Bounds on 2-Query Codeword Testing

Eli Ben-Sasson[1], Oded Goldreich[2], and Madhu Sudan[3]

[1] Division of Engineering and Applied Sciences, Harvard University and Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA.
eli@eecs.harvard.edu

[2] Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel.
oded@wisdom.weizmann.ac.il

[3] Laboratory for Computer Science, Massachusetts Institute of Technology, 200 Technology Square, Cambridge, MA 02139.
madhu@mit.edu

Abstract. We present upper bounds on the size of codes that are locally testable by querying only two input symbols. For linear codes, we show that any 2-locally testable code with minimal distance δn over any finite field $\mathbb{F}$ cannot have more than $|\mathbb{F}|^{3/\delta}$ codewords. This result holds even for testers with two-sided error. For general (non-linear) codes we obtain the exact same bounds on the code size as a function of the minimal distance, but our bounds apply only for binary alphabets and one-sided error testers (i.e. with perfect completeness). Our bounds are obtained by examining the graph induced by the set of possible pairs of queries made by a codeword tester on a given code. We also demonstrate the tightness of our upper bounds and the essential role of certain parameters.

1 Introduction

Locally testable codes are error-correcting codes that admit very efficient codeword testers. Specifically, using a constant number of (random) queries, non-codewords are rejected with probability proportional to their distance from the code.

Locally testable codes arise naturally from the study of probabilistically checkable proofs, and were explicitly defined in [5] and systematically studied in [7]. The task of testing a code locally may also be viewed as a special case of the general task of property testing initiated by [9, 6], where the property being tested here is that of being a codeword. In this paper we explore codes that can be tested with constant number of queries.

We focus on codes $\mathcal{C} \subset \Sigma^n$ that have large distance (i.e., each pair of codewords differ in at least $\Omega(n)$ coordinates) and large size (i.e., at the very least, $|\mathcal{C}|$ should grow with n and $|\Sigma|$). Such codes are known to exist. Specifically, in [7] locally testable codes are shown such that $|\mathcal{C}| = |\Sigma|^k$ for $k = n^{1-o(1)}$. We highlight two of these results:

1. For $\Sigma = \{0, 1\}$, *three* queries are shown to suffice. Furthermore, these codes are *linear*.

2. For $|\Sigma| > 2$, *two* queries are shown to suffice.[1]

This raises the question of whether binary codes and/or linear codes can have codeword tests that make only *two queries.* In this paper, we show that the answer is essentially negative; that is, *for codes of linear distance, such codes can contain only a constant number of codewords.* More general statements are provided by Theorems 3.1 and 4.1, which address linear codes over arbitrary fields and non-linear binary codes, respectively. We also address the tightness of our upper-bounds and the essential role of certain parameters (i.e., our upper-bounds apply either to linear codes or to binary codes that have a tester of perfect completeness).

Organization: In Section 2 we present the main definitions used in this paper, and state our main results. In Section 3 we study *linear* codes that admit two-query codeword testers. In Section 4 we study general *binary* codes that admit two-query codeword testers of perfect completeness. Due to space considerations, the rests of our results appear only in our technical report [3]: In [3, Sec. 5] we show that our upper-bounds cease to hold for *ternary non-linear* codes (rather than for non-linear codes over much larger alphabets as considered in [7] and mentioned in Item 2 above). In [3, Sec. 5] we show that perfect completeness is essential for the results regarding *non-linear binary* codes (presented in Section 4).

2 Formal Setting

We consider words over an alphabet Σ. For $w \in \Sigma^n$ and $i \in [n]$, we denote by w_i the i-th symbol of w; that is, $w = w_1 \cdots w_n$.

2.1 Codes

We consider codes $\mathcal{C} \subseteq \Sigma^n$ over a finite size alphabet Σ. The blocklength of $\mathcal{C}$ is n, and the size of $\mathcal{C}$ is its cardinality $|\mathcal{C}|$. We use normalized Hamming distance as our distance measure; that is, for $u, v \in \Sigma^n$ the distance $\Delta(u, v)$ is defined as the number of locations on which u and v differ, divided by n (i.e., $\Delta(u, v) = |\{i : u_i \neq v_i\}|/n$). The relative minimal distance of a code, denoted $\delta(\mathcal{C})$, is the minimal normalized Hamming distance between two distinct codewords. Formally

$$\delta(\mathcal{C}) = \min_{u \neq v \in \mathcal{C}} \{\Delta(u, v)\}$$

The distance of a word w from the code, denoted $\Delta(w, \mathcal{C})$, is $\min_{v \in \mathcal{C}}\{\Delta(w, v)\}$.

A code is called redundant if its projection on some coordinate is constant (i.e., there exists $i \in \{1, \ldots, n\}$ such that for any two codewords w, w' it holds that

[1] We comment that these codes are "linear" in a certain sense. Specifically, Σ is a vector space over a field F, and the code is a linear subspace over F (rather than over Σ). That is, if $\Sigma = F^{\ell}$ then $\mathcal{C} \subset \Sigma^n$ is a linear subspace of $F^{n \cdot \ell}$ (but not of Σ^n, no matter what finite field we associate with Σ). In the coding literature such codes are called F-linear.

$w_i = w'_i$). A redundant code can be projected on all non-redundant coordinates, yielding a code with the same size and distance, but smaller blocklength. Thus, w.l.o.g., *we assume all codes to be non-redundant.*

Typically (in this paper) Σ is a finite field $\mathbb{F}$ and we view $\mathbb{F}^n$ as a vector space over $\mathbb{F}$. In particular, for $u, v \in \mathbb{F}^n$ the inner product of the two is $\langle v, u\rangle = \sum_{i=1}^{n} v_i \cdot u_i$ (all arithmetic operations are in $\mathbb{F}$). The weight of $v \in \mathbb{F}^n$, denoted $\mathrm{wt}(v)$, is the number of non-zero elements in v. In this case $\Delta(u, v) = \mathrm{wt}(u - v)/n$.

2.2 Testers and tests

By a codeword tester (or simply tester) with query complexity q, completeness c and soundness s (for the code $\mathcal{C} \subseteq \Sigma^n$) we mean a *randomized* oracle machine that given oracle access to $w \in \Sigma^n$ (viewed as a function $w : \{1, \ldots, n\} \to \Sigma$) satisfies the following three conditions:

- Query Complexity q: The tester makes at most q queries to w.
- Completeness: For any $w \in \mathcal{C}$, given oracle access to w the tester accepts with probability at least c.
- Soundness: For any w that is at relative distance at least $\delta(\mathcal{C})/3$ from $\mathcal{C}$, given oracle access to w, the tester accepts with probability at most s.[2]

If $\mathcal{C}$ has a codeword tester with query complexity q, completeness c and soundness s we say $\mathcal{C}$ is $[q, c, s]$-locally testable.

A deterministic test (or simply test) with query complexity q is a *deterministic* oracle machine that given oracle access to $w \in \Sigma^n$ makes at most q queries to w, and outputs 1 (= accept) or 0 (= reject). Any (randomized) tester can be described as a distribution over deterministic tests, and we adopt this view throughout the text.

A (deterministic) test is called adaptive if its queries depend on previous answers of the oracle, and otherwise it is called non-adaptive. A test has perfect completeness if it accepts all codewords. Both notions extend to (randomized) testers. Alternatively, we say that a tester is non-adaptive (resp., has perfect completeness) if all the deterministic tests that it uses are non-adaptive (resp., have perfect completeness resp.), and otherwise it is adaptive (resp., has non-perfect completeness).

2.3 Our results

We study 2-query codeword testers. Our main results are upper-bounds on the sizes of linear (resp., binary) codes admitting such testers (resp., testers of perfect completeness):

[2] We have set the *detection radius* of the tester at third its distance (i.e., for any w whose distance from $\mathcal{C}$ is at least $\frac{1}{3} \cdot \delta(\mathcal{C})$ the test rejects with probability at least s). As will be evident from the proofs, our results hold for any radius less than half the distance.

Theorem 2.1 *For any constants $c > s$, any $[2, c, s]$-locally testable* linear *code over Σ has at most $|\Sigma|^{3/\delta}$ codewords, where δ is its relative distance.*

Theorem 2.2 *For any constant $s < 1$, any $[2, 1, s]$-locally testable* binary *code has at most $2^{3/\delta}$ codewords, where δ is its relative distance.*

In contrast to the above, we state the following facts:

1. The upper-bounds stated in Theorems 2.1 and 2.2 are reasonablly tight: For some constants $s < 1$ and $\delta > 0$, and every finite field $\mathbb{F}$, there exists a linear $[2, 1, s]$-locally testable code of size $|\mathbb{F}|^{1/\delta}$ and minimal relative distance δ over $\mathbb{F}$ (see, Proposition 3.6).
2. Non-linearity of the code is essential to Theorem 2.1 and binary alphabet is essesntial to Theorem 2.2: there exists good *non-linear codes over ternary alphabets* that have 2-query codeword testers (of perfect completeness). That is, for some constants $s < 1$ and $\delta > 0$, there exists a $[2, 1, s]$-locally testable *ternary* code of relative distance δ that has size that grows almost linearly with the blocklength (see [3, Thm. 5.6]).
3. Perfect completeness is essesntial to Theorem 2.2: there exists good *non-linear codes over binary alphabets* that have 2-query codeword testers of *non-perfect completeness*. That is, for some constants $c > s > 0$ and $\delta > 0$, there exists a $[2, c, s]$-locally testable *binary* code of relative distance δ that has size that grows almost linearly with the blocklength (see [3, Thm. 6.1]).
4. Regarding the difference between linearity and "semi-linearity" (as in Footnote 1), we note that there exists good *$GF(2)$-linear codes over $\{0, 1\}^2$* that have 2-query codeword testers (of perfect completeness): (see [3, Thm. 5.7]).

We mention that some of our results are analogous to results regarding probabilistic checkable proof (PCP) systems. In particular, let $\mathcal{PCP}^{\Sigma}_{c,s}[\log, q]$ denote the class of languages having PCP systems with logarithmic randomness, making q queries to oracles over the alphabet Σ, and having completeness and soundness bounds c and s respectively. Then, it is known that $\mathcal{PCP}^{\{0,1\}}_{1,s}[\log, 2] = \mathcal{P}$ for every $s < 1$, whereas $\mathcal{PCP}^{\{0,1\}}_{c,s}[\log, 2] = \mathcal{NP}$ for some $c > s > 0$ and $\mathcal{PCP}^{\{0,1,2\}}_{1,s}[\log, 2] = \mathcal{NP}$ for some $s < 1$.[3] Folllowing [7], we warn that the translation between PCPs and locally-checkable codes is not obvious. In particular, we do not know whether it is possible to obtain our coding results from the known PCP results or the other way around.

3 Linear Codes

In this section we show that $[2, c, s]$-locally testable *linear* codes with constant minimal relative distance must have very small size. Throughout this section $\mathbb{F}$ is a finite field of size $|\mathbb{F}|$. A code $\mathcal{C} \subseteq \mathbb{F}^n$ is called linear if it is a linear subspace of $\mathbb{F}^n$. The main result of this section is the following.

[3] The first two results are proven in [2], whereas the third result is lolklore that is based on the NP-Hardness of approximating Max3SAT as established in [1].

Theorem 3.1 (Theorem 2.1, restated): *Let $\mathcal{C} \subset \mathbb{F}^n$ be a $[2, c, s]$-locally testable* linear *code with minimal relative distance δ. If $c > s$ then*

$$|\mathcal{C}| \leq |\mathbb{F}|^{3/\delta}$$

We start by pointing out that, when considering testers for linear codes, the tester can be assumed to be non-adaptive and with perfect completeness. This holds by the following result of [4].

Theorem 3.2 [4]: *If a linear code (over any finite field) is $[q, c, s]$-locally testable using an adaptive tester, then it is $[q, 1, 1 - (c - s)]$-locally testable using a non-adaptive tester.*

Notice that if we start off with a tester having completeness greater than soundness ($c > s$), then the resulting non-adaptive, perfect-completeness tester (guaranteed by Theorem 3.2) will have soundness strictly less than 1. Thus, in order to prove Theorem 3.1 it suffices to show the following.

Theorem 3.3 *Let $\mathcal{C} \subseteq \mathbb{F}^n$ be a $[2, 1, s]$-locally* (non-adaptively) *testable linear code, with $s < 1$, and let the minimal relative distance be δ. Then:*

$$|\mathcal{C}| \leq |\mathbb{F}|^{3/\delta}$$

In the rest of the section we prove Theorem 3.3. The proof idea is as follows. Each possible test of query complexity 2 and perfect completeness imposes a constraint on the code, because all codewords must pass the test. Thus, we view the n codeword coordinates as *variables* and the set of tests as inducing constraints on these variables (i.e., codewords correspond to assignments (to the variables) that satisfy all these constraints). Since the code is linear, each test imposes a *linear* constraint on the pair of variables queried by it. (A linear constraint on the variables x, y has the form $ax + by = 0$ for some fixed $a, b \in \mathbb{F}$). We will show that in a code of large distance, these constraints induce very few satisfying assignments. Specifically, we look at the graph in which the vertices are the (n) codeword-coordinates (or variables) and edges connect two vertices that share a test. The main observation is that in any codeword, the values of all variables in a connected component are determined by the value of any one variable in the component; that is, the assignment to a single variable determines the assignment to the whole component. By perfect completeness, any word that satisfies all constraints in all connected components will pass *all* tests. Hence there cannot be many variables in small connected components, for then we could find a word that is far from the code and yet is accepted with probability 1. But this means that the code is essentially determined by the (small number of) large connected components, and hence the size of the code is small. We now give the details, starting with a brief discussion of *dual codes* which is followed by the proof.

3.1 Linearity Tests and Dual Codes

Recall that $\mathcal{C} \subseteq \mathbb{F}^n$ is linear iff for all $u, v \in \mathcal{C}$ we have $u + v \in \mathcal{C}$. In this case $\delta(\mathcal{C}) = \min_{w \in \mathcal{C}}\{\mathrm{wt}(w)/n\}$. As pointed out in [8], codeword tests for linear codes are intimately related to the "dual" of the code. For a linear code $\mathcal{C}$, the *dual code* $\mathcal{C}^\perp$ is defined as the subspace of $\mathbb{F}^n$ orthogonal to $\mathcal{C}$, i.e.

$$\mathcal{C}^\perp = \{v : v \perp \mathcal{C}\}$$

where $v \perp \mathcal{C}$ iff for all $u \in \mathcal{C}$, $v \perp u$ (recall $v \perp u$ iff $\langle v, u\rangle = 0$).

The support of a vector v, denoted $\mathbf{Supp}(v)$, is the set of indices of non-zero entries. Similarly, the support of a test T is the set of indices it queries. Notice that a non-adaptive test with query complexity q has support size q. For $v, u \in \mathbb{F}^n$ we say that v covers u if $\mathbf{Supp}(v) \supseteq \mathbf{Supp}(u)$. A test is called trivial if it always accepts. Elementary linear algebra gives the following claim.

Proposition 3.4 *The support of any* non-trivial perfect-completeness test *for $\mathcal{C}$ covers an element of $\mathcal{C}^\perp \setminus \{0^n\}$.*

Proof: Let T be a test and $\mathcal{C}_T$ be the projection of (the linear space) $\mathcal{C}$ onto $\mathbf{Supp}(T)$. The projection is a linear operator, so $\mathcal{C}_T$ is a linear space over $\mathbb{F}$. The linear space $\mathcal{C}_T$ must be a strict subspace of $\mathbb{F}^{\mathbf{Supp}(T)}$, because $|\mathcal{C}_T| = |\mathbb{F}^{\mathbf{Supp}(T)}|$ (i.e. $\mathcal{C}_T$ includes all vectors in $\mathbb{F}^{\mathbf{Supp}(T)}$) implies that either T reject some valid codeword in $\mathcal{C}$ (in violation of perfect completeness) or T always accepts (in violation of non-triviality). It follows that $(\mathcal{C}_T)^\perp$ has a non-zero element, denoted w. However, $\mathbf{Supp}(w) \subseteq \mathbf{Supp}(T)$ and $w \in \mathcal{C}^\perp$, completing the proof. □

Clearly one can assume that all tests used by a tester are non-trivial. We also assume $\mathcal{C}^\perp$ has no element of weight 1, because otherwise $\mathcal{C}$ is redundant. Since we consider only testers that make two queries, it follows that all tests they use have support size exactly two. Furthermore, without loss of generality, all the tests are linear.[4]

3.2 Upper Bounds on Code Size

By the above discussion (i.e., end of Section 3.1), we may assume (w.l.o.g.) that the $[2, 1, s]$-tester for $\mathcal{C}$ is described by a distribution over

$$\mathcal{C}_2^\perp \stackrel{\mathrm{def}}{=} \{v \in \mathcal{C}^\perp : \mathrm{wt}(v) = 2\}$$

The test corresponding to $v \in \mathcal{C}_2^\perp$ refers to the orthogonality of v and the oracle w; that is, the test accepts w if $v \perp w$ and rejects otherwise.[5] We now look at $\mathcal{C}_2^\perp$ and bound the size of $(\mathcal{C}_2^\perp)^\perp$. Our theorem will follow because $\mathcal{C} \subseteq (\mathcal{C}_2^\perp)^\perp$.

[4] In genenral, without loss of generality, a one-sided tester for a property P accepts y if and only if its view of y is consistent with its view of some $x \in P$. In our case P is a linear space, so consistecy means satisfying a linear system. For further details see Appendix.

[5] Notice that since $\mathrm{wt}(v) = 2$ such a test amounts to two queries into w.

The set $\mathcal{C}_2^\perp$ gives rise to a natural graph, denoted $G_\mathcal{C}$. The vertex set of $G_\mathcal{C}$ is $V(G_\mathcal{C}) = \{1, \ldots, n\}$ and $(i,j) \in E(G_\mathcal{C})$ iff there exists $v_{ij} \in \mathcal{C}_2^\perp$ with $\mathbf{Supp}(v_{ij}) = \{i,j\}$.

The key observation is that, for any edge $(i,j) \in E(G_\mathcal{C})$ there is some $c_{ij} \in \mathbb{F} \setminus \{0\}$ such that for any $w \in \mathcal{C}$ it holds that $w_i = c_{ij} \cdot w_j$. To see this, notice the constraint corresponding to (i,j) can be written as $a_{ij}w_i + b_{ij}w_j = 0$, where $a_{ij}, b_{ij} \in \mathbb{F} \setminus \{0\}$ (if either a_{ij} or b_{ij} are 0 then v_{ij} has support size one, meaning $\mathcal{C}$ is redundant). So, by transitivity, the value of w on all variables in the connected component of i, is determined by w_i. (Moreover, all these values are non-zero iff $w_i \neq 0$.) Assuming that the number of connected components is k, this implies that there can be at most $|\mathbb{F}|^k$ different codewords (because there are only k degrees of freedom corresponding to the settings (of all variables) in each of the k components). To derive the desired bound we partition the components into big and small ones, and bound the number of codewords as a function of the number of big components (while showing that the small components do not matter).

Let $C_1, \ldots, C_k$ be the connected components of $G_\mathcal{C}$. We call a component small if its cardinality is less than $\delta n/3$. Without loss of generality, let $C_1, \ldots C_s$ be all the small components, and let $S = \bigcup_{i=1}^s C_i$ denote their union.

Claim 3.5 $|S| \leq 2\delta n/3$.

Proof: Otherwise there exists $I \subset \{1, \ldots, s\}$ such that

$$\delta n/3 \leq \sum_{i \in I} |C_i| < 2\delta n/3$$

For every $i \in I$, we consider a vector $w^i \in (\mathcal{C}_2^\perp)^\perp$ with $\mathbf{Supp}(w^i) = C_i$. To see that such a vector exists, set an arbitrary coordinate of C_i to 1 (which is possible because the code is not redundent) and force non-zero values to all other coordinates in C_i (by virtue of the above discussion). Furthermore, note that this leaves all coordinates out of C_i unset, and that the resulting w^i satisfy all tests in $\mathcal{C}_2^\perp$ (where the tests that correspond to the edges in C_i are satified by our setting of the non-zero values, whereas all other tests refer to vertices out of C_i and are satisfied by zero values). Now, define $w = \sum_{i \in I} w^i$. By definition, we have $\mathbf{Supp}(w) = \cup_{i \in I} C^i$, and $\delta n/3 \leq \mathrm{wt}(w) < 2\delta n/3$ follows by the hypothesis. Hence, $\Delta(w, \mathcal{C}) \geq \delta/3$.

On the other hand, w is orthogonal to $\mathcal{C}_2^\perp$. To see this, consider any $v \in \mathcal{C}^\perp$. If $\mathbf{Supp}(v) \subseteq C_i$, for some $i \in I$, then the "view v has of w" (i.e. the values of the coordinates v queries) is identical to the view v has of the codeword w^i, and so $\langle v, w \rangle = \langle v, w^i \rangle = 0$. Otherwise (i.e., $\mathbf{Supp}(v)$ has empty intersection with S), by definition v "sees" only zeros, and so $\langle v, w \rangle = 0$.

We conclude w is $\frac{\delta}{3}$-far from $\mathcal{C}$, yet it passes all possible tests of query complexity two. This contradicts the soundness condition, and the claim follows. □

Proof (of Theorem 3.3): *Assume for the sake of contradiction* that

$$|\mathcal{C}| > |\mathbb{F}|^{3/\delta}$$

Recall that (by the "key observation") the values of all variables in a connected component are determined by the value of a single variable in this component. Since there are at most $3/\delta$ large connected components in $G_{\mathcal{C}}$ (because each has cardinality at least $\delta n/3$), the contradiction hypothesis implies that there exist two codewords $x \neq y$ that agree on all variables that reside in the large connected components. Indeed, these two codewords $x \neq y$, may differ on variables that reside in the small connected components (i.e., variables in S), but Claim 3.5 says that there are few such variables (i.e.. $|S| \leq 2\delta n/3$). By linearity $x - y \in \mathcal{C}$ (but $x - y \neq 0^n$), and so $0 < \mathrm{wt}(x - y) \leq |S| < \delta n$. We have reached a contradiction (because $\mathcal{C}$ has distance δ), and Theorem 3.3 follows. ▌

3.3 Tightness of the Upper Bound

We remark that our upper bound is quite tight. For any $\delta < 1$, consider the following code $\mathcal{C}_n \subset \mathbb{F}^n$ formed by taking $1/\delta$ elements of $\mathbb{F}$ and repeating each one of them δn times. Thus, a codeword in $\mathcal{C}_n$ is formed of $1/\delta$ blocks, each block of the form $e^{\delta n}$ for some $e \in \mathbb{F}$ (here e^k means k repetitions of e).

Proposition 3.6 *$\mathcal{C}_n$ is a linear $[2, 1, 1 - \frac{2\delta}{3|\mathbb{F}|}]$-locally testable code with minimal relative distance δ and size $|\mathbb{F}|^{1/\delta}$.*

For instance, taking $\mathbb{F} = GF(2)$, the soundness parameter in the proposition is $1 - \delta/3$.

Proof: The linearity, distance and size of $\mathcal{C}_n$ are self-evident. Consider the following natural tester for $\mathcal{C}_n$: Select a random block, read two random elements in it, and accept iff the two are equal. This tester has perfect completeness and query complexity 2. As to the soundness, let $k = 1/\delta$ and write $v \in \mathbb{F}^n$ as $(v^{(1)}, \ldots, v^{(k)})$, where $v^{(i)}$ is the i-th block of v (i.e., $|v^{(i)}| = \delta n$). The Hamming distance of v from $\mathcal{C}_n$ is the sum of the Hamming distances of the individual blocks $v^{(i)}$ from the code $B = \{e^{\delta n} : e \in \mathbb{F}\}$.

Suppose v has relative distance at least $\delta/3$ from $\mathcal{C}_n$. Let δ_i denote the relative distance of $v^{(i)}$ from B. Then, $\frac{1}{k}\sum_{i=1}^{k} \delta_i \geq \delta/3$ (and $\delta_i \leq 1 - \frac{1}{|\mathbb{F}|}$). The acceptance probability of the tester equals

$$\begin{aligned} \frac{1}{k}\sum_{i=1}^{k}\left(\delta_i^2 + (1-\delta_i)^2\right) &= 1 - \frac{2}{k}\sum_{i=1}^{k}(1-\delta_i)\cdot\delta_i \\ &\leq 1 - \frac{2}{k|\mathbb{F}|}\sum_{i=1}^{k}\delta_i \\ &\leq 1 - \frac{2\delta}{3|\mathbb{F}|} \end{aligned}$$

where the first inequality is due to $\delta_i \leq 1 - \frac{1}{|\mathbb{F}|}$. Thus, the soundness parameter is as claimed. □

4 Non-Linear Codes

In this section we provide upper bounds on the code size of arbitrary (i.e., possibly non-linear) 2-locally testable codes. Our bounds apply only to binary codes and testers with perfect completeness, and with good reason: There exist good 2-testable binary codes with non-perfect completeness and there exist good 2-testable codes with perfect completeness over ternary alphabets (see Section 5 in our technical report [3]). Our main result is:

Theorem 4.1 (Theorem 2.2, restated): *If $\mathcal{C} \subseteq \{0,1\}^n$ is a $[2,1,s]$-locally testable code with minimal relative distance δ and $s < 1$, then*

$$|\mathcal{C}| \leq 2^{3/\delta}$$

The proof (presented below) generalizes that of the *binary* linear case (binary means $\mathbb{F} = GF(2)$), with some necessary modifications, which we briefly outline now. In the binary linear case a test querying x_i and x_j forces $x_i = x_j$ for all codewords (this is the only possible linear constraint of size two over $GF(2)$). In that case, the set of all tests corresponds to an undirected graph in which each connected component forces all variables to have the same value. In the non-linear case a test (adaptive or non-adaptive) corresponds to a 2-CNF. (Recall that in both cases we deal with perfect completeness testers.) The set of all tests (which is itself a 2-CNF) corresponds to a *directed* graph of constraints on codewords, where the constraint $x_i \vee x_j$ translates to the pair of directed edges $\bar{x}_i \to x_j$ and $\bar{x}_j \to x_i$. In the resulting *directed* graph, a *strongly* connected component takes the role played by the connected component in the linear case. Namely, for any codeword, all variables in a strongly connected component are fixed by the value of a single variable in the component. As in the linear case, we use the properties of the code and its tester (i.e., the code's large distance and the fact that the tester rejects any word that is far from the code with non-zero probability) to show that the weight of the *small* strongly connected components is small. Hence, the code is determined by a small number of *large* connected components.

Proof of Theorem 4.1

Again, we view the n codeword coordinates as *variables* and the set of tests (which are 2-CNFs) as inducing constraints on these variables. We stress that each test (even an adaptive one) can be represented by a 2-CNF.[6] Let $\mathcal{F}$ be the conjunction of all non-trivial deterministic tests that are used by a 2-query tester that has perfect completeness with respect to $\mathcal{C}$. We look at the satisfying assignments of $\mathcal{F}$, and use this to bound the size of $\mathcal{C}$. If $\mathcal{F}$ includes a clause of

[6] In general, an adaptive test querying k variables is a decision tree of depth k. It is easy to verify that (the function computed by) such a tree can be represented both as a k-CNF and as a k-DNF.

size 1 then $\mathcal{C}$ is redundant. Thus, assuming non-redundancy of $\mathcal{C}$ implies that $\mathcal{F}$ can be represented by a 2-CNF in which each clause has *exactly* two literals.

We examine the following directed graph $G_{\mathcal{F}}$. The vertex set of $G_{\mathcal{F}}$ is the set of literals $\{x_1, \bar{x}_1 \ldots, x_n, \bar{x}_n\}$. For each clause $(\ell \vee \ell') \in \mathcal{F}$ we introduce in $G_{\mathcal{F}}$ one directed edge from $\bar{\ell}$ to ℓ', and one from $\bar{\ell}'$ to ℓ. We use the notation $\ell \rightsquigarrow \ell'$ to indicate the existence of a directed path from ℓ to ℓ' in $G_{\mathcal{F}}$. We use the notation $w(\ell)$ to denote the value of literal ℓ under assignment w to the underlying variables. Identifying $True$ with 1 and $False$ with 0, we have

Claim 4.2 (folklore): *The following two conditions are equivalent*

1. *The assignment w satisfies $\mathcal{F}$.*
2. *For every directed edge $\ell \rightsquigarrow \ell'$ it holds that $w(\ell) \leq w(\ell')$.*

A strongly connected component in a directed graph G is a maximal set of vertices $C \subseteq V(G)$ such that for any $v, v' \in C$ it holds that $v \rightsquigarrow v'$. For two strongly connected components C and C' in G, we say $C \rightsquigarrow C'$ iff there exist $v \in C$ and $v' \in C'$ such that $v \rightsquigarrow v'$. (Indeed, this happens iff for all $v \in C, v' \in C'$ it holds that $v \rightsquigarrow v'$.)

By Claim 4.2, w satisfies all constraints corresponding to edges of a strongly connected component C iff $w(\ell) = w(\ell')$ for all $\ell, \ell' \in C$. So, any satisfying assignment w either sets to 1 *all* literals in C, or sets them all to 0. In the first case we say that $w(C) = 1$ and in the latter we say $w(C) = 0$.

Let L be the set of literals belonging to large strongly-connected components, where a component is called large iff its cardinality is at least $\delta n/3$. Consider an arbitrary assignment ρ' to the variables of L that can be extended to a satisfying assignment (to $\mathcal{F}$). In particular, ρ' does not falsify any clause of $\mathcal{F}$ (i.e., no clause of $\mathcal{F}$ is set to 0 by ρ'). A literal $\ell \notin L$ is said to be forced by ρ' if there exists $\ell' \in L$ such that $\ell' \rightsquigarrow \ell$ and $\rho'(\ell') = 1$. This is because any satisfying assignment to $\mathcal{F}$ that extends ρ' must set ℓ to 1 (since for such an assignment ρ it must holds that $\rho(\ell) \geq \rho(\ell') = 1$. Indeed, the complementary literal (i.e., $\bar{\ell}$) is forced to 0. Let ρ be the *closure* of ρ' obtained by (iteratively) fixing all forced literals to the value 1 (and their complementary literals to 0). By definition, ρ does not falsify $\mathcal{F}$. Let S_ρ be the set of unfixed variables under ρ.

Claim 4.3 *For any closure ρ of an assignment that satisfies L, it holds that $|S_\rho| \leq 2\delta n/3$.*

Proof: Otherwise, let $C_1, \ldots, C_k$ be a topological ordering of the unfixed strongly connected components comprising S_ρ, where the ordering is according to $\rightsquigarrow$ (as defined above). (Indeed, the digraph defined on the C_i's by $\rightsquigarrow$ is acyclic.) For $j = 0, \ldots, k$, let $v^{(j)}$ be the assignment extending ρ defined by:

$$v^{(j)}(C_i) = \begin{cases} 0 & i \leq j \\ 1 & i > j \end{cases}$$

By Claim 4.2, each assignment $v^{(j)}$ satisfies $\mathcal{F}$. Since $\mathcal{C}$ is 2-locally testable with soundness $s < 1$, each word that is at distance at least $\delta/3$ from $\mathcal{C}$ must falsify some clause in $\mathcal{F}$. But since $v^{(j)}$ satisfies $\mathcal{F}$, it must be that $v^{(j)}$ is within distance $\delta/3$ from some codeword, denoted $w^{(j)}$. By the constradiction hypothesis, we have $\Delta(v^{(0)}, v^{(k)}) = |S_\rho|/n > 2\delta/3$, which implies $w^{(0)} \neq w^{(k)}$ (because $\Delta(v^{(0)}, v^{(k)}) \leq \Delta(v^{(0)}, w^{(0)}) + \Delta(w^{(0)}, w^{(k)}) + \Delta(w^{(k)}, v^{(k)})$, which is upper-bounded by $2 \cdot (\delta/3) + \Delta(w^{(0)}, w^{(k)})$). It follows that

$$\Delta(v^{(k)}, w^{(0)}) \geq \Delta(w^{(k)}, w^{(0)}) - \Delta(w^{(k)}, v^{(k)}) \geq \delta - (\delta/3) = 2\delta/3$$

On the other hand, recall that $\Delta(v^{(0)}, w^{(0)}) \leq \delta/3$. Since, for each j, it holds that $\Delta(v^{(j)}, v^{(j+1)}) < \delta/3$ (because $|C_j| < \delta n/3$), there must be $j \in \{0, 1, \ldots, k\}$ such that $\delta/3 \leq \Delta(v^{(j)}, w^{(0)}) \leq 2\delta/3$. For this j, it holds that $\Delta(v^{(j)}, \mathcal{C}) \geq \delta/3$. But $v^{(j)}$ satisfies $\mathcal{F}$ and so it is accepted by the tester with probability 1, in contradiction to the soundness condition. □

Our proof is nearly complete. As in the proof of Theorem 3.3, assume for the sake of contradiction that

$$|\mathcal{C}| > 2^{\delta/3}$$

In this case, there must be two distinct codewords $w \neq u$ that agree on all large connected components. Let ρ' be the restriction of w to the variables of the large connected components. That is, ρ' agrees with w and with u on the assignment to all variables in L and is unfixed otherwise. Let ρ be the closure of ρ' (obtained by forcing as above). Note that w and u are satisfying assignments to $\mathcal{F}$ that agree on ρ', so they also must agree on ρ (which is forced by ρ'). Thus, by Claim 4.3

$$0 < \Delta(u, w) \leq |S_\rho|/n < \delta$$

This contradicts the hypothesis that the minimal distance of $\mathcal{C}$ is δ, and the theorem follows. ∎

Acknowledgments

Eli Ben-Sasson was supported by NSF grants CCR-0133096, CCR-9877049, CCR 0205390, and NTT Award MIT 2001-04. Oded Goldreich was supported by the MINERVA Foundation, Germany. Madhu Sudan was supported in part by NSF Awards CCR 9912342, CCR 0205390, and NTT Award MIT 2001-04.

References

1. S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Intractability of Approximation Problems. *Journal of the ACM*, Vol. 45, pages 501–555, 1998.
2. M. Bellare, O. Goldreich and M. Sudan. Free Bits, PCPs and Non-Approximability – Towards Tight Results. *SIAM Journal on Computing*, Vol. 27, No. 3, pages 804–915, 1998.
3. E. Ben-Sasson, O. Goldreich and M. Sudan. Bounds on 2-Query Codeword Testing. ECCC, TR03-019, 2003.

4. E. Ben-Sasson, P. Harsha, S. Raskhodnikova. Some 3-CNF Properties are Hard to Test. In *35th STOC*, 2003.
5. K. Friedl and M. Sudan. Some Improvements to Total Degree Tests. In *Proc. of ISTCS*, pages 190-198, 1995.
6. O. Goldreich, S. Goldwasser, D. Ron. Property Testing and its connection to Learning and Approximation. *Journal of the ACM*, 45(4):653–750, July 1998.
7. O. Goldreich and M. Sudan. Locally Testable Codes and PCPs of Almost-Linear Length. In *43rd FOCS*, pages 13–22, 2002.
8. M. Kiwi. *Probabilistically Checkable Proofs and the Testing of Hadamard-like Codes.* Ph.D. Thesis, MIT, 1996.
9. R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, Vol. 25 (2), pages 252–271, 1996.

Appendix: A general proposition regarding property testing

In Section 3.1, we used the fact that, without loss of generality, a perfect-completeness codeword-tester for a linear code makes only linear tests. This fact is a special case of the following general (folklore) proposition:

Proposition A.1 *Let M be an oracle machine for the promise problem $(\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$ such that for every $x \in \Pi_{\mathrm{YES}}$ it holds that $\mathbf{Pr}[M^x = 1] = 1$ (i.e., M has perfect completeness). Then, modifying M such that it outputs 1 if and only if its view is consistent with some $x' \in \Pi_{\mathrm{YES}}$ may only improve its performance. That is, denoting the modified machine by $\widetilde{M}$, we have $\mathbf{Pr}[\widetilde{M}^x = 1] = 1$ for every $x \in \Pi_{\mathrm{YES}}$ and $\mathbf{Pr}[\widetilde{M}^x = 1] \leq \mathbf{Pr}[M^x = 1]$ for every x.*

In our case, the property being tested is belonging to a certain linear subspace, and thus in our case consistecy (among two answers) means satisfying a linear condition.

Proof: Let us fix a contents r to the random-tape of M, and denote by $\mathrm{view}_M^x(r)$ the view of machine M on random-tape r and access to oracle x. Then, machine $\widetilde{M}$ accepts on random-tape r and access to oracle x if and only if $\mathrm{view}_M^x(r)$ equals $\mathrm{view}_M^{x'}(r)$ for some $x' \in \Pi_{\mathrm{YES}}$ (where the condition may be determined by scanning all $x' \in \Pi_{\mathrm{YES}}$ and computing the corresponding $\mathrm{view}_M^{x'}(r)$'s). Clearly, $\mathbf{Pr}[\widetilde{M}^x = 1] = 1$ for every $x \in \Pi_{\mathrm{YES}}$ (by considering $x' = x$). On the other hand, for every x and r, if $M^x(r) \neq 1$ then by the one-sided feature of M it must be that $\mathrm{view}_M^x(r)$ differs from $\mathrm{view}_M^{x'}(r)$ for all $x' \in \Pi_{\mathrm{YES}}$. It follows that $\widetilde{M}^x(r) \neq 1$ too. Thus, $\mathbf{Pr}[\widetilde{M}^x \neq 1] \geq \mathbf{Pr}[M^x \neq 1]$, and the proposition follows. $\square$