Quantum Error Correction using Surface Codes

Muhammad Ubaid Ur Rehman

September 10, 2025

Abstract

This paper provides a concise overview of surface codes which are an important class of topological error-correcting codes first introduced by Kitaev and has since become a leading contender for practical Quantum Error Correction.

1 Introduction

Since the discovery of quantum algorithms in the 1980s, there has been a steady increase in interest and the amount of effort put into quantum computation due to its promise of pushing the boundaries of computation, and its application to security, defense, scientific research and simply uncovering what was previously not even known to be unknown. In the last two decades, there has been significant improvement in the physical realization of quantum systems and considerable challenges have been overcome. However, one challenge remains above all and has proven itself to be the most difficult yet of prime importance.

All of the physical systems that are used to build quantum circuits and qubits are inherently noisy and hence not suitable for efficient computation. The task of building qubits that are resistant to noise and decoherence to a sufficient degree is currently the biggest challenge to achieving scalable and efficient quantum computing such that it provides a considerable gain over classical computing.

2 Quantum Error Correction

Quantum Error Correction is an umbrella term used to describe different approaches employed to protect quantum information against noise and decoherence. Initially it was believed that quantum error correction may never be possible due to two major reasons. First one being the no-cloning theorem which prohibits copying unknown quantum state to multiple qubits and secondly, the fact that errors in nature are continuous, or to say analogue whereas our computation is quantized or digital.

$$\hat{U}\,|\psi\rangle\,|0\rangle \neq |\psi\rangle\,|\psi\rangle$$

However, both these challenges were ovecome in 1990s when Peter Shor introduced the first ever quantum error correction algorithm. His algorithm, like classical ones, used redundancy to secure information but instead of copying information on multiple qubits unlike classical error correction, the same quantum state was spread over multiple physical qubits which collectively acted as a single logical qubit. This allowed us to circumvent the no-clonning theorem while using multiple qubits to increase redundancy and higher resistance to noise. The second challenge was even simpler to solve, as all physical errors could be digitized in terms of Pauli operator basis $\{$I, X, Y, Z$\}$.

In his algorithm, Shor used 9 physical qubits to encode 1 logical qubit and his code was able to correct any single qubit Pauli error. The details of Shor code are beyond the scope of this article but the logical states are encoded as following in the 9 physical qubits using an encoder circuit with ancillas:

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

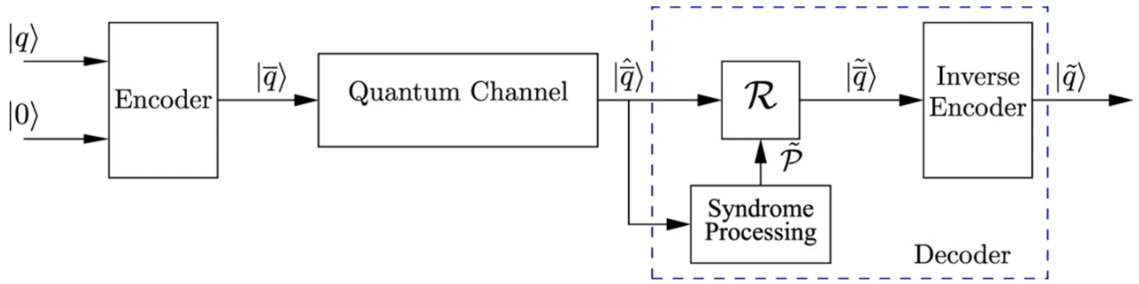


Figure 1: General Error correction scheme via stabilizers: the state of interest is encoded into logical qubits using an encoder circuit, which then pass through a noise channel. Syndrome measurements are made which output only classical information about errors which is then used to apply a recovery operation to undo the noise channel effects. As a last step, the encoded state is decoded back into the original basis as an output.

3 Stabilizer Formalism

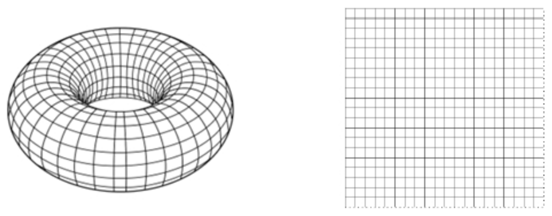
In quantum error correction, stabilizer formalism is a systematic method of defining various error-correcting codes in terms of operators called stabilizers and corresponding code states called stabilizer codes. A (binary) stabilizer code can be characterized as the simultaneous eigenspace with eigenvalue one of a set of mutually commuting check operators (or "stabilizer generators"), where each generator is a "Pauli operator."

4 Surface Codes

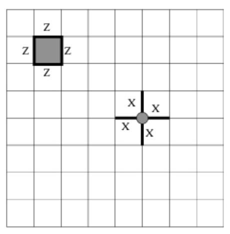
Surface codes are a type of topological stabilizer codes that are defined on two-dimensional spin lattice. Originally, the surface code was defined on a 2D surface with periodic boundary which gave it a shape of a torus (donut) and hence the original name, Toric codes. However, it has been proved that periodic boundary is not a necessary condition and planar surface codes have been developed with parallel qubit lattices. However, well only discuss the toric code here. Surface codes have been at the very forefront of quantum error correction research for a number of reasons. First one being their local nature. Since qubits don't need to interact with distant neighbours and most operations include only the nearest neighbours, they are much easier to physically implement. They also aren't limited to square lattice but rather any tessalition will correspond to a quantum code. Ancila qubits are also much readily available to them on a parallel surface. Moreover, they have a very high fault tolerance threshold of approximately 1% which is significantly larger then the competition, for example code contention which is 10^{-4}.

4.1 Toric Code

Toric code, first introduced by Alexei Kitaev, models qubits as arranged on a 2D square lattice. It employs the topological properties of a torus which has a genus 1 hole as an invariant. To understand, consider an $L \times L$ square lattice with periodic boundary. The periodic boundry condition in 2D forces our plane to take form of a torus. We start by joining opposite ends of our plane to form a pipe which ensures periodicity in one of the dimensions. Next, to ensure periodicity in the second dimension, we join togather the opposite ends of the pipe by bending it in the shape of a torus (or a donut if that's what you like) as shown in figure.2a



((a)) Z and X check operators on lattice



((b)) Torus tessellated by a periodic square lattice

Figure 2

It is convenient to consider this $L \times L$ lattice as a dual lattice, where dual lattice is formed by connecting the midpoints of edges of original lattice. Lets call original lattice $\mathcal{L}$ and dual $\mathcal{L}'$. Each link of lattice $\mathcal{L}$ is a data qubit whereas check operators (qubits) $\{X_s, Z_p\}$ are represented by vertices of $\mathcal{L}$ and $\mathcal{L}'$ respectively.[1] For a toric code with $L \times L$ lattice, there are $2L^2$ links or edges, and hence $2L^2$ data qubits as well as $2(L^2-1)$ check qubits.

Now lets take a more closer look at action and properties of these check operators. Each check operator $\{X_s, Z_p\}$ located at site $s_{\mathcal{L}}$ and $s'_{\mathcal{L}'}$ respectively acts only on four of it's neighboring data qubits and hence action of these check operators is very local. For simplicity, we can restrict ourselves to only $\mathcal{L}$, in which case Z_p check operators act on four qubits located at each edge surrounding it and thus form a plaquette. As a result, these check operators are simply tensor products of X's and Z's acting on neighboring qubits as following:

$$X_s = \otimes_{i \in s} X_i$$

$$Z_P = \otimes_{i \in P} Z_i$$

These check operators obviously commute with their type, i.e X_s with $X_{s'}$ and Z_P with $Z_{P'}$. A pair of X_s and Z_P operators either acts on two completely disjoint sets of 4 qubits or they share exactly 2 qubits in between them. In both cases, they still commute as in first case, they are acting on different qubits whereas in second case, the two minus signs cancel each-other. For example:

$$\begin{aligned}
[X_1X_2X_3X_4, Z_1Z_2Z_5Z_6] &= (X_1X_2X_3X_4)(Z_1Z_2Z_5Z_6) - (Z_1Z_2Z_5Z_6)(X_1X_2X_3X_4) \\
&= (-Y_1)(-Y_2)X_3X_4Z_5Z_6 - Y_1Y_2X_3X_4Z_5Z_6 \\
&= Y_1Y_2X_3X_4Z_5Z_6 - Y_1Y_2X_3X_4Z_5Z_6 \\
&= 0
\end{aligned}$$

Since these check operators commute with each other, they form the stabilizer group for our code.

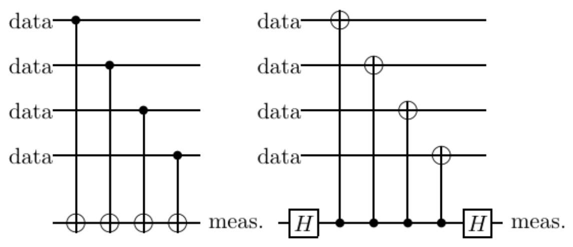


Figure 3: Circuits for Z and X check operators

4.2 Error Correction

Now we bring our attention to the elephant in the room, the errors on data qubits. Before we jump into what kind of errors can be corrected, we need some concepts from algebraic topology. In $\mathbb{Z}^2 = \{0, 1\}$ homology, a *0-chain* is a map that assigns elements of $\mathbb{Z}^2$ to vertices or sites of our lattice, *1-chain* is mapping from $\mathbb{Z}^2$ to links or edges and *2-chain* is a mapping from $\mathbb{Z}^2$ to plaquette. To put simply, 0 and 1 correspond to absence and presence of a quantity respectively. Another thing to consider is that *0-chains* form boundary of *1-chains* and *1-chains* form boundary of *2-chains*. We can use boundary operators ∂_2 to go from *2-chain* to *1-chain* and ∂_1 to got from *1-chain* to *0-chain*. In our case, *0-chains* are a mapping of defects founds at sites, *2-chains* are a map of defects at plaquettes and *1-chains* are a map of Z errors at links in $\mathcal{L}$ and X errors at links in $\mathcal{L}'$.

We know from our previous discussion on stabilizer formalism, for a Pauli operator error to be detectable and correctable, it must commute with all the check operators and preserve the code space or anti-commute with one of the stabilizers. However if an error commutes with the check operators but is not contained in the code space, it becomes a logical error. A Z error on any link commutes with plaquette operators however Z errors don't always commute with site operators. For a Z error to commute with site operator, Z errors must be acting on even number of qubits at that site. As we require it to commute with all the check operators, it

[1]There are other equivalent ways to define our lattice, for example where data qubits might be located at vertices instead of edges and lattice may have diamond shape etc. However, our convention suffices for this article.

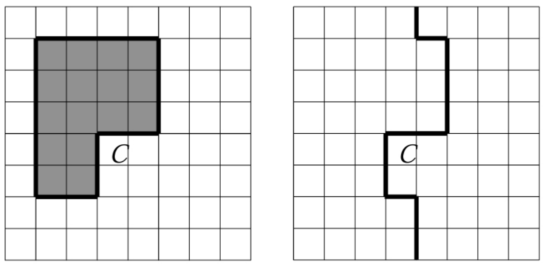
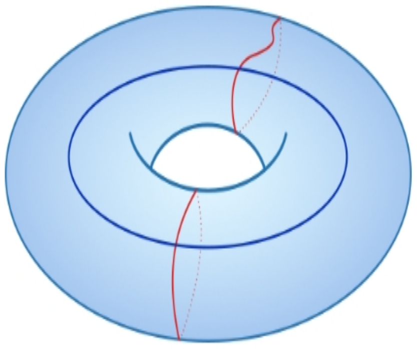


Figure 4: Trivial and Non Trivial 1-cycles

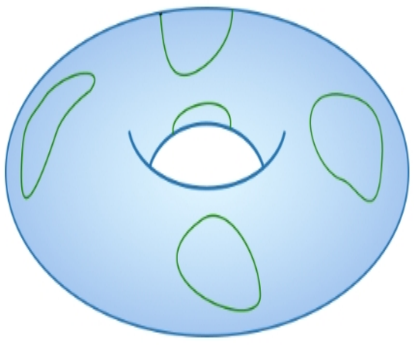
must therefore be a cyclic loop. Similarly, in dual lattice, X errors commute with all the plaquette operators but not with the site operators unless even number of Z errors act on qubits adjacent to site. This also implies that our *1-chain* representing the map of errors must also be cycles in $s_{\mathcal{L}}$ and $s'_{\mathcal{L}'}$ for Z and X respectively.

These cycles can be of two types, homologically trivial and homologically non-trivial. To understand these, take a look at figures below. A *1-chain* cycle is called trivial if it can form the boundary of a *2-chain*. This cycle can therefore be tiled by plaquettes and Z's acting on the boundary formed by product of plaquettes operators enclosed. Since these operators form the stabilizer group, the product is also an element of the group and the cycle is stabilized. Similarly, in dual-lattice, a product of X plaquette operators acting on trivial cycle boundary form an element within the stabilizer group. On the other hand, non-trivial cycles cannot form boundary of any object. They still commute with all the check operators however they don't lie within stabilizer group. Hence these form logical operators $\bar{Z}_1$ and $\bar{Z}_2$. In the dual space, these form logical $\bar{X}_1$ and $\bar{X}_2$ errors acting on the two encoded qubits.

Another way to look at these cycles is on the torus. A trivial loop is one which can be deformed to a single point however non-trivial loops cannot be deformed to single points. In the figure below, green loops are present on surface of the torus and can be reduced to a single point again lying on the surface. However, for red and blue loops, it is not possible to reduce them as doing so requires one to move across a hole which is not permissible.



((a)) Homoligically Non-trivial loops



((b)) Homoligically trivial loops

Figure 5

4.3 Recovery Operation

We need to be able to apply recovery operation after the syndrome measurement. However mapping the defect output at measurement with the correct error is a challenging task. For two defect points, many chains can share the same boundary or end points and result in the same syndrome. However, this ambiguity should not be a concern as applying recovery gates to any chain of links with same boundary will remove the error as we will get back a trivial cycle which resides within group. However if we end up applying a recovery that makes a non-trivial cycle, that wraps around the torus, then we have an irrecoverable logical error.

For example, in the figure below, applying second recovery when actual error is first one still corrects the error as we get a closed loop. However, applying last recovery operation creates a non-trivial loop that wraps around the torus and becomes a logical error.

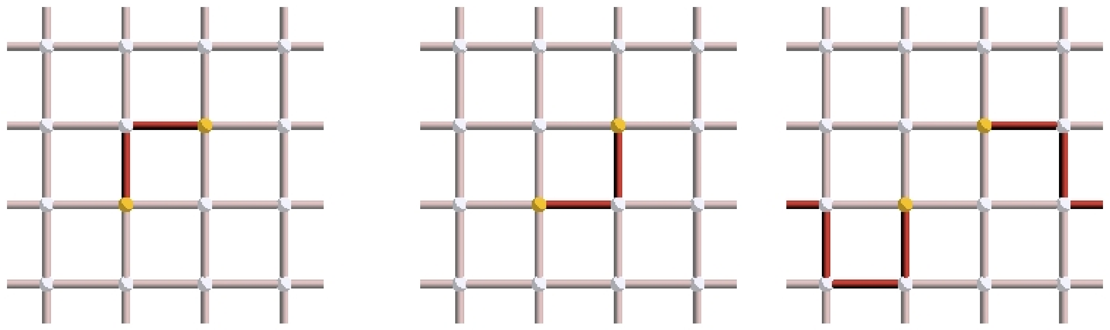


Figure 6: Example

Currently research is being conducted in optimizing techniques for decoding the error syndrome and mapping it to correct recovery operation. Two of the leading contenders are Maximum Likelihood Decoding and Minimum Weight Perfect Matching.They both work by extracting information from what we have, error syndrome and prior probabilities. MLD works by finding set of most logical paths between the two defects as any coset of original error would resolve it, whereas MWPM works by assigning weight to different vertices in the lattice and then choosing the path that minimizes this weight.

Further research is being conducted in optimizing these algorithms and surface codes have become a very promising candidate in race towards practical quantum computers.

5 References

1.Dennis, Eric, et al. “Topological Quantum Memory.” Journal of Mathematical Physics, vol. 43, no. 9, Sept. 2002, pp. 4452–4505, https://doi.org/10.1063/1.1499754.

2. Fowler, Austin G., et al. “Surface Codes: Towards Practical Large-Scale Quantum Computation.” Physical Review A, vol. 86, no. 3, 18 Sept. 2012, p. 032324, arxiv.org/abs/1208.0928, https://doi.org/10.1103/PhysRevA.86.032324.

3.Gaspari, Andrea. Quantum Error Correction and the Toric Code Relatore: Dott. Davide Vodola Presentata Da.

4. Google Quantum AI. Quantum Error Correction below the Surface Code Threshold Google Quantum AI and Collaborators. 27 Aug. 2024.

5. Lidar, Daniel A. “Lecture Notes on the Theory of Open Quantum Systems.” ArXiv.org, 2019, arxiv.org/abs/1902.00967. 21 Feb. 2020.