Symmetric LDPC codes are not necessarily locally testable

Eli Ben-Sasson, Ghid Maatouk, Amir Shpilka and Madhu Sudan

June 8, 2011

Locally testable codes (LTC)

- Linear code $\mathcal{C}$, subspace of $\mathbb{F}^N$
- Given $f \in \mathbb{F}^N$, can determine if $f \in \mathcal{C}$ with a constant number of queries.
- Wlog, $\mathcal{C}$ is q-testable if there exists a *canonical* (q, δ, s)-*tester* for $\mathcal{C}$.

Canonical Tester

- A *canonical* (q, δ, s)*-tester* for a linear code $\mathcal{C}$ is specified by a distribution μ over $\mathcal{C}^{\perp}_{\leq q}$.
- operates by sampling u according to μ and accepting f if and only if $\langle u, f \rangle = 0$, where $\langle u, f \rangle = \sum_{i=1}^{n} u(i) f(i)$.
- comes with a guarantee to reject words which are δ-far from $\mathcal{C}$ with probability at least s.
- Think about δ and s being 0.1.

LTC are LDPC

- If $\mathcal{C}$ has a k-local tester, then $\mathcal{C}^{\perp}$ is spanned by words of weight $\leq k$
- If $\mathcal{C}$ has a k-local tester, $\mathcal{C}$ is a k-LDPC (Low Density Parity Check) code
- Converse: does k-LDPC imply k-testability?

LTC are LDPC

- If $\mathcal{C}$ has a k-local tester, then $\mathcal{C}^{\perp}$ is spanned by words of weight $\leq k$
- If $\mathcal{C}$ has a k-local tester, $\mathcal{C}$ is a k-LDPC (Low Density Parity Check) code
- Converse: does k-LDPC imply k-testability?

  [BHR05] Not for random LDPC codes!

LTC are LDPC

- If $\mathcal{C}$ has a k-local tester, then $\mathcal{C}^{\perp}$ is spanned by words of weight $\leq k$
- If $\mathcal{C}$ has a k-local tester, $\mathcal{C}$ is a k-LDPC (Low Density Parity Check) code
- Converse: does k-LDPC imply k-testability?

  [BHR05] Not for random LDPC codes!
- Intuition: random LDPC codes lack "structure".

Codes with symmetries

- Let π be a permutation of $\{1, \ldots, N\}$
- $\mathcal{C} \circ \pi = \{(x_{\pi(1)}, \ldots, x_{\pi(N)}) \mid (x_1, \ldots, x_N) \in \mathcal{C}\}$
- $\mathcal{C}$ *invariant* under π if $\mathcal{C} = \mathcal{C} \circ \pi$
- Automorphism group of $\mathcal{C}$ is $\{\pi \mid \mathcal{C} = \mathcal{C} \circ \pi\}$.

Examples of automorphism groups

- Symmetric group S_N: too restrictive

Examples of automorphism groups

- Symmetric group S_N: too restrictive
- **Affine group**:

$$\mathrm{Affine}_{\mathbb{K}} = \{\pi_{a,b} : x \mapsto ax + b\}$$

- Index coordinates $\{1, \ldots, N\}$ by elements of a field $\mathbb{K}$

Examples of automorphism groups

- Symmetric group S_N: too restrictive
- **Affine group**:

$$\mathrm{Affine}_{\mathbb{K}} = \{\pi_{a,b} : x \mapsto ax + b\}$$

- Index coordinates $\{1, \ldots, N\}$ by elements of a field $\mathbb{K}$
- Example of affine-invariant codes: Reed-Muller codes
  - $RM[m, d]_{\mathbb{K}}$ is set of evaluations of m-variate polynomials of degree d over $\mathbb{K}$
  - invariant under affine transformations $\bar{x} \mapsto A \cdot \bar{x} + \bar{b}$.

Conjecture [AKKLR05]

- For general $\mathcal{C}$, automorphism group is trivial (only identity permutation)

Conjecture [AKKLR05]

- For general $\mathcal{C}$, automorphism group is trivial (only identity permutation)
- Large/rich automorphism group gives hope for testability

Conjecture [AKKLR05]

- For general $\mathcal{C}$, automorphism group is trivial (only identity permutation)
- Large/rich automorphism group gives hope for testability
- **Conjecture** [AKKLR05]: If $\mathcal{C}$ has a 2-transitive automorphism group and a low-weight dual codeword, $\mathcal{C}$ is LTC

Conjecture [AKKLR05]

- For general $\mathcal{C}$, automorphism group is trivial (only identity permutation)
- Large/rich automorphism group gives hope for testability
- **Conjecture** [AKKLR05]: If $\mathcal{C}$ has a 2-transitive automorphism group and a low-weight dual codeword, $\mathcal{C}$ is LTC
- 2-transitivity: lots of (weight-preserving) permutations in automorphism group. Hopefully, $\mathcal{C}$ is LDPC..

Conjecture [AKKLR05]

- For general $\mathcal{C}$, automorphism group is trivial (only identity permutation)
- Large/rich automorphism group gives hope for testability
- **Conjecture** [AKKLR05]: If $\mathcal{C}$ has a 2-transitive automorphism group and a low-weight dual codeword, $\mathcal{C}$ is LTC
- 2-transitivity: lots of (weight-preserving) permutations in automorphism group. Hopefully, $\mathcal{C}$ is LDPC..
- Conjecture refuted in [GKS08]: there exist such codes which are not even LDPC.

Conjecture (revisited)

- **Conjecture**[AKKLR05]: If $\mathcal{C}$ is LDPC and has a 2-transitive automorphism group, then $\mathcal{C}$ is LTC
- Known LTC are symmetric LDPC
  - RM codes
  - "Sparse" affine-invariant codes ($O(|\mathbb{K}|^{\ell})$ codewords) [GKS09]
  - "Single-orbit" affine-invariant codes [KS05]

Conjecture (revisited)

- **Conjecture**[AKKLR05]: If $\mathcal{C}$ is LDPC and has a 2-transitive automorphism group, then $\mathcal{C}$ is LTC
- Known LTC are symmetric LDPC
  - RM codes
  - "Sparse" affine-invariant codes ($O(|\mathbb{K}|^{\ell})$ codewords) [GKS09]
  - "Single-orbit" affine-invariant codes [KS05]
- **Our result**: the conjecture is false. There exists an infinite family of 2-transitive LDPC codes which is not testable with a constant number of queries.

Main result

Theorem (Symmetric LDPC codes are not necessarily LTC)

For every prime p there exists a positive integer k and an infinite family of positive integers $\mathcal{N}$ such that for every $n \in \mathcal{N}$ the following holds:

- *There is an affine-invariant code $\mathcal{C}^{(n)} \subseteq \{\mathbb{F}_{p^n} \to \mathbb{F}_p\}$.*
- *$\mathcal{C}^{(n)}$ is a k-LDPC code.*
- *$\mathcal{C}^{(n)}$ is not $o(\log n / \log \log n)$-locally testable.*

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes
- Problem: these families are also LTC!

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes
- Problem: these families are also LTC!
- Second attempt: use them as building blocks.

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes
- Problem: these families are also LTC!
- Second attempt: use them as building blocks.
  - Taking sums ($\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$): preserves symmetry and LDPC, but also LTC [BGMSS11]

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes
- Problem: these families are also LTC!
- Second attempt: use them as building blocks.
  - Taking sums ($\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$): preserves symmetry and LDPC, but also LTC [BGMSS11]
  - Taking intersections ($\mathcal{C} = \mathcal{C}_1 \cap \mathcal{C}_2$):
    - Intersection of k-LDPC codes is also k-LDPC
    - Intersection of k-LTC codes not necessarily k-LTC!

Constructing a symmetric LDPC non-LTC

- Known symmetric LDPC codes:
  - $RM[m, d]_{\mathbb{K}}$
  - Sparse codes
- Problem: these families are also LTC!
- Second attempt: use them as building blocks.
  - Taking sums ($\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$): preserves symmetry and LDPC, but also LTC [BGMSS11]
  - Taking intersections ($\mathcal{C} = \mathcal{C}_1 \cap \mathcal{C}_2$):
    - Intersection of k-LDPC codes is also k-LDPC
    - Intersection of k-LTC codes not necessarily k-LTC!
    - **BUT** Simply taking intersection of sparse/RM codes will not work.

Lifting affine-invariant codes

- Defining a new operation on affine-invariant codes
- Let $\mathbb{F} \leq \mathbb{L}$ and $\mathcal{C} \leq \mathbb{F}^{|\mathbb{L}|}$
- Recall: we view coordinates of codeword as field elements
- View elements of $\mathcal{C}$ as functions from $\mathbb{L}$ to $\mathbb{F}$: $\mathcal{C} \subseteq \{\mathbb{L} \to \mathbb{F}\}$.

Lifting affine-invariant codes

- Defining a new operation on affine-invariant codes
- Let $\mathbb{F} \leq \mathbb{L}$ and $\mathcal{C} \leq \mathbb{F}^{|\mathbb{L}|}$
- Recall: we view coordinates of codeword as field elements
- View elements of $\mathcal{C}$ as functions from $\mathbb{L}$ to $\mathbb{F}$: $\mathcal{C} \subseteq \{\mathbb{L} \to \mathbb{F}\}$.

- Now let $\mathbb{F} \leq \mathbb{L} \leq \mathbb{K}$ and $\mathcal{C} \subseteq \{\mathbb{L} \to \mathbb{F}\}$
- Want to "lift" $\mathcal{C}$ to a code $\mathcal{C}' \subseteq \{\mathbb{K} \to \mathbb{F}\}$
- $\mathrm{Lift}(\mathcal{C}) = \{f : \mathbb{K} \to \mathbb{F} \mid \mathrm{Trace}(f) \in \mathcal{C}\}$
- Recall $\mathrm{Trace}_{\mathbb{L},\mathbb{K}}(x) := \sum_{i=0}^{[\mathbb{K}:\mathbb{L}]} x^{p^i}$
- Any test for $\mathcal{C}$ can be viewed as a test for $\mathrm{Lift}(\mathcal{C})$.

The idea

- Work on an extension field $\mathbb{K} := \mathbb{F}_p^n$ with many subfields
- $n := p_1 \cdot \dots \cdot p_\ell$, $\mathbb{L}_i := \mathbb{F}_p^{p_i}$.
- Start with (sparse) symmetric LDPC testable codes $\tilde{\mathcal{C}}_i \subseteq \{\mathbb{L}_i \to \mathbb{F}_p\}$
- "Lift" the $\tilde{\mathcal{C}}_i$ to codes $\mathcal{C}_i \subseteq \{\mathbb{K} \to \mathbb{F}_p\}$
  - $\mathcal{C}_i$ no longer sparse, but still LDPC and testable
- Intersection of $\mathcal{C}_i$'s is our code
  - Keep the LDPC property (easy), lose the testability (harder).

$\mathcal{C}$ is not locally testable

- Proof relies on analysis of structure of symmetric sparse codes (developed in [KS05], [GKS08], [GKS09], [BS10], [KL10],...)
- Constraints lead to the construction of certain rectangular matrices over finite field, whose kernels we analyze
- Combine with [BHR05] strategy for proving non-testability: defining a distribution on faraway words that fools canonical testers.

Thank you!
Questions?