Surface code related paper survey

Group 28

Abstract

Today, Quantum computers have high error rates compared to the classical computer. In order to make quantum computers useful, error rates have to be as low as 1 in a trillion. While a typical transistor in a microprocessor can run for about a billion years at a billion operations per second, without ever suffering a hardware fault due to any form of interference. A huge improvement in performance is needed, since the typical quantum bits become randomized in about one one-thousandth of a second. Classical error correction employ redundacy, however, it is impossible for quantum code due to no-cloning theorem. In this review, we will summarize one of the promising method – surface code. Besides, the novel improvement of the algorithm and decoder will also be provided.

Keywords: Quantum Computation, Quantum Error Correction, Surface Code, Stabilizer codes

1.0 Introduction

Quantum computing is a rapidly developing field, and its future is full of exciting possibilities. With breakthroughs from various promising teams, quantum computers are fast approaching the point where they can start to benefit our society. However, in order to fully explore the potential of quantum computers, complex design flows have to be carefully designed and employed to boost the accuracy and efficiency of quantum computing development. Among the different methodologies, error correction is a critical component in ensuring the reliable operation of quantum computers. One of the most prominent error correction schemes is the surface code, which has garnered significant attention due to its robustness and scalability. Surface codes, a class of topological quantum codes, play a vital role in protecting quantum information from decoherence and operational errors. The surface code utilizes a two-dimensional lattice of qubits with stabilizers to detect and correct errors. Recent advancements have focused on improving the efficiency and performance of surface code decoders, stabilizers, and associated algorithms, which are pivotal for practical quantum computing. Generally, implementing and improving surface codes is a complex task. Unlike classical error correction, quantum error correction deals with qubits that exist in superpositions, making the detection and correction of errors more challenging. The design and optimization of decoders and stabilizers, which are responsible for identifying and rectifying errors, are crucial for enhancing the fidelity of quantum computations.

In Section II, we provide a comprehensive review of the surface code, including its foundational principles and its importance in quantum error correction. In Section III, we delve into recent improvements in surface code decoders, examining various approaches to enhance their accuracy and efficiency. Section IV discusses advancements in stabilizer codes and their role in the implementation of the surface code. Finally, in Section V, we explore algorithmic developments that leverage the surface code, highlighting their potential impact on the future of quantum computing.

1.1 From classical to quantum error correction

1.1.1 Classical error correction

The basic principle behind classical error correction is to increase the bits used to encode information. The exact method to introduce redundancy and retrieve information is known as a correction code. A simple example is the three-bit repetition code: the bit 0 is encoded into three bits as 000, and the bit 1 is encoded into 111, the encoded bit strings are referred to as codewords.

If the message is subject to a single-bit error, we can still retrieve the correct information via a majority vote. However, it is easy to see that this code is vulnerable to two or more bit-flip errors.

The distance d of a code is the minimum number of errors to transform a codeword into another, and its relationship to the number of errors the code can correct t is given by $d = 2t + 1$. Error correction codes are described in terms of the $[n, k, d]$ notation, where n is the total number of bits per codeword, k is the number of encoded bits and d is the code distance.

1.1.2 Classical bits to qubits

Qubits have some distinct characteristics different from classical bits. Its values can be visualized as a point on a sphere, namely the Bloch sphere, and can take on infinite numbers of values.

It may seem like qubits are subject to infinite numbers of errors, however, we can decompose any error by expanding it in terms of the Pauli basis.

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$U(\delta\theta, \delta\phi)\,|\psi\rangle = \alpha_I \mathbb{1}\,|\psi\rangle + \alpha_X X\,|\psi\rangle + \alpha_Y Y\,|\psi\rangle + \alpha_Z Z\,|\psi\rangle$$

The error correction process involves performing projective measurements that cause the above superposition to collapse to a subset of its terms, and by checking abnormality in parties, we can find a suitable correction process to restore the original quantum information.

1.1.3 Challenges in quantum error correction

Qubits states are confined to the no-cloning theorem, thus making it impossible for us to add redundancy by simply copying states and using the tensor product. The second complication in quantum coding arises from the fact that qubits are susceptible to both bit-flips (X-errors) and phase-flips (Z-errors), and we need to design codes to detect both errors simultaneously. The last challenge in quantum error correction is the fact that quantum states collapse after measurement, thus the error correction procedure must be carefully chosen so as not to cause the wave function to collapse and erase the encoded information.

1.2 Quantum redundancy and stabilizer measurements

In quantum codes, redundancy is added by expanding the Hilbert space in which the qubits are encoded. Taking the two-qubit code as an example, the encoding process is shown below, noting that this process does not disobey the no-cloning theorem:

$$\begin{aligned} |\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \quad &\xrightarrow{\text{two-qubit encoder}} \quad |\psi\rangle_L \\ = \alpha\,|00\rangle + \beta\,|11\rangle &= \alpha\,|0\rangle_L + \beta\,|1\rangle_L\,, \end{aligned}$$

After encoding, the logical qubit occupies a four-dimensional Hilbert space

$$|\psi\rangle_L \in \mathcal{H}_4 = \text{span}\{|00\rangle\,, |01\rangle\,, |10\rangle\,, |11\rangle\}$$

More specifically, the logical qubits are defined within a two-dimensional subspace called the codespace, and bit-flip errors rotate the state into another subspace called the error subspace.

$$|\psi\rangle_L \in C = \text{span}\{|00\rangle\,, |11\rangle\} \subset \mathcal{H}_4$$

$$X_1\,|\psi\rangle_L \in \mathcal{F} \subset \mathcal{H}_4$$

To differentiate between the codespace and the error space, a projective measurement of Z_1Z_2 is performed. The Z_1Z_2 operator yields a (+1) eigenvalue when applied to the logical state, and is said to stabilize the logical qubit. On the other hand, The Z_1Z_2 operator yields a (-1) eigenvalue when applied to errored states. Note that the coefficients stay undisturbed during this parity check.

$$Z_1Z_2\,|\psi\rangle_L = Z_1Z_2(\alpha\,|00\rangle + \beta\,|11\rangle) = (+1)\,|\psi\rangle_L$$

The figure shows the circuit implementation of the two-qubit code. Following the error stage, an ancilla qubit $|0\rangle_A$ is introduced to measure the Z_1Z_2 stabilizer. The ancilla's outcome is referred to as a syndrome, and we can construct a syndrome table that shows the outcome of the syndrome corresponding to possible errors.

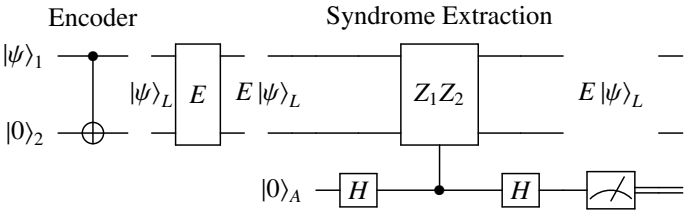


Table 1: The syndrome table for the two-qubit code.

Error	**Syndrome,** S
I_1I_2	0
X_1I_2	1
I_1X_2	1
X_1X_2	0

Note: The syndrome S is a bit string where each bit corresponds to the outcome of a stabiliser measurement.

In this correction scheme, we can conclude that the logical qubit is subject to a bit-flip error if we measure the ancilla and get the syndrome 1. It requires simultaneous bit-flip errors on both qubits such that we do not notice any error, thus suppressing the error rate. Note that we could not determine which qubit had been subject to the bit-flip mistake even if we acknowledge the existence of it, thus more sophisticated codes must be designed to fulfill the purpose of error detection and correction.

1.3 Stabilizer codes and operators

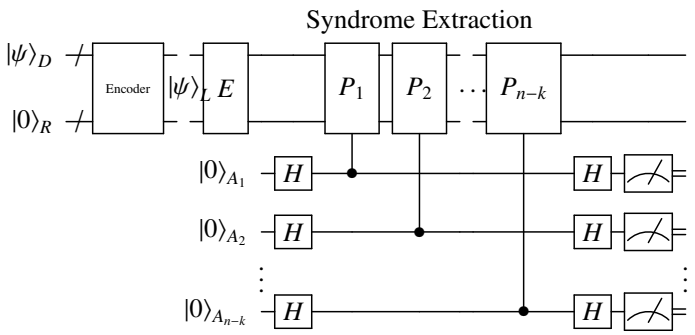
This section briefly discusses the operation of the general $[[n, k, d]]$ stabilizer code, discussing basic concepts sufficient for readers to understand key concepts of recent papers we have surveyed.

The circuit below shows the basic structure of an $[[n, k, d]]$ stabilizer code. A register of k data qubits is entangled with $m = n-k$ redundancy qubits to create a logical qubit, errors can then be detected by performing m stabilizer measurements Π.

Constructing a good code involves finding stabilizers that anti-commute with the errors to be detected so that the presence of errors will be detected via syndrome.

The result of the m stabilizer measurements gives us an m-bit syndrome, we then deduce the best recovery operation to restore the logical state to the codespace using various decoding algorithms, since for large numbers of qubits it is impossible to

enumerate errors for all possible syndromes.



1.3.1 Properties of the stabilizers

The stabilizers Π must satisfy some properties. Firstly, they must stabilize all logical states, which means when the stabilizers act on any logical state, they must return the eigenvalue 1. Secondly, all the stabilizers of a code must commute with one another, this is necessary so that the stabilizers can be measured simultaneously.

1.3.2 Logical operators

An $[[n, k, d]]$ stabilizer code has $2k$ logical Pauli operators that allow for logical states to be modified without having to decode and then re-encode. Each logical operator must commute with all the code stabilizers.

Finding sets of stabilizers and logical operators is not trivial, and needs to be found with effort.

1.3.3 Quantum error correction with stabilizer codes

The figure below shows the general error correction procedure for a single cycle of an $[[n, k, d \geq 3]]$ stabilizer code.

Circuit here
Use the qcircuit package to draw the circuit

After the logical qubit is subject to an error, stabilizer measurements are performed to produce an m-bit syndrome $\mathcal{S}$. The next step, referred to as decoding, involves processing the syndrome to determine the best unitary operation $\mathcal{R}$ to return the logical state to the codespace. The decoding step is successful if the combined action of $\mathcal{R}E$ on the code state is as follows

$$\mathcal{R}E\,|\psi\rangle_L = (+1)\,|\psi\rangle_L$$

The decoding step fails if the recovery operation maps the code state as follows

$$\mathcal{R}E\,|\psi\rangle_L = L\,|\psi\rangle_L$$

The recovery process rotates the qubit into the codespace but leads to a change in the encoded information.

1.4 Surface code

Commuting sets of stabilizers enable the ability to detect different errors simultaneously without disturbing information. Finding such sets is non-trivial, and special codes have to be designed.

The general design principle behind the surface code is that the code is built up by patching repeated elements. This approach ensures that the surface code can be straightforwardly scaled whilst ensuring stabilizer commutativity. One advantage of the surface code is that it requires only nearest-neighbor interactions, as high-fidelity long-range interactions are difficult to maintain.

1.4.1 The surface code four-cycle

The basic element in the surface code is shown in the figure below, squares represent ancilla qubits, while circles represent data qubits. Black edges represent controlled X gates, each controlled on an ancilla qubit A and acting on a data qubit D. Likewise, the dashed edges represent controlled-Z operations, each controlled by an ancilla qubit and acting on a data qubit. For example, ancilla A_1 measures the stabilizer XD_1XD_2.

Circuit here
Use the qcircuit package to draw the circuit

The surface code four-cycle is not useful itself as it encodes $k = n - m = 0$ qubits, but working detection and correction codes can be formed by tiling together multiple four-cycles to form square lattices.

1.4.2 The [[5, 1, 2]] surface code

The figure below shows the five-qubit surface code formed by tiling four four-cycles in a square.

Circuit here
Use the qcircuit package to draw the circuit

The stabilizers can be read off to give

The logical operators of the surface code can be defined as chains of Pauli operators along the edges of the boundaries of the surface code. For example, XD_1XD_4 and ZD_1ZD_2 are both valid operators.

Circuit here
Use the qcircuit package to draw the circuit

From the above we see that the minimum weight of the logical operators is 2, meaning the $[[5, 1, 2]]$ code is a detection code with $d = 2$.

1.4.3 Scaling

The distance of a surface code can be increased simply by scaling the size of the lattice. The figure shown below is the $[[13, 1, 3]]$ surface code, stabilizers and operators can be read off just by inspection.

Circuit here
Use the qcircuit package to draw the circuit

1.5 Practical considerations

1.5.1 Decoding algorithms

Given a code syndrome, the role of the decoder is to find the best recovery operation to restore the quantum information into the codespace. For small code examples, it is possible to compute the lookup table for all combination of errors. However, as the code size increases, the number of possible syndromes grows exponentially, and it becomes impractical to use such decoding strategy.

In place of lookup tables, decoding algorithms are developed to decode syndromes. For surface codes, one technique known as minimum weight perfect matching (MWPM) can be used for decoding, which works by identifying error chains between positive syndrome measurements.

For these approximate inference techniques, the logical error rate of a quantum error correction code will depend heavily on the decoder used, and the efficiency of the decoder is also a topic that researchers opt to optimize.

1.5.2 Experimental implementation

The experimental implementation of the surface code represents a significant milestone in quantum computing, aiming to achieve fault-tolerant logical qubits. Researchers at institutions like Google, IBM, and TU Delft are at the forefront, developing superconducting devices to realize this goal. The surface code known for its high threshold for error rates under realistic noise conditions, demands sophisticated hardware capable of maintaining low error rates.

1.5.3 Fault tolerant

In the part where we introduce basic working principles of the surface code, we have assumed that errors happen in certain parts of the circuit. In realistic environments, however, this is not the case as errors could also appear at two-qubit gates or stabiliser measurements

A fault-tolerant code is a code that could handle errors up to the code distance occurring at any location in the circuit. This involves modifying quantum circuits, which increases overhead by requiring additional ancilla qubits.

2.0 Review of the surface code

section2

3.0 Decoder

section3

4.0 Stabilizer

section4

5.0 Algorithm developments

section5