Electronic Colloquium on Computational Complexity, Report No. 58 (2011)

Linear time decoding of regular expander codes

Michael Viderman*
Computer Science Department
Technion — Israel Institute of Technology
Haifa 32000, Israel
viderman@cs.technion.ac.il

April 15, 2011

Abstract

Sipser and Spielman (IEEE IT, 1996) showed that any (c, d)-regular expander code with expansion parameter $> \frac{3}{4}$ is decodable in *linear time* from a constant fraction of errors. Feldman et al. (IEEE IT, 2007) proved that expansion parameter $> \frac{2}{3} + \frac{1}{3c}$ is sufficient to correct a constant fraction of errors in *polynomial time* using LP decoding.

In this work we give a simple combinatorial algorithm that achieves even better parameters. In particular, our algorithm runs in *linear time* and works for any expansion parameter $> \frac{2}{3} - \frac{1}{6c}$.

1 Introduction

Linear codes play an important role in numerous interconnections between the area of error-correcting codes and complexity theory. A linear code $C \subseteq \mathbb{F}^n$ can be specified by its parity check matrix or in other words, the set of linear constraints that each codeword must satisfy. These constraints can be naturally viewed in terms of a bipartite graph with codeword positions on one side and parity checks on the other side, and adjacency reflecting which symbols are involved in which parity check constraints. This graph is called a parity check graph. When this graph has constant degree, the underlying codes are called low-density parity check codes (LDPC codes). Binary as well as q-ary LDPC codes were introduced and studied in Gallager's amazing work more than four decades ago [9, 10]. They have been studied extensively in information theory (cf. [3]). Binary LDPC codes motivated Margulis' explicit construction of graphs of large girth [18], the precursor of his construction of Ramanujan graphs [15]. The following two remarkable works stressed the importance of LDPC codes in coding theory. Zyablov and Pinsker [28] proved that for random LDPC codes, with high probability over the choice of the code, Gallager's algorithm corrects a constant fraction of worst-case errors. Tanner [25] presented an important generalization of Gallager's construction and his decoding algorithms, which was later used in the work on linear time decodable expander codes [22]. A special class of LDPC codes is regular LDPC codes where the underlying parity check graph is both left-regular and right-regular. Regular LDPC codes were in fact the variant originally studied by Gallager [9, 10], as well as in the works of Mackay and Neal [16, 17] and Sipser and Spielman [22, 23] that sparked the resurgence of interest in LDPC codes after over 30 years since Gallager's work.

[*] Part of this work was done while the author was a summer intern at Microsoft Research New England. The research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 240258 and by grant number 2006104 by the US-Israel Binational Science Foundation.

ISSN 1433-8092

We turn to mention the well-known and widely investigated class of decoding algorithms for LDPC codes, called message passing algorithms. At each round of these algorithms the messages are passed from codeword nodes to (parity) check nodes, and from check nodes back to codeword nodes. One important subclass of message passing algorithms is the *belief propagation algorithm*. This algorithm is presented in Gallager's work [10], and it is also used in the Artificial Intelligence community [19]. The messages passed along the edges in this algorithm are probabilities, or beliefs. More precisely, the message passed from a codeword node v to a check node c is the probability that v has a certain value given the observed value of this codeword node, and all the values communicated to v in the prior round from check nodes incident to v other than c. On the other hand, the message passed from c to v is the probability that v has a certain value given all the messages passed to c in the previous round from message nodes other than v.

The central algorithmic problem in coding theory is the explicit construction of error-correcting codes with best possible parameters together with fast encoding and decoding algorithms. Recently, this area has benefited enormously from insights and viewpoints originating in complexity theory, and numerous interconnections between codes and complexity theory have been discovered, which are surveyed for example in [24, 26]. The former survey [24] focuses on a notion called list-decoding and the second survey [26] focuses mainly on sub-linear algorithms for local testing and local decoding. Basically, there are two different kinds of noise models: adversarial and probabilistic. In this paper we consider an *adversarial noise model* where we only assume a bound on the number of errors and not how they are distributed.

Over the last decade, significant new developments have taken place on this problem using combinatorial constructions that exploit the power of *expander graphs*. The role of expander graphs in theoretical computer science is by now certainly well-appreciated. Expander graphs have been the subject of much study in combinatorics and computer science, and have found applications in diverse context. There have been significant breakthroughs in explicit construction of expanders [4, 20]. Informally, the error-correcting codes constructed from expander graphs are called expander codes. Expander codes are the only known construction of asymptotically good error-correcting codes which can be decoded in linear time when a constant fraction of symbols are corrupted. In particular case, when the parity check graph of the regular LDPC code has good expansion properties the associated code is called regular expander code. Regular expander codes and their decoding algorithms are often (implicitly) involved as basic building blocks in the constructions of asymptotically good codes which are linear-time decodable. The celebrated result of Sipser and Spielman [22] shows that regular expander codes with expansion parameter $> 3/4$ are decodable in linear time from constant fraction of errors. They achieved this by defining extremely simple decoding algorithm, called Flip Algorithm (see Section 3.1). Then Spielman [23], using the result of [22], showed that expander code with an extremely high expansion properties can be used to construct asymptotically good error-correcting codes which can be encoded and decoded in linear time. Since the works of [22, 23] the (regular) expander codes have been studied extensively in the area of LDPC codes and many different constructions and improvements have been achieved (see e.g., [1, 2, 11, 12, 13, 14, 21, 27]).

Sipser and Spielman [22] pointed out that their work leaves many natural questions opened. One of these questions is what is the minimal expansion requirement for the regular expander codes that is sufficient for linear-time decoding from a constant fraction of errors. In spite of the broad research in the area this question remained open.[1] Feldman et al. [8] showed that the linear programming (LP) decoding corrects a constant fraction of errors in *polynomial time* if the expansion parameter $> \frac{2}{3} + \frac{1}{3c}$. In this paper we show (Theorem 3.1) a simple combinatorial algorithm, called Find Erasures and Decode that decodes regular expander codes in *linear time* if the expansion parameter $> \frac{2}{3}$ from the same number of errors as in

[1] Best to our knowledge, since the result of [22] the minimal known expansion parameter required for the *linear-time* decoding of regular expander codes was $3/4$.

[8]. Find Erasures and Decode algorithm converts some "suspicious bits" to *erasures* and then decode the underlying word from the erasures, and this is possibly a midway between "hard belief propagation" (Flip Algorithm) and the full belief propagation (BP), which has resisted analysis in the case of adversarial noise model. Then, in Theorem B.1 we show that the expansion parameter $> 2/3 - \Omega(1)$ is sufficient for the linear-time decoding from a constant fraction of errors.

We also show in Theorem A.1 that all regular expander codes with expansion parameter $> 1/2$, decodable by Flip Algorithm, are linear-time decodable by our algorithm (Find Erasures and Decode), and hence this paper presents (arguably) a better algorithm (Find Erasures and Decode) than the famous Flip Algorithm with respect to linear time decoding of regular expander codes (see Section 3.1.1 for full comparison). Furthermore, we show in Theorem 3.4 that the expansion parameter $> 1/2$ is sufficient to correct a sublinear number ($n^{\Omega(1)}$) of errors and the expansion parameter $\leq 1/2$ is insufficient (for any decoding algorithm) to correct even 2 errors. Given the importance of analyzing BP for decoding, we believe that Find Erasures and Decode algorithm takes an important step.

Organization of the paper. We begin with some quick background on codes and expanders in Section 2. In Section 3 we present our main results. We prove our main theorem (Theorem 3.1) in Section 4. We end by raising Conjecture 5.1 in Section 5.

The rest of the material is postponed to Appendix. In particular, in Section A we show that all regular expander codes with expansion parameter $> 1/2$, decodable by Flip Algorithm, are linear-time decodable by our algorithm Find Erasures and Decode (presented in Section 4). We continue in Section B and show that expansion parameter $> 2/3 - \Omega(1)$ is sufficient for the linear-time decoding. Then we prove Theorem 3.4 in Section C. The tightness of Theorem 3.4 is discussed in Section D.

2 Preliminaries

Let $\mathbb{F}$ be a finite field and $[n]$ be the set $\{1, \ldots, n\}$. In this work, we consider only linear codes. We start with a few definitions.

Let $C \subseteq \mathbb{F}^n$ be a linear code over $\mathbb{F}$. The dimension of C is denoted by $\dim(C)$. For $w \in \mathbb{F}^n$, let $\operatorname{supp}(w) = \{i \in [n] \mid w_i \neq 0\}$ and $|w| = |\operatorname{supp}(w)|$. We define the *distance* between two words $x, y \in \mathbb{F}^n$ to be $\Delta(x, y) = |\{i \mid x_i \neq y_i\}|$ and the relative distance to be $\delta(x, y) = \frac{\Delta(x,y)}{n}$. The distance of a code is denoted by $\Delta(C)$ and defined to be the minimal value of $\Delta(x, y)$ for two distinct codewords $x, y \in C$. Similarly, the relative distance of the code is denoted $\delta(C) = \frac{\Delta(C)}{n}$. For $x \in \mathbb{F}^n$ and $C \subseteq \mathbb{F}^n$, let $\delta(x, C) = \min\limits_{y \in C} \{\delta(x, y)\}$ denote the relative distance of x from the code C. We note that $\Delta(C) = \min\limits_{c \in C \setminus \{0\}} \{|c|\}$. If $\delta(x, C) \geq \epsilon$, we say that x is ϵ-far from C and otherwise x is ϵ-close to C. The vector inner product between u_1 and u_2 is denoted by $\langle u_1, u_2 \rangle$. The dual code $C^{\perp}$ is defined as $C^{\perp} = \{u \in \mathbb{F}^n \mid \forall c \in C : \langle u, c \rangle = 0\}$. In a similar way we define $C^{\perp}_{\leq t} = \left\{u \in C^{\perp} \mid |u| \leq t\right\}$ and $C^{\perp}_{t} = \left\{u \in C^{\perp} \mid |u| = t\right\}$. For $T \subseteq \mathbb{F}^n$ we say that $w \perp T$ if for all $t \in T$ we have $\langle w, t \rangle = 0$.

For $w \in F^n$ and $S = \{j_1, j_2, \ldots, j_m\} \subseteq [n]$, where $j_1 < j_2 < \ldots < j_m$, let $w|_S = (w_{j_1}, w_{j_2}, \ldots, w_{j_m})$ be the *restriction* of w to the subset S. Let $C|_S = \{c|_S \mid c \in C\}$ denote the restriction of the code C to the subset S.

2.1 Expander Codes

Now we define expanders and expander codes. Informally, expanders are sparse graphs with excellent connectivity properties. We start from the definition of *expanders* and then proceed to the definition of *expander codes*.

Definition 2.1 (Regular Expander Codes)**.** Let $C \subseteq \mathbb{F}_2^n$ be a linear code and let $G = (L, R, E)$ be its parity check graph, where $L = [n]$ represents the codeword positions and $R \subseteq C^{\perp}$ represents the parity check constraints. Note that for every $\mathrm{x} \in \mathbb{F}_2^n$ we have $\mathrm{x} \perp R$ if and only if $\mathrm{x} \in C$. For $l \in L$ and $r \in R$ it holds that $\{l, r\} \in E$ if and only if $l \in \mathrm{supp}(r)$.

The graph G is called (c, d)-regular if every vertex $l \in L$ has degree c and every vertex $r \in R$ has degree d. For $L_0 \subseteq L$ and $x \in L_0$, let

- $N(L_0) = \{r \in R \mid \{l, r\} \in E \text{ for some } l \in L_0\}$ be the set of neighbors of L_0,
- $N(x) = N(\{x\})$ be the set of neighbors of the node x,
- $N^1(L_0) = \{r \in R \mid \{l, r\} \in E \text{ for a unique } l \in L_0\}$ be the set of unique neighbors of L_0,
- $N^{\geq 2}(L_0) = N(L_0) \setminus N^1(L_0)$,
- $N^1_{L_0}(x) = N^1(L_0) \cap N(x)$, i.e., the unique neighbors of L_0, which are neighbors of x.

Let $\epsilon, \delta > 0$ be constants. Then,

- G is called a (c, d, ϵ, δ)-*expander* if G is (c, d)-regular and for all subsets $S \subseteq L$ s.t. $|S| \leq \delta n$ we have $|N(S)| \geq \epsilon \cdot c|S|$;
- G is called a (c, d, ϵ, δ)-*unique expander*[2] if G is (c, d)-regular and for all subsets $S \subseteq L$ s.t. $|S| \leq \delta n$ we have $|N^1(S)| \geq \epsilon \cdot c|S|$.

We say that a code C is a (c, d, ϵ, δ)-*(unique) expander code* if it has a parity check graph that is a (c, d, ϵ, δ)-(unique) expander. Throughout the paper we let $S_C = R$ be a set of d-weight constraints in the parity check graph of the expander code C.

3 Main Results

In this section we state our main results. In Section 3.1 we discuss and compare our results to the related works. The results presented in this paper are stated over the binary field but can be easily extended to any finite field.

The main result of this paper is stated in the following theorem (Theorem 3.1) and improves the result of Sipser and Spielman [22] (see Section 3.1 for discussion). In particular, we show that expansion parameter $\frac{2}{3}$ is sufficient for linear-time decoding.

Theorem 3.1. *Let $c, d, \epsilon, \delta > 0$ be constants and let $C \subseteq \mathbb{F}_2^n$ be a (c, d, ϵ, δ)-expander code. Assume that $\epsilon > 1/2$ and $\epsilon c + h - c > 0$, where $h = \lceil (2\epsilon - 1)c \rceil$. Then C is decodable in* linear time *from $\frac{\epsilon c + h - c}{h} \lfloor \delta n \rfloor$ errors.*

The proof of the theorem is postponed to Section 4.

[2] Usually such expanders are called unique neighbor expanders, but to shorten the notation we call them unique expanders.

Remark 3.2. If $\epsilon > 2/3$ then $\epsilon c + h - c > 0$ and hence C is decodable in *linear time* from $\frac{\epsilon c + h - c}{h} \lfloor \delta n \rfloor \geq \frac{3\epsilon - 2}{2\epsilon - 1} \lfloor \delta n \rfloor$ errors.

Remark 3.3. Corollary E.3 shows that if C is a (c, d, ϵ, δ)-expander code, where $\epsilon > 1/2$ then $\Delta(C) \geq 2\epsilon\delta n$ and this is the best distance bound one can achieve based only on the expansion property of C. Hence the value $2\epsilon\delta n$ is called a design distance of the code C.

We notice that when ϵ converges to 1 the design distance converges to $2\delta n$ and the number of corrected errors $\left(\frac{3\epsilon - 2}{2\epsilon - 1} \lfloor \delta n \rfloor\right)$ converges to δn, i.e., a half of the design distance of C.

Looking at Theorem 3.1 one might conjecture that $2/3$ is a minimal expansion required for the linear time decoding. In Section B we refute such conjecture and prove that even if the expansion parameter is more than $2/3 - 1/(6c) = 2/3 - \Omega(1)$ then C is linear time decodable from the constant fraction of errors.

Expansion parameter $> \frac{1}{2}$ We want to determine the minimal expansion parameter $\epsilon > 0$ such that for all constants $c, d, \delta > 0$ it holds that a (c, d, ϵ, δ)-expander code is linear time decodable from $\Omega_{c,d,\delta,\epsilon}(n)$ errors. Section D shows that $\epsilon \leq \frac{1}{2}$ is insufficient for the decoding even from 2 errors. In Theorem 3.4 we show that the expansion parameter $\epsilon > 1/2$ is sufficient for the linear time decoding from $n^{\Omega(1)}$ errors.

Theorem 3.4. *Let $c, d, \delta > 0$ and $\epsilon > 1/2$ be constants. Then there exists $\alpha > 0$ which depends only on c, d, ϵ, δ, such that if $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-expander code then C is linear-time decodable from n^α errors.*

Proof. We have $2\epsilon - 1 > 0$. Proposition E.1 implies that C is a $(c, d, 2\epsilon - 1, \delta)$-unique expander code. The theorem follows from Theorem 3.5, stated next. □

Theorem 3.5. *Let $c, d, \epsilon, \delta > 0$ be constants. Then there exists $\alpha > 0$ which depends only on c, d, ϵ, δ, such that if $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-unique expander code then C is linear-time decodable from n^α errors.*

The proof of Theorem 3.5 is postponed to Section C. We prove the tightness of Theorem 3.4 in Section D, i.e., we show that the expansion parameter $1/2$ is insufficient for decoding even from 2 errors.

In Section A we prove Theorem A.1 which states that all regular expander codes with expansion parameter $> 1/2$ and decodable by Flip Algorithm of [22], are linear-time decodable by Find Erasures and Decode.

3.1 Related works

We start from recalling the result of Sipser and Spielman [22]. Let us define

$$N^{sat}_{S,w}(i) = \{u \in S \mid \langle u, w \rangle = 0 \ \text{ and } \ i \in \text{supp}(u)\},$$

and similarly, we define $N^{unsat}_{S,w}(i) = \{u \in S \mid \langle u, w \rangle \neq 0 \ \text{ and } \ i \in \text{supp}(u)\}$.

The Flip Algorithm from [22] is defined as follows.

```
Flip Algorithm
  • Input: $w \in \mathbb{F}^n$ and $S \subseteq C_d^\perp$
  • While there exists $i \in [n]$ such that $|N^{sat}_{S,w}(i)| < |N^{unsat}_{S,w}(i)|$:
      – Flip bit $i$ of $w$.
```

It can be easily verified that the Flip Algorithm achieves the following result (it is stated in a more general form than in [22]). For the sake of completeness we give a proof sketch of their result.

Theorem 3.6 ([22]). *Let C be a (c, d, ϵ, δ)-expander, where $\epsilon > \frac{3}{4}$. Then C is decodable in linear time by* Flip Algorithm *from less than $(2\epsilon - 1)\lfloor \delta n \rfloor$ errors.*

Proof Sketch: Let $w \in \mathbb{F}_2^n$ be an input word such that $\Delta(w, C) < (2\epsilon - 1)\lfloor \delta n \rfloor$ and let $S \subseteq C_d^{\perp}$ be a subset of constraints from the parity check graph of C. The observation made in [22] said that if $\epsilon > \frac{3}{4}$ then while $\Delta(w, C) \leq \delta n$ but $w \notin C$ there exists $i \in [n]$ such that $|N_{S,w}^{sat}(i)| < |N_{S,w}^{unsat}(i)|$. Hence, while the underlying word is δ-close to the code C, every iteration of Flip Algorithm decreases the number of unsatisfied constraints from S.

Hence the only possible bad scenario is that during the run of Flip Algorithm there exists an iteration where the number of the corrupted codeword symbols is $\lfloor \delta n \rfloor$. So, in this iteration the number of unsatisfied constraints from S is at least $(2\epsilon - 1) \cdot c \cdot \lfloor \delta n \rfloor$ (see Proposition E.1). However, initially the number of unsatisfied constraints from S was at most $c \cdot \Delta(w, C) < (2\epsilon - 1) \cdot c \cdot \lfloor \delta n \rfloor$, and by definition of Flip Algorithm, the number of the unsatisfied constraints from S is decreased (at least by one) each iteration. Contradiction. □

Feldman et al. [8] proved the following result using the linear programming (LP) decoding [5, 6, 7].

Theorem 3.7 ([8]). *If $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-expander, where $\epsilon > \frac{2}{3} + \frac{1}{3c}$. Then C is decodable in* polynomial time *from $\frac{3\epsilon - 2}{2\epsilon - 1}\lfloor \delta n \rfloor$ errors.*

3.1.1 Comparison to the previous works

In this section we summarize the improvements made in this paper over two previous results: [22] and [8]. We take into consideration the following aspects: a minimal required expansion parameter, a number of errors that the decoding algorithm is able to correct and the running time of the decoding algorithm.

We notice that the number of errors Theorem 3.1 deals with is $\frac{\epsilon c + h - c}{h} \geq \frac{3\epsilon - 2}{2\epsilon - 1}$, recalling that $h = \lceil (2\epsilon - 1)c \rceil$. Note also that for $\epsilon > 3/4$ it holds that $\frac{3\epsilon - 2}{2\epsilon - 1} > 2\epsilon - 1$. Thus the algorithm Find Erasures and Decode, presented in the proof of Theorem 3.1, corrects at least as many errors as in Theorem 3.7, and strictly more errors than in Theorem 3.6.

	Required expansion	Run Time	Number of corrected errors
Our Result (Theorem 3.1)	$\epsilon > \frac{2}{3}$	linear	$\frac{3\epsilon - 2}{2\epsilon - 1}\lfloor \delta n \rfloor$
Feldman et al. [8]	$\epsilon > \frac{2}{3} + \frac{1}{3c}$	polynomial	$\frac{3\epsilon - 2}{2\epsilon - 1}\lfloor \delta n \rfloor$
Sipser and Spielman [22]	$\epsilon > \frac{3}{4}$	linear	$(2\epsilon - 1)\lfloor \delta n \rfloor$

4 Expansion $> 2/3$ — Proof of Theorem 3.1

We start this section by presenting our Find Erasures and Decode algorithm, which receives an input word $w \in \mathbb{F}_2^n$ and a constraints set $S_C \subseteq C_d^{\perp}$ from the parity check graph of C and is required to return the closest codeword $\mathrm{x} \in C$ to w. Find Erasures and Decode is composed from two sub-algorithms defined later in Sections 4.1 and 4.2, respectively.

```
Find Erasures and Decode
    • Input: $w \in \mathbb{F}_2^n$ and $S_C \subseteq C_d^\perp$
    • $L'$ := Find Erasures $(w, S_C)$
    • $x'$ := Decode From Erasures $(w, L', S_C)$
    • Return $x'$
```

We recall that $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-expander code, where $c, d, \delta > 0$ are constants, $\epsilon > 1/2$ and $\epsilon c + h - c > 0$ for $h = \lceil (2\epsilon - 1)c \rceil$. For the rest of this section we assume that the (fixed) input word is $w \in \mathbb{F}_2^n$ and the closest codeword to w is $\mathrm{x} \in C$ such that $\Delta(w, \mathrm{x}) \leq \frac{\epsilon c + h - c}{h} \lfloor \delta n \rfloor$. Let $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$ be the subset of corrupted variables and note that $|\mathrm{corr}| \leq \frac{\epsilon c + h - c}{h} \lfloor \delta n \rfloor \leq \lfloor \delta n \rfloor$.

In the rest of the paper we prove Lemmas 4.1 and 4.2.

Lemma 4.1. *The algorithm* Find Erasures *(defined in Section 4.1) returns* $L' \subseteq [n]$ *such that* $\mathrm{corr} \subseteq L'$ *and* $|L'| \leq \delta n$. *Moreover, the algorithm* Find Erasures *runs in linear time.*

Lemma 4.2. *If for* $L' \subseteq [n]$ *we have* $\mathrm{corr} \subseteq L'$ *and* $|L'| \leq \delta n$ *then the algorithm* Decode From Erasures *(defined in Section 4.2) on the input* w, L' *and* S_C *returns* x. *Moreover, the algorithm* Decode From Erasures *runs in linear time.*

The proofs of Lemmas 4.1 and 4.2 are postponed to Sections 4.1 and 4.2, respectively. The proof of Theorem 3.1 follows immediately from Lemmas 4.1 and 4.2.

Proof of Theorem 3.1. Lemma 4.1 proves that Find Erasures returns L' such that $\mathrm{corr} \subseteq L'$ and $|L'| \leq \lfloor \delta n \rfloor$. Lemma 4.2 implies that the codeword x will be returned by Decode From Erasures. Moreover, Find Erasures and Decode algorithm runs in linear time since Lemmas 4.1 and 4.2 imply that Find Erasures and Decode From Erasures run in linear time. □

4.1 How to find erasures — Proof of Lemma 4.1

In this section we present an algorithm Find Erasures and then prove Lemma 4.1. Roughly we maintain at each iteration sets $R' = R'(t)$ and $L' = L'(t)$ which grow slowly. For analysis it is better to denote sets separately, but later we will suppress it, to get an efficient implementation.

```
Find Erasures
  • Input: $w \in \mathbb{F}_2^n$ and $S \subseteq C_d^{\perp}$
  • Initialize:
      – $h := \lceil (2\epsilon - 1) \cdot c \rceil$
      – $R_0 := \{u \in S \mid \langle u, w \rangle \neq 0\}$
      – $R'(0) := R_0$
      – $L'(0) := \{i \in [n] \mid |N(i) \cap R_0| \geq h\}$
      – $t := 0$
  • While there exists $i \in [n] \setminus L'(t)$ such that $|N(i) \cap R'(t)| \geq h$ do
      – $L'(t+1) := L'(t) \cup \{i\}$
      – $R'(t+1) := R'(t) \cup N(i)$
      – $t := t + 1$
  • return $L'(t)$
```

Intuition behind the Algorithm The intuition behind the algorithm Find Erasures is as follows. As we have already mentioned, during the algorithm we maintain $R' = R'(t)$ and $L' = L'(t)$. We think of $R'(t)$ as a subset of untrustful constraints and of $L'(t)$ as a subset of untrustful indices (erasures). Initially, $L'(0)$ is an empty set and $R'(0)$ contains all unsatisfied constraints (i.e., $u \in S$ such that $\langle u, w \rangle \neq 0$. Each iteration t, an index $i \notin L'(t-1)$ is declared as untrustful (erasure) if it is touched by "many" untrustful constraints (i.e., those from $R'(t-1)$) and then all constraints that touch i are declared untrustful. We repeat until we see no new untrustful indices. Finally, the algorithm returns a set of all untrustful indices (erasures).

Now we present Propositions 4.3 and 4.4. The proof of Lemma 4.1 will follow from these propositions.

Proposition 4.3 (L' includes all corrupted bits)**.** *Assume the algorithm* Find Erasures *returns $L'(t)$ for some $t \geq 0$. Then* corr $\subseteq L'(t)$.

Proposition 4.4 (L' is small)**.** *For all $t \geq 0$ we have $|L'(t)| < \lfloor \delta n \rfloor$.*

The proofs of Propositions 4.3 and 4.4 are postponed to Sections 4.1.2 and 4.1.1, respectively. We are ready to prove Lemma 4.1.

Proof of Lemma 4.1. Correctness follows from Propositions 4.4 and 4.3. The linear runtime is shown in Section 4.1.3. □

4.1.1 Proof of Proposition 4.3

Proof of Proposition 4.3. Assume the contrary, i.e., the algorithm stops and returns $L'(t)$ for some $t \geq 0$ such that corr $\setminus L'(t) \neq \emptyset$. Let $good$ = corr $\cap L'(t)$ and bad = corr $\setminus L'(t)$. Then corr $= bad \cup good$, $good \subseteq L'(t)$, $bad \cap L'(t) = \emptyset$ and $bad \neq \emptyset$. Note that $|bad| \leq |\mathrm{corr}| \leq \delta n$. We claim that there exists $i \in bad$ such that $N^1_{bad}(i) \geq h$. This is true since by Proposition E.1 we have $|N^1(bad)| \geq (2\epsilon - 1) \cdot c \cdot |bad|$ and hence at least one index $i \in bad$ sees at least $h = \lceil (2\epsilon - 1) \cdot c \rceil$ unique neighbor constraints, i.e., $|N^1_{bad}(i)| \geq h$.

Next we claim that for all $u \in N^1_{bad}(i)$ we have $u \in R'(t)$. Let $u \in N^1_{bad}(i)$ and note that there are two cases: $\langle u, w \rangle \neq 0$ or $\langle u, w \rangle = 0$. If $\langle u, w \rangle \neq 0$ then $u \in R_0 \subseteq R'(t)$ and we are done. Otherwise, we have $\langle u, w \rangle = 0$ and thus $u \notin N^1(\text{corr})$[3] but $u \in N^1(bad)$. Hence $\text{supp}(u) \cap good \neq \emptyset$, i.e., $u \in N(good)$. By definition of Find Erasures and the fact that $good \subseteq L'(t)$[4] it follows that $N(good) \subseteq R'(t)$ and thus $u \in R'(t)$.

We conclude that there exists $i \in [n] \setminus L'(t)$ such that $|N(i) \cap R'(t)| \geq h$. Contradiction to the fact that the algorithm was stopped. □

4.1.2 Proof of Proposition 4.4

Recall that $\epsilon > 1/2$, $h = \lceil (2\epsilon - 1)c \rceil$ and $\epsilon c + h - c > 0$. Let $\text{corr}_0 = \{i \in \text{corr} \mid |N(i) \cap R_0| \geq h\}$ and $\text{conf} = \{i \in [n] \setminus \text{corr} \mid |N(i) \cap R_0| \geq h\}$. By definition of corr_0 we have $\text{corr}_0 \subseteq L'(0) \subseteq L'(t)$ for every $t \geq 0$. Note that $\text{corr}_0 \subseteq \text{corr}$ and $|\text{corr}| < \frac{\epsilon \cdot c + h - c}{h} \cdot \lfloor \delta n \rfloor$.

We first present Claims 4.5, 4.6 and 4.7. Then we prove Proposition 4.4 using these claims.

Claim 4.5. *It holds that $|L'(0)| < \lfloor \delta n \rfloor$.*

Claim 4.6. *It holds that $|N(\text{corr}_0) \setminus R_0| \leq (c - h)\,|\text{corr}_0| - |R_0| + h\,|\text{corr}|$.*

Claim 4.7. *For every $t \geq 0$ it holds that $|N(L'(t) \setminus \text{corr}_0) \setminus (R_0 \cup N(\text{corr}_0))| \leq (c-h)(|L'(t)| - |\text{corr}_0|)$.*

We are ready to prove Proposition 4.4.

Proof of Proposition 4.4. Assume the contrary. Then there exists $t > 0$ such that $|L'(t)| = \lfloor \delta n \rfloor$ since by Claim 4.5 we have $|L'(0)| < \lfloor \delta n \rfloor$ and for all $t' > 0$ we have $|L'(t')| = |L'(t'-1)| + 1$. It holds that

$$\epsilon \cdot c \cdot \left|L'(t)\right| \leq \left|N(L'(t))\right| \leq |R_0| + \left|N(L'(t)) \setminus R_0\right| \leq$$

$$|R_0| + |N(\text{corr}_0) \setminus R_0| + \left|N(L'(t) \setminus corr_0) \setminus (R_0 \cup N(\text{corr}_0))\right| \leq$$

$$|R_0| + ((c-h)|\text{corr}_0| - |R_0| + h|corr|) + (|L'(t)| - |corr_0|)(c-h) = |L'(t)| \cdot (c-h) + |corr| \cdot h,$$

where the last inequality follows from Claims 4.6 and 4.7. We conclude that
$\epsilon \cdot c \cdot |L'(t)| \leq |L'(t)| \cdot (c-h) + |\text{corr}| \cdot h$ and $|L'(t)|(\epsilon \cdot c - c + h) \leq |\text{corr}| \cdot h$. Since $|\text{corr}| < \frac{\epsilon \cdot c + h - c}{h} \cdot \lfloor \delta n \rfloor$ it holds that $|L'(t)| < \frac{h}{\epsilon \cdot c - c + h} \cdot |\text{corr}| < \lfloor \delta n \rfloor$. Contradiction. □

Now we prove Claims 4.5, 4.6 and 4.7.

Proof of Claim 4.5. We show that $|\text{corr} \cup \text{conf}| < \lfloor \delta n \rfloor$ and this will imply that $|L'(0)| \leq |\text{corr} \cup \text{conf}| < \lfloor \delta n \rfloor$. Assume the contrary, i.e., $|\text{corr} \cup \text{conf}| \geq \lfloor \delta n \rfloor$. Let $\text{conf}' \subseteq \text{conf}$ be such that $|\text{corr} \cup \text{conf}'| = \lfloor \delta n \rfloor$. It holds that

$$\epsilon \cdot c \cdot \lfloor \delta n \rfloor \leq |N(\text{corr} \cup \text{conf}')| \leq c \cdot |\text{corr}| + (c-h)|\text{conf}'| = c \cdot |\text{corr}| + (c-h)(\lfloor \delta n \rfloor - |\text{corr}|).$$

This implies that $\lfloor \delta n \rfloor \cdot (\epsilon c + h - c) \leq |\text{corr}|(c - c + h)$ and hence $|\text{corr}| \geq \frac{\epsilon c + h - c}{h} \cdot \lfloor \delta n \rfloor$. Contradiction. □

[3] Note that for all $u' \in N^1(\text{corr})$ we have $\langle u', w \rangle \neq 0$.

[4] Find Erasures is defined such that for every $t \geq 0$ it holds that $N(L'(t)) \subseteq R'(t)$.

Proof of Claim 4.6. We first prove that $\sum\limits_{j\in\text{corr}_0} |N(j)\cap R_0| \geq |R_0| + h|\text{corr}_0| - h|\text{corr}|$. We have

$$\left(\sum_{j\in\text{corr}_0} |N(j)\cap R_0|\right) - h|\text{corr}_0| =$$

$$\left(\sum_{j\in\text{corr}_0} |N(j)\cap R_0|\right) + \left(\sum_{j\in(\text{corr}\setminus\text{corr}_0)} |N(j)\cap R_0|\right) - \left(\sum_{j\in(\text{corr}\setminus\text{corr}_0)} |N(j)\cap R_0|\right) - h|\text{corr}_0| \geq$$

$$\left(\sum_{j\in\text{corr}} |N(j)\cap R_0|\right) - h(|\text{corr}| - |corr_0|) - h|\text{corr}_0| \geq |R_0| - h|\text{corr}|,$$

where we used $\sum\limits_{j\in\text{corr}} |N(j)\cap R_0| \geq |R_0|$ since for all $u\in R_0$ we have $\text{supp}(u)\cap\text{corr}\neq\emptyset$ and $u\in N(\text{corr})$. We conclude that $\sum_{j\in\text{corr}_0} |N(j)\cap R_0| - h|\text{corr}_0| \geq |R_0| - h|\text{corr}|$ and $\sum_{j\in\text{corr}_0} |N(j)\cap R_0| \geq |R_0| + h|\text{corr}_0| - h|\text{corr}|$. Thus $|N(\text{corr}_0)\setminus R_0| \leq \sum_{j\in\text{corr}_0} (N(j)\setminus R_0) \leq c|\text{corr}_0| - \left(\sum_{j\in\text{corr}_0}(N(j)\cap R_0)\right) \leq (c-h)|\text{corr}_0| - |R_0| + h|\text{corr}|$. □

Proof of Claim 4.7. For all $t\geq 0$ we have $\text{corr}_0 \subseteq L'(0) \subseteq L'(t)$ and hence $(c-h)\,|L'(t)\setminus\text{corr}_0| = (c-h)\,(|L'(t)| - |\text{corr}_0|)$. Thus, it is sufficient to prove that $|N(L'(t)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))| \leq (c-h)|L'(t)\setminus\text{corr}_0|$ and we prove this by induction on t. Note that $\text{conf} = L'(0)\setminus\text{corr}_0$.

We first prove the base case: $|N(L'(0)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))| \leq |L'(0)\setminus corr_0|\,(c-h)$. By definition of conf it holds that $\sum_{i\in\text{conf}} |N(i)\setminus R_0| \leq |\text{conf}|(c-h) = |L'(0)\setminus\text{corr}_0|\,(c-h)$ and hence

$$\left|N(L'(0)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))\right| \leq |N(\text{conf})\setminus R_0| \leq \sum_{i\in\text{conf}} |N(i)\setminus R_0| \leq \left|L'(0)\setminus\text{corr}_0\right|(c-h).$$

For the induction step, assume the correctness for t, and let us prove that

$$\left|N(L'(t+1)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))\right| \leq (c-h)|L'(t+1)\setminus\text{corr}_0|.$$

The definition of the algorithm implies that $L'(t+1) = L'(t)\cup\{i\}$ for some $i\in[n]\setminus L'(t)$ such that $|N(i)\cap N(L'(t))| \geq h$. Recalling that $|N(i)| = c$ we have $|N(L'(t+1))| \leq |N(L'(t))| + |N(i)| - |N(i)\cap N(L'(t))| \leq |N(L'(t))| + (c-h)$.

We know that $L'(t)\subset L'(t+1)$ and $N(L'(t))\subseteq N(L'(t+1))$. Hence $|N(L'(t))\cap(R_0\cup N(\text{corr}_0))| \leq |N(L'(t+1))\cap((R_0\cup N(\text{corr}_0))|$. It follows that

$$\left|N(L'(t+1))\setminus(R_0\cup N(\text{corr}_0))\right| = \left|N(L'(t+1))\right| - \left|N(L'(t+1))\cap((R_0\cup N(\text{corr}_0))\right| \leq$$

$$\left|N(L'(t))\right| + (c-h) - \left|N(L'(t))\cap(R_0\cup N(\text{corr}_0))\right| = \left|N(L'(t))\setminus(R_0\cup N(\text{corr}_0))\right| + (c-h).$$

But

$$\left|N(L'(t+1))\setminus(R_0\cup N(\text{corr}_0))\right| = \left|N(L'(t+1)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))\right|$$

and

$$\left|N(L'(t))\setminus(R_0\cup N(\text{corr}_0))\right| = \left|N(L'(t)\setminus\text{corr}_0)\setminus(R_0\cup N(\text{corr}_0))\right|.$$

We conclude that

$$\left|N(L'(t+1) \setminus \mathrm{corr}_0) \setminus (R_0 \cup N(\mathrm{corr}_0))\right| \leq \left|N(L'(t) \setminus \mathrm{corr}_0) \setminus (R_0 \cup N(\mathrm{corr}_0))\right| + (c-h) \leq$$

$$\leq (c-h)|L'(t) \setminus \mathrm{corr}_0| + (c-h) = (c-h)|L'(t+1) \setminus \mathrm{corr}_0|,$$

where the last inequality follows from the induction assumption and the last equality holds since $L'(t+1) = L'(t) \cup \{i\}$ for $i \notin corr_0 \subseteq L'(t)$. This completes the induction step and proves the Claim. □

4.1.3 Find Erasures runs in Linear Time

It can be readily verified that the following algorithm is equivalent to Find Erasures defined in Section 4.1. The only difference is that we do not maintain the variable t indicating the iteration number.

```
Algorithm Find Erasures
  • Input: $w \in \mathbb{F}^n$ and $S \subseteq C_d^\perp$
  • Initialize:
      – $h := \lceil (2\epsilon - 1) \cdot c \rceil$
      – $R_0 := \{u \in S \mid \langle u, w \rangle \neq 0\}$
      – $R' := R_0$
      – $L' := \{i \in [n] \mid |N(i) \cap R_0| \geq h\}$
  • While there exists $i \in [n] \setminus L'$ such that $|N(i) \cap R'| \geq h$ do
      – $L' := L' \cup \{i\}$
      – $R' := R' \cup N(i)$
  • return $L'$
```

Note that $|S| = \frac{c \cdot n}{d} \leq O(n)$ and $|R_0| \leq |S|$. Hence R_0 can be constructed in linear time. Similarly, the sets L' and R' can be initialized in linear time. During the run of the algorithm each iteration of "while" can be executed in $O(1)$ time. The "while" loop stops after at most n iteration since $L' \subseteq [n]$ and each iteration its size grows by 1. Thus the run time of the algorithm is $O(n)$.

4.2 Decoding Expander Codes From Erasures — Proof of Lemma 4.2

First let us recall that Proposition E.1 implies that for any $\epsilon' > 1/2$ a $(c, d, \epsilon', \delta)$-expander code is a $(c, d, 2\epsilon' - 1, \delta)$-unique expander code. To prove Lemma 4.2 it is sufficient to assume that C is a (c, d, ϵ, δ)-unique expander code for some $\epsilon > 0$, i.e., we prove even a stronger claim than needed for Lemma 4.2.

Hence for the rest of this section we assume that $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-unique expander code, where $c, d, \delta, \epsilon > 0$ are constants and $S_C \subseteq C_d^\perp$ is an associated set of local constraints from the parity check graph of C. Let us define the algorithm Decode From Erasures that on the input word $w \in \mathbb{F}_2^n$, a set of constraints S_C and a subset $L' \subseteq [n]$ such that $|L'| \leq \delta n$ and $\mathrm{supp}(w - \mathrm{x}) \subseteq L'$ for some codeword $\mathrm{x} \in C$, outputs the codeword x.

```
Decode From Erasures
  • Input: $w \in \mathbb{F}^n$, $L' \subseteq [n]$ and $S_C \subseteq C_d^{\perp}$
  • $Unique := \{u \in S_C \mid |L' \cap \mathrm{supp}(u)| = 1\}$.
  • While there exists $i \in L'$ and $u \in Unique$ such that $\mathrm{supp}(u) \cap L' = \{i\}$:
      – If $\langle u, w \rangle \neq 0$ then flip bit $i$ of $w$
      – $L' := L' \setminus \{i\}$
      – For all $u' \in N(\{i\})$ do
          ∗ If $|\mathrm{supp}(u') \cap L'| = 1$ then $Unique := Unique \cup \{u'\}$
          ∗ Otherwise, $Unique := Unique \setminus \{u'\}$.
      – (Note that now we have $Unique := \{v \in S_C \mid |L' \cap \mathrm{supp}(v)| = 1\}$)
  • Return $w$.
```

Intuition behind the algorithm Decode From Erasures. Let $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$ be a set of corrupted bits and note that initially $\mathrm{corr} \subseteq L'$. We notice that it might be that $L' \setminus \mathrm{corr} \neq \emptyset$. Since C is a unique expander, there exists (at least one) constraint $u \in S_C$ such that $\mathrm{supp}(u) \cap L' = \{i\}$ for some $i \in L'$. This constraint u can be used to determine (in time $O(1)$) whether w_i is corrupted, i.e., $i \in \mathrm{corr}$ and it is needed to flip this bit. After the algorithm handled the bit i (and flipped it, if needed) this bit is removed from L' since it is not corrupted more, and the decoding is continued recursively on $L' \setminus \{i\}$.

Now we prove Lemma 4.2.

Proof of Lemma 4.2. Let $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$ and think of corr as a set of corrupted bits. Note that initially $\mathrm{corr} \subseteq L'$. The algorithm constructs $Unique = \{u \in S_C \mid |L' \cap \mathrm{supp}(u)| = 1\}$, i.e., the set of unique neighbor constraints for the set L'.

Each iteration the algorithm selects some $u \in Unique$ and $i \in L'$ such that $\mathrm{supp}(u) \cap L' = \{i\}$. Note that the fact that $|\mathrm{supp}(u) \cap L'| = 1$ implies that $|\mathrm{supp}(u) \cap \mathrm{corr}| \leq 1$. We argue that $\mathrm{supp}(u) \cap \mathrm{corr} = \emptyset$ iff $\langle u, w \rangle = 0$, or in words, $\mathrm{supp}(u)$ does not contain a corrupted bit iff the constraint u is satisfied. This is true since if $\mathrm{supp}(u) \cap \mathrm{corr} = \emptyset$ then $\langle u, w \rangle = \langle u, \mathrm{x} \rangle = 0$, but if $|\mathrm{supp}(u) \cap \mathrm{corr}| = 1$ then $\langle u, w \rangle \neq \langle u, \mathrm{x} \rangle = 0$. Hence $i \in \mathrm{corr}$ if and only if $\langle u, w \rangle \neq 0$, and thus the algorithm flips i only if this bit is corrupted. After that the algorithm removes i from L', since i is not corrupted more and updates the set $Unique$ to contain the unique neighbor constraints of the modified L'. Hence the Decode From Erasures algorithm flips only the corrupted bits ($i \in \mathrm{corr}$) and finally outputs x.

We turn to explain that the run-time of the algorithm Decode From Erasures is $O(n)$. The appropriate data structure containing the set S_C can be constructed in $O(n)$ time since $|S_C| = \frac{c \cdot n}{d} = O(n)$. For every $u \in S_C$ it is possible to check in time $O(1)$ whether $|L' \cap \mathrm{supp}(u)| = 1$, by checking for every $j \in \mathrm{supp}(u)$ whether $j \in L'$ since $|\mathrm{supp}(u)| = d = O(1)$. E.g., this can be done by maintaining a boolean array of size n, where a cell i is set to 1 iff $i \in L'$. Note also that the set $Unique$ is modified in each "while" iteration in time $O(1)$, since the algorithm reconsiders only c constraints of S_C, and for every such $u \in S_C$ checks whether $|L' \cap \mathrm{supp}(u)| = 1$. So, each "while" iteration can be executed in time $O(1)$. Since the number of "while" iterations is upper-bounded by the blocklength n we conclude that the total run-time of Decode From Erasures is $O(n)$. □

5 Open Questions

The main open question which remains unsolved after this work is following.

Conjecture 5.1. *Let $c, d, \delta > 0$ and $\epsilon > 1/2$ be constants. Then there exists a constant $\alpha > 0$, which depends only on c, d, ϵ, δ, such that if $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-expander code then C is decodable in linear time from αn errors.*

Recall that our suggested algorithm Find Erasures and Decode is composed from Find Erasures and Decode From Erasures algorithms, where Decode From Erasures works for every expansion parameter $\epsilon > 1/2$. So, Find Erasures algorithm remains to be the bottleneck in this sense.

We feel that a more tight analysis or improvement of Find Erasures algorithm would play a crucial role in proving Conjecture 5.1, however in this paper we were unable to do it.

Acknowledgements

The author thanks Eli Ben-Sasson and Madhu Sudan who refused to co-author the paper for many invaluable discussions and ideas. In particular, we would like to thank Eli Ben-Sasson for valuable comments on an earlier draft and endless encouragement. The author is grateful to Madhu Sudan for asking about the minimal expansion requirement that is sufficient for linear-time decoding, which led to the results presented in this paper. The author is much indebted to Madhu Sudan for close supervision during the internship of the author in Microsoft Research, New England. We thank David Steurer for pointers to the literature.

Many thanks to Microsoft Research, New England for its hospitality.

References

[1] A. Barg and G. Zémor, "Error exponents of expander codes," *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1725–1729, 2002.

[2] ——, "Distance properties of expander codes," *IEEE Transactions on Information Theory*, vol. 52, no. 1, pp. 78–90, 2006. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TIT.2005.860415

[3] A. Bennatan and D. Burshtein, "On the application of LDPC codes to arbitrary discrete-memoryless channels," *IEEE Transactions on Information Theory*, vol. 50, no. 3, pp. 417–438, 2004.

[4] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson, "Randomness conductors and constant-degree lossless expanders," in *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC-02)*. New York: ACM Press, May 19–21 2002, pp. 659–668.

[5] J. Feldman, "Decoding error-correcting codes via linear programming," Ph.D. dissertation, 2003. [Online]. Available: http://hdl.handle.net/1721.1/42831

[6] ——, "LP Decoding," in *Encyclopedia of Algorithms*, M.-Y. Kao, Ed. Springer, 2008. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-30162-4_216

[7] J. Feldman and D. R. Karger, "Decoding turbo-like codes via linear programming," *J. Comput. Syst. Sci*, vol. 68, no. 4, pp. 733–752, 2004. [Online]. Available: http://dx.doi.org/10.1016/j.jcss.2003.11.005

[8] J. Feldman, T. Malkin, R. A. Servedio, C. Stein, and M. J. Wainwright, “LP decoding corrects a constant fraction of errors,” *IEEE Transactions on Information Theory*, vol. 53, no. 1, pp. 82–89, 2007. [Online]. Available: http://dx.doi.org/10.1109/TIT.2006.887523

[9] R. G. Gallager, *Information Theory and Reliable Communication*. Wiley, New York, 1968.

[10] ——, *Low-Density Parity-Check Codes*. Cambridge, MA, USA: The M.I.T. Press, 1963.

[11] V. Guruswami and P. Indyk, “Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets,” in *STOC*, 2002, pp. 812–821. [Online]. Available: http://doi.acm.org/10.1145/509907.510023

[12] ——, “Linear time encodable and list decodable codes,” in *STOC*. ACM, 2003, pp. 126–135. [Online]. Available: http://doi.acm.org/10.1145/780542.780562

[13] ——, “Linear-time list decoding in error-free settings,” in *ICALP*, ser. Lecture Notes in Computer Science, J. Díaz, J. Karhumäki, A. Lepistö, and D. Sannella, Eds., vol. 3142. Springer, 2004, pp. 695–707. [Online]. Available: http://springerlink.metapress.com/openurl.asp?genre=article&amp;issn=0302-9743&amp;volume=3142&amp;spage=695

[14] ——, “Linear-time encodable/decodable codes with near-optimal rate,” *IEEE Transactions on Information Theory*, vol. 51, no. 10, pp. 3393–3400, 2005. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TIT.2005.855587

[15] A. Lubotzky, R. Phillips, and P. Sarnak, “Ramanujan graphs,” *Combinatorica*, vol. 8, no. 3, pp. 261–277, 1988.

[16] D. J. C. MacKay and R. M. Neal, “Near Shannon Limit Performance of Low Density Parity Check Codes,” *Electronic Letters*, vol. 32, no. 18, pp. 1645–1646, Aug. 1996.

[17] ——, “Good codes based on very sparse matrices,” in *Proceedings of the 5th IMA Conference on Cryptography and Coding, (Cirencester, United Kingdom, December 18-20, 1995)*, ser. LNCS, C. Boyd, Ed. Berlin-Heidelberg-New York-London-Paris-Tokyo-Hong Kong-Barcelona-Budapest: Springer-Verlag, 1995, vol. 1025, pp. 100–111. [Online]. Available: http://springerlink.metapress.com/openurl.asp?genre=article&issn=0302-9743&volume=1025&spage=100

[18] G. A. Margulis, “Explicit constructions of graphs without short cycles and low density codes,” *Combinatorica*, vol. 2, no. 1, pp. 71–78, 1982.

[19] J. Pearl, *Probabilistic Inference in Intelligent Systems. Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[20] Reingold, Vadhan, and Wigderson, “Entropy waves, the zig-zag graph product, and new constant-degree expanders,” *ANNALSMTH: Annals of Mathematics*, vol. 155, 2002.

[21] T. J. Richardson and R. L. Urbanke, “Efficient encoding of low-density parity-check codes,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 638–656, 2001.

[22] M. Sipser and D. A. Spielman, “Expander codes,” *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710–1722, 1996, preliminary version appeared in FOCS 1994.

[23] D. A. Spielman, “Linear-time encodable and decodable error-correcting codes,” *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1723–1731, 1996.

[24] M. Sudan, “List decoding: algorithms and applications,” *SIGACT News*, vol. 31, no. 1, pp. 16–27, 2000. [Online]. Available: http://doi.acm.org/10.1145/346048.346049

[25] R. M. Tanner, “A recursive approach to low complexity codes,” *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[26] Trevisan, “Some applications of coding theory in computational complexity,” in *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 2004.

[27] G. Zémor, “On expander codes,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 835–837, 2001.

[28] V. V. Zyablov and M. S. Pinsker, “Estimation of the error-correction complexity of gallger low-density codes.” *Problems of Information Transmission*, vol. 11(1), pp. 18–28, 1976.

A Find Erasures and Decode **Algorithm extends** Flip Algorithm

In this section we prove Theorem A.1, which says that all regular expander codes with expansion parameter $\epsilon > 1/2$, decodable by Flip Algorithm, are decodable by Find Erasures and Decode algorithm. Recall that in Section D we show that the expansion parameter $\epsilon \leq 1/2$ is insufficient for the decoding even from 2 errors.

To prove Theorem A.1 we consider the following version of Find Erasures algorithm which is identical to Find Erasures, defined in Section 4.1, with a single change that the threshold h is set to $c/2+1/2$. For the rest of this section $C \subseteq \mathbb{F}_2^n$ denotes a (c, d, ϵ, δ)-expander code and $S_C \subseteq C_d^{\perp}$ denotes a set of low-weight constraints from the parity check graph of C.

```
Find Erasures
  • Input: $w \in \mathbb{F}_2^n$ and $S_C \subseteq C_d^{\perp}$
  • Initialize:
      – $h := c/2 + 1/2$
      – $R'(0) := \{u \in S_C \mid \langle u, w \rangle \neq 0\}$
      – $L'(0) := \{i \in [n] \mid |N(i) \cap R'(0)| \geq h\}$
      – $t := 0$
  • While there exists $i \in [n] \setminus L'(t)$ such that $|N(i) \cap R'(t)| \geq h$ do
      – $L'(t+1) := L'(t) \cup \{i\}$
      – $R'(t+1) := R'(t) \cup N(i)$
      – $t := t + 1$
  • return $L'(t)$
```

We stress that the algorithm Decode From Erasures remains the same as it was defined in Section 4.2 and the algorithm Find Erasures and Decode remains the same as it was defined in Section 4, i.e., the combination of Find Erasures, defined above, and Decode From Erasures.

Theorem A.1. *Let $C \subseteq \mathbb{F}_2^n$ be a (c, d, ϵ, δ)-expander code, where $\epsilon > 1/2$. Assume C is decodable from m errors using* Flip Algorithm*. Then C is linear time decodable from $\min\{m, (\epsilon - 1/2 + 1/(2c))\lfloor \delta n \rfloor\}$ errors using algorithm* Find Erasures and Decode.

Before proving this theorem, we first present Proposition A.2 which will be used in the proof of the Theorem A.1.

Proposition A.2. *All instances of the algorithm* Find Erasures *on the same input output the same subset L'.* [5]

We are ready to prove Theorem A.1.

Proof of Theorem A.1. Let $w \in \mathbb{F}_2^n$ such that $\delta(w, \mathrm{x}) \leq \min\{m, (\epsilon - 1/2 + 1/(2c))\lfloor \delta n \rfloor\}$, where $\mathrm{x} \in C$ is the closest codeword to w. Let $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$ and think of corr as a subset of corrupted variables. By assumption w can be decoded to x using Flip Algorithm. Consider a run of Flip Algorithm on w and let $I = \{i_1, ..., i_k\} \subseteq [n]$ be a *multiset* of bits indices of w which were flipped during the run of Flip Algorithm on w. Note that it might be the case that for some $j_1 < j_2$ we have $i_{j_1} = i_{j_2}$, i.e., the same bit was flipped more than once. Since w is decoded to x we know that $\mathrm{corr} \subseteq I$.

We argue that Find Erasures and Decode algorithm on the input w will output x, and note that the run-time of Find Erasures and Decode is linear as was explained in Section 4.1.3 and Lemma 4.2. It is sufficient to prove that an algorithm Find Erasures on w will output $L'(t) \subseteq [n]$ such that $|L'(t)| \leq \delta n$ and $\mathrm{corr} \subseteq L'(t)$. In this case, Lemma 4.2 implies that Decode From Erasures (on the input w, $L'(t)$ and S_C) will output x and so Find Erasures and Decode succeeds.

We first argue that Find Erasures outputs L' such that $|L'| < \lfloor \delta n \rfloor$. Assume the contrary, but then there exists an iteration, where $|L'(t)| = \lfloor \delta n \rfloor$ since $|L'(0)| < \lfloor \delta n \rfloor$ (see Claim 4.5) and $L'(\cdot)$ grows by 1 each iteration. We know that $\epsilon \cdot c \cdot |L'(t)| \leq |N(L'(t))|$ because of the expansion properties of C. Note that for every iteration $t' + 1$ if the algorithm Find Erasures executes $L'(t'+1) := L'(t') \cup \{i\}$ then it holds that $|N(i) \cap R'(t')| \geq h$, i.e., $|N(i) \setminus R'(t')| \leq (c - h)$. Recall that $R'(t'+1) = R'(t') \cup N(i)$ and hence $|R'(t'+1)| \leq |R'(t')| + (c - h)$, i.e., every iteration $R'(\cdot)$ grows at most by $(c - h)$. We conclude that $|N(L'(t))| \leq |R'(0)| + (c - h)|L'(t)|$. Hence $\epsilon \cdot c \cdot |L'(t)| \leq |N(L'(t))| \leq |R'(0)| + (c - h)|L'(t)|$. Then,

$$(\epsilon \cdot c - (c/2 - 1/2))|L'(t)| \leq |R'(0)| \leq |\mathrm{corr}| \cdot c \quad \text{and}$$

$$|L'(t)| \leq \frac{|\mathrm{corr}|}{\epsilon - 1/2 + 1/(2c)} < \lfloor \delta n \rfloor \text{ since } |\mathrm{corr}| < (\epsilon - 1/2 + 1/(2c))(\lfloor \delta n \rfloor)$$

Contradiction.

It is remained to prove that Find Erasures on the input w and S_C will output $L' \subseteq [n]$ such that $\mathrm{corr} \subseteq L'$. By Proposition A.2 it is sufficient to show at least one such instance of the algorithm Find Erasures. We show that there exists an instance of Find Erasures that on the input w and S_C outputs L' such that $I \subseteq L'$.[6] Recall that $\mathrm{corr} \subseteq I$.

We argue that there exists an instance of Find Erasures such that each iteration when Flip Algorithm flips a bit i_j, the algorithm Find Erasures executes $L'(t+1) := L'(t) \cup \{i_j\}$ for some $t \geq 0$ (of course, if

[5] The proposition is correct regardless of the initialization of the threshold h.

[6] Note that L' is a set but not a multiset.

$i_j \notin L'(t)$). Recall that Flip Algorithm modifies the underlying word w (by flipping its bits) and so, let us denote $w_0 = w$ and for every $j \geq 0$ let w_j be a word after jth iteration of Flip Algorithm, i.e., after flipping bit i_j. Note that for all $j \geq 1$ we have $\mathrm{supp}(w_j - w_{j+1}) = \{i_{j+1}\}$. We prove by induction on the iteration number j that there exists an instance of the algorithm Find Erasures such that for every $j \leq |I| = k$ it holds that $\{i_1, ..., i_j\} \subseteq L'(j)$ and for all $u \in S_C$ such that $\langle u, w_j \rangle \neq 0$ we have $u \in R'(j)$.

This is true for $L'(0)$ and w_0, i.e., $\emptyset \subseteq L'(0)$ and for all $u \in S_C$ such that $\langle u, w_0 \rangle \neq 0$ we have $u \in R'(0)$. Assume the correctness for j, i.e., $\{i_1, ..., i_j\} \subseteq L'$ and for all $u \in S_C$ such that $\langle u, w_j \rangle \neq 0$ we have $u \in R'(j)$. By assumption, Flip Algorithm flips the bit i_{j+1} in the iteration $j+1$ and thus it follows that there are more than $c/2$ of $u \in S_C$ such that $i_{j+1} \in \mathrm{supp}(u)$ and $\langle u, w_j \rangle \neq 0$.[7]

We first prove that $i_{j+1} \in L'(j+1)$. If $i_{j+1} \in L'(j)$ we are done since $L'(j) \subseteq L'(j+1)$. Thus assume that $i_{j+1} \notin L'(j)$. By the inductive assumption, for all $u \in S_C$ such that $\langle u, w_j \rangle \neq 0$ we have $u \in R'(j)$, hence there are more than $c/2$ of $u \in R'(j)$ such that $i_{j+1} \in \mathrm{supp}(u)$. This implies that $|N(i_{j+1}) \cap R'(j)| > c/2$ and $|N(i_{j+1}) \cap R'(j)| \geq c/2 + 1/2 = h$. We conclude that Find Erasures can execute $L'(j+1) := L'(j) \cup \{i_{j+1}\}$ and $R'(j+1) := R'(j) \cup N(i_{j+1})$ in the iteration $j+1$.

We argue that for every $u \in S_C$ such that $\langle u, w_{j+1} \rangle \neq 0$ we have $u \in R'(j+1)$. To see this, note that if $\langle u, w_{j+1} \rangle \neq 0$ then either $(\langle u, w_j \rangle \neq 0 \text{ and } u \in R'(j) \subseteq R'(j+1))$ or $(\langle u, w_j \rangle = 0 \text{ and } \langle u, w_{j+1} \rangle \neq 0)$. In the last case, where $\langle u, w_j \rangle = 0$ and $\langle u, w_{j+1} \rangle \neq 0$ we have $i_{j+1} \in \mathrm{supp}(u)$ since $\mathrm{supp}(w_j - w_{j+1}) = \{i_{j+1}\}$ and thus $u \in R'(j+1)$.

Hence we showed that $\mathrm{corr} \subseteq I \subseteq L'$ and this completes the proof of Theorem A.1. □

Now we prove Proposition A.2.

Proof of Proposition A.2. Let $w \in \mathbb{F}_2^n$ be an input word. Recall that Find Erasures maintains subsets $L'(\cdot)$ and $R'(\cdot)$.

Assume two different instances of Find Erasures: the instance A_1 which maintains subsets $L'_1(\cdot)$, $R'_1(\cdot)$ and the instance A_2 which maintains $L'_2(\cdot)$, $R'_2(\cdot)$, respectively. Assume also that A_1 outputs L'_1 and A_2 outputs L'_2. We have $R'_1(0) = R'_2(0)$ and $L'_1(0) = L'_2(0)$. We know that for every $i \geq 0$ it holds that $L'_1(i) \subseteq L'_1$, $L'_2(i) \subseteq L'_2$, $R'_1(i) \subseteq R'_1$, $R'_2(i) \subseteq R'_2$.

We prove by induction that for every $i \geq 0$ we have $L'_1(i) \subseteq L'_2$, $L'_2(i) \subseteq L'_1$, $R'_1(i) \subseteq R'_2$ and $R'_2(i) \subseteq R'_2$. This will imply that $L'_1 = L'_2$.

It holds that $R'_1(0) = R'_2(0)$ and $L'_1(0) = L'_2(0)$, and hence $L'_1(0) \subseteq L'_2$, $L'_2(0) \subseteq L'_1$, $R'_1(0) \subseteq R'_2$ and $R'_2(0) \subseteq R'_1$. Assume by induction that $L'_1(i) \subseteq L'_2$, $L'_2(i) \subseteq L'_1$, $R'_1(i) \subseteq R'_2$ and $R'_2(i) \subseteq R'_2$. Consider the iteration $i+1$ of the first instance A_1 and assume that it executes $L'_1(i+1) := L'_1(i) \cup \{j\}$. It follows that $|N(j) \cap L'_2| \geq |N(j) \cap L'_1(i)| \geq h$ and hence $|N(j) \cap L'_2(t)| \geq h$ for some $t \geq 0$. Then $j \in L'_2(t')$ for some $t' \geq t$ since otherwise the instance A_2 would not stop. Hence $L'_1(i+1) \subseteq L'_2$ and $R'_1(i+1) \subseteq R'_2$ due to the same reason. Similarly, $L'_2(i+1) \subseteq L'_1$ and $R'_2(i+1) \subseteq R'_1$. This completes the induction and the proof of the proposition. □

B Decoding below the expansion $2/3$

We show that $2/3$ was not a principle barrier and a similar algorithm to Find Erasures and Decode can be applied for the regular expander codes with expansion parameter $> 2/3 - 1/(6c) = 2/3 - \Omega(1)$ to correct $\Omega(n)$ errors. [8]

[7] Recall that since C is (c,d)-regular we have $\{u \in S_C \mid i_{j+1} \in \mathrm{supp}(u)\}$.

[8] We do not try to optimize the number of errors the algorithm will correct.

Theorem B.1. *Let $C \subseteq \mathbb{F}_2^n$ be a (c,d,ϵ,δ)-expander code such that $c,d,\delta > 0$ are constants and $\epsilon > 2/3 - 1/(6c)$. Then C is linear-time decodable from $\Omega_{c,d,\epsilon,\delta}(n)$ errors.*

First we explain that in the following cases we have already done, by Theorem 3.1.

- If $\epsilon > 2/3 - 1(6c)$ and $c \mod 3 = 2$ then $\epsilon > 1/2$ and $h = \lceil (2\epsilon - 1)c \rceil \geq c/3 - 1/3 + 2/3 = c/3 + 1/3$, where the inequality follows due to ceiling. Hence $\epsilon c + h - c > (2/3)c - 1/6 + c/3 + 1/3 - c = 1/6 > 0$ and we are done by Theorem 3.1.
- If $\epsilon > 2/3 - 1(6c)$ and $c \mod 3 = 1$ then $h = \lceil (2\epsilon - 1)c \rceil > c/3 - 1/3$. But $c/3 - 1/3$ is an integer and h is an integer by definition. Hence $h \geq c/3 - 1/3 + 1 = c/3 + 2/3$. So $\epsilon > 1/2$ and $\epsilon c + h - c > (2/3)c - (1/6) + (c/3) + (2/3) - c = 1/2 > 0$ and we are done by Theorem 3.1.
- If $\epsilon > 2/3 - 1(6c)$, $c \mod 3 = 0$ (so $c \geq 3$) and $h = \lceil (2\epsilon - 1)c \rceil > c/3$ then $h \geq c/3 + 1$ since $c/3$ and h are integers. So, we have $\epsilon c + h - c > 1/2$ and we are done by Theorem 3.1.

So, to prove Theorem B.1 it is remained to consider a case, where $\epsilon > 2/3 - 1(6c)$, $c \mod 3 = 0$ and $h = \lceil (2\epsilon - 1)c \rceil = c/3$[9]. We prove this case of Theorem B.1 in Section B.1.

B.1 Proof of Theorem B.1

In this section we assume that $C \subseteq \mathbb{F}_2^n$ is a (c,d,ϵ,δ)-expander code, where $\epsilon > 2/3 - 1/(6c)$, $c \mod 3 = 0$ and $h = \lceil (2\epsilon - 1)c \rceil = c/3$. We prove that C is decodable in linear time from $\Omega(n)$ errors.

Now, we (re)define an algorithm Find Erasures. The only difference between this version of the algorithm and the version defined in Section 4 is that in the current version of Find Erasures each iteration an erasure subset L' can grow by 1 or 2.

```
Find Erasures
  • Input: $w \in \mathbb{F}^n$ and $S \subseteq C_d^\perp$
  • Initialize:
      – $h := \lceil (2\epsilon - 1)c \rceil$
      – $R_0 := \{u \in S \mid \langle u, w \rangle \neq 0\}$
      – $R' := R_0$
      – $L' := \emptyset$
  • While true do
      – $Temp := \{i \in [n] \setminus L' \mid |N(i) \cap R'| \geq h\}$
      – If Find Subset$(Temp, h, S, R') == Failure$ then return $L'$
      – Otherwise:
          ∗ $Subset :=$ Find Subset$(Temp, h, S, R')$
          ∗ $L' := L' \cup Subset$
          ∗ $R' := R' \cup N(Subset)$
```

[9] Note that for $\epsilon > 2/3 - 1(6c)$ and $c \mod 3 = 0$ we have $\lceil (2\epsilon - 1)c \rceil \geq c/3$

Now we define the algorithm Find Subset.

Find Subset

- Input: $Temp \subseteq [n]$, h ,S and R'
- If three exists $i \in Temp$ such that $|N(i) \cap R'| \geq h+1$ then return $\{i\}$
- Otherwise, if there exists $u \in S \setminus R'$ such that for some $i_1 \neq i_2 \in \mathrm{supp}(u)$ we have $|N(i_1) \cap R'| \geq h$ and $|N(i_2) \cap R'| \geq h$, then return $\{i_1, i_2\}$
- Otherwise, return Failure

For the rest of this section let $w \in F_2^n$ be a (fixed) input word such that for some (fixed) $\mathrm{x} \in C$ it holds that $\Delta(w, \mathrm{x}) < (3\epsilon - 2 + 1/(2c))\lfloor(\delta/2)n\rfloor$. Let $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$ and note that $|\mathrm{corr}| < (3\epsilon - 2 + 1/(2c))\lfloor(\delta/2)n\rfloor$.

Lemma B.2. Find Erasures *algorithm runs linear time and returns* $L' \subseteq [n]$ *such that* $\mathrm{corr} \subseteq L'$ *and* $|L'| < \delta n$.

We postpone the proof of Lemma B.2 to Section B.1.1 and turn to prove Theorem B.1.

Proof. It can be easily verified that Find Erasures and Decode algorithm, defined in Section 4 with the only difference that it invokes the current version of Find Erasures algorithm, obtains the desired results. This is true since by Lemma B.2 the algorithm Find Erasures returns L' such that $\mathrm{corr} \subseteq L'$ and $|L'| < \delta n$. Then by Lemma 4.2 the algorithm Decode From Erasures returns x. The linear runtime follows immediately from the Lemmas B.2 and 4.2. □

B.1.1 Proof of Lemma B.2

We present Propositions B.3 and B.4. The proof of Lemma B.2 will follow from these propositions.

Proposition B.3. *Algorithm* Find Erasures *returns* L' *such that* $|L'| < \lfloor \delta n \rfloor - 1$.

We now prove that when algorithm returns L' we have $\mathrm{corr} \subseteq L'$.

Proposition B.4. *Algorithm* Find Erasures *returns* L' *such that* $\mathrm{corr} \subseteq L'$.

We are ready to prove Lemma B.2.

Proof of Lemma B.2. Proposition B.3 implies that $|L'| < \delta n$. Proposition B.4 implies that $\mathrm{corr} \subseteq L'$. It can be verified that the runtime of Find Erasures is linear because of the similar reasons discussed in Section 4.1.3. □

Now we prove Propositions B.3 and B.4.

Proof of Proposition B.3. We first prove an auxiliary claim.

Claim B.5. *If Algorithm* Find Subset *returns subset* $T \subseteq [n]$ *then* $|N(T) \setminus R'| \leq |T|(c - h - 1/2)$.

Proof. Recall that $|N(i)| = c$. If Find Subset returns $\{i\}$ then $|N(i) \cap R'| \geq h + 1$ and hence $|N(\{i\}) \setminus R'| \leq (c - h - 1) \leq |\{i\}|(c - h - 1/2)$. Otherwise, Find Subset returns $\{i_1, i_2\}$ such that for some $u \in S \setminus R'$ we have $i_1 \neq i_2 \in \mathrm{supp}(u)$, $|N(i_1) \cap R'| \geq h$ and $|N(i_2) \cap R'| \geq h$. Then $|N(\{i_1, i_2\}) \setminus R'| \leq (2c - 2h - 1) = |\{i_1, i_2\}|(c - h - 1/2)$. □

We continue to prove the proposition. Assume the contrary. Since L' grows at most by 2 in each iteration this means for some iteration we have $\lfloor \delta n \rfloor - 1 \leq |L'| \leq \lfloor \delta n \rfloor$. Claim B.5 implies that each iteration R' is increased at most by $(c - h - 1/2)$ per index in L', i.e., R' is bounded by $|R'| \leq |R_0| + |L'|(c - h - 1/2)$. It holds that $\epsilon c|L'| \leq |N(L')| \leq |R'| \leq |R_0| + |L'|(c - h - 1/2)$ and $|R_0| \leq c|\mathrm{corr}|$. Thus $|L'|(\epsilon c - c + h + 1/2) \leq c|\mathrm{corr}|$. We conclude that $|L'| \leq \frac{c|\mathrm{corr}|}{\epsilon c - c + h + 1/2} \leq \frac{|\mathrm{corr}|}{3\epsilon - 2 + 1/(2c)}$. Note that if $\epsilon > 2/3 - (1/(6c))$ and $|\mathrm{corr}| < (3\epsilon - 2 + 1/(2c))\lfloor(\delta/2)n\rfloor$ then $|L'| \leq (\delta/2)n < \lfloor \delta n \rfloor - 1$. Contradiction. □

Proof of Proposition B.4. Assume the contrary, i.e., Find Erasures returns L' but $\mathrm{corr} \setminus L' \neq \emptyset$. Let $bad = \mathrm{corr} \setminus L'$ and $good = \mathrm{corr} \cap L'$. Note that $\mathrm{corr} = good \cup bad$ and $good \cap bad = \emptyset$. Since the algorithm Find Erasures was stopped it follows that for all $i \in bad$ we have $|N(i) \cap bad| \leq h$. Let $bad_h = \{i \in bad \mid |N(i) \cap bad| = h\}$ and $bad_l = \{i \in bad \mid |N(i) \cap bad| < h\}$. It is sufficient to show that there exists $u \in S \setminus R'$ such that $|\mathrm{supp}(u) \cap bad_h| \geq 2$. This comes in contradiction to the fact that the algorithm Find Erasures was stopped.

So, we argue that there exists $u \in S \setminus R'$ such that $|\mathrm{supp}(u) \cap bad_h| \geq 2$. Recall that $Temp = \{i \in [n] \setminus L' \mid |N(i) \cap R'| \geq h\}, c \mod 3 = 0$ (so $c \geq 3$) and $h = c/3$. We have $|N(Temp)| \geq \epsilon c|Temp|$ that implies $N^1(Temp) \geq (2\epsilon - 1)c|Temp| > (c/3 - 1/3)$ (Proposition E.1). Hence an average index $i \in Temp$ sees more than $(c/3 - 1/3)$ unique neighbors. Moreover, we know that for all $i \in Temp$ it holds that $|N^1_{Temp}(i)| \leq h = c/3$. We conclude that $|bad_h| > (2/3)|bad|$.

However, for all $i \in bad_h$ we have $c - h = (2/3)c$ vectors $u \in S$ such that $i \in \mathrm{supp}(u)$ and $u \notin N^1_{bad}(i)$. This implies that for all $i \in bad_h$ we have at least $(2/3)c$ vectors $u \in S$ such that $i \in \mathrm{supp}(u)$ and $|\mathrm{supp}(u) \cap bad| \geq 2$. Assume by contradiction that for all these vectors u we have $|\mathrm{supp}(u) \cap bad_h| = 1$ but then $|\mathrm{supp}(u) \cap bad_l| \geq 1$ since $u \notin N^1_{bad}(i)$. We conclude that $\sum_{i \in bad_l} |N(i)| \geq (2/3)c|bad_h| > (4/9)c|bad|$. Hence $|bad_l| > 4/9|bad|$ with contradiction to the fact that $|bad_h| > (2/3)|bad|$ and $|bad_l| + |bad_h| = |bad|$.

This completes the proof of the proposition. □

C Proof of Theorem 3.5

Assume that $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-unique expander code, where $c, d, \epsilon, \delta > 0$ and $S \subseteq C_d^\perp$ is a set of local constraints from the parity check graph of C. In this section, for $T \subseteq [n]$ we let $N(T) = \{u \mid \mathrm{supp}(u) \cap T \neq \emptyset\}$ be the neighbors of T. With some abuse of notation, for $U \subseteq C_d^\perp$ let $\mathrm{supp}_\epsilon(U) = \{i \in [n] \mid |N(i) \cap U| \geq \epsilon c\}$.

Now we define an algorithm Simple Find Erasures, which is an instance of the algorithm Find Erasures. Simple Find Erasures is very simple to analyze and works for all unique expander codes but corrects only a sublinear number of errors.

```
Simple Find Erasures
  • Input: $w \in \mathbb{F}^n$, integer $i$ and $S \subseteq C_d^{\perp}$
  • $N^{unsat} := \{u \in S \mid \langle u, w \rangle \neq 0\}$
  • $L'(1) := \mathrm{supp}_{\epsilon}(N^{unsat})$
  • For $j = 2, \ldots i$ do:
      – $L'(j) = L'(j-1) \cup \mathrm{supp}_{\epsilon}(N(L'(j-1)))$
  • Return $L'(i)$
```

We turn to analyze this algorithm and prove Theorem 3.5. We let $\alpha > 0$ be such that $\log_{1-\epsilon} n^{-\alpha} < \lfloor \log_{(d/\epsilon)} \delta n^{1-\alpha} \rfloor$. Clearly, such α exists and it depends only on d, ϵ, δ.

For the rest of this section let $w \in \mathbb{F}^n$ such that $\Delta(w, \mathrm{x}) \leq n^{\alpha}$ for some $\mathrm{x} \in C$ and $\mathrm{corr} = \mathrm{supp}(w - \mathrm{x})$. Note that $|\mathrm{corr}| \leq n^{\alpha}$ and we have the following inequality.

$$\log_{(1-\epsilon)} \frac{1}{|\mathrm{corr}|} < \left\lfloor \log_{(d/\epsilon)} \frac{\delta n}{|\mathrm{corr}|} \right\rfloor \tag{1}$$

First of all we present the following lemma. Then we prove Theorem 3.5.

Lemma C.1. *If $i = \lfloor \log_{(d/\epsilon)} \delta n^{1-\alpha} \rfloor$ then algorithm* Simple Find Erasures *on the input (w, i, S) runs linear time and outputs $L'(i) \subseteq [n]$ such that $\mathrm{corr} \subseteq L'(i)$ and $|L'(i)| \leq \delta n$.*

The proof of the lemma is postponed to Section C.1. We are ready to prove Theorem 3.5.

Proof of Theorem 3.5. The following algorithm runs linear time and decodes w to the closest codeword $\mathrm{x} \in C$.

```
Decoding Algorithm
  • Input: $w \in \mathbb{F}_2^n$, $S \subseteq C_d^{\perp}$
  • $L'$ := Simple Find Erasures $(w, i, S)$
  • $x$ := Decode From Erasures $(w, L', S)$
  • Return $x$
```

Lemma C.1 says that $\mathrm{corr} \subseteq L'$ and $|L'| \leq \delta n$. Lemma 4.2 implies that Decode From Erasures returns x. □

C.1 Proof of Lemma C.1

We first prove Claim C.2 and Proposition C.3. Then we prove Lemma C.1.

Claim C.2. *If Algorithm* Find Erasures *outputs $L'(i)$, where $i \leq \log_{(d/\epsilon)} \frac{\delta n}{n^{\alpha}}$, then $|L'(i)| \leq \delta n$.*

Proof. We prove by induction on i that for every $i \geq 1$ we have $|L'(i)| \leq (d/\epsilon)^i|\mathrm{corr}|$. It holds for $i = 1$ since $|N^{unsat}| \leq c|\mathrm{corr}|$ and hence $|L'(1)| \leq (c|\mathrm{corr}|) \cdot d/(\epsilon c) = (d/\epsilon)|\mathrm{corr}|$. Assume the correctness for $i - 1$. Then $|L'(i)| \leq |L'(i-1)| \cdot (d/\epsilon) \leq (d/\epsilon)^i|\mathrm{corr}|$, where last inequality follows from the induction assumption, and the first inequality follows since $|L'(i)| \leq |\mathrm{supp}_\epsilon(N(L'(i-1)))| \leq |L'(i-1)| \cdot (cd/(\epsilon c))$.

We conclude that $|L'(i)| \leq (d/\epsilon)^i \cdot n^\alpha$, and in particular, if $i \leq \log_{(d/\epsilon)} \frac{\delta n}{n^\alpha}$ then $|L'(i)| \leq \delta n$. □

Proposition C.3. *If Algorithm* Simple Find Erasures *outputs $L'(i)$, where $i > \log_{(1-\epsilon)} n^{-\alpha}$, then* $\mathrm{corr} \subseteq L'(i)$.

Proof. Let $i > \log_{(1-\epsilon)} n^{-\alpha} \geq \log_{(1-\epsilon)} |\mathrm{corr}|^{-1}$ then Algorithm returns $L'(i)$ such that $\mathrm{corr} \subseteq L'(i)$. We have $|\mathrm{corr}| \leq \delta n$. The proposition follows by examining the unique neighbor structure of the expander code.

For all $j = 1, \ldots, i$ let $\mathrm{corr}_j = L'(j) \cap \mathrm{corr}$. Note that $\mathrm{corr}_j \subset \mathrm{corr}_{j+1}$.

For $j = 1, \ldots, i$ we prove by induction that sets corr_j satisfy

- if $\mathrm{corr} \setminus \mathrm{corr}_j \neq \emptyset$ then $|\mathrm{corr} \setminus \mathrm{corr}_{j+1}| \leq (1-\epsilon) \cdot |\mathrm{corr} \setminus \mathrm{corr}_j|$.

Therefore $|\mathrm{corr} \setminus L'(i)| = |\mathrm{corr} \subseteq \mathrm{corr}_i| \leq (1-\epsilon)^i |\mathrm{corr}| < 1$, where $i > \log_{(1-\epsilon)} |\mathrm{corr}|^{-1}$. We conclude that $\mathrm{corr} \subseteq L'(i)$. This will complete the proof of the proposition.

For the base case, we argue that $|\mathrm{corr} \cap L'(1)| \geq \epsilon|\mathrm{corr}|$. To see this note that $|N^1(\mathrm{corr})| \geq \epsilon c|\mathrm{corr}|$ because of the expansion properties of C and since $|\mathrm{corr}| \leq \delta n$. But for every index $l \in \mathrm{corr}$ we have at most c constraints $u \in N^1(\mathrm{corr})$ such that $i \in \mathrm{supp}(u)$. Hence for at least ϵ-fraction of indices $i \in \mathrm{corr}$ we have $|\{u \in N^{unsat} \mid i \in \mathrm{supp}(u)\}| \geq \epsilon c$. We also know that for all $u \in N^1(\mathrm{corr})$ we have $\langle u, w\rangle \neq 0$ and hence $u \in N^{unsat}$. So, for at least ϵ-fraction of indices $i \in \mathrm{corr}$ we have $|\{u \in N^{unsat} \mid i \in \mathrm{supp}(u)\}| \geq \epsilon c$. Thus $|\mathrm{corr} \setminus \mathrm{corr}_1| = |\mathrm{corr} \setminus \mathrm{supp}_\epsilon(N^{unsat})| \leq (1-\epsilon)|\mathrm{corr}|$. This completes the base case.

Assume correctness up to $j - 1$ and let us prove it for j, where $j \geq 1$. Assume $\mathrm{corr} \setminus \mathrm{corr}_{j-1} \neq \emptyset$. We know that $|\mathrm{corr} \setminus \mathrm{corr}_{j-1}| \leq |\mathrm{corr}| \leq \delta n$ and hence $|N^1(\mathrm{corr} \setminus \mathrm{corr}_{j-1})| \geq \epsilon c|\mathrm{corr} \setminus \mathrm{corr}_{j-1}|$.

So, for at least ϵ-fraction of indices $i \in \mathrm{corr} \setminus \mathrm{corr}_{j-1}$ we have $|\{u \in N^1(\mathrm{corr} \setminus \mathrm{corr}_{j-1}) \mid i \in \mathrm{supp}(u)\}| \geq \epsilon c$. Moreover, for every $u \in N^1(\mathrm{corr} \setminus \mathrm{corr}_{j-1})$ we have $u \in N(L'(j-1))$.

We conclude that $|\mathrm{corr} \setminus \mathrm{corr}_j| \leq \epsilon|\mathrm{corr} \setminus \mathrm{corr}_{j-1}|$. This completes the induction and the proof of Proposition C.3. □

Now we prove Lemma C.1.

Proof of Lemma C.1. It holds that $\log_{(1-\epsilon)} |\mathrm{corr}|^{-1} < i = \left\lfloor \log_{(d/\epsilon)} \delta n^{1-\alpha} \right\rfloor \leq \log_{(d/\epsilon)} \frac{\delta n}{|\mathrm{corr}|}$.

We have $|\mathrm{corr}| \leq n^\alpha$. Proposition C.3 implies that $\mathrm{corr} \subseteq L'(i)$. Claim C.2 implies that $|\mathrm{corr}| \leq \delta n$. Moreover, it can be readily verified that Algorithm Find Erasures runs time $O(n)$. □

D Tightness of Theorem 3.4

We show that for sufficiently large constants $c, d \geq 2$ there exists a $(c, d, 1/2, \delta)$-expander code (where $\delta > 0$ is a constant) with distance 2.[10] We conclude that the expansion parameter $1/2$ is insufficient for the decoding, even from 2 errors.

[10] This statement might be folklore. We did not verify this.

Proposition D.1. *There exist constants $c, d \geq 2$, constant $0 < \delta < 1$ and infinitely many n such that there exists a $(c+1, d, \frac{1}{2}, 0.9\delta)$-expander code $C' \subseteq \mathbb{F}_2^{n+2}$ but $\Delta(C') = 2$. Hence C' can not be decodable even from 2 errors.*

Proof. Take any $(c, d, \frac{3}{4}, \delta)$-expander code $C \subseteq \mathbb{F}_2^n$, where $c + 1 < d$ (e.g., a random expander code will have these parameters). Let $S_C \subseteq \mathbb{F}_2^n$ be a set of d-weight constraints from the parity check graph (which is a $(c, d, \frac{3}{4}, \delta)$-expander) of C and note that $|S_C| = c \cdot n/d$. Since C is (c, d)-regular it holds that for every $u \in S_C$ we have $|u| = d$ and for every $i \in [n]$ we have $|\{u \in S_C \mid i \in \text{supp}(u)\}| = c$. The fact that C is a $(c, d, \frac{3}{4}, \delta)$-expander code implies that for every $L_0 \subseteq [n]$, $|L_0| \leq \delta n$ we have $|\{u \in S_C \mid \text{supp}(u) \cap L_0 \neq \emptyset\}| \geq \frac{3}{4} \cdot c \cdot |L_0|$. Let $S_C^{(00)} = \{u \odot 00 \mid u \in S_C\}$ be a set of vectors obtained from S_C by appending 00 to every vector in S_C. Note that $S_C^{(00)} \subseteq \mathbb{F}_2^{n+2}$, $S_C^{(00)}|_{[n]} = S_C$ and for every $u \in S_C^{(00)}$ we have $|u| = d$.

In the rest of the proof we will define $C' \subseteq \mathbb{F}_2^{n+2}$ by defining the d-weight constraints set $S_{C'} \subseteq \mathbb{F}_2^{n+2}$, i.e., $C' = (\text{span}(S_{C'}))^{\perp}$. Let $U = \{u_1, \ldots, u_c, u_{c+1}\} \subseteq \mathbb{F}_2^{n+2}$ be a set of constraints such that for every $i \in [c+1]$ we have

$$\text{supp}(u_i) = \{(i-1)(d-2)+1, (i-1)(d-2)+2, \ldots, (i-1)(d-2)+d-2\} \cup \{n+1, n+2\}.$$[11]

In particular, it holds that for every $i \in [c+1]$ we have $|u_i| = d$, and for every $i_1 \neq i_2 \in [c+1]$ we have $\text{supp}(u_{i_1}) \cap \text{supp}(u_{i_2}) = \{n+1, n+2\}$.

Let $t = \frac{n-(d-2)\cdot(c+1)}{d}$ (assume w.l.o.g. that t is an integer) and $U' = \{u'_1, \ldots, u'_t\}$ such that for every $u'_i \in U'$ we have $u'_i \in \mathbb{F}_2^{n+2}$ and

$$\text{supp}(u'_i) = \{(d-2)\cdot(c+1) + (i-1)\cdot d + 1, \ldots, (d-2)\cdot(c+1) + (i-1)\cdot d + d\}.$$

Note that for every $u' \in U'$ we have $|u'| = d$.

We let $S_{C'} = S_C^{(00)} \cup U \cup U'$ and the code $C' \subseteq \mathbb{F}_2^{n+2}$ is defined by its constraint set $S_{C'}$, i.e., $C' = (\text{span}(S_{C'}))^{\perp}$. Note that for all $u \in S_{C'}$ we have $|u| = d$ and for all $i \in [n+2]$ it holds that $|\{u \in S_{C'} \mid i \in \text{supp}(u)\}| = c+1$. We conclude that C' is a $(c+1, d)$-regular code.

For the rest of the proof, given a set $T \subseteq [n+2]$ we define (with some abuse of notation)

$$N_C(T) = \left\{u \in S_C^{(00)} \mid \text{supp}(u) \cap T \neq \emptyset\right\}$$

be the neighborhood of T regarding the constraints set $S_C^{(00)}$ and

$$N_{C'}(T) = \{u \in S_{C'} \mid \text{supp}(u) \cap T \neq \emptyset\}$$

be the neighborhood of T regarding the constraints set $S_{C'}$.

We argue that for all $L_0 \subseteq [n+2]$ such that $|L_0| \leq (0.9\delta) \cdot (n+2) \leq \delta n$ it holds that $|N_{C'}(L_0)| \geq 1/2 \cdot (c+1) \cdot |L_0|$. Let $part_1 = L_0 \cap [n]$ and $part_2 = L_0 \cap \{n+1, n+2\}$. Note that $L_0 = part_1 \cup part_2$ and $part_1 \cap part_2 = \emptyset$. It follows that $|part_1| \leq \delta n$ and hence $|N_C(part1)| \geq 3/4c|part_1|$ by definition of $S_C^{(00)}$. Hence $|N_C(part_1)| \geq 3/4c|part_1| = (c/2 + c/4)|part_1| \geq (c/2 + 1/2)|part_1| = 1/2(c+1)|part_1|$, where the second inequality holds since $c \geq 2$.

Note that $|part_2| \leq 2$. By definition of C' we conclude that if $|part_2| = 0$ then $|N_{C'}(part_2)| = 0$ and if $|part_2| \geq 1$ then $|N_{C'}(part_2)| = c+1$. Thus it holds that $|N_{C'}(part_2)| \geq 1/2(c+1)|part_2|$.

[11] Note that the support of a vector defines the vector since the field is binary.

Note that by construction of C' and $S_{C'}$ it holds that $N_C(part_1) \cap N_{C'}(part_2) = \emptyset$ and $N_C(part_1) \subseteq N_{C'}(part_1)$. We conclude that

$$|N_{C'}(L_0)| = |N_{C'}(part_1 \cup part_2)| \geq |N_C(part_1)| + |N_{C'}(part_2)| \geq$$

$$1/2 \cdot (c+1) \cdot |part_1| + 1/2 \cdot (c+1) \cdot |part_2| = 1/2 \cdot (c+1) \cdot |L_0|.$$

Thus we showed that C' is a $(c+1, d, 1/2, 0.9\delta)$-expander code. However, it holds that $0^n 11 \perp S_{C'}$ by construction of $S_{C'}$. Hence $0^n 11 \in C'$ and $\Delta(C') = 2$ (by construction $\Delta(C') > 1$). □

E Expander Codes — Auxiliary Statements

The following folklore proposition says that high expansion implies unique neighbor expansion.

Proposition E.1 (Folklore). *If $C \subseteq \mathbb{F}_2^n$ is a (c, d, ϵ, δ)-expander code then C is a $(c, d, 2\epsilon - 1, \delta)$-unique expander code.*

Proof. Let $L_0 \subset [n]$ such that $|L_0| \leq \delta n$. We show that $|N^1(L_0)| \geq (2\epsilon - 1) \cdot c|L_0|$.

We know that $\epsilon c|L_0| \leq |N(L_0)| = |N^1(L_0)| + |N^{\geq 2}(L_0)|$ and hence $|N^{\geq 2}(L_0)| \geq \epsilon c|L_0| - |N^1(L_0)|$. But $|L_0| \cdot c \geq |N^1(L_0)| + 2|N^{\geq 2}(L_0)|$ which implies that $|L_0| \cdot c \geq |N^1(L_0)| + 2(\epsilon c|L_0| - |N^1(L_0)|) = -|N^1(L_0)| + (2\epsilon)c|L_0|)$. We conclude that $|L_0| \cdot c \geq 2\epsilon c|L_0| - |N^1(L_0)|$ and $|N^1(L_0)| \geq (2\epsilon - 1) \cdot c|L_0|$. □

The following simple claim bounds an expansion of subsets larger that δn.

Claim E.2 (Expansion beyond δn). *Let $C \subseteq \mathbb{F}_2^n$ be a (c, d, ϵ, δ)-expander code, where $\epsilon > 1/2$ and let $S \subseteq [n]$. Then,*

- *if $|S| \leq \delta n$ then $|N^1(S)| \geq (2\epsilon - 1)c|S|$,*
- *if $\delta n < |S| \leq 2\epsilon\delta n$ then $|N^1(S)| \geq (2\epsilon\delta) \cdot c \cdot n - |S| \cdot c$.*

Proof. The first item follows from Proposition E.1. Hence we assume that $\delta n < |S| \leq 2\epsilon\delta n$ and prove the second item.

Let $S_0 \subseteq S$ such that $|S_0| = \delta n$. By Proposition E.1 we have $|N^1(S_0)| \geq (2\epsilon - 1) \cdot c \cdot |S_0|$. For every $u \in N^1(S_0)$ it holds that either $u \in N(S \setminus S_0)$ or $u \in N^1(S)$. Hence we have

$$|N^1(S)| \geq |N^1(S_0)| - |N(S \setminus S_0)| \geq (2\epsilon - 1) \cdot c \cdot |S_0| - c \cdot (|S| - |S_0|) = (2\epsilon\delta) \cdot c \cdot n - |S| \cdot c.$$

□

We show that Claim E.2 implies the bound on the distance of the expander code.

Corollary E.3. *Let $C \subseteq \mathbb{F}_2^n$ be a (c, d, ϵ, δ)-expander code, where $\epsilon > 1/2$. Then $\delta(C) \geq 2\epsilon\delta$.*

Proof. Assume the contrary. Then there exist $c_1 \neq c_2 \in C$ such that $\delta(c_1, c_2) < 2\epsilon\delta$. Let $c = c_1 - c_2$. Note that $c \in C$ and $|\operatorname{supp}(c)| < 2\epsilon\delta n$. Claim E.2 implies the existence of $u \in C_d^{\perp}$ such that $|\operatorname{supp}(u) \cap \operatorname{supp}(c)| = 1$. Hence $\langle u, c \rangle \neq 0$ and thus $c \notin C$. Contradiction. □