ACKNOWLEDGEMENTS

xx

Poramet Pathumsoot

DEVELOPMENT OF SECOND GENERATION NETWORK OF QUANTUM REPEATERS.

PORAMET PATHUMSOOT 6337776 SCPY/M

M.Sc. (PHYSICS)

THESIS ADVISORS : SUJIN SUWANNA, Ph.D. (MATHEMATICS), XXX, Ph.D. (XXX)

ABSTRACT

xxx

KEY WORDS : XX / XX /
XXX / XXX
XXX / XXX

48 pages

CONTENTS

CONTENTS (cont.)

CONTENTS (cont.)

LIST OF FIGURES

Figure **Page**

CHAPTER I
INTRODUCTION

Comments from proposal examination.

- (RVM) Research content wmust be clearly stated, and goals spelled out explicitly.
- (RVM) Please read many more research papers, and summarize the content, motivation and pain points accordingly. For example, we must explain why a classical internet can/cannot be used for a network of quantum computers.
- (RVM) For thesis writing, must add solid background on quantum mechanics, and relevant concepts.
- (RVM) Please be specific with noise models: gate noises, memory effects, stochastic noise, etc.
- (Michal) Please take smaller states to benchmark your calculation anlytically.
  1. test convergence of stabilizer counting methods. Increase the number of measurements, and experimental trials.
  2. Compare stabilizer counting with analytical method
  3. Do state tomography and compare results.
  4. Benchmarking waiting time.
  5. need to describe extraction of stabilizer values
  6. need to explain how BSA works, especially why it should be at the center, under what condition? Keyword is distinguishability.
- (Puwis) Is it possible to compare results with other simulators?
- (SS) There is no need to consider non-linear network topology.

- (SS) Discuss roles of BSA, and other methods of distributing Bell pairs, e.g. in relation to satellite QKD.
- Stabilizer counting seems to valid only for Q4 model. How should we modify it?
- Explain notation SS-Dp more clearly in thesis.

$$\exp\left\{x^2 + xy + y^2\right\}, \cos\left(xy + zx\right)$$

CHAPTER II
BACKGROUND THEORY

2.1 Quantum information

A bit of flow in of quantum information from its carrier, qubit, to mathematical description.

2.1.1 Mathematical representation of qubit

Qubit is a quantum object. There are number of approaches that could describe a state and evolution of qubits through mathematical model. For ideal case, Dirac's braket notation could perfectly model system of qubits with ease. The "ideal" case is refer to a quantum state that is a pure state, while for non-ideal case, Dirac notation fail to describe such a state which refer as mixed state. Density matrix approach however is more general than Dirac notation as it could model mixed state without a problem. Let's us begin with an ideal system with Dirac notation first, while the generalisation will be given later.

As analogy to classical bit, qubit could also represent bit information of zero and one with also superposition of them. An general mathematical expression of a state of one qubit, $|\psi\rangle$, is

$$|\psi\rangle = a\begin{bmatrix}1\\0\end{bmatrix} + b\begin{bmatrix}0\\1\end{bmatrix} = a|0\rangle + b|1\rangle, \tag{2.1}$$

where eq. (2.1) is a wave function of qubit. A vector representation is column vector of two rows where the first row is a probability amplitude of bit zero while another row is for bit one. For a system of more than one qubit, the vector representation grow exponentially. For convenience, Dirac notation is used to represent quantum state whereas those two representation could be map to each other. Upon observation of quantum state, there is probability of observes bit zero of $|a|^2$ and bit one of $|b|^2$.

An important note of observing quantum system is that, a projective measurement as much used in quantum communication collapse quantum state to the observed readout hence, destroy quantum information previously contained. Quantum computing and communication in someway is then an art of probability manipulation. Other type of measurement is out of slope for this work.

For system of more than one qubit, such as two qubits system, tensor product between each qubit is used to represent the state. The tensor product of qubits often omitted as,

$$|\psi\rangle \otimes |\psi\rangle = a^2|0\rangle \otimes |0\rangle + ab|0\rangle \otimes |1\rangle + ba|1\rangle \otimes |0\rangle + b^2|1\rangle \otimes |1\rangle \tag{2.2}$$

$$= a^2|00\rangle + ab|01\rangle + ba|10\rangle + b^2|11\rangle. \tag{2.3}$$

However, if a system of qubits could not write as a product of each individual subsystem, then those subsystem is said to be entangle with each other. An important example of entangle state of two qubits is Bell states which are,

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.4}$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{2.5}$$

$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |01\rangle) \tag{2.6}$$

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |01\rangle). \tag{2.7}$$

Bell state could not write as a tensor product of two qubit hence, those two qubit are entangled. Especially the Bell state in section 2.1.1, this state is called Bell pair and it is a building block of quantum communication as a wood necessary for an old school house building.

2.1.2 Quantum operator

Without an ability to control quantum state as desired, novel quantum technologies would still be a novel. Quantum operators are essential to manipulate quantum state, they transform one quantum state to another. Just as in classical computation, quantum operator is an analogy version of logic gate in classical computer. Quantum operators used in quantum information processing are unitary matrix. Consider an operator, U, that create Bell pair from zeros state of two qubit,

$$U = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}. \tag{2.8}$$

Applying U on zeros state of two qubit yields,

$$|\phi\rangle = |00\rangle \tag{2.9}$$
$$\Rightarrow U|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{2.10}$$

Unfortunately building physical device for every possible U is not practical. However with just a subset of U, it is possible to realize any U by a mean of assembling smaller U. A set of building block quantum operators that could used to build any U is called set of universal quantum gates. One example of the universal set is Clifford + T gates which consist of controlled-NOT gate ($CNOT$), Hadamard gate (H), S gate and T gate.

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.11}$$

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.12}$$

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{2}} \end{bmatrix} \tag{2.13}$$

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}. \tag{2.14}$$

S and T gate are phase gate, $P(\theta)$ with a rotation of phase of state as,

$$P(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix} \tag{2.15}$$

$$P(\frac{\pi}{2}) = S \tag{2.16}$$

$$P(\frac{\pi}{4}) = T. \tag{2.17}$$

Pauli gates are also an essential ingredients of quantum information processing too which could also construct from Clifford gates as,

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = P(\pi) = S^2 \tag{2.18}$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = HZH \tag{2.19}$$

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = iXZ. \tag{2.20}$$

Unitary operator in eq. (2.8) could be construct from Clifford group as,

$$U = CNOT \cdot H \otimes I, \tag{2.21}$$

where I is a identity operator of 2×2 dimension. The application of identity operator on quantum state change nothing but necessary for dimension consistency of operator U.

In quantum information processing, a visualization could help illustrate a flow of evolution of qubits. Quantum circuit is one of the most chosen.

2.1.3 Measurement

2.1.4 Mixed state

Mixed state is a linear combination of pure state ψ_i with classical probability of p_i. A density matrix describe mixed state could be write as,

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|. \tag{2.22}$$

Mixed state is a nightmare. It could occur in many ways, such as from imperfection of quantum gate. Consider a faulty Hadamard gate that will do its job with probability of p and do nothing with probability of $1 - p$. The application of that H gate on state,

$|\psi\rangle = |0\rangle$ will be,

$$\rho = p\left[\frac{1}{2}(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| + |1\rangle\langle 1|)\right] + (1-p)\left[|0\rangle\langle 0|\right] \tag{2.23}$$

2.1.5 Fidelity

In order to evaluate quality of quantum state created to the ideal one, fidelity is a suitable metric used. Fidelity is a metric measure how close of two quantum state ρ and σ given as,

$$F(\sigma,\rho) = \mathrm{Tr}\left[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right]^2. \tag{2.24}$$

If $\sigma = \rho$ then $F(\sigma,\rho) = 1$. Fidelity often used for quantum channel quality evaluation.

2.1.6 State tomography

Evaluation of quantum channel is important task, while using fidelity as metric, given that σ is an ideal state, a state, ρ, construct through channel is needed. As rho is a quantum state which is collapse to classical information upon projective measurement, a process of recovering quantum state ρ is also needed. State tomography is such a process to approximate quantum state ρ. To perform state tomography, one has to prepare identical quantum states and measures them with certain combination of bases. An accuracy of approximation could be increase by increase number of experiments.

2.1.7 Quantum Network

Quantum computers offer exclusive algorithms that could solve problems takes unreasonable long time for classical computer to solve. Those problems could bring impact to our daily day life in some ways. However, such a game changing algorithms also required powerful a quantum computer for execution. Yet, that powerful and useful single quantum computer is not likely to appear in a near future. To compensate a lack of quality qubits within single system, researchers propose technique to utilize many quantum computers with a network of quantum computers. Although we found work a round for complication in building quantum computers that could efficiently control many qubits, we still have to handle a complication of connecting two quantum computers.

Classical computers have our classical internet, naturally, quantum computers also have their internet too! Just as in classical internet, quantum internet should be able to send quantum information around. However even if we use photon qubit that could travel faster than anything else in universe, there is still a good chance that our qubit will be lost in hostile environment if it has to cover a distance too far away, that kids could not find their way back home either. To prevent that, stations that reduce a distance that photon qubit would exposed to noise for a long time is proposed, those station is called, quantum repeater as illustrate in fig. 2.1. Originally, quantum repeaters role are to establish a connection between adjacency nodes along the path, such a connections are to share Bell pair between nodes. After sharing those pairs, entanglement swapping are perform to produce an end-to-end Bell pair shared between source and destination nodes. The Bell pair could be used to perform quantum teleportation which destroy quantum information in the source node and at a cost of that end of the Bell pair at source via Bell-measurement, the quantum information will directly transfer to remaining end of the Bell pair at the destination node. Unfortunately, the magic could not be perfectly complete without classical communication of measurement result from the source, since measurement results are needed to correct the quantum information which now exist in the new qubit. Quantum teleportation is not the only one that use Bell pair as a resource. Distributed quantum computing required an ability to apply CNOT gate between qubits place separately in remote quantum computers, this could be done via teleported-CNOT gate or Non-local CNOT gate, while the gate also required the Bell pair for each application of the gate. Moreover, some quantum key distribution protocols also based on entangled pair such as E91 cite E91.

Nonetheless, after proposal of the quantum repeater, cite Gens proposed a further improvement of the scheme. Quantum repeater is now divided into three generations, where the next generation required more advance technologies than the previous generations. First generation quantum repeater, 1G, use Bell pair as a resource, hence relies on entanglement swapping to extent its connection. 1G increase its connection

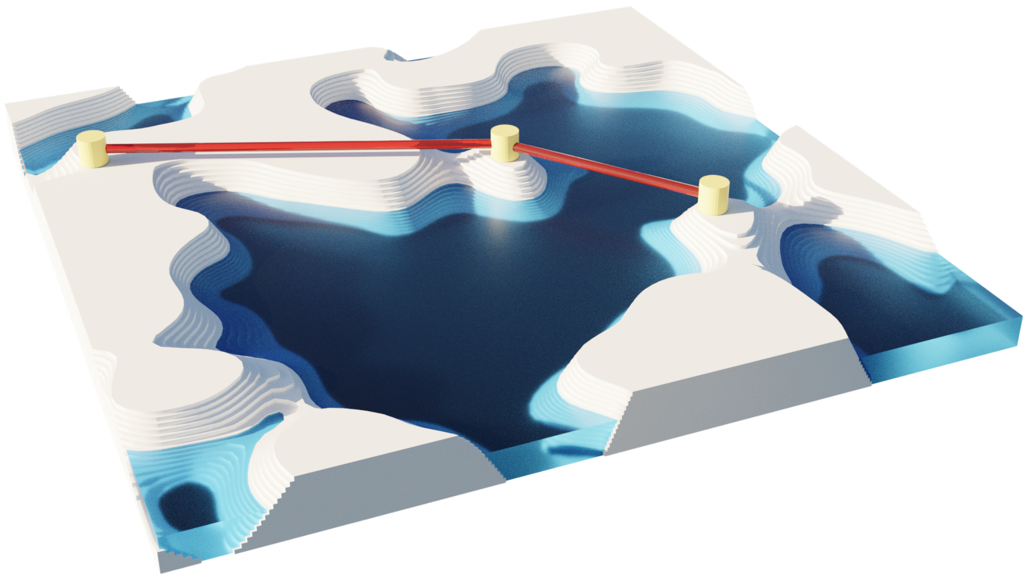

Figure 2.1 Caption

fidelity via entanglement purification (entanglement distillation) protocol. 2G use quantum error correction instead of entanglement purification. A resource to be consume for any application in 2G quantum network is therefore logical Bell pairs. However, 2G still use entanglement swapping to extend its connection. While, 3G use even more advance technique in quantum error correction. The technologies required for 3G are so advance such that quantum information could be encoded and send its directly from repeater to repeater without aid of entanglement swapping protocol. An illustration of generations of quantum repeater is present in fig. 2.2.

2.1.8 Quantum teleportation

Good news is teleportation is possible! Perhaps, bad news is only quantum information could be teleported. Anyway, quantum teleportation still useful in many ways as its concept used in another algorithm such as entanglement swapping. Here, we explain quantum teleportation for one qubit in its simplest form. Suppose that Prin want to teleport an arbitrary quantum information express as in eq. (2.1) to Glen located in far away place. Prin and Glen first needed to share a Bell pair, which is precisely why we need a quantum network! For a moment, let's skip how to share a Bell pair and to focus

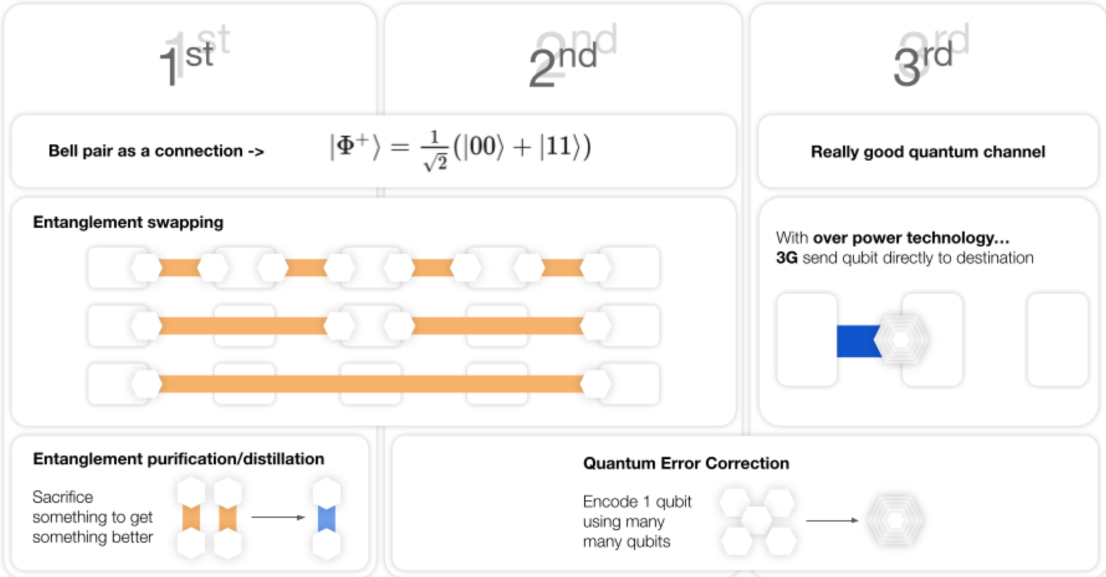

Figure 2.2 Caption

on quantum teleportation itself. Prin have to measure his qubit and one-end of a their Bell pair with Bell basis. A mathematical description of the scenario is,

$$|\psi\rangle|\Phi^+\rangle = (a|0\rangle + b|1\rangle)(\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)) \tag{2.25}$$

$$= \frac{1}{\sqrt{2}}(a|000\rangle + a|011\rangle + b|100\rangle + b|111\rangle) \tag{2.26}$$

$$\text{Change basis} \Rightarrow \frac{1}{\sqrt{2}}(a|+00\rangle + a|+11\rangle + b|-10\rangle + b|-01\rangle) \tag{2.27}$$

$$= \frac{1}{2}(a|000\rangle + a|100\rangle + a|011\rangle + a|111\rangle + b|010\rangle - b|110\rangle + b|001\rangle - b|101\rangle) \tag{2.28}$$

Depend on measurement results, quantum state of the final qubit that Glen hold is,

$$\{0, 0\} \Rightarrow (a|0\rangle + b|1\rangle) \tag{2.29}$$

$$\{0, 1\} \Rightarrow (a|1\rangle + b|0\rangle) \tag{2.30}$$

$$\{1, 0\} \Rightarrow (a|0\rangle - b|1\rangle) \tag{2.31}$$

$$\{1, 1\} \Rightarrow (a|1\rangle - b|0\rangle). \tag{2.32}$$

One could easily see that except for state corresponding to {0, 0 }, other state are not information that we desired. We could fix the final state with information of measurement results on Glen's qubit to turn it into the correct state. Prin has to send his measurement result to Glen; he then need to apply a following quantum gate on his qubit, if the information carrier qubit measurement result is one then apply Z and if one-end of Bell pair is one apply X. Thus, quantum information in Prin's qubit was successfully teleport to Glen's qubit. The most important note is that this is impossible without the aid of classical communication.

2.1.9 Entanglement Swapping

The first and second generation quantum repeater extend their connection by a mean of entanglement swapping protocol. Typically, the protocol involve three spacial separated parties, which will be refer as left, center, and right node where the center node place somewhere in between other two nodes. The requirement of the protocol is that left and center have to share one Bell pair and center and right node also have to share another Bell pair. The goal of the protocol is to finish with Bell pair shared between left and right node. An illustrate of the scheme is shown in add figure for es. A quantum state of the system is,

$$|\Phi^+\rangle_{AB}|\Phi^+\rangle_{CD} = (\frac{1}{\sqrt{2}}(|0_A\rangle|0_B\rangle + |1_A\rangle|1_B\rangle))(\frac{1}{\sqrt{2}}(|0_C\rangle|0_D\rangle + |1_C\rangle|1_D\rangle)), \tag{2.33}$$

where left node hold qubit A entangled with qubit B hold by center node and right node hold qubit D entangled with qubit C hold by center node too. Center node start the protocol by measure qubit B and C with Bell basis; a measurement result of those two qubits, c and d are then send to either node. The node receive measurement result is then apply quantum gate which may refer as correction operation on their qubit currently in one the four possible Bell state, $|\mathrm{UQS}\rangle$ which will left with Bell pair of the desired state as,

$$|\Phi^+\rangle_{AD} = Z^c X^d |\mathrm{UQS}\rangle_{AD}. \tag{2.34}$$

This is yet again, impossible without the help of classical communication! It is worth noting that entanglement swapping could be seen as the quantum teleportation of one-end of Bell pair to another qubit.

2.1.10 Link-level Model

As quantum teleportation and entanglement swapping relies on Bell pair, at very least in an early era of quantum network, distribute physical Bell pair across adjacency nodes is an essential task.

A general idea of distributing Bell pair for link level, the layer of adjacency nodes, is involve light-matter entanglement swapping. Consider first, type of qubit that suitable for travel along a great distance with lighting speed is obviously photon qubit. Secondly, the qubit that suit for computation however, is a matter qubit. Ideally, one might want to uses two types of qubit together to gain the full advantages. In that case, the simplest schematic involve two parties is to have each node generate their own Bell pair from photon and matter qubit, they then send their photon qubit to a Bell State Analyzer (BSA) for Bell measurement. After the measurement and application of correction operation, matter qubits will be entangle. This is nothing more than entanglement swapping of light-matter Bell pairs. In a case of static quantum stations place spatially separate from each other, BSA could be place at one side of the node. The node holding the BSA will has to wait for photon qubit from another node to arrive first then emitted its photon qubit. The most important part is to synchronize the timing of operation, so that two photon could not be distinguishable in BSA. This schematic is refer to as sender-receiver model. The placement of BSA could be in somewhere between both node such as right in the middle too, this is of course, requiring that the BSA node should capable of sending the measurement results to both nodes. While, another possible scheme including more Bell pair which is photon-photon Bell pair. A node at the center has to use EPPS to generate photon-photon Bell pair and distributed them to both sides. At a considerate timing, both nodes have to emitted their photon qubit entangled with their matter qubit to the BSA. This way the entanglement swapping is preform for both sides, with a appropriate correction operator, matter qubit of both sides will be entangled. This method is refer as EPPS model, which is likely to be used over a longer distance involve where a satellite is holding a role of photon-photon Bell pair generator.

2.2 Entanglement purification

A purification of entanglement in this case refer to purification of Bell pair which is a system of qubits entangled with each other, this process might refer to as entanglement distillation as well. The general idea of entanglement purification in quantum network is to scarifies some Bell pairs to check if another Bell pair is "seem" to be healthy or not.

2.2.1 X and Z purification

Let proceed with an example of a simple purification of X-type error, this treatment also work just fine with Z-type error with the magical Hadamard gate!

1 The procedure begin with resource Bell pair and ancilla Bell pair shared between two parties.

2 They operate CNOT gate on their ends of both pairs with resource-end as a control gate while using ancilla as a target qubit.

3 They both measure ancilla qubit in Z-basis. Results then shared to determine if resource pair is kept or not. If the measurement result agree then keep the resource pair while discard otherwise.

This could be done as an error propagates from control-qubit to target qubit, however there is some probability that resource qubits are both subject to errors simultaneously. With probability, p, of X-error occurring on each qubit independently, all possible scenario are present in table 2.1.

2.2.2 Single selection Double purification

2.3 Quantum Error correction

2.3.1 Pauli noise

One of the noise that might compromise the computation in quantum realm is Pauli noise, which typically refer as a Pauli operator X, Y, Z randomly apply on qubit. X-type error, or Pauli-X is corresponding to bit-flip of classical realm in computational

Error		probability	measurement result
qubit 1	qubit 2		
I	I	$(1-p)(1-p)$	Agree
X	I	$p(1-p)$	Disagree
I	X	$(1-p)p$	Disagree
X	X	p^2	Agree

Table 2.1 X-purification

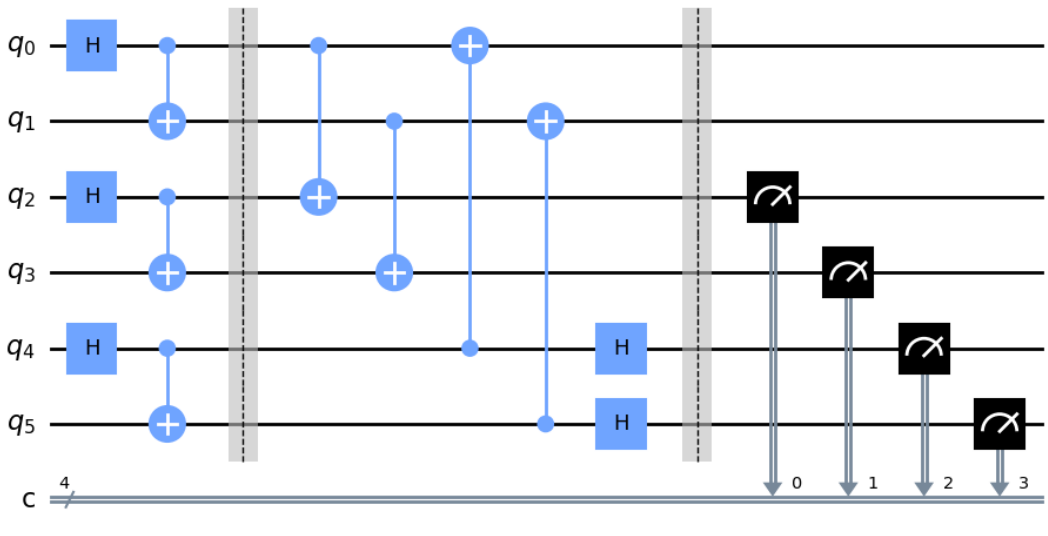

Figure 2.3 Quantum circuit for Single selection Double purification (Ss-Dp).

basis as it turn state zero to one and vice versa. If X-error occur on state in eq. (2.1) then, the state become,

$$X|\psi\rangle = a|1\rangle + b|0\rangle. \tag{2.35}$$

While Z-type error or Pauli-Z is a phase flip error as it flip the sign of state one from plus to minus and reverse. The state in eq. (2.1) will become,

$$Z|\psi\rangle = a|0\rangle - b|1\rangle, \tag{2.36}$$

in the present of Z-error.

As for Y-error, Pauli-Y could decompose into X and Z as, $Y = iXZ$, this affect the eq. (2.1) as,

$$Y|\psi\rangle = iXZ|\psi\rangle = ia|1\rangle + ib|0\rangle. \tag{2.37}$$

2.3.2 3-qubits code

Three-qubits quantum error correction code is a small that could only one type of error, which is why this code is not practical in real life as quantum error has both bit flip and phase flip error but it is useful as an entry-level code to study.

Suppose that there is quantum information to be encoded,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle. \tag{2.38}$$

Using quantum circuit in the first block to encode section 2.3.2 into the codeword, the quantum state become,

$$|\bar{\psi}\rangle = \alpha|000\rangle + \beta|111\rangle \tag{2.39}$$

$$= \alpha|\bar{0}\rangle + \beta|\bar{1}\rangle. \tag{2.40}$$

If error, say bit-flip occur on the first qubit, the quantum state is

$$|\bar{\psi}\rangle = \alpha|100\rangle + \beta|011\rangle, \tag{2.41}$$

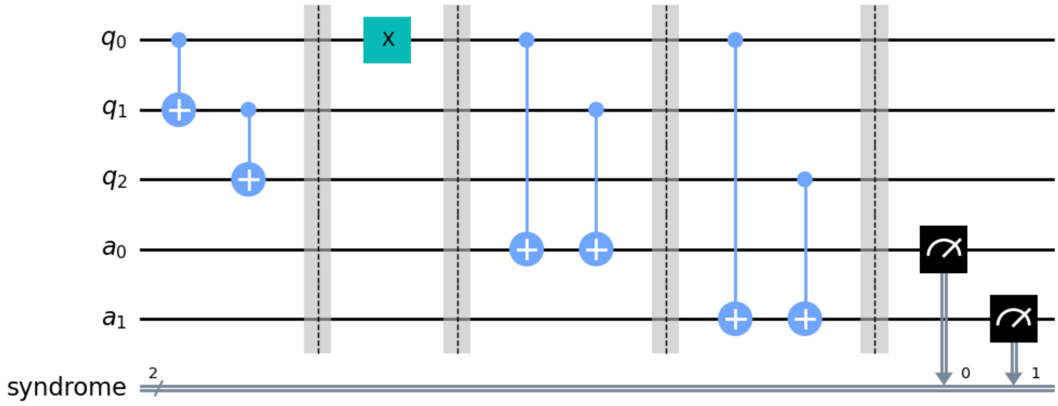

Figure 2.4 Quantum circuit to encode logical qubit using 3-qubits code. First block uses to encode quantum information in the first qubit, q_0, with addition qubits q_1 and q_2. The next block represent quantum error that could occur to any encoding qubit. The following two blocks are circuit for syndrome measurement.

measuring those three qubits would yield either $|100\rangle$ or $|011\rangle$ with probability of $|\alpha|^2$ and $|\beta|^2$. Either cases, a natural cause of action is to interpret that bit-flip has occur on the first qubit, therefore correction on codeword would yield $|000\rangle$ or $|111\rangle$ which is legal codeword. However, if bit-flip error would occur on any of two encoded qubits, the measurement readout would be the same as one error case. An important note of this scenario is that quantum error correction codes have limit of tolerance to noise.

If one has to measure the logical qubit itself to perform error correction, this might not useful at all, as qubit could experience error multiple time before the final measurement, which might resulting in failure in quantum computation. The power of quantum error correction is that syndrome measurement and correction can be done without knowledge of quantum information encoded with aid of ancilla qubits.

Syndrome measurement is to measure eigenvalue of generator of the code. More detail of generator and stabilizer will be further explained in section 2.3.3. For simplicity, operating an generator to codewords would not changes codeword at all and if only legal codewords are measured, the measurement result will be +1 eigenvalue of the generator.

In 3-qubits code case, the generators are, $G_1 = ZZI$ and $G_2 = ZIZ$. Quantum circuit corresponding to syndrome measurement of each generator are shown in third and fourth blocks in fig. 2.4. Quantum state is therefore,

$$|\bar{\psi}\rangle = \alpha|100\rangle_q|11\rangle_a + \beta|011\rangle_q|11\rangle_a, \tag{2.42}$$

measurement result will be 11 which reveal that an error that happen on encoded qubit flip the codeword such that q_0 has different parity with q_1 and q_0 has different parity with q_2. The result show that the q_0 experience bit-flip error, so the correction is of course to apply X gate to q_0.

2.3.3 Stabilizer

2.3.4 Error propagation

Those Pauli errors are vicious indeed as quantum gate applying on qubit could transform the error from one to another! A prime example is when one apply Hadamard gate in a present of an error. The H gate will transform bit-flip error to phase-flip error and vice versa. Suppose that there is an X-error beforehand on qubit of state eq. (2.1), applying H gate will then,

$$HX|\psi\rangle = HXHH|\psi\rangle \tag{2.43}$$

$$= ZH|\psi\rangle, \tag{2.44}$$

in which X error become Z error. An idea of this treatment is to determine a final result state of the ideal stabilizer-circuit subject to random Pauli-errors at any points of expected operations.

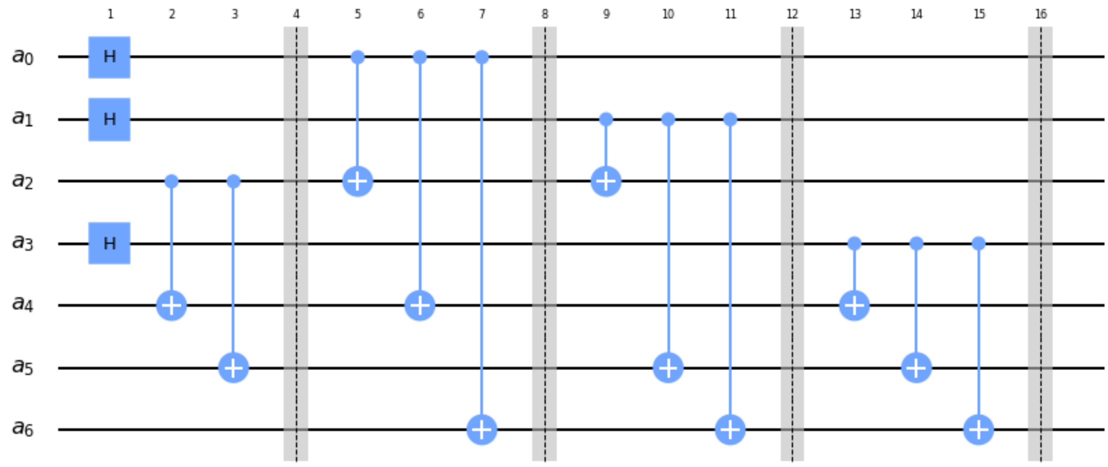

Figure 2.5 Steane Code [7, 1, 3] encoding circuit where q_2 is an input state.

2.3.5 Steane [7, 1, 3] code

In this work, we use $[7, 1, 3]$ Steane code which encode logical qubit in codeword,

$$\begin{aligned} |\bar{0}\rangle &= |0000000\rangle + |1010101\rangle + |0110011\rangle \\ &+|1100110\rangle + |0001111\rangle + |1011010\rangle \\ &+|0111100\rangle + |1101001\rangle \qquad (2.45) \\ |\bar{1}\rangle &= |1111111\rangle + |0101010\rangle + |1001100\rangle \\ &+|0011001\rangle + |1110000\rangle + |0100101\rangle \\ &+|1000011\rangle + |0010110\rangle. \qquad (2.46) \end{aligned}$$

A quantum circuit for encoding quantum state is present in figure 2.5.

During the computational process, error correction can be done at any time. For Steane code, the straight forward non fault-tolerance error detection circuit is present in figure 2.6. The measurement results of three ancilla qubits will be represent the location of Pauli-X error in binary form while if there is no error occur, the measurement result will be all-zero.

The logical operation for logical qubit depend on code used. For Steane code, the logical operation can be implement by apply logic gate on each physical qubits, e.g.

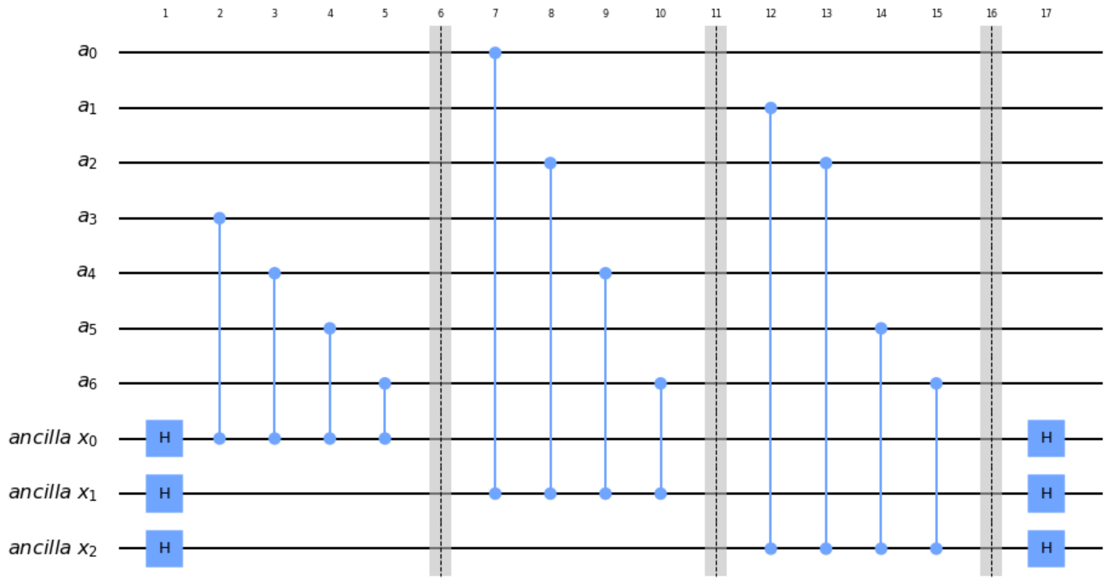

Figure 2.6 The quantum circuit that introduced 3 additional ancilla qubits to perform an error detection of a Pauli-X error. By measuring the ancilla qubits, the readout will give an index of the qubit subject to the X error in base 2 form.

applying Hadamard gates on all 7 physical qubits is equivalent to the logical Hadamard gate H on logical qubit. For logical $CNOT$ gate, one need to apply $CNOT$ gates to the pair of physical control and target qubits transversely.

The measurement readout for logical qubit using Steane code can be obtain a number of way. First way is to apply decode operation to the logical qubit and measure the input qubit which represent readout of logical qubit, where decoding is applying encoding inversely. Second way is to just measure all encoded physical qubits, the measurement result should be one of the codeword in 2.45. The later is a preferable way since apply decoder might induce error during the operation. Further more, Steane code is an CSS code which adapt the idea of parity check from classical error correction, i.e. one can use parity matrix H to detect and correct a bit-flip error on the readout result There is a larger Parity check matrix which should be able to check the phase flip error

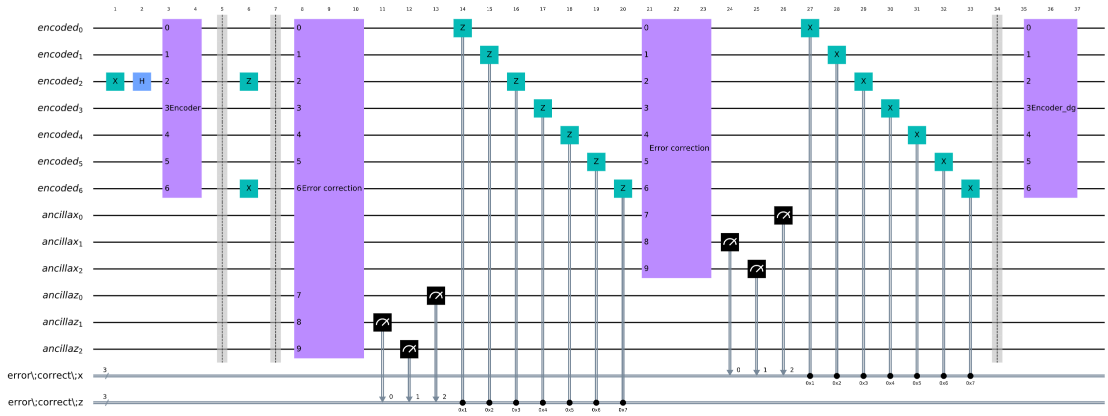

Figure 2.7 The full quantum circuit for quantum error correction using Steane code, where an encoder block should be replace with the circuit in figure 2.5 and error correction block is replace by circuit in figure 2.6. Note that the quantum circuit of phase flip error detection is like in bit flip error detection but replace CZ gates with CNOT gate instead.

too. But I'm not sure how to use it, where

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{2.47}$$

A full quantum circuit for quantum correction using Steane code is shown in figure

CHAPTER III

LITERATURE REVIEW

3.1 XXX

3.1.1 xxx

CHAPTER IV
DESIGN OF QWANTA

`Qwanta` is a python software design for simulate quantum network based on discrete-event simulation using `simpy`. Despite the slow performance compare to other programming languages such as C++, python offer an easy to read code style suit for prototyping concept and theories. One of a important design principle of `qwanta` is to let user easily define their our custom events and they could easliy fit in the package. User could provide topology, processes, and system configurations needed to be simulate and execute a code to their heart content!.

4.1 Simpy

As mentioned above, `simpy` is discrte-event simulation package for python. In the following subsection, related concept of `simpy` used for simulate process in real-world quantum network will be explained.

4.1.1 Resource and Process

The main idea of `simpy` is to process event within environment. The event could be define as a function which could trigger other events as well. `simpy` will process all of the events until there are no events left to process or until time in the simulation progress to the set limit. The unit of time in `simpy` is arbitrary, in this work, we set the unit to second. Event may require resource for further processing. Resource in `simpy` could be any python object. If there are no resources to be consume when event request it, the event will wait until there are to be consumed. If however, resources are never available, then the simulation will impossible to finish which may cause an error, depend on design of process.

In `qwanta`, resources typically are qubits or certain system of qubits such as physical and Logical Bell pair and stored in resource tables which is `simpy.FilterStore`. While `simpy.Store` could be used for other propose such as communication line. Events or processes are defined as a function to process resources and introduced time delay to simulate a real world devices where task is not done instantly, they may consume and produce other type of resources. One could set a limit to processes which is basically set how many time it needed to loop over or with an infinite loop is possible too, which we refer as *limited* and *unlimited* process respectively. One important note is that if all of the processes of experiment are set to be unlimited process then, time limit needed to be provided. If at least one process is limited process then, the simulation will terminate when that process is finish. List of main processes are present in section 4.1.1

Other ancillary processes could be introduce for applying noise, attack from eavesdropper, and etc. More detail on design of some main process will be explained in add example section

4.1.2 Device modeling

Pre-defined process, model devices in a way that simplify all devices related to particular task into one function. For example, `PrototypeGenerateLogicalResource` is implement as a processing unit that have access to share memory which could allocate and use resources that match the criteria. Regardless of physical form of a qubit, the processing unit could also perform quantum operations and measurement on qubit as well as sending classical message to other nodes. However `qwanta` allow for more detail implementation of the process, such as at individual device. One good example is a `PrototypeGeneratePhysicalResourcePulse` which consists of `Emitter` responsible for allocate physical qubit to let it emit entangled photon out to quantum channel, `Detector` responsible for detect any incoming photons and send a measurement result to relate unit (This could interpret as implement of BSA, however for simplicity, the actual implementation is more simpler which will be explain later in add me), and `ClassicalMessageHandler` responsible for process incoming classical message. Depend on a level of interest, user could defied their own function to reflect the device and process needed to be simulate. In low-level point of view, `qwanta` is just

Desription	options	Consume	Produce
o produce physical Bell pair at link-level whenever there are nal physcial qubits available. Physical qubits from both nodes used to emit a photons and sent to BSA somewhere of either with some pulse interval as SenderReciver model.		Physical qubit	physical Bell pair
o produce physical Bell pair at link-level whenever there are external cial qubits available. Physical qubits from both nodes will used to emit otons and sent to BSA at the middle node or entangled photon pair will uce in the middle node and sent to BSA at both nodes as EPPS model.	(1) EPPS	Physical qubit	physical Bell pair
	(2) BSA at middle node		
ume physical Bell pair and internal physical qubit to encode logical Bell pair.	(1) Non-local CNOT (Steane code)	7 Physical Bell pairs, 7 internal physical qubit	Logical Bell pair
	(2) Purified-encoded (Steane Code)	1 Physical Bell pair, 6 internal physical qubits	
o purify physical Bell pair using one or more physical Bell pairs	(1) Ss-Dp	3 Physical Bell pairs	Physical Bell pair
	(2) X-purfication	2 Physical Bell pairs	
	(3) Z-purification	2 Physical Bell pairs	
g Bell pairs from two adjency edges to perform entanglement swapping, uce Bell pair for end-nodes	(1) Physical	2 Physical Bell pairs	Physical Bell pair
	(2) Logical	2 Logical Bell pairs	Logical Bell pair
rm measurement for state-tomography using Bell pair	(1) Physical	Physical Bell pair	Physical qubits
	(2) Logical	Logical Bell pair	

a package that provide a framework for processing and passing through of the resources in network.

4.2 Error Model

Evolution of quantum system is important to track throughout the simulation, as its needed to interact with each other and measure to collect value of interest. Although, density matrix formalism allow the most general description of quantum system, the matrix of system grow exponentially with number of qubits. For small system, it is possible to simulate small system with classical computer but the computational resources price needed to be pay still huge. Sadly, I'm indeed very poor. Still with Gottesman–Knill theorem, to simulate only Clifford gates and Pauli error, classical computer can simulate that system of qubits efficiently. `Qwanta` focus on Pauli error and perform only Clifford gates. Further simplification let `qwanta` store only Boolean value represent presence of Pauli-X and Pauli-Z error. Performing quantum gate on qubits is then a transformation and propagation of error. Measurement is simplified as a function that return `bool` indicate a presence of error. Following the basic setting of error management system in `qwanta`, interesting tasks fall to how one define timing of error that could occur on qubit as will be describe in the following subsection.

4.2.1 Gate Error

In real-world, the devices used to control qubit are likely to be imperfect, at least in the moment. Current implementation of `qwanta` interpret gate error as a probabilistic event. With probability p_{gate}, the device will perform quantum gate O imperfectly as it apply depolarizing channel to qubit. The channel could be describe as,

$$\epsilon_{\text{gate}}(\rho) = (1 - p_{gate})O\rho O^{\dagger} + \frac{p_{gate}}{4} \sum_{i=\{I,X,Y,Z\}} G_i \rho G_i^{\dagger}. \tag{4.1}$$

For two-qubit gate, quantum channel is define as,

$$\epsilon_{\text{gate}}(\rho) = (1 - p_{gate})O\rho O^\dagger + \frac{p_{gate}}{16} \sum_{i,j=\{I,X,Y,Z\}} G_i G_j \rho G_i^\dagger G_j^\dagger. \tag{4.2}$$

4.2.2 Depolarizing channel

At a link-level generation of physical Bell pair, the process involve sending photon qubit through medium which could corrupt the qubit along the way. Current implementation simplify the process of measuring photonic qubit which their errors would then propagate in a form of measurement result and hence performing correction quantum gate would transfer error from photonic qubit to matter qubit instead by directly apply depolarizing channel on matter qubit after classical message exchange of successful notification. The depolarizing channel is defined as,

$$\epsilon_{\text{depolarizing}}(\rho) = (1 - p_{\text{depolarizing}})I\rho I^\dagger + \frac{p_{\text{depolarizing}}}{3}(X\rho X^\dagger + Y\rho Y^\dagger + Z\rho Z^\dagger), \tag{4.3}$$

where $p_{\text{depolarizing}}$ is probability of error occur on matter qubit.

4.2.3 Memory error

Depending on a physical form of qubit, a lifetime, τ of qubit may varies. However, one may define memory error as an exponential decay probability of no error based on τ, $p_{\text{mem}} = \frac{e^{-t/\tau}+3}{4}$ as a function of time which in `Qwanta` defined as,

$$\epsilon_{mem}(\rho) = p_{\text{mem}} I\rho I^\dagger + \frac{(1 - p_{\text{mem}})}{3}(X\rho X^\dagger + (Y\rho Y^\dagger + (Z\rho Z^\dagger). \tag{4.4}$$

The memory error would be applied right before any measurement of that qubit.

4.2.4 Measurement error

Measurement is a process that extract information from qubit to classical bit, like the other processes, measurement is not perfect. A definition of measurement error in `qwanta` is, there is probability, p_{meas}, that measurement result will subject to bit flip error after the readout.

4.2.5 Photon loss

In `qwanta`, photon loss is interpret as a probability of photon emitting from single photon source and arrive at the detector, p_{loss}, under value of loss, l in unit of dB/km. This however, could easily extent to include an efficiency of instruments, η. The p_{loss} is then calculate using loss, l, and distance photon needed to cover, d,

$$p_{\text{loss}} = \eta 10^{-l \cdot d/10}. \tag{4.5}$$

4.3 Stabilizer measurement

In order to avoid dealing with density matrix formalism, a work a round for calculate fidelity could be done by following method in this work [1] with some modification.

4.3.1 Physical Bell pair

Bell pair could be decompose into Pauli operator,

$$|\Phi^+\rangle\langle\Phi^+| = \frac{1}{4}(II + XX - YY + ZZ). \tag{4.6}$$

While, the quantum state right before measurement could be express as,

$$\rho = \sum_{i,j=\{I,X,Y,Z\}} p_{i,j} G_i G_j |\Phi^+\rangle\langle\Phi^+| G_i^\dagger G_j^\dagger. \tag{4.7}$$

Fidelity of section 4.3.1 compares to eq. (4.7) is

$$\begin{aligned} F(\rho, |\Phi^+\rangle\langle\Phi^+|) &= \langle\Phi^+|\rho|\Phi^+\rangle && (4.8)\\ &= \mathrm{Tr}(\rho|\Phi^+\rangle\langle\Phi^+|) && (4.9)\\ &= \frac{1}{4}\mathrm{Tr}(\rho + \rho XX - \rho YY + \rho ZZ) && (4.10)\\ &= \frac{1}{4}(1 + \langle XX\rangle - \langle YY\rangle + \langle ZZ\rangle) && (4.11) \end{aligned}$$

Consider first $\langle XX \rangle$,

$$\langle XX \rangle = \sum_{i,j=\{I,X,Y,Z\}} p_{i,j} \langle \Phi^+ | G_i^\dagger G_j^\dagger XX G_i G_j | \Phi^+ \rangle \tag{4.12}$$

We know that there are two possible outcome which are 1 if $[G_i G_j, XX] = 0$ and -1 if $\{G_i G_j, XX\} = 0$, therefore,

$$\langle XX \rangle = \sum p_{\text{commute}} - \sum p_{\text{anti-commute}}. \tag{4.13}$$

This is also true for $\langle ZZ \rangle$. For $\langle YY \rangle$, because $YY = -XXZZ$ which yield,

$$\langle YY \rangle = -\sum p_{\text{commute}} + \sum p_{\text{anti-commute}}. \tag{4.14}$$

Therefore, fidelity become,

$$F = \frac{1}{4}(1 + \sum_{i=\{XX,YY,ZZ\}} (p^i_{\text{commute}} - p^i_{\text{anti-commute}})) \tag{4.15}$$

Each term in eq. (4.15) could be interpret as measurement in a basis that directly affect fidelity. For non-stabilizer measurement-basis, any Pauli noises on Bell pair have no effect on readout distribution of Bell pair.

Furthermore, only bit-flip error of measured basis is visible and has an negative effect to fidelity. For example, if one measure qubit in XX-basis then the quantum operator before measurement is XI, IX, or XX, this particular measurement would increase p^{XX}_{commute}. however if the operator is ZI or IZ, then this measurement would increase $p^{XX}_{\text{anti-commute}}$. To be more specific, follow the method in section 2.3.4, X is bit flip error in basis Z, Z is bit-flip error in basis X, and either X or Z are bit-flip error in basis Y but XZ is not bit-flip error in Y-basis. The interpretation provide a way to includes measurement error after the measurement in certain basis in `qwanta`.

4.3.2 Logical Bell pair

Even if logical Bell pair is a system of 14-qubits for Steane code case, a fidelity of interest is still be 2-qubit physical Bell pair. However, upon measuring a codeword, measurement result needed to be classical corrected and therefore decoded. The decoded result will then be used to reconstruct density matrix of 2-qubits system. After that fidelity is calculated from reconstructed density matrix and ideal 2-qubits Bell pair.

Error-basis formalism allow us to use error operator to do classical error correction in a similar fashion as an actual codeword. The procedure involves mapping integer, 1, with boolean, `True` and 0 with `False`. With mapping, classical error correction can be done normally, where the corrected result is used to decode for logical information. If there are odd 1 then, the logical readout would be a result with bit-flip error, namely, anti-commute with the measurement basis, otherwise, the logical readout will commute with the measurement basis.

4.4 How to simulate thing

`Qwanta` lets user create resource-based timeline in excel or csv file format for ease of visualization. Resource-based timeline is simply table that specify how fundamental resource, physical qubits, are transform through processes. Name of nodes have to be the same as nodes-edges information in python format. Both are then used to initialize `configuration` object or `Experiment` which is object to executes multiple `configuration` under the hood.

4.4.1 Flow of resource and process as table

First of all, let's look at a figure fig. 4.1 illustrate how resource and process interact in network and corresponding table table 4.1. `Qwanta` processes any process as soon as resources required are acquired. At the beginning of a simulation, only physical qubit is available, which mean `PrototypeGeneratePhysicalResource` processes are generally the first process to be completed. The process convert physical qubit resources

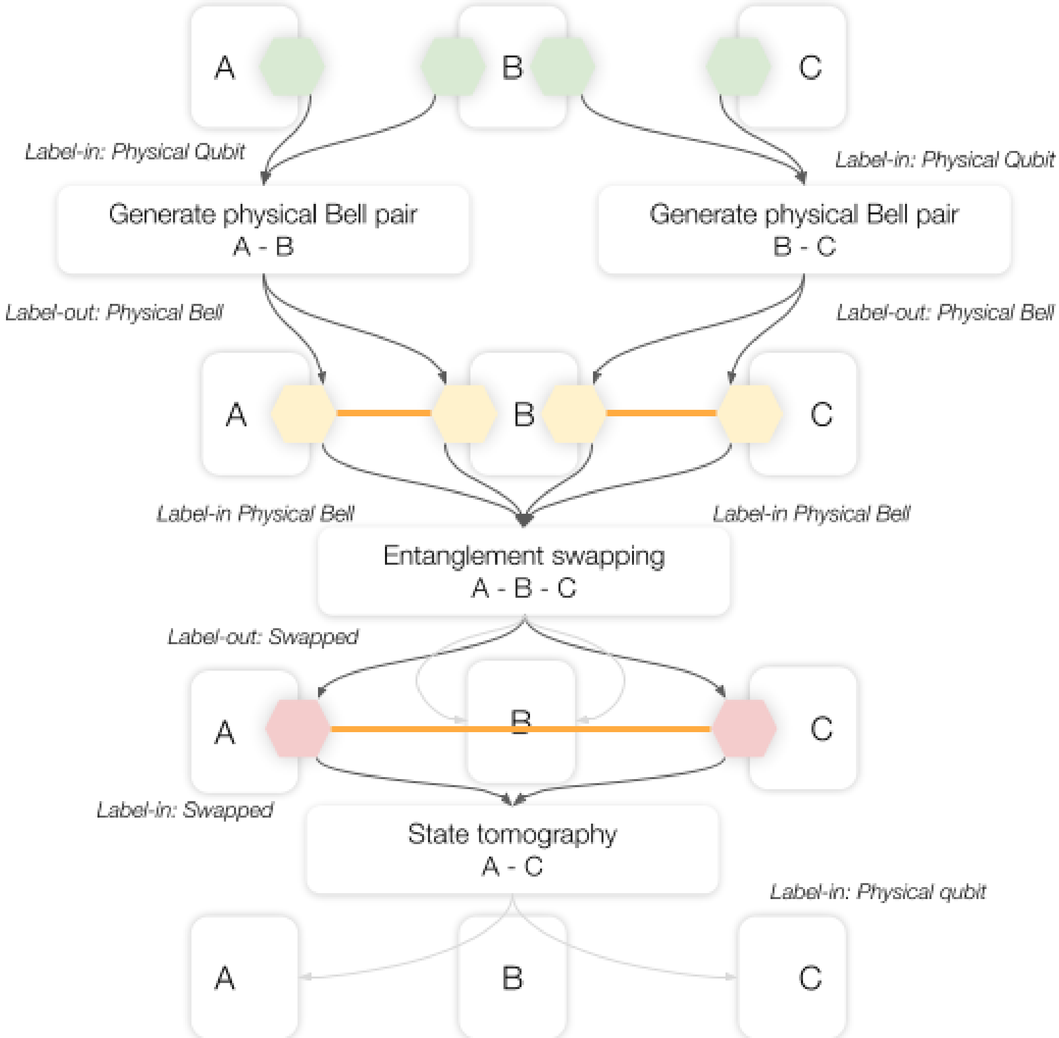

Figure 4.1 Resource flow in simulation

into physical Bell pairs which convert to other resources via other processes. In table table 4.1, there are four processes aim to do state-tomography using 9,000 physical Bell pair for node A and C where there is Node B in the middle. Generate Physical resource processes independently generate physical Bell pairs for edge A-B and edge B-C, the physical Bell pairs generated are label as Physical Bell. Entanglement swapping process is then uses `Physical Bell` to produce Bell pair for edge A-C and label them as `Swapped`. Finally, State tomography measures `Swapped` Bell pair to calculate fidelity.

Main Process	Edges	Label in	Label out	Protocol	Resource type
Generate Physical resource	A - B	Physical qubit	Physical Bell	standard	physical
Generate Physical resource	B - C	Physical qubit	Physical Bell	standard	physical
Entanglement swapping	A - B - C	Physical Bell	Swapped	standard	physical
State tomography	A - C	Swapped	Physical qubit	standard	physical

Table 4.1 Resource-based timeline table

4.4.2 Configuration of dynamics/static position and simulation time limit

Noises setting can easily define as a python object,

- **Loss** has to be a float in a unit of dB/km
- **Depolarizing Channel** has to be provide as a list of probability of length four corresponding to probability of applying Pauli I, X, Y, and Z respectively.
- **Gate error** has to be a list of probability of length two which corresponding to perform a gate properly and apply depolarizing channel to mixed qubit.
- **Measurement error** has to be a probability of float
- **Memory error** should be defined as a function that accept the first argument as a time in unit of second and return a probability list of length four corresponding to probability of applying Pauli I, X, Y, and Z.

Below is a code example of noise,

```
loss = .1 # dB/km # Photon loss
p_dep = [0.97, 0.01, 0.01, 0.01] # Depolarizing Channel
gateErr =  [0.99, 0.01] # Gate error -> [Do, Mix]
measurement_error = 0.01 # Measurement error

# Memory error
def memError(time, tau=10**(-3)):
    p = (np.e**(-1*(time/tau)))/4 + 0.75
    return [p, (1- p)/3, (1- p)/3, (1- p)/3]
```

```

# Position function
def position_function(time):
    return (time, 0, 0)
```

Listing 4.1: Simple configuration

4.4.3 Nodes and Edges information

An information of nodes and edges are provide as a following format.

Generally, information of nodes should provided as a dictionary. Position of node has to provide as a value of `coordinate` key where the value should be tuple of position (x, y, z) for static node and function as dynamic node. Other information are number of qubit of node for each edge.

Edges information should provide as dict of each edge, where each value should be a dict contain information of that particular edge such as noises of that channel, speed of light, pulse interval, and etc.

```
nodes_info_exp = [{
    'Node A': {'coordinate': (0, 0, 0)},
    'Node B': {'coordinate': (100, 0, 0)},
    'Node C': {'coordinate': (200, 0, 0)},
    # 'Node C': {'coordinate': position_function},
    'numPhysicalBuffer': 20,
    'numInternalEncodingBuffer': 20,
    'numInternalDetectingBuffer': 10,
    'numInternalInterfaceBuffer': 2,
}]

Quantum_topology = [{
        ('Node A', 'Node B'): {
        'connection-type': 'Space',
        'function': p_dep,
        'loss': loss,
        'light speed': 300000,
        'Pulse rate': 0.0001,
    },
```

```
        ('Node B', 'Node C'): {
        'connection-type': 'Space',
        'function': p_dep,
        'loss': loss,
        'light speed': 300000,
        'Pulse rate': 0.0001,
    }
}]
```

Listing 4.2: Simple nodes and edges configuration

4.4.4 Quick start

Including code snippets 4.1, 4.2, and 4.3, assuming that timeline is stored in `exper_id10_selectedStats_2hops.xlsx`, a simulation is ready to execute.

```

timelines = {}
for exp_name in exp_names:
    e_tl, vis_a = Experiment.read_timeline_from_csv(f'
   exper_id10_selectedStats_2hops.xlsx', excel=True, sheet_name=
   exp_name)
    timelines[exp_name] = e_tl

nodes_information = {exp_name: nodes_info_exp[index] for index,
   exp_name in enumerate(exp_names)}
networks = {exp_name: Quantum_topology[index] for index, exp_name in
   enumerate(exp_names)}
mem_func = {exp_name: memError for exp_name in exp_names}
measurement_error = {exp_name: measurement_error for exp_name in
   exp_names}
gate_error = {exp_name: gateErr for exp_name in exp_names}
sim_time = {exp_name: None for exp_name in exp_names}
labels = {exp_name: 'Physical' for exp_name in exp_names}

p = [0]
exper = Experiment(networks, timelines, nodes_info=nodes_information,
                   memFunc=mem_func, gateError=gate_error,
```

```
                    simTime=sim_time,
                    measurementError=measurement_error,
                    parameters_set=p, collect_fidelity_history=True,
                    repeat=1, label_records=labels,path='exp_ID10',
                    message_log='exp_ID10_selectedStats_2hops',
                    progress_bar=True)

# Execute simulation
exper.run()
```

Listing 4.3: Simple configuration

4.5 Customization

In this section, I will explain how to create and include customize process and parameter of interest into `qwanta`.

4.5.1 Design of custom process

Under the hood each processes of `qwanta` is a python generator in a certain template. `CustomProcess` should define in separate python file which could be import into `qwanta.py`. For example if code snippet 4.6 is defined in python file under directory of `QuantumProcess/_CustomProcess.py`, user should register the custom process by first import the process at the begining of `qwanta.py` then include the process as a method of `QuantumNetwork` object in `qwanta.py`

```
# In qwanta.py

from QuantumProcess import _CustomProcess

class QuantumNetwork(_GeneratePhyscialResource.Mixin,
                     _EntanglementPurification.Mixin,
                     _EntanglementSwapping.Mixin,
                     _GenerateLogicalResource.Mixin,
                     _VirtualStateTomography.Mixin,
```

```
                    _CustomProcess.Mixin, # <--- Include here
                    _TimeLag.Mixin):
```

Listing 4.4: Simple configuration

The next step is to specify a name of custom process to be used in resource-based timeline and tell `qwanta` which process to execute. In a code snippet 4.5, show an example of the mapping step. `process` is a python dictionary that store information of each process corresponding to each row in resource-based timeline table where key is a column name and entire is an entry.

```
    elif process['Main Process'] == 'Custom Process':
                p =[self.env.process(self.CustomProcess(process[
                'Edges'][0], process['Edges'][1],
                label_in=process['Label in'],
                label_out=process['Label out'],
                num_required=process['Num Trials']))]
```

Listing 4.5: Simple configuration

In code snippet 4.5, `process['Main Process'] == 'Custom Process'` is precisely where user define how do they refer to their custom process in resource-based timeline table. Please note that `process['isSuccess'] = 0` will be initialized automatically.

Let's now proceed to an actual definition of `CustomProcess` itself.

```
class Mixin:
    def CustomProcess(self, process, node1, node2,
                      num_required=1,
                      label_in='IN',
                      label_out='OUT'):

        # Valiate node order
        node1, node2 = self.validateNodeOrder(node1, node2)

        table = self.resourceTables['physicalResourceTable']

        # Loop the process until some conditions meet.
        # In this case, just
```

```
        while process['isSuccess'] < num_required:

            # Get all resource needed before further processing
            event = yield simpy.AllOf(self.env,
                        [table[f'{node1}-{node2}'].
                        get(lambda bell: bell[2] == label_in[0]))

            # Do resource processing

            # <--- OPTIONAL --->
            # Wait for 1 second before further processing
            yield self.env.timeout(1)
            # <--- END OPTIONAL --->

            # <--- OPTIONAL --->
            # Trigger another independent process with
            # resources previously got.

            # Pack necessary variables into single variable
            info = (Bells, node1, node2, table, label_out, num_required, process)
            self.env.process(self._independentProcess(info))

            # <--- END OPTIONAL --->

            # Increment number of success,
            # this could be move to independent process
            process['isSuccess'] += 1
```

Listing 4.6: Simple configuration

Code snippet 4.7 show an example template of independent process triggered from custom process in code snippet 4.6.

```
def _independentProcess(self, info):

        # Unpack variables
        Bells, node1, node2, table, label_out, num_required, process = info
```

```

        # Process resources
        Success = True # False

        # Put resource back to fundamental resource table
        # or/and update resource with new label.
        if Success == True:
            # Register resource with new label
            self.createLinkResource(node1, node2, Bells[0],
            Bells[1], table, label=label_out)

            if num_required is not True:
                process['isSuccess'] += 1
        else:
            # Put resource back to fundamental table
            Bells[0].setFree(); Bells[1].setFree()
            self.QubitsTables[Bells[0].table][Bells[0].qnics_address]
                    [f'QNICs-{Bells[0].qubit_node_address}'].
                    put(Bells[0])
            self.QubitsTables[Bells[1].table][Bells[1].qnics_address]
                    [f'QNICs-{Bells[1].qubit_node_address}'].
                    put(Bells[1])
```

Listing 4.7: Simple configuration

A placement of `process['isSuccess'] += 1` is not necessary fix, it is depend on the user design of the process for example, if the process should execute with a fix number of time regardless of success or fail of the processing, `process['isSuccess'] += 1` could be place outside the if condition as well. A reasoning of if condition before updating of `process['isSuccess']` is that if `num_required = True` then the process should execute until the simulation terminates due to simulation time out or limited processes are done. Therefore if this process is unlimited process, namely, `num_required = True`, the condition of while loop will be true until simulation terminates as $0 < 1 =$ true.

4.5.2 parameters of interest

To collect parameters of interest in `qwanta`, user first need to define the paramter in `QuantumNetwork` object in `qwanta.py` as illustrate in 4.8.

```
# qwanta.py

class QuantumNetwork(_GeneratePhyscialResource.Mixin,
                     _EntanglementPurification.Mixin,
                     _EntanglementSwapping.Mixin,
                     _GenerateLogicalResource.Mixin,
                     _VirtualStateTomography.Mixin,
                     _TimeLag.Mixin):

    def __init__(self, configuration):

        .
        .
        .

        # you can store parameter of any type!
        self.customParameter = Any
```

Listing 4.8: Simple configuration

In order to actually store the parameter in simulation result, user have to insert a line of code as in snippet 4.9. `qwanta` uses `dill` to store python object in `.pkl` format, this should enable various python objects to be easily stored and read back correctly.

```
# qwanta.py

def run(self, save_tomography=False, save_result=True):

    .
    .
    .

```

```
    config['Custom parameter'] = self.customParameter
```

Listing 4.9: Simple configuration

Lastly, user could uses `self.customParameter` in any processes, for example as shown in 4.10.

```
def _independentProcess(self, info):

        .
        .
        .

        self.customParameter = "Here is what I want to store"
```

Listing 4.10: Simple configuration

CHAPTER V

XXX

5.1 Problem definition

5.1.1 Motivation of problem

Due to the loss of sending photon qubit through quantum channel, the probability of successfully established link-level Bell pair is decrease exponentially.

Problem might arise in the scenario of entanglement swapping with asymmetric value of losses of each edges. For example, in the simplest case where node 0 needed to check quality of quantum channel with node 2 which entanglement swapping with node 1 is necessary for direct connection and loss in the environment is fixed to some value. However, distance between node 0 and node 1 is relatively shorter than another. Therefore, edge $\{0, 1\}$ can share Bell pair relatively easier as the probability of photon arrive at the detector is relatively high compare to edge $\{1, 2\}$.

To uses `qwanta` simulates this problem, consider 2G network using non-local CNOT gate for entangling two remote logical qubits, seven physical Bell pair is needed for each logical Bell pair. Each adjacency edge first try to share seven physical Bell pairs and then each node initialize their logical qubit, non-local CNOT gate is performed subsequently, resulting logical Bell pair. Whenever both edges have a logical Bell pair, those pair will used for entanglement swapping. In this work, logical Bell pairs obtained from entanglement swapping are used for state-tomography. State-tomography also serve as resource recycling process and a main propose of connection. The reason for using non-local CNOT approach is to asses asymmetric waiting time problem at the most Bell pairs demanding protocol currently implemented in `qwanta`.

5.1.2 Evaluation

Due to memory error, the longer the qubits needed to wait before consumed, the higher the probability that they will useless. Waiting time of qubits are quantity of interest, and therefore to be minimized. In this scenario, waiting time of each physical qubits used for logical Bell pair state-tomography are values to be investigated.

At the moment that seven physical Bell pairs is shared, let $t^m_{i,k} = \frac{1}{7}\sum_{n=1}^{7} t^{m,n}_{i,k}$ be an average initialization time of physical qubits used to encode logical qubit measure from the start of simulation, where k is kth set of physical qubit qubits sued to encode kth logical Bell pair; n is an index of each physical qubits in the encoding set; m is an index of each simulated trajectory. We then collect an average measurement time of physical qubits used to encode logical qubit measure from the start of simulation, $t^m_{f,k} = \frac{1}{7}\sum_{n=1}^{7} t^{m,n}_{i,k}$. Hence, an average waiting time of physical qubits (AWTPQ) used to encode the kth logical Bell pair is defined as,

$$\text{AWTPQ} = \frac{1}{N}\sum_{m=1}^{N}(t^m_{f,k} - t^m_{i,k}). \tag{5.1}$$

To empathise effect of link asymmetry, four configurations are simulated where placement of quantum node in middle of network is shift further to the left. Let (x_0, x_1, x_2) be position in km of node 0, 1, and 2 respectively, those four configurations are $(0, 25, 200)$, $(0, 50, 200)$, $(0, 75, 200)$, and $(0, 100, 200)$ for a reference as a symmetry configuration.

A simulation results are present in fig. 5.1, where each configuration uses two rows and each row shows a histogram of AWTPQ. Up-row shows AWTPQ of node 0 and down-row show AWTPQ of node 2. A vertical axis is limit to maximum of 2,000 counts. Gray distributions are histogram of ATWPQ without any adjustment while the colored distributions will be discussed in section 5.2.

In fig. 5.1, without any adjustment shown by gray histogram, distributions of AWTPQ are diverge further as the relative distance between edges is higher. The phenomenon is expected as the short edge can share physical Bell pair faster, hence logical Bell pair is prepared earlier than of longer edge. Therefore, after entanglement swapping, physical qubits of node 0 are older. Which is why AWTPQ of qubits of node 2 is short as they are used for state-tomography as soon as entanglement swapping is finish resulting in stack bar in fig. 5.1. While, for $(0, 100, 200)$, the distributions of both nodes are symmetry as a timings of getting logical Bell pair of both edges are approximately the same.

5.2 Solution: Adjustment of pulse interval

To delivery a young Bell pair to the end-nodes, performance of edges along the path needed to be synchronized. The problem is that, shorter edge, to be precise, the higher Bell pair generation rate, have to wait for the low edges.

The very core idea of a solution in this work is that, instead of work as hard as you can, let just slow down, so that we could proceed to the better together.

Using probability of photon arrive at detector from eq. (4.5), pulse interval of shorter edge, t^{s}_{interval}, that would give Bell pair generation rate approximately the same as the longer edge, t^{l}_{interval}, could be calculate from,

$$t^{(s)}_{\text{interval}} = t^{(l)}_{\text{interval}} \frac{\eta_s}{\eta_l} 10^{\frac{(d_l - d_s)x}{10}}, \tag{5.2}$$

where η is some efficiency constant, d is distance of the edge, x is a loss of quantum channel.

After an adjustment, AWTPQ histograms are present in fig. 5.1 as colored distributions.

The simulation result show that distributions AWTPQ of qubits in node 0 are shift to the left as well as distributions of node 2 which are no longer stack bars as before. A misalignment might come from a classical communication waiting time.

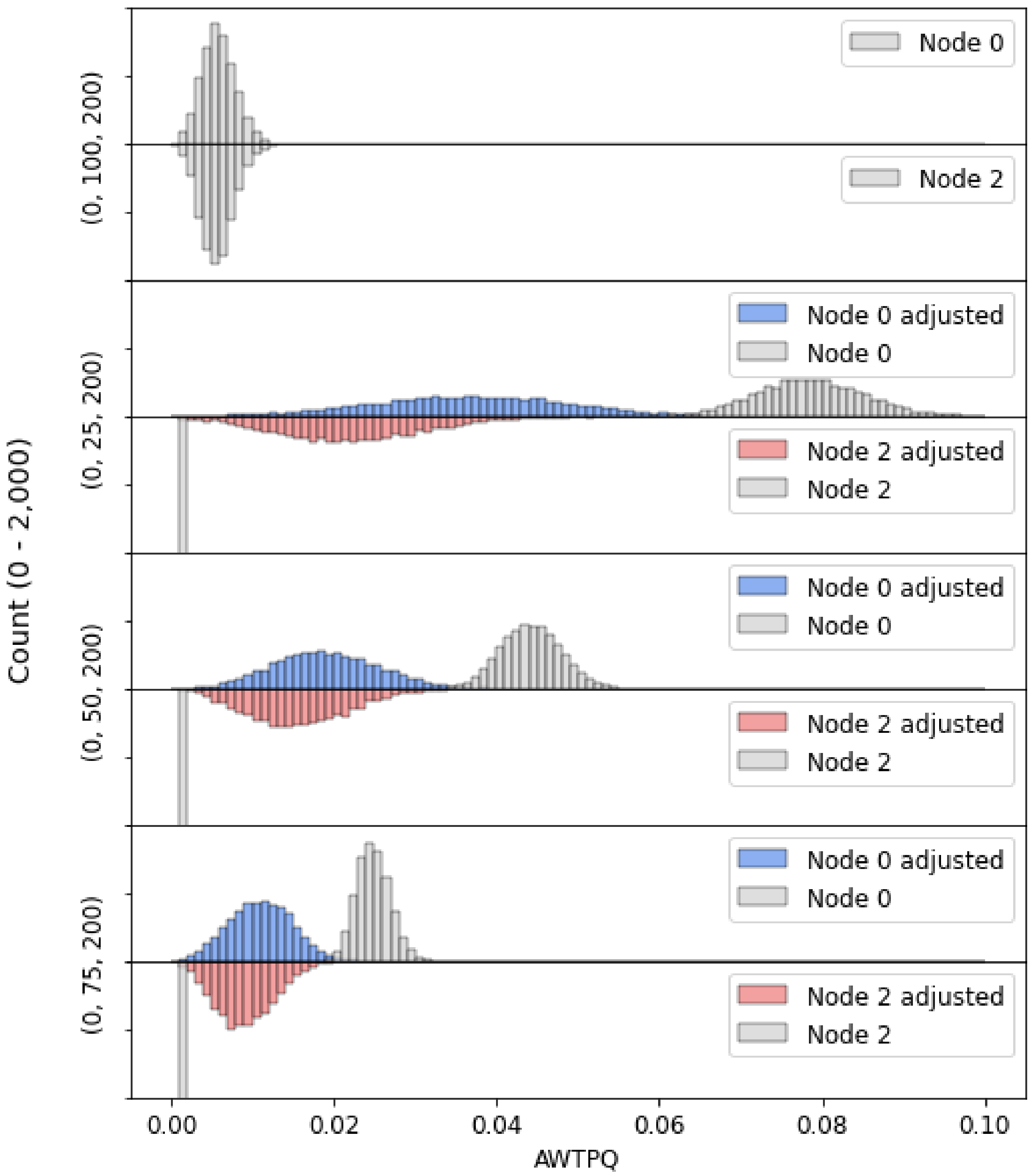

Figure 5.1 Caption

CHAPTER VI

XXX

CHAPTER VII

XXX

7.1 Conclusion

7.2 Outlook

REFERENCES

[1] S. T. Flammia and Y. K. Liu, “Direct fidelity estimation from few Pauli measurements,” *Physical Review Letters*, vol. 106, no. 23, pp. 1–9, 2011.

BIOGRAPHY

NAME	Mr. Poramet Pathumsoot
DATE OF BIRTH	xxth xxx
PLACE OF BIRTH	xxx, Thailand
INSTITUTIONS ATTENDED	Mahidol University, 20xx–20xx
	Bachelor of Science (Physics)
	Mahidol University, 20xx–20xx
	Master of Science (Physics)
SCHOLARSHIP	xx
	Faculty of Science, Mahidol University, 20xx
HOME ADDRESS	xx
	xxx
E-MAIL	xxx.xx@student.mahidol.ac.th