Study of Quantum Error Correction on IBM Quantum Platform

Amrita[1*]

ABSTRACT

The effective and safe communication of information using quantum systems is one of the important facets of ongoing Quantum technology. In the quantum information theory, a quantum channel is used which is based on the phenomena of entanglement and superposition. The entangled pairs retain a non-local correlation between them making them useful as quantum channel. The information transfer involves key distribution scheme in which data is encoded and sent through public channel. A private key is shared amongst the two parties using which they can retrieve the information. The channel cannot be purely isolated and noise free. So, there occurs a change in the encoded qubits which needs to be corrected. In order to identify and mitigate the errors, the error correction codes are required.

With the help of quantum computing on real quantum systems, the quantum phenomenon, quantum algorithms and correction codes can be designed. IBM offers its Quantum Information Science toolkit known as qiskit which is an open cloud based sdk. The advanced problems with classical computers with particular algorithms cannot be solved more efficiently and quickly, but it can be done with the help of quantum computers. The paper is about the design of error correction schemes on qiskit and their implementation on openly accessible quantum computers.

Keywords : *Error correction, Information theory, Quantum channel, Qiskit.*

1. Introduction

The secure and error free transfer of information is important in the information processing and communication theory. The entangled pair of photons can be used as a quantum channel in the communication. In the initial stage of quantum communication theory, the systems as well as channel were taken to be isolated and dissipation free. But there is a possibility of noise in actual physical systems which are used to encode and transmit data in the form of qubits. If the channel is noisy, there are methods to measure the errors occuring in the channel mathematically.

The quantum error correction schemes involves the encoding, error detection and decoding of the qubits. There are a number of researchers working on error correction schemes as it's an important issue of concern. In a recent paper [1], phase flip and bit flip errors were estimated on IBM Quantum platform. It is a cloud-based platform in which user can access and work on a number of Quantum hardwares situated at various parts of the world. The largest real world quantum computers are able to handle 100-200 noisy qubits. Error mitigation constitutes a class of promising techniques using which the computational errors can be reduced to a minimum[2].

With the help of quantum neural networks, in the form of quantum autoencoders active detection and correction of errors, including spatially correlated computational errors as well as qubit loss can be done [3]. The inherent symmetry of the physical system can effect the error correction as a constraint. An approximate QEC with covariant symmetry and fidelity of the system was studied in work by Dai [4]. A number of techniques for Quantum error corrections have been discussed , such as probabilistic QEC[5], gauge symmetry [6],circuit engineering [7], IBM computing [1]. In the present paper, I have tried to discuss the repetitive method and circuits for noise detection, encoding and decoding of the information.

2. Repitition Method of Error Detection

Bit-flip error codes are a type of quantum error correction code used to protect quantum information against errors that flip the values of individual qubits (i.e., change 0 to 1 or vice versa). The most well-known bit-flip code is the 3-qubit repetition code which can be used to correct bit-flip errors. Initially, the qubit is encoded by creating multiple copies (usually three) of the same quantum state. For example, if you want to encode the logical qubit state |0>, you create three physical qubits in the |0> state. The next step is to take into consideration the different types and means of errors, during quantum operations or due to external factors. Bit-flip errors might flip some of the qubits, changing |0> to |1> or vice versa. The detection

1. *Assistant Professor, Department of Physics, Patna Women's College, Patna.*
* *Corresponding Author ✉ amritaphy@gmail.com*

Received: 28 December, 2023
Available online: 30 March, 2024

method involves comparing the values of the three qubits. If two or more qubits agree, it's likely that the majority value is correct. Then the minority value can be flipped to match the majorit to corrects the bit-flip errors. Finally, the qubits were measured to retrieve the corrected logical qubit.

The simplest type of errors is independent single bit random errors that have the same probability for every bit value. The probability of a bit flip error in each bit can be taken as p, which is same for both 0 and1. So, the probability that bit value remains unchanged is 1-p. Another little complicated case is that in which the bit flip error probabilities are not the same for different bit values. i. e. for 1, the probability equals p_1 and for 1, the probability equals p_2. Then the probabilities of no errors are $(1-p_1)$ and $(1-p_2)$ for 0 and 1 respectively [8,9].

The most general case includes multibit or correlated errors, meaning uncontrollable changes in several bits at once. Such errors are schematically depicted by a certain logical elements. The action of such a logical element can be represented as a set of certain logical operations, which can be both single bit or multibit i.e. each operation ξ_i has probability p_i. If we have a model of errors, then we can choose encoding methods that will make the information more resistant to this type of errors.

The information to be sent or stored is converted into larger qubits by multiples copies of the same so as to protect it from noise. A qubit can have two possible states |0> or |1>, so the encoded data can be |000> or |111>. There are two possible errors, one is the gate error which can occur due to a wrong operation and other is the measurement error which can flip the data bits. In order to design a python code for this, let us consider the probability of each noise type and create a function which measures them.[10, 11]

The simplest change that can occur is the bit flip or phase flip which can be considered as operation of Pauli I, X, Y and Z operators on the data qubit. pauli_error is a function from Noise Model which shows the change of state due to any of the pauli operators whereas depolarizing _error implements the same in all the encoded qubits. p_meas and p_ gate represent the probability of errors. The errors which can change the state is corrected by applying a X gate and then a controlled X gate [12].

```
from qiskit.providers.aer.noise import NoiseModel
from qiskit.providers.aer.noise.errors import pauli_error, depolarizing_error

def get_noise(p_meas,p_gate):

    error_meas = pauli_error([('X',p_meas), ('I', 1 - p_meas)])
    error_gate1 = depolarizing_error(p_gate, 1)
    error_gate2 = error_gate1.tensor(error_gate1)

    noise_model = NoiseModel()
    noise_model.add_all_qubit_quantum_error(error_meas, "measure")
    noise_model.add_all_qubit_quantum_error(error_gate1, ["x"])
    noise_model.add_all_qubit_quantum_error(error_gate2, ["cx"])
    return noise_model
```

The different noise values can be put in the model and then the measurement of the encoded qubits can be done. In the code below, both the errors are taken to be 2 percent and the quantum circuit of 3 qubits with initial |000> is measured 1024 times. The result shows the different possible states and their counts, the encoded data qubit has the largest possibility of occurrence. This is a repetitive method of detecting and measuring the data qubits.

```
noise_model = get_noise(0.02,0.02)
from qiskit import QuantumCircuit, execute, Aer
qc1 = QuantumCircuit(3,3,name='0')
qc1.measure(qc1.qregs[0],qc1.cregs[0])
counts=execute(qc1,Aer.get_backend('qasm_simulator'),noise_model=noise_model).result().get_counts(
print(counts)
```

```
{'001': 16, '100': 19, '010': 23, '000': 966}
```

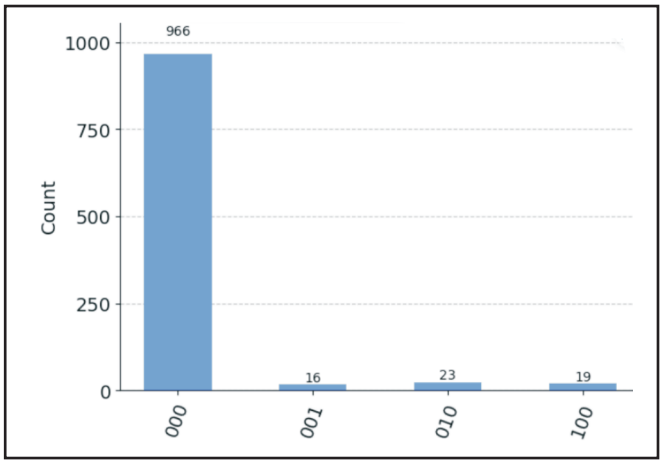


Figure 1 : *Histogram of probability of possible states when noise percentage is low*

If the error considered is high then the encoded qubit would be difficult to obtain. Let us consider the case when p_gate is 10 percent and p_meas is 50 percent in the noise model and apply on the encoded qubit |111>. This is observed from the histogram of all the possible states and their probability.

```
noise_model = get_noise(0.5,0.1)
from qiskit import QuantumCircuit, execute, Aer
qc1 = QuantumCircuit(3,3,name='0')
qc1.x(qc1.qregs[0])
qc1.measure(qc1.qregs[0],qc1.cregs[0])
counts=execute(qc1,Aer.get_backend('qasm_simulator'),noise_model=noise_model).result().get_counts()
print(counts)
from qiskit.visualization import plot_histogram
plot_histogram(counts)
```

```
{'101': 131, '001': 140, '100': 137, '011': 119, '111': 123, '110': 115, '010': 124, '000': 135}
```

Thus noise results in complete change of the expected state at the receiver's end.

When the qubits need to be stored then they can also interact amongst themselves and the surroundings, resulting in change of their states. In order to keep track of

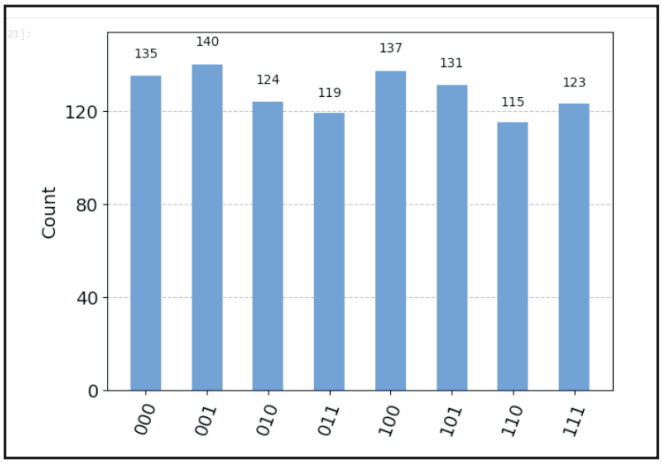
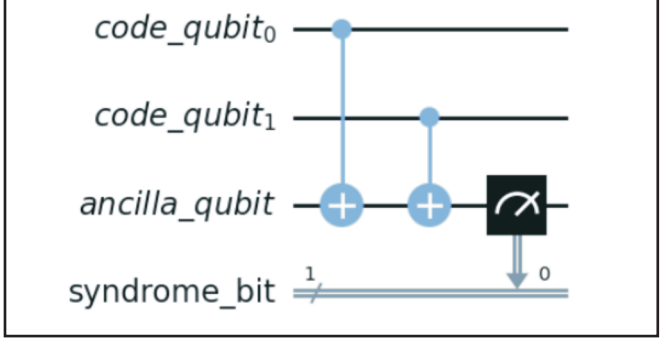


Figure 2 : *Histogram of probability of possible states when noise percentage is high*

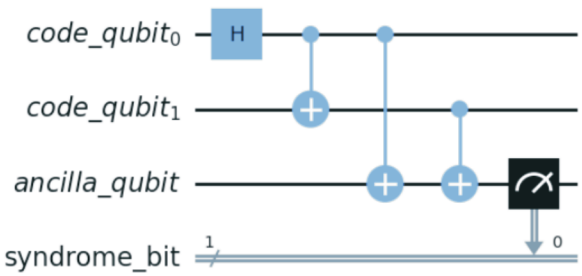
the states , we can use one or two ancilla qubits and two syndrome bits for making measurement of the encoded qubits. For the simplest case, consider two data qubits and one ancilla qubit which is in state |0>. Both the data qubits serve as control for the ancilla qubit and the measurement of the same is done on syndrome bit. This can be done in a quantum circuit as follows:

```
from qiskit import QuantumRegister, ClassicalRegister

cq = QuantumRegister(2, 'code_qubit')
lq = QuantumRegister(1, 'ancilla_qubit')
sb = ClassicalRegister(1, 'syndrome_bit')
qc = QuantumCircuit(cq, lq, sb)
qc.cx(cq[0], lq[0])
qc.cx(cq[1], lq[0])
qc.measure(lq, sb)
qc.draw()
```



In the circuit, if both the code qubits are in |0> then the syndrome bit will give the result 0 as the ancilla qubit will retain its state. If we initialize the code qubits to |1> by applying X gate, then also the measurement will give 0. If one of the code qubits is set to superposition state by applying H gate and CNOT is applied between the two code qubits then they are entangled to one another. The first code qubit state will be |0>+|1> / √2 and the second code qubit will be |00>+|11> / √2. As the ancilla qubit is controlled by the two code qubits it changes its state twice, its final measurement will give 0. If another X gate is applied on first code qubit after CNOT, then the output will be 1 which shows the presence of error.

```
qc_init = QuantumCircuit(cq, lq, sb)
qc_init.h(cq[0])
qc_init.cx(cq[0], cq[1])

qc_init.compose(qc).draw()
```




```
counts = execute(qc_init.compose(qc), Aer.get_backend('qasm_simulator')).result().get_counts()
print('Results:', counts)
```

```
Results: {'0': 1024}
```

Out of the four different possible states of code qubits, three are providing a correct result justifying the repetitive code. Github possesses a repetitive code and topological code which performs the repetition by just specifying the number and syndrome bits.

```
from topological_codes import RepetitionCode
from topological_codes import lookuptable_decoding
from topological_codes import GraphDecoder
```

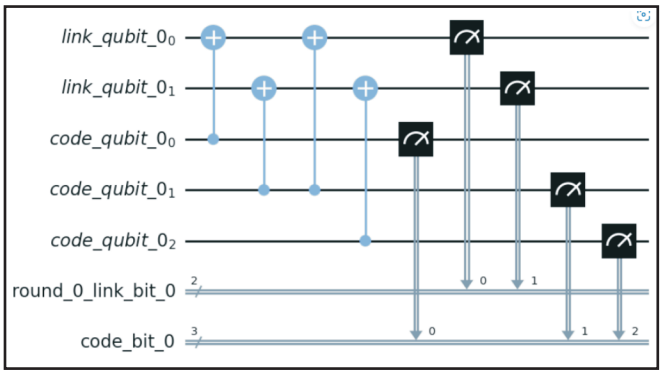
```
from qiskit import QuantumCircuit, execute, Aer
from qiskit.providers.aer import AerSimulator
qubits = 3
circ = QuantumCircuit(qubits, qubits)
circ.h(0)
circ.cx(0, 1)
circ.x(0)
circ.cx(0, 1)
circ.cx(0, 2)
circ.measure(range(qubits), range(qubits))
circ.draw()
simulator = AerSimulator()
result = execute(circ, simulator, shots=1).result()
counts = result.get_counts(circ)
print(counts)
```

```
{'111': 1}
```

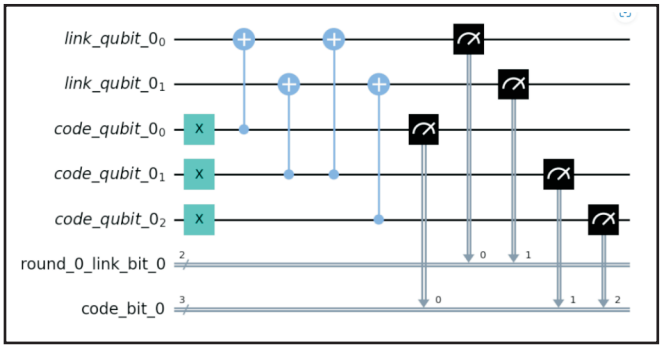
Once the topological and repetition code is installed and imported , the number of repeititions n and the number of syndrome bits T is to be specified. In the circuit below, three data qubits in states |0> and |1> are the targets of two ancillary qubits in pairs and finally all the qubits are measured.

```
n = 3
T = 1

code = RepetitionCode(n, T)
```

```
code.circuit['0'].draw()
```




```
code.circuit['1'].draw()
```




```
def get_raw_results(code,noise_model=None):
    circuits = code.get_circuit_list()
    raw_results = {}
    for log in range(2):
        job = execute( circuits[log], Aer.get_backend('qasm_simulator'), noise_model=noise_model)
        raw_results[str(log)] = job.result().get_counts(str(log))
    return raw_results

raw_results = get_raw_results(code)
for log in raw_results:
    print(f'Logical {log}: {raw_results[log]}')


Logical 0: {'000 00': 1024}
Logical 1: {'111 00': 1024}
```

The output shows the initial qubit states along with the ancillary qubits, when both the circuits are run 1024 times. If the number of syndrome bits is increased to 4, then also the qubit state is recovered alongwith the ancillary qubits.

```
code = RepetitionCode(n, 4)

raw_results = get_raw_results(code)
for log in raw_results:
    print(f'Logical {log}: {raw_results[log]}')


Logical 0: {'000 00 00 00 00': 1024}
Logical 1: {'111 00 00 00 00': 1024}
```

```
code = RepetitionCode(5, 4)

raw_results = get_raw_results(code)
for log in raw_results:
    print(f'Logical {log}: {raw_results[log]}')


Logical 0: {'00000 0000 0000 0000 0000': 1024}
Logical 1: {'11111 0000 0000 0000 0000': 1024}
```

3. Conclusions and Discussion

Qiskit offers a user interface to run circuits of different quantum machines. It include the algorithm design and development for studying entanglement , communication theory, teleportation, quantum key distribution and so on. In the present paper, the method of Bit flip error and its correction is discussed using the qiskit programming. The quantum repetition code and its circuit has been studied using the quantum platform. The noise model and its effect on the qubit states has been shown pictorially. The decoding and measurement of states can be done using the inbuilt topographic codes.

References

1. Sangat Sharma, Suresh Basnet and Raju Khanal, "Implementation of Error correction on IBM quantum computing devices", Journal of Nepal Physical Society, Vol 8, No.2,2022
2. Yifeng Xiong, "Quantum error mitigation for error-resilient quantum computation", University of Southampton, Doctoral Thesis.
3. David F. Locher , Lorenzo Cardarelli , Markus M¨uller , "Quantum Error Correction with Quantum Autoencoders", Quantum , Vol 7, pp942, 2023.
4. Hao Dai, "Approximate quantum error correction, covariance symmetry and their relation" https://doi.org/10.48550/arXiv.2305.02162
5. Ryszard Kukulski, Łukasz Pawela, and Zbigniew Puchała, "On the probabilistic quantum error correction", IEEE Transactions on Information theory, Vol 69, Issue 7, 2023.
6. Abhishek Rajput, Alessandro Roggero and Nathan Wiebe, "Quantum error correction with gauge

symmetries", npj Quantum Information, 9, 41, 2023; https://doi.org/10.1038/s41534-023-00706-8.

7. Muhammad Ahsan, Syed Abbas Zilqurnain Naqvi, and Haider Anwer, "Quantum Circuit Engineering for Correcting Coherent Noise", Phys. Rev. A 105, 022428, 2022.
8. Kang Xiao, "Review of Quantum Computer Development", Highlights in Science, Engineering and Technology, Vol 34, 2023.
9. Belal Ehsan Baaquie & Leong ChuanKwek, "Quantum Error correction", Quantum Computers – Theory and Algorithms , Springer Pub, 2023.
10. Suguru Endo, Simon C. Benjamin, and Ying Li; Practical Quantum Error Mitigation for Near-Future Applications. Phys.Rev. X 7, 021050 (2017).
11. Suguru Endo and Qi Zhao, "Mitigating algorithmic errors in a Hamiltonian simulation." Physical Review A 99, 012334 (2019).
12. Hiroshi Ohno, "A direct error correction method for quantum machine learning, Quantum Information Processing", Vol 22, 2023.
13. Vladimir Silva, "Enter the IBM Q experience: A one of a kind platform for Quantum computing in the cloud" , Practical Quantum Computing for developers, Springer, 2018, pp77-141.

❖ ❖ ❖