \in E$ uniformly at random
    Add an edge connecting u and v
    $E \leftarrow E \setminus \{(v, j)\}$
  end for
end for

We claim the following property of this process and omit its proof.

Lemma A.1. *The probability distribution of the graph generated by Algorithm 6 is independent of the order in which the pairs $(u, i) \in L \times [c]$ are enumerated.*

A.2 A Combinatorial Lemma

Lemma A.2. *Let $m > 0$ and $i \geq 0$ be integers. Let $k > 1$ be a real number such that $i \leq km - m$. Then,*

$$\left(1 - \frac{1}{k}\right)^i \geq \frac{\binom{km-i}{m}}{\binom{km}{m}} = \left(1 - \frac{1}{k}\right)^i - O_{i,k}\left(\frac{1}{m}\right),$$

where the $O(\cdot)$ term is with respect to the growing parameter m.

Proof. The ratio can be expressed as:

$$\frac{\binom{km-i}{m}}{\binom{km}{m}} = \frac{(km-i)(km-i-1)\dots((k-1)m-i+1)}{km(km-1)\dots((k-1)m+1)}.$$

Since the product $(km-i)\cdots((k-1)m+1)$ appears in both the numerator and denominator, by factoring out the $\left(1-\frac{1}{k}\right)$ terms, we can simplify the ratio as follows:

$$\frac{\binom{km-i}{m}}{\binom{km}{m}} = \prod_{j=0}^{i-1} \frac{(k-1)m-j}{km-j} = \left(1-\frac{1}{k}\right)^i \cdot \prod_{j=0}^{i-1}\left(1-\frac{j}{(k-1)(km-j)}\right) \leq \left(1-\frac{1}{k}\right)^i.$$

Further, we bound the additional factor:

$$\begin{aligned}\prod_{j=0}^{i-1}\left(1-\frac{j}{(k-1)(km-j)}\right) &\geq 1-\sum_{j=0}^{i-1}\frac{j}{(k-1)(km-j)} \\ &\geq 1-\frac{i(i-1)}{2(k-1)(km-i)}.\end{aligned}$$

Therefore, we have

$$\begin{aligned}\frac{\binom{km-i}{m}}{\binom{km}{m}} &\geq \left(1-\frac{1}{k}\right)^i\left(1-\frac{i(i-1)}{2(k-1)(km-i)}\right) \\ &\geq \left(1-\frac{1}{k}\right)^i - \frac{i(i-1)}{2(k-1)(km-i)} \\ &= \left(1-\frac{1}{k}\right)^i - O_{k,i}\left(\frac{1}{m}\right).\end{aligned}$$

This completes the proof. □

A.3 Properties of the Size-Expansion Function

Next, we restate and prove Lemma 5.3.

Lemma 5.3. *The functions f_δ and $f_{d,\delta}$ satisfy the following properties:*

1. *f_δ and $f_{d,\delta}$ are non-increasing.*
2. *$f_{d,\delta}(k) = \max\left\{f_\delta(k), \frac{1}{d}\right\}$.*
3. *$f_\delta(k) \geq \frac{\delta}{k}$.*
4. *Omitting the $O\left(\frac{1}{n}\right)$ term in the constraint (11) increases the value of $f_{d,\delta}(k)$ by at most $O\left(\frac{1}{n}\right)$.*
5. *$f_\delta(k)$ is continuous with $\lim_{k\to 1} f_\delta(k) = \delta$ and $\lim_{k\to\infty} f_\delta(k) = 0$.*

Proof. We prove these properties separately.

Monotonicity. Let $k' > k$, and let $(\hat{\beta}_i)_{i\in\mathbb{N}^+}$ denote the optimal solution to the LP defining $f_\delta(k)$. Then, $(\frac{k'}{k}\hat{\beta}_i)_{i\in\mathbb{N}^+}$ is a feasible solution to the LP defining $f_\delta(k')$, and the corresponding value of the objective function is

$$\frac{1}{k'}\sum_{i=1}^{\infty}\frac{k'}{k}\hat{\beta}_i = \frac{1}{k}\sum_{i=1}^{\infty}\hat{\beta}_i = f_\delta(k).$$

Therefore, $f_\delta(k') \leq f_\delta(k)$. The same argument applies to $f_{d,\delta}$. This proves Item 1.

Relation between f_δ and $f_{d,\delta}$. Recall the definition of $f_\delta(k)$ from Definition 5.2, which is the minimum of the following LP:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{k}\sum_{i=1}^{\infty}\beta_i \\ \text{subject to} \quad & \sum_{i=1}^{\infty} i\cdot\beta_i = k, \\ & \sum_{i=1}^{\infty}\left(1-\left(1-\frac{1}{k}\right)^i\right)\cdot\beta_i \geq \delta, \\ & \beta_i \geq 0, \quad \forall i. \end{aligned} \tag{21}$$

We now show that the optimal solution is achieved by $\hat{\beta} = (\hat{\beta}_i)_{i\in\mathbb{N}^+}$ that is supported on at most two adjacent indices. The proof follows [CCLO23, Lemma 3.2].

The dual of this LP is:

$$\begin{aligned} \text{maximize} \quad & kx + \delta y \\ \text{subject to} \quad & x\cdot i + y\left(1-\left(1-\frac{1}{k}\right)^i\right) \leq \frac{1}{k}, \quad \forall i\in\mathbb{N}^+ \\ & y \geq 0. \end{aligned} \tag{22}$$

Assume, for contradiction, that $\hat{\beta}_i, \hat{\beta}_j \neq 0$ with $j > i+1$. By the complementary slackness conditions, the corresponding constraints in Equation (22) must be satisfied as equalities:

$$x\cdot i + y\left(1-\left(1-\frac{1}{k}\right)^i\right) = \frac{1}{k}$$

and

$$x\cdot j + y\left(1-\left(1-\frac{1}{k}\right)^j\right) = \frac{1}{k}.$$

These two equalities imply that $y \neq 0$.

The function $1-\left(1-\frac{1}{k}\right)^i$ is convex with respect to i. In particular:

$$\left(1-\left(1-\frac{1}{k}\right)^i\right) - \left(1-\left(1-\frac{1}{k}\right)^{i-1}\right) > \left(1-\left(1-\frac{1}{k}\right)^{i+1}\right) - \left(1-\left(1-\frac{1}{k}\right)^i\right).$$

From the two equalities derived from the slackness conditions, $y > 0$, and the convexity of the function, we obtain:

$$x\cdot(i+1) + y\left(1-\left(1-\frac{1}{k}\right)^{i+1}\right) > \frac{1}{k}.$$

This contradicts the constraints of the dual program. Thus, we have shown that $\hat{\beta}$ must be supported on at most two adjacent indices.

Now, assume that the support of $\hat{\beta}$ is contained in $\{i, i+1\}$ for some $i \in \mathbb{N}^+$. Then, the LP (21) simplifies to:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{k}(\beta_i + \beta_{i+1}) \\ \text{subject to} \quad & \beta_i \cdot i + \beta_{i+1} \cdot (i+1) = k, \\ & \beta_i \cdot \left(1 - \left(1 - \frac{1}{k}\right)^i\right) + \beta_{i+1} \cdot \left(1 - \left(1 - \frac{1}{k}\right)^{i+1}\right) \geq \delta, \\ & \beta_i, \beta_{i+1} \geq 0. \end{aligned}$$

The optimal value of this problem is given by:

$$\begin{cases} \frac{\delta - (1-\frac{1}{k})^i}{k - (k+i)(1-\frac{1}{k})^i}, & \text{for } \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i+1}\right)}{i+1} \leq \delta \leq \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i}\right)}{i}, \\ \frac{1}{i+1}, & \text{for } \delta < \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i+1}\right)}{i+1}, \\ \text{infeasible}, & \text{otherwise.} \end{cases} \tag{23}$$

By comparing the values above for different choices of i, we find the value of $f_\delta(k)$ is given by:

$$f_\delta(k) = \frac{\delta - (1-\frac{1}{k})^i}{k - (k+i)(1-\frac{1}{k})^i} \quad \text{for } i \in \mathbb{N}^+ \text{ such that } \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i+1}\right)}{i+1} \leq \delta \leq \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i}\right)}{i}. \tag{24}$$

The same analysis applies to the LP defining $f_{d,\delta}$, with the only difference being that the choice of i is restricted to the range $[d-1]$. Therefore, we have the expression for $f_{d,\delta}(k)$ as follows:

$$f_{d,\delta}(k) = \begin{cases} \frac{1}{d}, & \text{for } \delta < \frac{k\left(1-\left(1-\frac{1}{k}\right)^{d}\right)}{d}, \\ \frac{\delta - (1-\frac{1}{k})^i}{k - (k+i)(1-\frac{1}{k})^i}, & \text{for } i \in [d-1] \text{ such that } \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i+1}\right)}{i+1} \leq \delta \leq \frac{k\left(1-\left(1-\frac{1}{k}\right)^{i}\right)}{i}. \end{cases} \tag{25}$$

Next, we consider the condition $f_\delta(k) \neq f_{d,\delta}(k)$, which implies that the support of the optimal solution $\hat{\beta}$ of the LP (21) has the form $\{i+1\}$ or $\{i, i+1\}$ with $i \geq d$. Therefore, comparing (25) with (24) proves Item 2.

Improvement over the trivial bound $\frac{\delta}{k}$. By substituting $x = 0$ and $y = \frac{1}{k}$ into the dual program (22), we see that $f_\delta(k) \geq \frac{\delta}{k}$, which establishes Item 3.

Effect of omitting the $O\left(\frac{1}{n}\right)$ term. Let $f^*_{d,\delta}(k)$ be the size-expansion trade-off defined by the LP without omitting the error term. The only difference from the LP defining $f_{d,\delta}(k)$ is that the constraint becomes $\sum_{i=1}^{d}\left(1 - \left(1-\frac{1}{k}\right)^i\right)\beta_i \geq \delta - O\left(\frac{1}{n}\right)$, and the dual objective function changes to $kx + \left(\delta - O\left(\frac{1}{n}\right)\right)y$.

Let $(\hat{x}, \hat{y})$ be the optimal solution for the dual program for $f_{d,\delta}(k)$, so $f_{d,\delta}(k) = k\hat{x} + \delta\hat{y}$. Since $\hat{x}, \hat{y}$ also satisfy the dual program for $f^*_{d,\delta}(k)$, we obtain the bound:

$$f^*_{d,\delta}(k) \geq f_{d,\delta}(k) - \hat{y} \cdot O\left(\frac{1}{n}\right).$$

Next, we bound $\hat{y}$. The constraint in (22) with $i = 1$ states that $\hat{x} + \frac{\hat{y}}{k} \le \frac{1}{k}$, i.e., $\hat{x} \le \frac{1-\hat{y}}{k}$. Therefore,

$$f_{d,\delta}(k) = k\hat{x} + \delta\hat{y} \le 1 - \hat{y} + \delta\hat{y}.$$

By Item 3, $f_{d,\delta}(k) \ge \frac{\delta}{k} \ge 0$. So $y \le \frac{1}{1-\delta} = O(1)$. Thus, $f^*_{d,\delta}(k) \ge f_{d,\delta}(k) - O\left(\frac{1}{n}\right)$, which proves Item 4.

Continuity and range. The continuity of f_δ can be established via the explicit description of f_δ given in (24). When k is located at the endpoint of two pieces, namely, when

$$\frac{k\left(1 - \left(1 - \frac{1}{k}\right)^{i+1}\right)}{i+1} = \delta$$

for some positive integer i, we have

$$\lim_{x \to k^-} f_\delta(x) = \frac{\delta - \left(1 - \frac{1}{k}\right)^i}{k - (k+i)\left(1 - \frac{1}{k}\right)^i} = \frac{1}{i+1} = \frac{\delta - \left(1 - \frac{1}{k}\right)^{i+1}}{k - (k+i+1)\left(1 - \frac{1}{k}\right)^{i+1}} = \lim_{x \to k^+} f_\delta(x).$$

Thus, f_δ is continuous.

To prove that $\lim_{k \to 1} f_\delta(k) = \delta$, note that for bounded k, the value of i chosen in (24) satisfies $i = O(1/\delta)$. Thus,

$$\lim_{k \to 1} f_\delta(k) = \lim_{k \to 1} \frac{\delta - (1 - \frac{1}{k})^i}{k - (k+i)(1 - \frac{1}{k})^i} = \lim_{k \to 1} \frac{\delta}{k} = \delta.$$

Next, we prove that $\lim_{k \to \infty} f_\delta(k) = 0$. For any fixed i, we have

$$\lim_{k \to \infty} \frac{k\left(1 - \left(1 - \frac{1}{k}\right)^{i+1}\right)}{i+1} = 1.$$

Thus, as $k \to \infty$, the value of i chosen in (24) also tends to infinity. Therefore, the value of the function satisfies

$$\lim_{k \to \infty} f_\delta(k) = \lim_{k \to \infty} \frac{\delta - \left(1 - \frac{1}{k}\right)^i}{k - (k+i)\left(1 - \frac{1}{k}\right)^i} \le \lim_{k \to \infty} \frac{1}{i} = 0,$$

which completes the proof of Item 5. □