The Hebrew University of Jerusalem
The Rachel and Selim Benin School of Computer Science and Engineering

Thesis Title

Author's Name

Thesis submitted in partial fulfillment of the requirements
for the Master of Sciences degree
in Computer Science

Under the supervision of **Prof./Dr. Supervisor's Name**

Month year

Contents

List of Figures

Chapter 1

Introduction

Chapter 2

Codes

2.1 Introduction

Coding theory has emerged due to the need to transfer information in noisy communication channels. By embedding a message in a higher-dimensional space, one can guarantee robustness against possible faults. The ratio of the original content length to the transmitted message *length* is the *rate* of the code, and it measures how consuming our communication protocol is. Additionally, the *distance* of the code quantifies how many faults the scheme can absorb such that the receiver can recover the original message. We can consider the code as a collection of all strings that satisfy specified restrictions.

Non-formally, a code is good if its distance and rate scale linearly with the encoded message length. In practice, one is also interested in implementing these checks efficiently. We say that a code is an LDPC if any bit is involved in a constant number of restrictions, each of which is a linear equation, and if any restriction contains a fixed number of variables.

Moreover, another characteristic of the code is its testability, which is the complexity of the number of random checks one must do to verify that a given candidate is in the code. Besides being considered efficient in terms of robustness and overhead, good codes are also vital components in establishing secure multiparty computation [**MultiParty**] and have a deep connection to probabilistic proofs.

In Section 2, we state the notations, definitions, and formal theorem. Then, in Sections 3 and 4, we review past results and provide their proofs to make this paper self-contained. Readers familiar with the basic concepts of LDPC, Tanner, and Expanders codes construction may consider skipping directly to Section 5, in which we provide our proof. Readers familiar with the basic concepts of LDPC, Tanner, and Expanders codes construction may skip Sections 2, 3, and 4 and proceed directly to Section 5, where we provide our proof.

2.2 Codes in General. Notations and Definitions.

Here we focus only on linear binary codes, which one could think about as linear subspaces of $\mathbb{F}_2^n$. A common way to measure resilience is to ask how many bits an evil entity needs to flip such that the corrupted vector will be closer to another vector in that space than the original one. Those ideas were formulated by Hamming [**Hamming**], who presented the following definitions.

2.3 Locally Testable Codes.

In addition to distance and rate, we are also interested in ensuring that the checking process is robust. Specifically, we want to guarantee that a single missed check will only have a tiny probability of compromising the entire computation, even in the presence of significant errors. Consider a code C a string x, and denote by $\xi(x)$ the fraction of the checks in which x fails. C will be called a ***local-testabile*** *$f(n)$ If there exists $\kappa > 0$ such that*

$$\frac{d(x,C)}{n} \leq \kappa \cdot \xi(x) f(n)$$

Next, we will introduce the polynomial code, a testable code. We will then demonstrate a proof that emphasizes the concept of restriction degeneration, meaning that if more than a certain number of restrictions are satisfied, then almost all the remaining restrictions must also be satisfied. It is also worth mentioning that the polynomial code has a nice quantum variant which we will present in the next chapter.

2.4 Polynomial Code.

Consider the field $\mathbb{F}_q$ for an arbitrary prime power q greater than n. The polynomial codes rely on the fact that any two different polynomials in the ring $\mathbb{F}_q[x]$ of degree at most d differ by at least $q-d+1$ points. For example, consider a pair of polynomials of degree 1, namely two linear straight lines. If they are not identical, then they have at most one intersection point, and they disagree on each of the $n-1$ remaining points. By defining the code to be the subspace containing all polynomials of degree at most d, such that any codeword is an image of such a polynomial encoded by n numbers, we can guarantee a lower bound on the code's distance. Formally, we define:

Now we are about to prove that the polynomial code is a testable code. The next claim will enable us to design a random test to certify polynomials. Consider the finte field $\mathbb{F}_q$ for prime q, denote by w the d root of 1, namely $w^d = 1$ then for any polynomial f with degree at most d it holds:

$$f(x) = \frac{1}{d}\sum_{t}^{d} f(x + w^t y) \quad \forall x, y \in \mathbb{F}_q \tag{2.1}$$

Definition 2.4.1 (Polynomial Code. [**Reed1960PolynomialCO**])**.** *Proof.* Denote by a_j the jth coffecicient of f. And recall that for any fiexd $l \in \mathbb{F}_q$ the summtiom $\sum_t^d w^{t\cdot l}$ equals:

$$\sum_{t}^{d} w^{t\cdot l} = \begin{cases} \frac{w^{l\cdot d}-1}{w^l-1} = 0 & l \neq 0 \\ d & l = 0 \end{cases}$$

Thus:

$$\begin{aligned}
&\sum_t f\left(x + w^t y\right) = \sum_{t,j} a_j \left(x + w^t y\right)^j = \sum_{t,j} a_j \left(\sum_l x^{j-l} w^{t\cdot l} y^l \binom{j}{l}\right) \\
&= \sum_j a_j \sum_l x^{j-l} y^l \binom{j}{l} \sum_t w^{t\cdot l} \\
&= \sum_j a_j x^j \cdot d = f(x) \cdot d
\end{aligned}$$

□

3.1 Introduction.

It is widely believed that quantum machines have a significant advantage over classical machines in a wide range of computational tasks [grover1996fast], [ahuja1999quantum]. Simple algorithms which can be interpreted as the quantum version of scanning all the options, reducing the running time by the square root of the classical magnitude.
Nevertheless, Shor has demonstrated a polynomial-depth quantum circuit that solves the hidden abelian subgroup problem [Shor˙1997], which is considered a breakthrough, as it made the computer science community believe that a quantum computer could offer an exponential advantage.
Despite a consensus on the superiority of the ideal quantum computing model, it is still uncertain whether it is possible to implement such a machine in a noisy

environment. Still, simply pointing out the existence of noise is not sufficient to negate the feasibility of computation. Evidence of this is that classical computers also experience a certain rate of errors. Therefore, to fully comprehend the difficulty, let us compare two main factors that made it a challenging task. First is the magnitude of the error rate; classical computers also have errors, and sometimes we witness system failures (e.g. the blue screen). The error rate of modern computers is so low that the probability of errors propagating stays negligible, even if the length of the computation is polynomial in the scale of what is considered a reasonable input size. It's worth mentioning that in exascale computing, when supercomputers perform around 10^{18} operations per second, it is difficult to overlook faults. In quantum computing, we become aware of their existence much earlier.

The second difference, which is a subtle point, is that quantum states are susceptible to additional types of errors.

In addition to the possibility of bit-flip errors, a quantum state may also experience a change in phase. For example, consider the initial state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and suppose that due to noise the state is transformed into $\frac{1}{\sqrt{4}}\left(\sqrt{3}\,|0\rangle + |1\rangle\right)$. Classical circuits are oblivious to such faults, meaning that their operation would remain unchanged as no error has occurred. Quantum circuits, however, are usually affected and may fail. Furthermore, when designing a decoder for quantum error correction codes, one must ensure that the decoding process does not introduce bit-flip errors if a classical code is used to protect against phase flips.

3.2 Quantum Noise.

*However, even though quantum noise is so violent, it has been proven that any ideal circuit of polynomial depth can be transformed into a robust circuit at poly-logarithmic cost [***aharonov1999faulttolera***In other words, there is a threshold: if physicists provide qubits and a finite gate set with a noise*

*rate below that threshold, then BQP, the class of polynomial-time ideal quantum computation, is feasible and can be computed on a realistic machine. The basic idea in [***aharonov1999faulttolerant***] was to show the existence of quantum error correction codes, which would enable logic operations to be performed in a way that prevents errors from propagating. This process involves separating the operation into two stages: the operation itself and an error correction stage. This comes with an additional cost in terms of space and time, but it can reduce the probability of the final state being faulty. The trade-off between the resources needed and the rate of decrease defines the threshold. If the balance is positive, then the process can be repeated in a recursive manner, and after log-log iterations, the failure probability will decay to zero. At the same time, the circuit will scale to a maximum of poly-logarithmic width and depth. Let's return to the repetition code presented in Chapter 2. We would like to have an analog; a first and natural attempt might consider duplicating copies of the state. Unfortunately, copying a general state is not a linear operation and therefore cannot be done in the circuit model (or any other believed to be feasible). In particular, there is no*

*circuit U which can simultaneously duplicate the states $|0\rangle, |1\rangle, |+\rangle, |-\rangle$. To overcome the issue, Shor came up with the nine-qubit code [***Ninequ***], which at first glance might seem a naive straightforward implementation of "duplication", but instead uses a clever insight about quantumness in general. Any operation can be seen as a linear (and even unitary) operation over a subspace embedded in a large enough dimension. The encoding is given as follows:*

$$|\overline{0}\rangle = \frac{1}{2\sqrt{2}}\left(|000\rangle + |111\rangle\right)^{\otimes 3}$$
$$|\overline{1}\rangle = \frac{1}{2\sqrt{2}}\left(|000\rangle - |111\rangle\right)^{\otimes 3} \quad .$$

For convenience, let us use the notation $|\mathbf{GHZ}^{\pm}\rangle = |0^m\rangle \pm |1^m\rangle$. We can also consider the Shor code over m^2 qubits, which is defined as above, such that any logical state contains m products over m qubits. Therefore, the state $|\overline{0}\rangle$ over m^2 qubits can be written as $|\mathbf{GHZ}^{+}\rangle^m$. We are now ready to prove a statement regarding the robustness.The Shor code over 9 qubits enable to correct a single either bit or phase flip.

Chapter 4

Good qLDPC Codes and LTC.

*The existence of good quantum LDPC codes and locally testable codes (LTCs) was considered an open problem for roughly two decades. Although they seemed to be related only by containing the word "code" in their names, they were proven to exist by the same construction. They first appeared in [***Dinur***] as good locally testable codes and not long after that in [***Pavel***], in which they also extended and derived the result to obtain the quantum code. We emphasize that even though they developed the same codes, their proofs are not similar at all. Here, we follow the [***leverrier2022quantun*** work, which simplifies the original proof and does not rely on any concept more complicated than what we have already seen in the previous chapters. They also coined the term "Quantum Tanner Codes" referring to the fact that C_X and C_Z are classical Tanner codes. Yet, the proof we present*

*is not exactly the same, as we use a small code that requires satisfying a stronger assumption (the w-robustness property 4.1.1) relative to the original work. The reason why they had to use a weaker assumption is because the existence of codes satisfying the stronger one was proven a year later [***kalache***. Relying on the stronger assumption allows us to simplify the proof even more and get rid of another requirement that the small code has to satisfy (The p-resistance to puncturing* **??***).*

4.1 Quantum Tanner Codes.

Recall our insight that for a pair of LDPC codes to define a good CSS code, they must both be poor codes in the sense that they must have a constant distance. Therefore, we understand that any codeword in C_X with small weight belongs to $C_Z^\perp$. To prove this, we will construct a proof such that if $x \in C_X$ and $|x|$ is small, then there is a small codeword $z \in C_Z^\perp$ such that $|x + z| < |x|$; by repeating this process recursively, it follows that $x \in C_Z^\perp$. To formulate this theorem, we will need to define more definitions.
The next two definitions are concerned with the properties of the small code that will be set on the

edges. Using them, one can characterize cases in which a local view can be reduced by subtracting a codeword from the dual code.

Both good quantum LDPC and good LTC had been open problems for more than decades, so the fact that they have been solved in one stroke raises two interesting questions. The first one is whether the construction that obtains each of them alone can also yield good quantum LTC? What is the exact challenge that prevents the inventors from proceeding into quantum testability? The second question is whether one of the codes can be obtained without giving the other. Either positive or negative answers to these questions will shed light on our understanding of what quantumness is. In this chapter, we present our attempts to provide answers. The chapter is divided into two parts. In the first, we demonstrate a variation of the classical Tanner code that defines many equations. We believe that this variation contains enough structure to guarantee degeneration of equations similarly to

what occurs in the polynomial code. In the second chapter, we show that the polynomial code is not w-robust. If it were not the case, then one could hope to obtain quantum testability by considering a Tanner code in which any vertex restricts its local view to be a codeword of a quantum code.

5.1 Local Majority =? Local Testability.

We begin by demonstrating that selecting C_0, the small code in the Tanner code, to have a large distance, which can also be considered as adding numerous restrictions, yields testability. However, the amount one will have to enlarge the distance to cannot be achieved with a rate greater than $\frac{1}{2}$ by the Singleton bound.

5.1.1 Almost LTC With Zero Rate

Suppose that G is an expander graph with a second eigenvalue λ, then For any layer U there exist

a layer U' such that:

(1) $|U'| \geq |U|$

(2) $w_{E/E'}(x|_{U'}) \geq \Delta|U'|\left(\delta_0 - \frac{2}{3} - \frac{2\lambda}{\Delta}\right)$

Theorem (Theorem 1)**.** *Proof.* Consider layer U and denote by U_{-1} and U_{+1} the preceding and the following layers to U in T. It follows from the expander mixing lemma that:

$$w_{E/E'}(x|_U) \geq \delta_0\Delta|U| - w\left(E(U_{-1}\bigcup U_{+1}, U)\right) \geq$$
$$\delta_0\Delta|U| - E(U_{-1}\bigcup U_{+1}, U)$$
$$\delta_0\Delta|U| - \Delta\frac{|U||U_{-1}|}{n} - \Delta\frac{|U||U_{+1}|}{n}$$
$$- \lambda\sqrt{|U||U_{-1}|} - \lambda\sqrt{|U||U_{+1}|}$$

5.1.2 Overcoming The Vanishing Rate.

Consider the following code; instead of associating each edge with pair of checks, let's define the vertices to be the checks of small codes over $q \in [0,1]$ fraction of their edges. That is, now each vertex defines only $(1-\rho_0)\,q\Delta$ restrictions.

Hence, the rate of the code is at least:

$$\begin{aligned}\rho\frac{1}{2}\Delta n &\geq \frac{1}{2}\Delta n - (1-\rho_0)\, q\Delta n \\ \Rightarrow \rho &\geq \left(2\rho_0 + \left(\frac{1}{q} - 2\right)\right) q \\ \rho_0 &\geq 1 - \frac{1}{2q}\end{aligned}$$

for example, if $q = 2/3$, then for having constant rate, it is enough to ensure that $\rho_0 \geq 1 - \frac{3}{4} = \frac{1}{4}$. Before expand the construction let's us justify why one should even expects that removing constraints preserves testability. Assume that is guaranteed that the lower bound of the flux on the trivial vertices remains up to multiplication by the fraction factor q, or put it differently, one could just stick q in every inequality without lose correctness, Then:

$$\begin{aligned}w_{E/E'}\left(x|_U\right) &\geq \delta_0 q\Delta|U| - qw\left(E(U_{-1}\bigcup U_{+1}, U)\right) \\ \Rightarrow |x| &\geq \left(\delta_0 - \frac{2}{3} - \frac{2\lambda}{\Delta}\right) q\Delta|U_{\max}|\end{aligned}$$

*As you can see, ireducable [**??**] words of the disagreement have a linear weight, despite that the original code has non-vanish rate. Yet, We still require more to prove a linear distance. By repeating on the* Singleton Bound **??** *proof it follows that the small code $\tilde{C}_0$ obtained by ignoring arbitrary*

$\left(q - \frac{1}{2}\right)\Delta$ coordinates yield a code with distance:

$$\left(\delta_0 q - \left(q - \frac{1}{2}\right)\right)\Delta$$

So assume that we could engineer an expander family such that the graphs obtained by removing $\frac{1}{2}$ of the edges connected for each vertex result also expanders, and in addition, regarding $\tilde{C}_0$ each edge is checked by both vertices on its support. Namely, a good Tanners Code could be defined on the restricted graphs; Then, any string that satisfies the original checks also has a linear weight. To achieve this property, we will restrict ourselves to a particular family of Cayley Graphs.There exist a constant $\alpha > 0$ and infinite family of codes which satisfies Theorem 1 and also good.

Intuition For Testability.

5.2 The Polynomial-Code Is Not w-Robust.

One idea for constructing is to use the polynomial code instead of C_0. This follows from the fact that if one picks a degree strictly greater than $\Delta/2$, then $C_0^{\perp} \subset C_0$ and therefore one could choose C_z to be the same code defined on the negative vertices of the graph. Here we prove that the dual-tensor code, in that case, is not w-robust, meaning that any such construction should be consid-

ered another way of proving the Reduction Lemma. Let C_0 *be the* $[\Delta, d, \Delta - d]$ *polynomial code. Then any code word in* $\left(C_0^{\perp} \otimes C_0^{\perp}\right)^{\perp}$ *is a polynomial in* $F[x, y]$ *at degree at most* $\Delta + d$

Definition 5.1.1 (Testability Tanner Code)**.** *Proof.* Consider base element $C_0 \otimes \mathbb{F}$, denote it by $c = g_i \otimes e_j$. And notice that c has representation in $F[x, y]$ of $\prod_{y' \neq j} (y - y') g_i(x)$. By the fact that $g_i(x) \in C_0$ we have that degree of c is at most $\Delta + \delta$. Hence any element in the subspace of $C_0 \otimes \mathbb{F}$ is a polynomial at degree at most $\Delta + d$. □