Quantum Error Correction with Stabilizer Codes

River McCubbin

December 12, 2024

1 Abstract

Quantum computing in the modern sense originates around 1980, with a number of important theoretical results published between 1985 and 2000. These results, such as Shor's algorithm for integer factorization, and Grover's search algorithm provide significant speedups from their classical counterparts. Since then, the difficulty has been to construct a quantum computer sufficiently large to use the algorithms in practice. One key factor to enabling these algorithms is to adequately correct errors that accumulate during quantum computation.

This report presents stabilizer codes - quantum error correcting codes based in subspace correction. We discuss bit-flip and phase flip errors, justify that correcting these errors is sufficient for correcting arbitray errors, and provide codes capable of correcting each of these. Finally, we provide Qiskit implementations of these codes that verify their correctness and feasibility.

2 Quantum Computing

Classical computing is built from a foundation of bits and logic gates. We use bits to represent data, and gates to manipulate that data in order to reach conclusions. The systems of quantum computing are quite similar. We represent data via qubits:

Definition 1 (Qubit)**.** A *qubit* is a unit vector $|\psi\rangle \in \mathbb{C}^2$.

We can more accurately describe an arbitrary qubit with reference to the computational basis:

Notation 1 (Computational Basis)**.** The *computational basis* of $\mathbb{C}^n$ is denoted $\{|0\rangle, \dots, |n-1\rangle\}$ where

$$|i\rangle = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{bmatrix}^T$$

with a 1 in the $i-1$th position and 0 elsewhere.
Note: Sometimes we write $|3\rangle$ to represent the 4th basis vector, but we often write $|11\rangle$ to represent the fourth basis vector in binary instead.

For example the computational basis of $\mathbb{C}^2$ is given by

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

With this basis we can represent an arbitrary qubit as a linear combination of these coefficients:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle .$$

While a single qubit is useful, in order to represent more complex data we'll need a way to combine multiple qubits. This is done via the tensor product:

Definition 2 (Tensor)**.** The tensor product of vectors (not necessarily single qubits) $|\psi\rangle \in \mathbb{C}^m$, $|\phi\rangle \in \mathbb{C}^n$ is denoted

$$|\psi\rangle \otimes |\phi\rangle \in \mathbb{C}^{mn}.$$

It is a bilinear map $\otimes : \mathbb{C}^m \otimes \mathbb{C}^n \to \mathbb{C}^{mn}$, i.e.

$$(\alpha |\psi\rangle) \otimes |\phi\rangle = \alpha(|\psi\rangle \otimes |\phi\rangle) = |\psi\rangle \otimes (\alpha |\phi\rangle)$$

$$(|\psi\rangle + |\phi\rangle) \otimes |\gamma\rangle = |\psi\rangle \otimes |\gamma\rangle + |\phi\rangle \otimes |\gamma\rangle$$

and

$$|\psi\rangle \otimes |\phi\rangle = |\phi\rangle \otimes |\psi\rangle .$$

Since the tensor product is a bilinear map, we can specify its action on the basis elements, which defines the action on the entire space by linearity. For example the tensor of basis elements of $\mathbb{C}^2$ provides a basis for $\mathbb{C}^4$:

$$\begin{aligned} |0\rangle \otimes |0\rangle &=: |00\rangle & |0\rangle \otimes |1\rangle &=: |01\rangle \\ |1\rangle \otimes |0\rangle &=: |10\rangle & |1\rangle \otimes |1\rangle &=: |11\rangle \end{aligned}$$

Referring back, we can see that this is the notation for the computational basis of $\mathbb{C}^4$.

With this method of representing data, we move on to operations that can be applied to the data. As mentioned previously, classical computing operations consist of combinations of logical gates. The quantum analogue of these gates are called quantum gates, and are modelled mathematically by unitary operators:

Definition 3 (Unitary Operator)**.** A *unitary operator* U is a linear operator that satisfies

$$U^*U = UU^* = I$$

where U^* denotes the adjoint (conjugate transpose in our context) of U.

For example, the quantum NOT gate is denoted X and acts by

$$X |0\rangle = |1\rangle \qquad\qquad X |1\rangle = |0\rangle$$

Notice that since these are linear operators on a vector space with a specified basis, there exist matrix representations for these operators. Continuing with the X gate example, we have matrix representation

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

which allows us to very simply check that $X = X^*$ and hence $XX^* = X^*X = X^2 = I$.

Notice that since quantum gates are linear operators, they are determined by their action on the basis elements and hence we typically specify only this. Other common quantum gates are listed in the appendix. Before continuing, a reader should be familiar with the Z gate and the CNOT gate at the very least.

Similarly to the way we combined qubits with the tensor product, we can combine operators with the tensor product as well. For example, if we want to apply an X gate to the second qubit only, we can apply $I \otimes X$:

$$\begin{aligned}(I \otimes X)(|0\rangle \otimes |0\rangle) &= I\,|0\rangle \otimes X\,|0\rangle \\ &= |0\rangle \otimes |1\rangle\end{aligned}$$

Some gates are representable as a tensor product of single qubit gates, but some are not representible in this way. One example of this is the CNOT (controlled-not) gate.

Though an arbitrary gate may not be a pure tensor product of single qubit gates, it must be a linear combination of I, X, Z, XZ gates. These gates form a basis for the space of single qubit operators, and hence linear combinations of their tensors forms a basis for any space of n-qubit gates. This will be a key observation that informs how we approach error correction.

3 Quantum Errors

In classical computation, errors can occur only as "bit-flips." In quantum computing, there is an analagous error:

Definition 4 (Bit-Flip Error)**.** A *bit-flip error* on a general qubit $|\psi\rangle$ is given by

$$\begin{aligned}X\,|\psi\rangle &= X(\alpha\,|0\rangle + \beta\,|1\rangle) \\ &= \alpha X\,|0\rangle + \beta X\,|1\rangle \\ &= \alpha\,|1\rangle + \beta\,|0\rangle\end{aligned}$$

Unfortunately, this is not the only type of error in quantum computing. The second type of error is the phase flip error:

Definition 5 (Phase Flip Error)**.** A *phase flip error* on a general qubit $|\psi\rangle$ is given by

$$\begin{aligned}Z\,|\psi\rangle &= Z(\alpha\,|0\rangle + \beta\,|1\rangle) \\ &= \alpha Z\,|0\rangle + \beta Z\,|1\rangle \\ &= \alpha\,|0\rangle - \beta\,|1\rangle\end{aligned}$$

An arbitrary error in quantum computing may appear different from these. Since any interaction with a quantum state can be modelled by applications of unitary operators, for which I, X, Z form a basis, it is possible to represent an arbitrary error as a linear combination of these errors. Hence any error can be corrected by correcting bit and phase flip errors [4].

In addition to two types of errors, there is an additional challenge to quantum error correction.

Theorem 1 (No Cloning Theorem)**.** *There does not exist a unitary operator U such that*

$$U(|\psi\rangle \otimes |0\rangle) = |\psi\rangle \otimes |\psi\rangle\,.$$

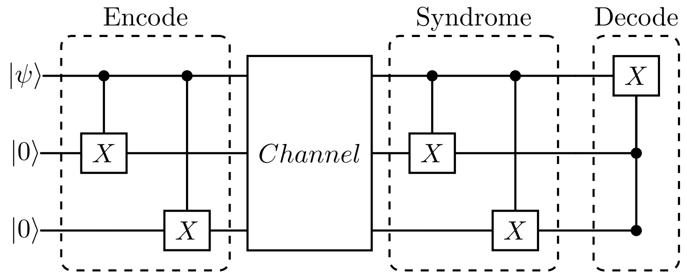
This was shown by James Park in 1970 [5]. This result means that classical techniques that involve the duplication of data (ex. repetition codes) are not possible to implement in a quantum setting. Despite that, as we will see in the following section, it is still possible to preserve protect quantum information over noisy channels.

4 Stabilizer Codes

With the types of errors we could encounter identified we can now look for methods of detecting and correcting these errors. The primary approach to quantum error correction is via stabilizer codes. These codes are in a sense the quantum analogue of the repetition code, aiming to preserve quantum information by sharing it across multiple qubits, whose errors can be individually corrected in order to reconstruct the original state. The name "stabilizer" comes from viewing the set of possible errors as a group generated by products of the X, Z Pauli gates. The subgroup that stabilizes the codespace $\mathcal{C} = \text{span}\{|0^n\rangle, |1^n\rangle\}$ defines the set of encoding and decoding operations. For each of the upcoming codes there is an implementation in Qiskit available in the appendix to demonstrate correctness and ease understanding.

4.1 Bit-Flip Code

The following circuit demonstrates the bit-flip code, which is capable of correcting 1 bit-flip error



The Encode segment of the above circuit takes an arbitrary state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

to the state

$$\alpha|000\rangle + \beta|111\rangle.$$

In the same way that a classical code shares information across additional bits, this shares the quantum information from the original state across three qubits.

After encoding, the state is sent through the channel which may introduce bit flip errors. We assume that the likelihood of more than one bit flip is negligible and analyze only the cases with 0 or 1 errors.

We then detect the syndrome of the resulting state. We can examine the state by cases:

Channel	State After Channel	State After Syndrome	Syndrome
$I \otimes I \otimes I$	$\alpha|000\rangle + \beta|111\rangle$	$\alpha|000\rangle + \beta|100\rangle$	00
$X \otimes I \otimes I$	$\alpha|100\rangle + \beta|011\rangle$	$\alpha|111\rangle + \beta|011\rangle$	11
$I \otimes X \otimes I$	$\alpha|010\rangle + \beta|101\rangle$	$\alpha|010\rangle + \beta|110\rangle$	10
$I \otimes I \otimes X$	$\alpha|001\rangle + \beta|110\rangle$	$\alpha|001\rangle + \beta|101\rangle$	01

The syndromes here are simply the states of the second and third qubits after applying the syndrome circuit.

We can then use these syndromes to determine the necessary correction. We have accurately detected errors in each state with non-zero syndrome. Notice though that only the state with syndrome 11 requires correction; in each other state we have that the first qubit (i.e. the state we wished to preserve) is in the correct state $\alpha|0\rangle + \beta|1\rangle$. Hence we decode by applying a doubly-controlled X gate to the first qubit. This ensures that we return to the original state of the first qubit: if the syndrome is 11 then we apply the X gate, resulting in the state

$$\alpha|011\rangle + \beta|111\rangle$$

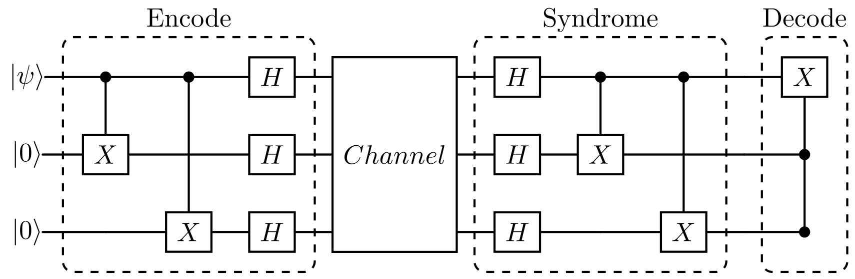
and in every other case nothing is done. After applying this operation the first qubit has returned to the sent state, and the second and third qubits match with both coefficients, and hence we can factor the state as

$$(\alpha|0\rangle + \beta|1\rangle) \otimes |q_2 q_3\rangle,$$

recovering our original state, as wanted.

4.2 Phase-Flip Code

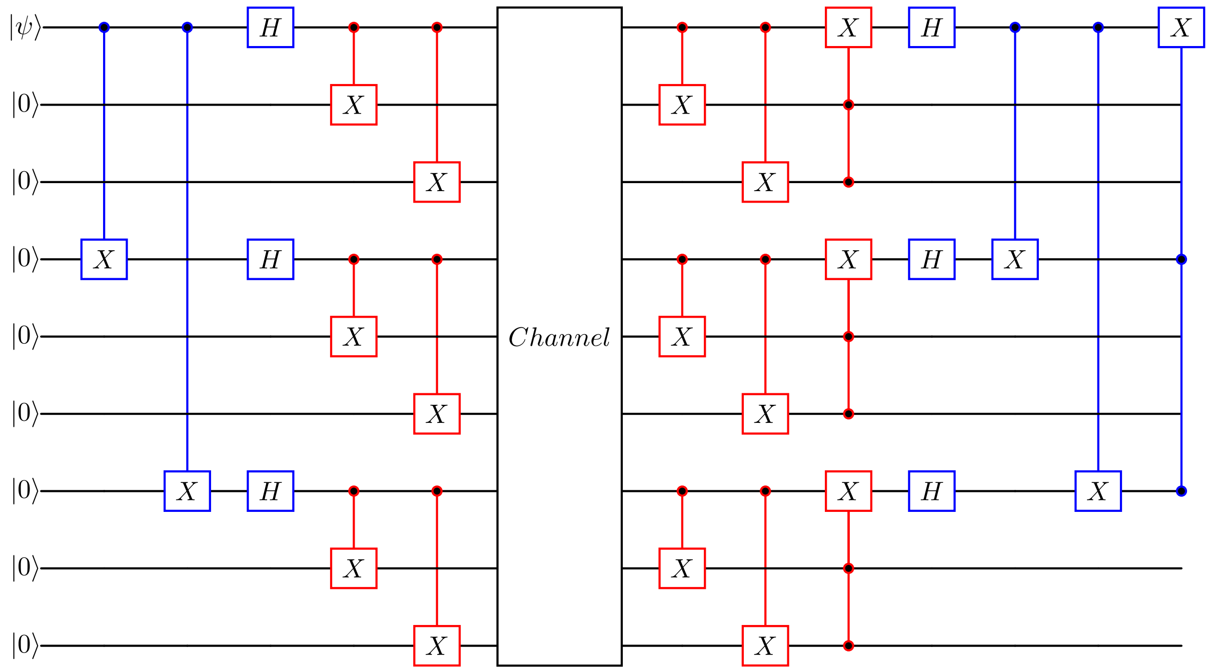
Having corrected a bit-flip error, we continue with the phase-flip error. Luckily, the method for correcting a phase-flip error is informed by our success with bit-flip errors. We find that the following circuit successfully corrects up to 1 phase-flip error:



The key here is the application of the Hadamard gate and the identity $HZH = X$. If a phase-flip error occurs in the channel, then the Hadamard gates applied on either side of the channel allow us to correct this phase flip error as though it were a bit-flip error. At the same time, if no error occurs then we have $HH = I$, leaving the state unchanged.

4.3 Shor Code

We can effectively combine the bit-flip code and the phase-flip code in order to correct an arbitrary error. The following code that accomplishes this was the first of its kind, and was introduced by Peter Shor in 1995 [7]. The code is shown below as a quantum circuit:



Notice that this acts as a sort of composition of the bit-flip and phase-flip codes: each block of three qubits implements the bit-flip code (red), and every third qubit implements the phase-flip code (blue). In this way, a single arbitrary error can be corrected by the Shor code.

5 Conclusion

With the increasing promise of practical quantum computation, the need for more effective quantum error correcting techniques is growing. Stabilizer codes provide the foundation for future work on quantum error correction, allowing the correction of arbitrary errors at the cost of a number of additional qubits. The challenges moving forward will be to increase the efficiency of these codes, reducing the number of additional qubits necessary to correct.

6 Bibliography

References

[1] Jason Crann. *MATH 5821 Lecture Notes*. 2024.

[2] Craig Gidney. *Quirk*. `https://github.com/Strilanc/Quirk`.

[3] Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. 1997. arXiv: `quant-ph/9705052 [quant-ph]`. URL: `https://arxiv.org/abs/quant-ph/9705052`.

[4] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[5] James L. Park. "The concept of transition in quantum mechanics". In: *Foundations of Physics* 1.1 (1970), pp. 23–33. DOI: `10.1007/BF00708652`. URL: `https://doi.org/10.1007/BF00708652`.

[6] Joschka Roffe. "Quantum error correction: an introductory guide". In: *Contemporary Physics* 60.3 (July 2019), pp. 226–245. ISSN: 1366-5812. DOI: `10.1080/00107514.2019.1667078`. URL: `http://dx.doi.org/10.1080/00107514.2019.1667078`.

[7] Peter W. Shor. "Scheme for reducing decoherence in quantum computer memory". In: *Phys. Rev. A* 52 (4 Oct. 1995), R2493–R2496. DOI: `10.1103/PhysRevA.52.R2493`. URL: `https://link.aps.org/doi/10.1103/PhysRevA.52.R2493`.

7 Appendix

7.1 Common Quantum Gates

Gate	Matrix	Action on zero state(s)	Action on 1 state(s)
X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$X\|0\rangle = \|1\rangle$	$X\|1\rangle = \|0\rangle$
Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	$Y\|0\rangle = i\|1\rangle$	$Y\|1\rangle = -i\|0\rangle$
Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$Z\|0\rangle = \|0\rangle$	$Z\|1\rangle = -\|1\rangle$
H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$H\|0\rangle = \|+\rangle := \frac{1}{\sqrt{2}}(\|0\rangle + \|1\rangle)$	$H\|1\rangle = \|-\rangle := \frac{1}{\sqrt{2}}(\|0\rangle - \|1\rangle)$
CNOT	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\text{CNOT}\|00\rangle = \|01\rangle$ $\text{CNOT}\|01\rangle = \|01\rangle$	$\text{CNOT}\|10\rangle = \|11\rangle$ $\text{CNOT}\|11\rangle = \|10\rangle$

7.2 Qiskit Implementation

Available at Github/kipawaa/Quantum-Error-Correction