

Master's Thesis (Academic Year 2019)

# Simulation of a Dynamic, RuleSet-based Quantum Network

Keio University  
Graduate School of Media and Governance

Takaaki Matsuo

# Simulation of a Dynamic, RuleSet-based Quantum Network

Similar to the classical Internet, the quantum Internet will require knowledge regarding link qualities used for purposes such as optimal route selection. This is commonly accomplished by performing link-level tomography with or without purification – a.k.a. quantum link bootstrapping. Meanwhile, the gate selection and the resource (Bell pair) selection for a task must be coordinated beforehand. This thesis introduces the RuleSet-based communication protocol aimed for supporting the autonomous coordination of quantum operations among distant nodes, with minimal classical packet transmission. This thesis also discusses the RuleSet-based quantum link bootstrapping protocol, which consists of recurrent purifications and link-level tomography, evaluated over a Markov-Chain Monte-Carlo simulation with noisy systems modeled on real world quality hardware. Given a 10km MeetInTheMiddle based two-node system, each with 100 memory qubits ideally connected to the optical fiber, the Recurrent Single selection - Single error purification (RSs-Sp) protocol is capable of improving the fidelity from an average input  $F_r = 0.675$  to approximately  $F_r = 0.865$ . The system gets noisier with longer channels, in which case errors may develop faster than the purification gain. For a noisier system with a longer channel length, the double selection-based purification shows an advantage for improving the fidelity.

Keywords :

1. Quantum purification, 2. RuleSet, 3. Quantum link bootstrapping,
4. Quantum repeater networks, 5. Coordinated quantum operations

Graduate School of Media and Governance  
Keio University

Takaaki Matsuo

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Background . . . . .	1
1.2	Research Contribution . . . . .	1
1.3	Thesis structure . . . . .	2
<b>2</b>	<b>Theory of Quantum Information<sup>1</sup></b>	<b>3</b>
2.1	Historical background of Quantum Computation . . . . .	3
2.2	Qubit . . . . .	4
2.2.1	Dirac notation . . . . .	4
2.2.2	Bloch sphere . . . . .	6
2.2.3	Eigenvalue and Eigenvector . . . . .	6
2.3	Composite quantum systems . . . . .	7
2.4	Quantum Gates . . . . .	8
2.4.1	Single Qubit Gates . . . . .	8
2.4.2	Measurement in a circuit . . . . .	10
2.4.3	Controlled Gates . . . . .	12
2.5	Alternative state representations . . . . .	13
2.5.1	Density matrix and Mixed States . . . . .	13
2.5.2	Stabilizer . . . . .	15
2.6	Entanglement . . . . .	16
2.6.1	Bell pair/Einstein-Podolsky-Rosen (EPR) pair . . . . .	16
2.6.2	W state . . . . .	17
2.6.3	Greenberger-Horne-Zeilinger (GHZ) state . . . . .	17
2.6.4	Cluster state . . . . .	17
2.7	Reading quantum circuits . . . . .	18
2.7.1	Understanding circuits using matrices . . . . .	18
2.7.2	Understanding circuits using stabilizers . . . . .	24
2.8	Quantum networking . . . . .	25
2.8.1	Quantum Teleportation . . . . .	25
2.8.2	Entanglement Swapping . . . . .	27
2.8.3	Entanglement Purification . . . . .	29
2.8.4	Quantum state tomography . . . . .	30

---

<sup>1</sup>Adapted from my bachelor's thesis, "Analysis of Measurement-based Quantum Network Coding over Repeater Networks under Noisy Conditions", Faculty of Environment and Information Studies, Keio University, 2017.

2.8.5	Verification of quantum correlation (entangled states) via CHSH inequality . . . . .	32
2.8.6	Quantum Repeaters . . . . .	34
2.8.7	Generations of Network Evolution . . . . .	35
2.9	Errors on quantum systems and its measure . . . . .	36
2.9.1	Error . . . . .	36
2.9.2	Fidelity . . . . .	38
2.10	Measurement-based Quantum Computing . . . . .	38
<b>3</b>	<b>Related Work</b>	<b>43</b>
3.1	Violation of a Bell inequality . . . . .	43
3.2	Long range entanglement distribution . . . . .	44
3.3	Physical link architecture . . . . .	44
3.3.1	MeetInTheMiddle . . . . .	44
3.3.2	SenderReceiver . . . . .	45
3.3.3	MidpointSource . . . . .	45
3.4	Quantum data link protocol . . . . .	46
3.5	Quantum link bootstrapping . . . . .	47
<b>4</b>	<b>Protocol Design</b>	<b>50</b>
4.1	Quantum data link protocol . . . . .	50
4.2	RuleSet-based communication protocol . . . . .	51
4.3	Pseudocode of Rules . . . . .	54
4.3.1	Single selection-Single error Purification (Ss-Sp) Rule . . . . .	54
4.3.2	Double selection-Single error Purification (Ds-Sp) Rule . . . . .	56
4.3.3	Full-state link-level tomography Rule . . . . .	58
<b>5</b>	<b>Quantum network simulator</b>	<b>60</b>
5.1	Design of a Quantum node architecture . . . . .	60
5.1.1	Network Interface Card (NIC) . . . . .	62
5.1.2	Quantum Network Interface Card (QNIC) . . . . .	62
5.1.3	Application module . . . . .	62
5.1.4	Hardware Monitor . . . . .	62
5.1.5	Routing Daemon . . . . .	63
5.1.6	Connection Manager . . . . .	65
5.1.7	Rule Engine . . . . .	65
5.1.8	Real-time Controller . . . . .	66
5.2	Design of a Bell State Analyzer (BSA) node/module architecture . . . . .	66
5.2.1	Beamsplitter & Photon detectors . . . . .	66
5.2.2	Controller . . . . .	67
5.3	Simulating errors . . . . .	67
5.3.1	Noisy memory errors . . . . .	67
5.3.2	Noisy channels . . . . .	68
5.3.3	Bell State Analysis (BSA) error . . . . .	68
5.3.4	Noisy quantum gates . . . . .	70

<b>6 Evaluation</b>	<b>71</b>
6.1 Performing simple tomography with different numbers of measurements . . . . .	72
6.2 Performing tomography with single shot purification over different distances	74
6.3 Bootstrapping using Ruleset-based recurrence purification and tomography	79
<b>7 Conclusion</b>	<b>87</b>
<b>Acknowledgements</b>	<b>89</b>
<b>A Other details regarding the Quantum networking simulator</b>	<b>97</b>
A.1 Simulator size . . . . .	97
A.2 How to build and run the simulator . . . . .	100
A.2.1 Run-time configurable parameters . . . . .	100
A.2.2 Output file format . . . . .	102
A.3 How to extend the simulator . . . . .	103
A.3.1 Adding a new Action and a new Clause . . . . .	103
A.3.2 Adding a new quantum gate . . . . .	104
A.3.3 How to add a new protocol (classical packet) . . . . .	104
A.4 Current state of the simulator . . . . .	105
A.4.1 Limitations . . . . .	105
A.4.2 Example run . . . . .	106

# List of Figures

2.1	Quantum State described on a Bloch Sphere . . . . .	6
2.2	Basic flow of quantum gate operation . . . . .	8
2.3	Circuit representation of a Pauli-X gate . . . . .	8
2.4	Circuit representation of a Pauli-Y gate . . . . .	9
2.5	Circuit representation of a Pauli-Z gate . . . . .	9
2.6	Circuit representation of an Hadamard gate . . . . .	9
2.7	Circuit representation of a S gate . . . . .	10
2.8	Circuit representation of a T gate . . . . .	10
2.9	Circuit representation of a Measurement gate and its byproduct operator . . . . .	11
2.10	Circuit representation of a Controlled-NOT gate . . . . .	12
2.11	Circuit representation of a Controlled-Z gate . . . . .	13
2.12	A quantum circuit generating a GHZ state of three qubits . . . . .	19
2.13	Example circuits of an Hadamard gate and a CNOT gate. . . . .	23
2.14	Propagation of stabilizers in the creation of a 3-qubit GHZ state . . . . .	25
2.15	Simple quantum circuit of quantum teleportation . . . . .	26
2.16	Simple quantum circuit of quantum teleportation without byproduct operations	27
2.17	Visualized model of quantum teleportation . . . . .	27
2.18	Simple quantum circuit of entanglement swapping . . . . .	28
2.19	Visualized model of entanglement swapping . . . . .	29
2.20	Visualized model of entanglement purification . . . . .	29
2.21	Architecture of a simple two-node quantum repeater network. . . . .	34
2.22	Bell measurement using a beam splitter and photon detectors. . . . .	35
2.23	Error propagation of Hadamard gate . . . . .	36
2.24	Error propagation of Controlled-Not gate . . . . .	37
2.25	Error propagation of Controlled-Z gate . . . . .	37
2.26	Error propagation of measurement . . . . .	37
2.27	2D resource state generation . . . . .	39
2.28	Quantum circuit for one-bit teleportation . . . . .	39
2.29	One-bit teleportation to Measurement-based Quantum Computing . . . . .	40
2.30	Measurement-based single qubit unitary operation . . . . .	40
2.31	Measurement-based CNOT operation . . . . .	40
2.32	Topological transition by measurements on 1D cluster states . . . . .	41
2.33	Topological transition by measurements on 2D cluster states . . . . .	42
3.1	Chinese satellite beaming down entangled photons to earth . . . . .	44
3.2	The MeetInTheMiddle model. . . . .	45

3.3	The SenderReceiver model. . . . .	45
3.4	The MidpointSource model. . . . .	46
3.5	Required functionality for each layer network stack . . . . .	46
3.6	Layered architecture for quantum communication . . . . .	47
3.7	Procedures of quantum link bootstrapping. . . . .	48
3.8	Procedure of quantum link bootstrapping with purification. . . . .	49
4.1	Flow chart of resource generation. . . . .	51
4.2	Node A generating and distributing RuleSets . . . . .	52
4.3	Bias in knowledge regarding the state of resources . . . . .	53
4.4	An example structure of a RuleSet . . . . .	53
4.5	Flow chart of the hardware monitor working on link-level tomography.x . . . . .	54
5.1	The Quantum Internet. Different networks are operated under different technologies. That includes the physical link architecture, data link protocol, routing protocol and any other necessary technology layered as in the OSI reference model [1]. Connecting those networks forms the Quantum Internet.	61
5.2	Quantum node architecture. . . . .	62
5.3	Message exchange of the routing daemon exchanging link qualities. . . . .	64
5.4	Example topology based on the routing table in Tab.5.3. . . . .	65
5.5	Flow chart of the connection setup including three nodes. . . . .	65
5.6	BSA node architecture. . . . .	66
5.7	Bell measurement with a Y error on the received photon. . . . .	69
5.8	Bell measurement with a lost error on the received photon. . . . .	70
6.1	Impact of the number of measurements on the reconstructed fidelity error over the MeetInTheMiddle link . . . . .	73
6.2	Change in the accuracy of the reconstructed fidelity relative to the actual fidelity. . . . .	74
6.3	Single selection - Single error (X) purification (Ss-Sp) . . . . .	75
6.4	Single selection - Double error (XZ) purification (Ss-Dp). . . . .	75
6.5	Double selection - Single error (X) purification (Ds-Sp) . . . . .	76
6.6	Double selection - Double error (XZ) purification (Ds-Dp). . . . .	76
6.7	Impact of the MeetInTheMiddle channel length on the reconstructed fidelity with 7000 measurement outcomes w/o, with purification. . . . .	77
6.8	Impact of the channel distance on the channel throughput with and without performing purification. . . . .	78
6.9	Recurrence purification protocol based on single selection X purification and Z purification. . . . .	80
6.10	Flowchart of the quantum link bootstrapping. The termination process due to timeout is omitted from the chart. We need $1 - F(\rho_{\text{reconstructed}}, \rho_{\text{real}}) \ll 1 - F(\rho_{\text{real}}, \rho_{\text{ideal}})$ in order to make high-quality decisions about next operations in real time. . . . .	81
6.11	Reconstructed fidelity with actual fidelity and actual error rates, and estimated throughput of protocols over $L = 10\text{km}$ MeetInTheMiddle link. . . . .	83

6.12	Impact of the memory buffer size to the maximum fidelity purified through the RSs-Sp protocol over a 10km MeetInTheMiddle link. . . . .	84
6.13	Reconstructed fidelity with actual fidelity and actual error rates, and estimated throughput of protocols over $L = 20\text{km}$ MeetInTheMiddle link. . . . .	85
6.14	Switching RDs-Sp to RSs-Sp over $L = 20\text{km}$ . . . . .	86
A.1	Network topology defined as Realistic_Layer2_Simple_MIM_MM_5km. .	108

# List of Tables

2.1	Different entangled states and the corresponding stabilizer sets . . . . .	16
2.2	Stabilizers of the input state. . . . .	24
2.3	Stabilizers of the output state after Step 1. . . . .	24
2.4	Stabilizers of the output state after Step 2. . . . .	24
2.5	Stabilizers of the output state after Step 3. . . . .	25
2.6	Byproduct operations to complete quantum teleportation . . . . .	26
2.7	Byproduct operations to complete entanglement swapping . . . . .	28
2.8	Entanglement swapping with noisy Bell pairs . . . . .	30
2.9	Bell measurement outcomes. The horizontal polarization $ H\rangle$ may be used for representing $ 0\rangle$ , and $ V\rangle$ for $ 1\rangle$ . The subscript <i>L</i> denotes the <i>Left photon</i> , and <i>R</i> denotes the <i>Right photon</i> . . . . .	35
2.10	Byproduct operation to complete one-bit teleportation . . . . .	39
5.1	Example of a table keeping the pair of the local QNIC id and its connected neighbor address. Neighboring nodes of Node 1 are assumed to be Node 2, 3 and 4 . . . . .	63
5.2	Example of a table keeping the link fidelity and the link throughout information. . . . .	63
5.3	Example quantum routing table of Node 1. . . . .	64
6.1	Default parameters used for all simulations (unless explicitly mentioned). .	72
A.1	Major Processor 0 specification. . . . .	108

# Chapter 1

## Introduction

This chapter focuses on the research background, novelty/contribution of this thesis and the structure of the remaining chapters<sup>1</sup>.

### 1.1 Background

The quantum Internet, interconnecting distinct quantum repeater networks [3], is a promising technology that can provide us new capabilities not reproducible by classical systems [4–7]. One of the best known examples is quantum key distribution [8–11], which is designed to utilize quantum mechanics to securely share strings of random bits suitable for use as encryption keys. Conducting a statistical experiment, such as Bell’s inequality [12], allows participants to detect the presence of an eavesdropper. Other example applications include quantum blind computing [13, 14] and quantum clock synchronization [15, 16].

Similar to the classical Internet, establishing a multi-hop quantum connection requires quantified link characteristics for purposes such as optimal route selection. Such characterization is commonly achieved by performing tomography over generated Bell pairs – a.k.a. quantum link bootstrapping. In any situation, distant nodes need to agree the selected operation and the targeted resource for completing the task. The simplest solution is to exchange classical messages at each attempt for guaranteeing synchronization but a more practical solution with minimal message transmission has not yet been discussed carefully in the field.

### 1.2 Research Contribution

This thesis introduces RuleSets, which can be used to autonomously coordinate quantum operations among multiple nodes. This thesis also adapts the above protocol to quantum link bootstrapping, and simulates the process based on a carefully designed Markov-Chain Monte-Carlo simulation modeled on currently available hardware. The network simulator dynamically executes the Physical Layer protocol, the Data Link Layer protocol and the Networking Layer protocol to realistically assess the capability of the available technology set. The simulation results successfully showed the importance of the selected purification method within the recurrence purification – the selected purification method may work the

---

<sup>1</sup>Portions of this thesis has been presented in [2]

best for a particular case, but not for others. This thesis also introduces a technique to switch the purification method in the middle of the recurrence purification, which can be beneficial in terms of fidelity and throughput. The bootstrapping process, therefore, also needs to check which purification works the best for the particular link.

### 1.3 Thesis structure

The remaining of the thesis is constructed as follows. In Chapter 2, basics of quantum information are provided as a preliminary to support the readers with minimal knowledge. In Chapter 3, current status of experimental work is introduced. In Chapter 4, several related publications are briefly explained. In Chapter 5, the protocol design of the RuleSet-based communication protocol is explained. In Chapter 6, the some details regarding the simulation is provided. In Chapter 7, main quantum link bootstrapping results of Markov-Chain Monte-Carlo simulations are discussed with various settings over multiple protocols. Finally in Chapter 7, this thesis is concluded with some discussions regarding the future work.

# Chapter 2

## Theory of Quantum Information<sup>1</sup>

### 2.1 Historical background of Quantum Computation

In 1982, a physicist named Richard Feynman noted [17] that it is generally not feasible to represent the results of quantum mechanics with a classical universal device. The newly introduced concept, quantum simulation, took the advantage of the puzzling quantum effects to effectively simulate physics, which cannot be handled by ordinary computers regardless of their computation power.

The term "Quantum Computer" was officially used in print for the first time in 1985, by physicist David Deutsch [18]. He proposed a mathematical concept of a strictly modeled universal computer based on quantum mechanics with many properties not reproducible by classical Turing machines, and generalized computing methods for quantum computers.

Contemporaneously, quantum networking appeared as a subfield of quantum computing. The algorithm named BB84, Quantum key distribution (QKD), was firstly proposed by Bennett and Brassard [19] in 1984, and came into the experimental forefront in 1989 [20]. The algorithm was widely recognized across the globe, as a result of its promising security by exploiting quantum mechanics compared to the classical technology.

In 1994, Peter W. Shor [21] at Bell Laboratories introduced a quantum algorithm that has the capability of factoring large numbers within polynomial time – known as Shor's algorithm. The algorithm essentially showed that quantum computers have the ability to break commonly used classical cryptography techniques based on prime number factorization such as the RSA cryptosystem.

The theoretical proposal of quantum teleportation, which is a technique to map an arbitrary state of qubit to another, was introduced by Bennett et al. [22] in 1993. Some years later, it was successfully demonstrated experimentally and became one of the essential ingredients for quantum networking [23,24]. In order to achieve long distance quantum communication, intermediate nodes called "quantum repeaters" were introduced by Briegel and Dür [3,25] in the late 1990s, as a tool for managing errors, creating entanglement and enabling multi-hop communications. Nevertheless, establishing a stable quantum communication over long distances still remains an outstanding challenge due to technical problems such as the operation

---

<sup>1</sup> Adapted from my bachelor's thesis, "Analysis of Measurement-based Quantum Network Coding over Repeater Networks under Noisy Conditions", Faculty of Environment and Information Studies, Keio University, 2017.

errors and qubit degradation known as decoherence.

## 2.2 Qubit

The indivisible unit of a classical information is known as *binary digit* or *bit*. A single bit has a single binary value that is not limited to, but in general expressed with *0* and *1*. The two values of a bit in a classical computer may be represented by the electric charge stored in a capacitor, the direction of a magnetic field, or anything else that is capable of physically representing two values. Similarly, the smallest unit of a quantum information is known as *quantum bit* or *qubit*<sup>2</sup>. Unlike a bit, a single qubit can be in a *superposition state* of two states, simultaneously representing *0* and *1*. The two-level system of a quantum computer may be represented by the vertical polarization and the horizontal polarization of a photon, the spin up and the spin down of an electron, or any other proposed state variables.

### 2.2.1 Dirac notation

The simplest quantum system is a two-state system, and the single qubit pure state can be expressed by using the *Dirac notation*:

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

$$|\alpha|^2 + |\beta|^2 = 1 \quad (2.1)$$

Coefficients  $\alpha$  and  $\beta$  are arbitrary complex numbers representing the probability amplitudes. The  $|0\rangle$  and  $|1\rangle$  are called *kets* and denote the two possible states, where the probability of the state being  $|0\rangle$  can be found by the quantity  $\alpha\alpha^* = |\alpha|^2$ , and  $\beta\beta^* = |\beta|^2$  for  $|1\rangle$ . As an example,  $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$  indicates that the state  $|\psi\rangle$  is in a superposition of two states  $|0\rangle$  and  $|1\rangle$  with equally weighted probabilities.

A quantum state can also be described by a vector in a two dimensional complex Hilbert space. State vectors of a single qubit may be:

Z-basis state

$$|0\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.2)$$

Therefore, the superposition state of a single qubit can be expressed by the following two-dimensional vector.

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \quad (2.3)$$

---

<sup>2</sup>The term qubit just had its 25th anniversary (2017).

The computational vectors introduced in equation 2.2 are called the *Z-basis states*, and a linear combination of them can be used to express any pure quantum state of one qubit. The alternative computational bases, the *X-basis states* and the *Y-basis states* are equally important. The *X-basis states* are based on the superposition states:

(2.4)

**X-basis state**

$$|+\rangle \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad |-\rangle \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \quad (2.5)$$

Similarly, the *Y-basis states* uses the complex bases:

(2.6)

**Y-basis state**

$$|+i\rangle \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ i \end{bmatrix} \quad |-i\rangle \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -i \end{bmatrix}. \quad (2.7)$$

Contrasting to *ket*,  $\langle \psi |$  is called *bra* which represents the conjugate transpose of  $|\psi\rangle$ .

$$|\alpha\rangle = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_n \end{bmatrix} \quad \langle \alpha | = [\alpha_1^* \quad \alpha_2^* \quad \alpha_3^* \quad \cdots \quad \alpha_n^*] \quad (2.8)$$

The inner product and the outer product of states are:

$$|\alpha\rangle \langle \alpha | = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_n \end{bmatrix} \cdot [\alpha_1^* \quad \alpha_2^* \quad \alpha_3^* \quad \cdots \quad \alpha_n^*] = \begin{bmatrix} \alpha_1\alpha_1^* & \alpha_1\alpha_2^* & \alpha_1\alpha_3^* & \cdots & \alpha_1\alpha_n^* \\ \alpha_2\alpha_1^* & \alpha_2\alpha_2^* & \alpha_2\alpha_3^* & \cdots & \alpha_2\alpha_n^* \\ \alpha_3\alpha_1^* & \alpha_3\alpha_2^* & \alpha_3\alpha_3^* & \cdots & \alpha_3\alpha_n^* \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_n\alpha_1^* & \alpha_n\alpha_2^* & \alpha_n\alpha_3^* & \cdots & \alpha_n\alpha_n^* \end{bmatrix} \quad (2.9)$$

$$\langle \alpha | \alpha \rangle = [\alpha_1^* \quad \alpha_2^* \quad \alpha_3^* \quad \cdots \quad \alpha_n^*] \cdot \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_n \end{bmatrix} = \alpha_1^*\alpha_1 + \alpha_2^*\alpha_2 + \alpha_3^*\alpha_3 + \cdots + \alpha_n^*\alpha_n \quad (2.10)$$

Note that the diagonal elements of  $|\alpha\rangle \langle \alpha |$  are real and non-negative.

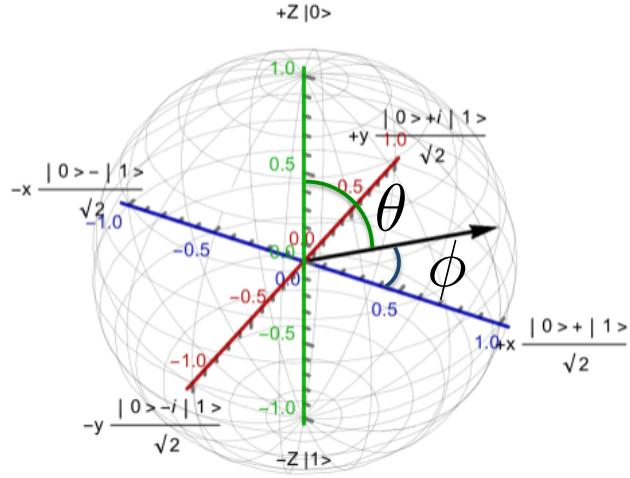


Figure 2.1: Quantum State described on a Bloch Sphere

### 2.2.2 Bloch sphere

The Bloch sphere is a geometric representation of a single qubit pure state as a unit vector pointing on the surface of a unit sphere.

An arbitrary single qubit state can be written:

$$|\psi\rangle = e^{i\lambda} \left( \cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle \right) \quad (2.11)$$

$$0 \leq \theta \leq \pi$$

$$0 \leq \phi \leq 2\pi$$

where variables  $\theta$ ,  $\phi$  and  $\lambda$  are real numbers. The number  $\theta$  represents the latitude with respect to the Z-axis and  $\phi$  represents the longitude with respect to the Y-axis. Together they define a point on the Bloch sphere surface. The variable  $e^{i\lambda}$  is known as the global phase of a quantum state, and has no observable effects. Therefore, equation 2.11 can simply be rewritten as:

$$|\psi\rangle = \cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle \quad (2.12)$$

Hence,  $\cos \frac{\theta}{2}$  and  $e^{i\phi} \sin \frac{\theta}{2}$  correspond to  $\alpha$  and  $\beta$  in equation 2.1 with  $\alpha$  constrained to be real.

### 2.2.3 Eigenvalue and Eigenvector

The *eigenvalue* and the *eigenvector* are important concepts for quantum information.

As an example, if an operator  $U$  acts on a vector  $|\psi\rangle$  and the result can be rearranged to a scalar  $\lambda$  and the same vector  $|\psi\rangle$ :

$$U|\psi\rangle = \lambda|\psi\rangle \quad (2.13)$$

Such  $\lambda$  is called the *eigenvalue* and the corresponding vector  $|\psi\rangle$  is called the *eigenvector*. In general, the eigenvalue is a complex number but here we often see it as a real number,  $\pm 1$ . For example, performing a Pauli-Z gate on a state  $|1\rangle$  will result in:

$$Z|1\rangle = -|1\rangle \quad (2.14)$$

where in this case, the eigenvector of  $Z$  is  $|1\rangle$  with an eigenvalue  $-1$ . We will see eigenvalues and eigenvectors when we discuss measurement (see Sec.2.4.2).

## 2.3 Composite quantum systems

In reality, a system may contain more than one qubit. An independent two-qubit system may have states:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (2.15)$$

$$|\phi\rangle = \gamma|0\rangle + \delta|1\rangle \quad (2.16)$$

Here, the joint system can be described by taking the tensor product of equation 2.15 and equation 2.16.

$$|\psi\rangle \otimes |\phi\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle = \begin{bmatrix} \alpha\gamma \\ \alpha\delta \\ \beta\gamma \\ \beta\delta \end{bmatrix} \quad (2.17)$$

$$|\alpha\gamma|^2 + |\alpha\delta|^2 + |\beta\gamma|^2 + |\beta\delta|^2 = 1$$

where  $\otimes$  is the tensor product of two vectors, and  $|00\rangle$  equates to  $|0\rangle|0\rangle$ . The tensor product of two vectors is:

$$\begin{bmatrix} a_1 \\ b_1 \end{bmatrix} \otimes \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_1 \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} \\ b_1 \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_1a_2 \\ a_1b_2 \\ b_1a_2 \\ b_1b_2 \end{bmatrix} \quad (2.18)$$

The vector representation of the two-qubit computational basis states is:

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (2.19)$$

Similarly, three qubits can be in eight states, and  $n$  qubits can be in a superposition of all  $2^n$  states simultaneously.

$$\sum_{i=0}^{2^n-1} \alpha_i |i\rangle \quad (2.20)$$

$$\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$$

## 2.4 Quantum Gates

As modern computers work based on Boolean logic gates, quantum computers performs similar gate operations to manipulate quantum information (see Figure 3.1 for a simplified model of gate operations). Such gates are often called unitary gates, as they give unitary transformation of the qubit states. Gates are unitary when  $U^\dagger U = UU^\dagger = I$ . Quantum gates are reversible and can be represented as unitary matrices.

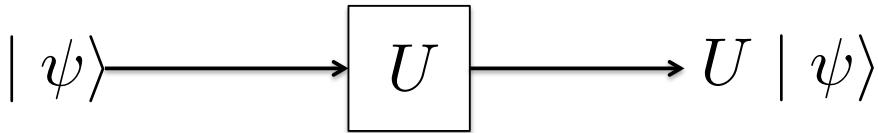


Figure 2.2: Basic flow of quantum gate operation

### 2.4.1 Single Qubit Gates

The most important operators for quantum computing are called the Pauli operators.

The Pauli-X gate is the equivalent of the classical NOT gate. The gate can be performed on a single qubit state and swaps the probability amplitude of  $|0\rangle$  and  $|1\rangle$ .

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (2.21)$$

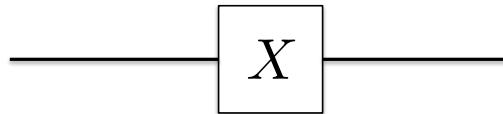


Figure 2.3: Circuit representation of a Pauli-X gate

Therefore,  $|0\rangle = X|1\rangle$  and  $|1\rangle = X|0\rangle$ .

The Pauli-Y gate changes the quantum state  $|0\rangle$  to  $i|1\rangle$ , and  $|1\rangle$  to  $-i|0\rangle$ .

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad (2.22)$$

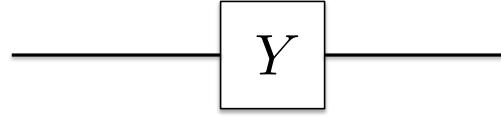


Figure 2.4: Circuit representation of a Pauli-Y gate

The Pauli-Z gate, or sometimes called the phase-flip gate, does not affect the basis state  $|0\rangle$ , but changes  $|1\rangle$  to  $-|1\rangle$ , and  $-|1\rangle$  to  $|1\rangle$ .

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (2.23)$$

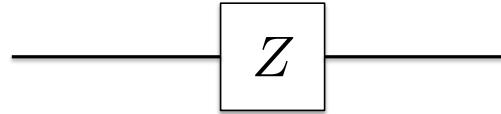


Figure 2.5: Circuit representation of a Pauli-Z gate

Besides Pauli operators, the Hadamard gate and the phase gate are equally important operators. These gates can be used to transform between the different basis states.

The Hadamard gate is as shown below.

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (2.24)$$

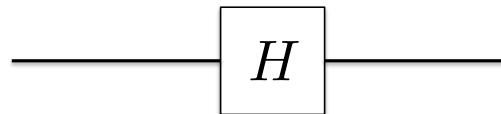


Figure 2.6: Circuit representation of an Hadamard gate

Applying an Hadamard gate to a qubit will result in:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle \quad (2.25)$$

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = |-\rangle \quad (2.26)$$

$$H|+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle \quad (2.27)$$

$$H|-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle \quad (2.28)$$

Notice that an Hadamard gate can bring up a qubit into a superposition state from a basis state, or vice versa. Moreover, the X gate can be constructed by conjugating two Hadamard gates and a Z gate:  $X = HZH$ .

Similar to the Z gate, the phase shift gate changes the phase of a quantum state, from  $|1\rangle$  to  $e^{i\phi}|1\rangle$ . Therefore, if  $\phi = \pi$ , the phase gate performs on a qubit the same way as the Z gate. The T gate and the S gate are for specific, defined values of  $\phi$ .

$$Z_\phi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \quad (2.29)$$

$$S = Z_{\frac{\pi}{2}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{bmatrix} \quad (2.30)$$

$$T = Z_{\frac{\pi}{4}} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix} \quad (2.31)$$

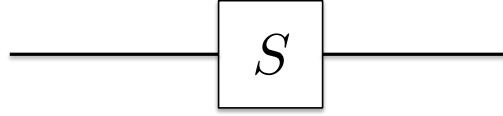


Figure 2.7: Circuit representation of a S gate

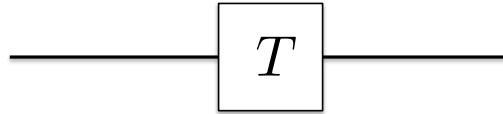


Figure 2.8: Circuit representation of a T gate

Two S gates and an X gate can be used to construct a Y gate:  $Y = SXS^\dagger$ . A gate that rotates the quantum state by 180 degrees is not affected by the rotational direction; clockwise or anti-clockwise. As an S gate performs a rotation of less than 180 degrees, the direction of rotation matters. The clockwise rotation is represented by the  $S$  gate, and the anti-clockwise rotation is represented by the dagger of the gate  $S^\dagger$  or sometimes  $S^-$ .

The generalized rotation operators, which rotate the Bloch vector about the X, Y and Z-axis by a given angle  $\theta$ , can be described as  $R_P(\theta) = e^{-i\frac{\theta}{2}P}$ , where P represents the axis.

$$R_X(\theta) = e^{-i\frac{\theta}{2}X} = \cos \frac{\theta}{2}I - i \sin \frac{\theta}{2}X = \begin{bmatrix} \cos \frac{\theta}{2} & -i \sin \frac{\theta}{2} \\ -i \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{bmatrix} \quad (2.32)$$

$$R_Y(\theta) = e^{-i\frac{\theta}{2}Y} = \cos \frac{\theta}{2}I - i \sin \frac{\theta}{2}Y = \begin{bmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{bmatrix} \quad (2.33)$$

$$R_Z(\theta) = e^{-i\frac{\theta}{2}Z} = \cos \frac{\theta}{2}I - i \sin \frac{\theta}{2}Z = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{bmatrix} \quad (2.34)$$

$$(2.35)$$

## 2.4.2 Measurement in a circuit

The measurement result of an arbitrary qubit, if entangled with another, decides the residual quantum state. That is to say, two post-measurement residue states may not equate depending

on the measurement results, even with two identical pre-measurement quantum systems. In some operations, measurement requires a classical feedforward operation to another qubit to fix the state to a wanted form. Sometimes these operators are applied to more than one qubit. Those operators are often called *byproduct operators* and are performed based on classically sent measurement results. As shown in the circuit representation in Figure 2.9, the classical message transmission is generally described with double lines interconnecting the measurement operator and the byproduct operator.

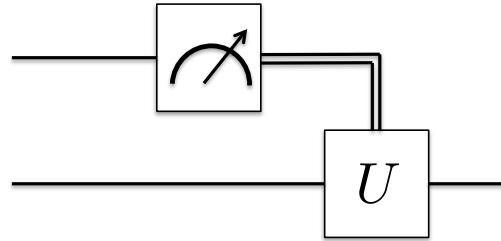


Figure 2.9: Circuit representation of a Measurement gate and its byproduct operator

### Measuring a qubit

Measuring a qubit will destroy the superposition state and determine the state in a classical fashion. The measurement outcome of a quantum state is the eigenvalue of the eigenvector.

As an example, if the qubit state is:

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \quad (2.36)$$

the probability amplitude is equally weighted with respect to the Z-basis. Therefore, measuring the state in Z-basis will return the observer either  $+Z$  or  $-Z$  with a probability of 50%. Changing the measurement basis may affect the measurement result. That is to say, if we use the X-basis for the measurement instead of Z, then the measurement result will always be  $+X$ .

If we measure the qubit  $|\psi\rangle$  in the Z-basis, measurement outcomes 0 and 1 can be obtained with probabilities:

$$P(0) = |\langle 0 | \psi \rangle|^2 = \text{Tr}[|0\rangle \langle 0 | \psi \rangle \langle \psi |] \quad (2.37)$$

$$P(1) = |\langle 1 | \psi \rangle|^2 = \text{Tr}[|1\rangle \langle 1 | \psi \rangle \langle \psi |] \quad (2.38)$$

More generally, the above equation can be expressed using measurement operators  $M_i$ .  $P(i)$  is the probability of obtaining the measurement outcome  $i$ .

$$P(i) = \langle \psi | M_i^\dagger M_i | \psi \rangle = \text{Tr}[M_i^\dagger M_i | \psi \rangle \langle \psi |] = \text{Tr}[E_i | \psi \rangle \langle \psi |] \quad (2.39)$$

where  $E_i$  is a set of positive operators such that  $\sum_i E_i = I$ .

The measurement operator can be obtained by:

$$M_i^\pm = \frac{1}{2}(I \pm U) \quad (2.40)$$

where  $U$  in this case is an element of the matrix set  $U = \{ Z, Y, X \}$ . As an example, the Z-basis measurement operators can be obtained by:

$$M_Z^+ = \frac{1}{2}(I + Z) = \frac{1}{2}(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad (2.41)$$

$$M_Z^- = \frac{1}{2}(I - Z) = \frac{1}{2}(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}) = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.42)$$

### 2.4.3 Controlled Gates

Not all gates work on a single qubit; some act on two or more qubits. One example of such a gate is the Controlled-NOT (CNOT) gate (see Fig. 2.10), which performs the X gate on one qubit (*the target qubit*), when the other qubit's (*the control qubit*) state is  $|1\rangle$ . The CNOT gate is defined by:

$$\Lambda_{c,t}(X) |i_c\rangle |j_t\rangle = |i_c\rangle |i \oplus j_t\rangle \quad (2.43)$$

$$i, j = 0, 1 \quad (2.44)$$

And the corresponding matrix is:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (2.45)$$

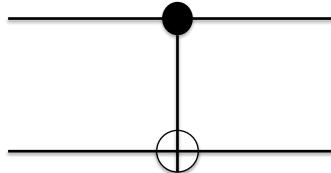


Figure 2.10: Circuit representation of a Controlled-NOT gate

Similarly, the Controlled-Z (CZ) gate performs the Z operation on the target qubit, when the controlled qubit's state is  $|1\rangle$ . The CZ gate is defined by:

$$\Lambda_{c,t}(Z) |i_c\rangle |j_t\rangle = (-1)^{ij} |i_c\rangle |j_t\rangle \quad (2.46)$$

$$i, j = 0, 1 \quad (2.47)$$

The CZ gate is symmetric,  $\Lambda_{c,t}(Z) = \Lambda_{t,c}(Z)$  and the corresponding matrix is:

$$CZ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \quad (2.48)$$

As shown in Fig. 2.11, the CZ gate can also be constructed by two Hadamard gates and one CNOT gate.

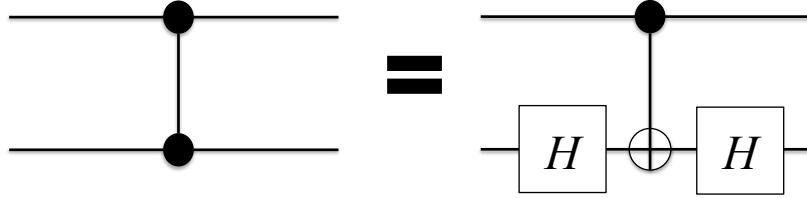


Figure 2.11: Circuit representation of a Controlled-Z gate

## 2.5 Alternative state representations

### 2.5.1 Density matrix and Mixed States

Quantum states are either mixed or pure. A pure state is in a closed system, with no interaction with the outside world. In contrast, a state is mixed when a part of the quantum system becomes entangled with or is acted upon in unknown ways by the environment. While any pure state can be written in state-vector form, mixed states can only be described using a *density matrix*. The density matrix of a pure state  $|\psi\rangle$  can be found by:

$$\rho = |\psi\rangle \langle \psi| \quad (2.49)$$

The corresponding probability of a state  $|j\rangle$  can be found by the diagonal entries of  $\rho$ :

$$P(|j\rangle) = \rho_{jj} |j\rangle \langle j| \quad (2.50)$$

If the state is pure,  $\rho^2 = \rho$  and  $\text{Tr}[\rho^2] = 1$  and if the state is mixed  $\rho^2 \neq \rho$  and  $\text{Tr}[\rho^2] < 1$ . For example, the density matrix of a pure state Bell pair is:

$$\begin{aligned} \rho &= |\psi\rangle \langle \psi| = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \frac{1}{\sqrt{2}}(\langle 00| + \langle 11|) \\ &= \frac{1}{2}(|00\rangle \langle 00| + |00\rangle \langle 11| + |11\rangle \langle 00| + |11\rangle \langle 11|) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix} \end{aligned} \quad (2.51)$$

An example of a completely mixed state of 2 qubits, which represents the classical dependent probabilities is:

$$\rho = \frac{1}{2}(|00\rangle\langle 00| + |11\rangle\langle 11|) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad (2.52)$$

With a completely mixed state of 2 qubits, there is no entanglement between the two qubits – each has a state,  $|0\rangle$  or  $|1\rangle$ . The off-diagonal elements are *quantum coherences* and can be complex and the diagonal elements must be real.

Given a set of probabilities per state, we can take the sum of each weighted density matrix to construct the entire system's density matrix. For example, the density matrix of a Bell pair (see Sec.2.6) with 10% bit flip error rate can be described as:

$$\rho = 0.9|\Psi^+\rangle\langle\Psi^+| + 0.1|\Phi^+\rangle\langle\Phi^+|. \quad (2.53)$$

More generally,  $\rho = \sum P(i)|\psi_i\rangle\langle\psi_i|$  such that  $\sum P(i) = 1$  where  $P(i)$  is the probability of the system being in the state  $|\psi_i\rangle$ . Note that the set  $\{|\psi_i\rangle\}$  may be but does not have to be a complete basis set.

### Calculating the density matrix of a left-over system after a measurement

Qubit measurement and its outcome affect the state of the left-over system. For example, a system may be composed of three qubits, entangled as a W state  $|\psi\rangle = \frac{1}{\sqrt{3}}(|0_A0_B1_C\rangle + |0_A1_B0_C\rangle + |1_A0_B0_C\rangle)$  (see section 2.6.2). The density matrix of the system is therefore:

$$\begin{aligned} \rho = |\psi\rangle\langle\psi| &= \begin{bmatrix} 0 \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ 0 \\ \frac{1}{\sqrt{3}} \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 & \frac{1}{\sqrt{3}} & 0 & 0 & 0 \\ \frac{1}{\sqrt{3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{3}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \quad (2.54)$$

When qubit A is measured, say along the Z axis, the measurement outcome (or its state) will be either  $|0_A\rangle$  or  $|1_A\rangle$  and qubit A will no longer be entangled with qubit B and C. As discussed in section 2.4.2,  $P(|0_A\rangle)$  and  $P(|1_A\rangle)$  can be derived as:

$$P(|0_A\rangle) = \text{Tr}[(M_{ZA}^+ \otimes I_B \otimes I_C)^\dagger \cdot (M_{ZA}^+ \otimes I_B \otimes I_C) \cdot \rho] = 2/3 \quad (2.55)$$

$$P(|1_A\rangle) = \text{Tr}[(M_{ZA}^- \otimes I_B \otimes I_C)^\dagger \cdot (M_{ZA}^- \otimes I_B \otimes I_C) \cdot \rho] = 1/3. \quad (2.56)$$

The overall density matrix of all three qubits, after performing the measurement on qubit A, can be derived as:

$$|\psi'\rangle = \begin{cases} (M_Z^+ \otimes I_B \otimes I_C) \rho (M_Z^+ \otimes I_B \otimes I_C)^\dagger, & A \rightarrow |0\rangle \\ (M_Z^- \otimes I_B \otimes I_C) \rho (M_Z^- \otimes I_B \otimes I_C)^\dagger, & A \rightarrow |1\rangle \end{cases} \quad (2.57)$$

$$(2.58)$$

The above density matrix still includes qubit A. The reduced density matrix of the left-over state,  $|\psi'_{B,C}\rangle$ , can also be calculated in a similar way:

$$|\psi'_{B,C}\rangle = \begin{cases} (\langle 0_A | \otimes I_B \otimes I_C) \rho (\langle 0_A | \otimes I_B \otimes I_C)^\dagger, & A \rightarrow |0\rangle \\ (\langle 1_A | \otimes I_B \otimes I_C) \rho (\langle 1_A | \otimes I_B \otimes I_C)^\dagger, & A \rightarrow |1\rangle \end{cases} \quad (2.59)$$

$$(2.60)$$

## 2.5.2 Stabilizer

A quantum state may be described by a state vector or a density matrix. If we use the ordinary vector notation, the size of the vector grows exponentially in the number of qubits, as we saw in Sec.2.3. The bra-ket notation allows us to dispense with writing down the non-zero terms. However, many states we create in a quantum computer have an exponentially large number of non-zero terms. An alternative way to represent a pure state is to use a set of stabilizers. A stabilizer  $S$  of a state  $|\psi\rangle$  is:

$$S |\psi\rangle = |\psi\rangle \quad (2.61)$$

Consequently, a stabilizer or a set of stabilizers uniquely determines a quantum state. Below are examples of stabilizers of different states:

$$Z |0\rangle = |0\rangle \quad -Z |1\rangle = |1\rangle \quad (2.62)$$

$$X |+\rangle = |+\rangle \quad -X |-\rangle = |-\rangle \quad (2.63)$$

$$Y |+i\rangle = |+i\rangle \quad -Y | -i\rangle = | -i\rangle \quad (2.64)$$

Stabilizers of major entangled states are shown in Tab.2.1.

Table 2.1: Different entangled states and the corresponding stabilizer sets

Quantum state	Stabilizer sets
$\frac{1}{\sqrt{2}}( 0_10_2\rangle +  1_11_2\rangle)$	$X_1X_2, Z_1Z_2$
$\frac{1}{\sqrt{2}}( 0_10_2\rangle -  1_11_2\rangle)$	$-X_1X_2, Z_1Z_2$
$\frac{1}{\sqrt{2}}( 0_11_2\rangle +  1_10_2\rangle)$	$X_1X_2, -Z_1Z_2$
$\frac{1}{\sqrt{2}}( 0_11_2\rangle -  1_10_2\rangle)$	$-X_1X_2, -Z_1Z_2$
$\frac{1}{\sqrt{2}}( 0_10_20_3\rangle +  0_10_20_3\rangle)$	$X_1X_2X_3, Z_1Z_2, Z_2Z_3$
$\frac{1}{\sqrt{2}}( 0_1+2\rangle +  1_1-2\rangle)$	$X_1Z_2, Z_1X_2$
$\frac{1}{2}( 0_10_2+3\rangle +  0_11_2-3\rangle +  1_11_2+3\rangle +  1_11_2-3\rangle)$	$X_1Z_2, Z_1X_2Z_3, Z_2X_3$

Not all states can be stabilized by the Pauli operators. So  $|\psi\rangle$  is the eigenvector of all stabilizers in the set. A quantum state with  $n$  qubits can be fully described by  $n$  stabilizers. Stabilizers can be helpful when describing a state consisting of many qubits, as the number of stabilizers grows only linearly while other methods grow exponentially. On the other hand, finding the correct stabilizer set representing a particular complex quantum state is difficult. Moreover, stabilizers may only be useful when representing states that are based on Clifford group operations (S gate, H gate and CZ gate). A state with  $n$  qubits has  $n$  degree of freedom. Each known stabilizer will decrease the degree of freedom linearly.

## 2.6 Entanglement

Two or more qubits can be in an *entangled* state. When states are entangled, each qubit's state cannot be described independently. That is to say, the collapse of the wave function of a qubit by measurement may immediately affect the other of the pair's state regardless of the physical distance between.

### 2.6.1 Bell pair/Einstein-Podolsky-Rosen (EPR) pair

One common example of an entangled state is called a *Bell pair* or sometimes called *EPR pair*:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = |\Phi^+\rangle \quad (2.65)$$

In the above example, note that each qubit state  $|00\rangle$  and  $|11\rangle$  has equally weighted probability, so that each qubit has a 50/50 probability of being found in each state but not independently. If one qubit's state is found to be 0, then the other qubit's state must be, and will be 0. Therefore, measuring one qubit will also decide the other qubit's state. Specifically, the four entangled states that can be used as a basis set are known as *Bell states*:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \quad (2.66)$$

$$|\Phi^-\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \quad (2.67)$$

$$|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \quad (2.68)$$

$$|\Psi^-\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \quad (2.69)$$

## 2.6.2 W state

The W state is an entangled quantum state that consists of 3 qubits, which has a similar state to the  $|\Phi^+\rangle$ . This state has an Hamming weight of 1.

$$|W\rangle = \frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle) \quad (2.70)$$

Measuring one qubit will result in either state:

$$|W^0\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \quad (2.71)$$

$$|W^1\rangle = |00\rangle \quad (2.72)$$

Depending on the measurement result, the residual system is entangled or unentangled.

## 2.6.3 Greenberger-Horne-Zeilinger (GHZ) state

The GHZ state is an entangled state of  $N \geq 3$  qubits as in equation 2.73.

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle^{\otimes N} + |1\rangle^{\otimes N}) = |GHZ\rangle \quad (2.73)$$

The simplest GHZ state includes 3 qubits:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle^{\otimes 3} + |1\rangle^{\otimes 3}) = \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) = |GHZ\rangle \quad (2.74)$$

Similar to a Bell pair, measuring an arbitrary qubit of the GHZ state decides the overall state to  $|000\rangle$  or  $|111\rangle$  with equally weighted probabilities. The GHZ state is not Local Operation and Classical Communication (LOCC) equivalent to the W state – no local operation can convert the GHZ state to the W state or vice versa when qubits are physically distant.

## 2.6.4 Cluster state

The cluster state is an example of qubits that are maximally entangled – a maximally entangled state has the maximum Von Neumann entropy obtainable by the number of qubits in the system. The Von Neumann entropy is a measure of entanglement:

$$S(\rho) = -\text{Tr } \rho \log_2(\rho) \quad (2.75)$$

A cluster state of  $n$  vertices (qubits) can be defined by:

$$|G\rangle = \prod_{(a,b) \in E} \Lambda_{a,b}(Z) |+\rangle^{\otimes n} \quad (2.76)$$

where  $E$  is the set of edges (entanglement) and  $a, b$  are the corresponding vertices (qubits). As an example, 3-qubit cluster state is:

$$|\psi\rangle = \frac{1}{2\sqrt{2}} (|0_1 0_2 0_3\rangle + |0_1 0_2 1_3\rangle + |0_1 1_2 0_3\rangle - |0_1 1_2 1_3\rangle + |1_1 0_2 0_3\rangle + |1_1 0_2 1_3\rangle - |1_1 1_2 0_3\rangle + |1_1 1_2 1_3\rangle) \quad (2.77)$$

where the subscript denotes the labeled qubit for identification. All states are equally weighted, and measuring qubit 1 will result in either one of the following system:

If the measurement result is 0

$$|\psi\rangle = \frac{1}{2} (|0_2 0_3\rangle + |0_2 1_3\rangle + |1_2 0_3\rangle - |1_2 1_3\rangle) \quad (2.78)$$

If the measurement result is 1

$$|\psi\rangle = \frac{1}{2} (|0_2 0_3\rangle + |0_2 1_3\rangle - |1_2 0_3\rangle + |1_2 1_3\rangle) \quad (2.79)$$

The equation 2.79 equates to equation 2.78 with an additional Z operation to qubit 2 as a byproduct,  $|\psi^0\rangle = Z_1 |\psi^1\rangle$ . Notice that measuring just one qubit does not fully decide the remaining state. The 2-qubit cluster state is also LOCC equivalent to the Bell pair since  $H_1 |\Phi_{1,2}^+\rangle = |G_{1,2}\rangle$  or  $H_2 |\Phi_{1,2}^+\rangle = |G_{1,2}\rangle$ .

## 2.7 Reading quantum circuits

We can track the state through a computation step-by-step by writing down the bra-ket expression. However, sometimes it is easier to understand the transition using a different mathematical representation.

A common quantum circuit consists of multiple gate operations. An example circuit that operates on three qubits (qubit A, qubit B and qubit C) initialized as  $|0\rangle$  is shown in Fig. 2.12. This particular circuit consists of three steps, with one Hadamard gate and two CNOT gates. This circuit uses three independent qubits as inputs, and outputs them as an entangled GHZ state.

### 2.7.1 Understanding circuits using matrices

It is sometimes easy to calculate the transition using matrices.

The input state  $|\psi_0\rangle = |0_A 0_B 0_C\rangle$  in vector form is

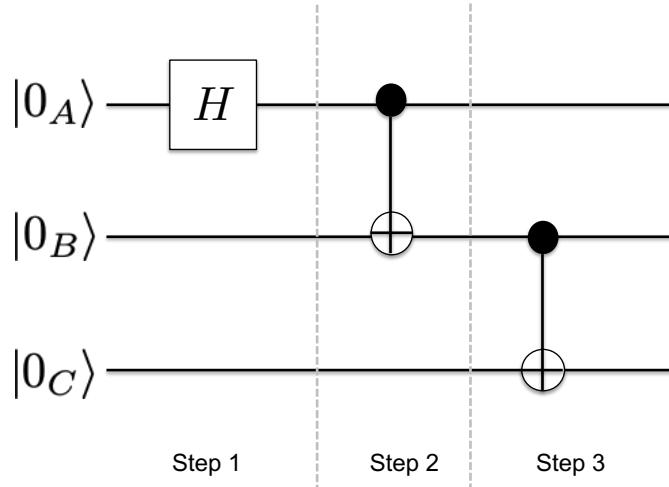


Figure 2.12: A quantum circuit generating a GHZ state (refer to subsection 2.6.3) of three qubits.

$$|\psi_0\rangle = |0_A 0_B 0_C\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad (2.80)$$

because each qubit has an initial state  $|0\rangle$ .

### Step 1

The first step in Fig.2.12 performs Hadamard gate, which can be represented as a  $2 \times 2$  matrix, on qubit A. Other two qubits in Step 1 are simply waiting, in which case we commonly use an Identity matrix as a matrix representation. The overall gate operation in Step 1 can be, therefore, seen as a set of three single qubit gate operations – an Hadamard gate and two Identity gates. The matrix representing the entire operation in Step 1,  $U_1$ , can be calculated by taking the tensor product of each matrix.

$$\begin{aligned}
U_1 &= H \otimes I \otimes I \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} \end{bmatrix} \quad (2.81)
\end{aligned}$$

The state after Step 1 ( $|\psi_1\rangle$ ), of course, can be calculated by multiplying the acquired matrix to the input state as:

$$|\psi_1\rangle = U_1 |\psi_0\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.82)$$

$$= \frac{1}{\sqrt{2}} (|0_A 0_B 0_C\rangle + |1_A 0_B 0_C\rangle). \quad (2.83)$$

## Step 2

$U_2$  can be calculated in a similar way as in Step 1. Hence,  $U_2$  is:

$$\begin{aligned}
U_2 &= CNOT \otimes I \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.84}
\end{aligned}$$

The state after Step 2,  $|\psi_2\rangle$ , is therefore:

$$|\psi_2\rangle = U_2 |\psi_1\rangle = U_2 U_1 |\psi_0\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{bmatrix} \tag{2.85}$$

$$= \frac{1}{\sqrt{2}} (|0_A 0_B 0_C\rangle + |1_A 1_B 0_C\rangle) \tag{2.86}$$

$$= \frac{1}{\sqrt{2}} (|0_A 0_B\rangle + |1_A 1_B\rangle) |0_C\rangle. \tag{2.87}$$

### Step 3

The last step in the circuit,  $U_3$ , is:

$$\begin{aligned}
U_3 &= I \otimes CNOT \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.88}
\end{aligned}$$

The output state of the circuit,  $|\psi_3\rangle$ , can be calculated using the acquired three matrices.

$$\begin{aligned}
U_3 |\psi_2\rangle &= U_3 U_2 |\psi_1\rangle = U_3 U_2 U_1 |\psi_0\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} \\
&= \frac{1}{\sqrt{2}} (|0_A 0_B 0_C\rangle + |1_A 1_B 1_C\rangle). \tag{2.89}
\end{aligned}$$

Notice that the entire circuit, composed of the discussed three steps, can be described using a single  $8 \times 8$  matrix, which is:

$$U = U_3 \cdot U_2 \cdot U_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 \end{bmatrix}. \tag{2.90}$$

### Other cases

For a different quantum circuit, a two-qubit gate may be performed between non-adjacent qubits (see Fig. 2.13).

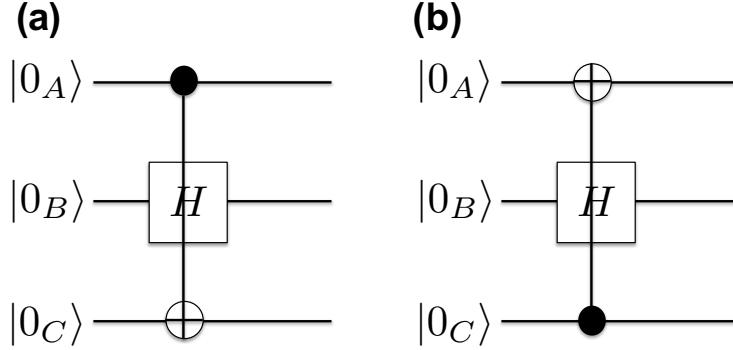


Figure 2.13: Example circuits of an Hadamard gate and a CNOT gate.

For such cases, we have to calculate the matrix separately based on the controlled qubit's state,  $|0\rangle$  and  $|1\rangle$ , and take the sum as:

$$\begin{aligned}
 U_{(a)} &= (|0_A\rangle\langle 0_A| \otimes H_B \otimes I_C) + (|1_A\rangle\langle 1_A| \otimes H_B \otimes X_C) \\
 &= \left( \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \otimes \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) + \left( \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right) \\
 &= \left[ \begin{array}{cccccccc}
 \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\
 \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\
 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\
 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\
 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0
 \end{array} \right] \quad (2.91)
 \end{aligned}$$

Similarly, for an inverted CNOT, we can calculate the matrix as:

$$\begin{aligned}
 U_{(b)} &= (I_A \otimes H_B \otimes |0_C\rangle\langle 0_C|) + (X_A \otimes H_B \otimes |1_C\rangle\langle 1_C|) \\
 &= \left( \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \right) + \left( \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \otimes \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right) \\
 &= \left[ \begin{array}{cccccccc}
 \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\
 \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\
 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\
 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 \\
 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0
 \end{array} \right] \quad (2.92)
 \end{aligned}$$

The required dimension of matrices will increase exponentially to the number of involved qubits.

### 2.7.2 Understanding circuits using stabilizers

Not only does the number of terms in the vector increase exponentially, but naturally the size of a full matrix representing the unitary operation on a set of qubits does as well. However, as discussed in section 2.5.2, the number of stabilizers only increases linearly. The transition of stabilizers through a circuit can be understood in a similar way as error propagations (see section 2.9.1). Here, we study the behavior based on the same GHZ state circuit shown in Fig. 2.12.

The Z operator stabilizes the state  $|0\rangle$ . Thus, the input state  $|\psi_i\rangle = |0_A 0_B 0_C\rangle$  in stabilizer formalism is as shown in Tab. 2.2. Note that each empty space in the table is actually a stabilizer I.

Table 2.2: Stabilizers of the input state.

	qubit A	qubit B	qubit C
1st stabilizer	Z		
2nd stabilizer		Z	
3rd stabilizer			Z

#### Step 1

The first step is an Hadamard gate on qubit A. Similar to the error propagation, the Hadamard gate converts the Z stabilizer of qubit A to X as in Tab. 2.3.

Table 2.3: Stabilizers of the output state after Step 1.

	qubit A	qubit B	qubit C
1st stabilizer	X		
2nd stabilizer		Z	
3rd stabilizer			Z

#### Step 2

The second step is a CNOT gate between qubit A and B. The X stabilizer of qubit A propagates to qubit B, and the Z stabilizer of qubit B propagates to qubit A.

Table 2.4: Stabilizers of the output state after Step 2.

	qubit A	qubit B	qubit C
1st stabilizer	X	X	
2nd stabilizer	Z	Z	
3rd stabilizer			Z

### Step 3

The third step is also a CNOT gate, but between qubit B and C. The X stabilizer propagates from qubit B propagates to qubit C, and the Z stabilizer propagates from qubit C propagates to qubit B.

Table 2.5: Stabilizers of the output state after Step 3.

	qubit A	qubit B	qubit C
1st stabilizer	X	X	X
2nd stabilizer	Z	Z	
3rd stabilizer		Z	Z

Thus, the stabilizer set of a three qubit GHZ state is  $\{X_A X_B X_C, Z_A, Z_B, Z_B Z_C\}$ . A pictorial transition of the stabilizers is summarized in Fig. 2.14.

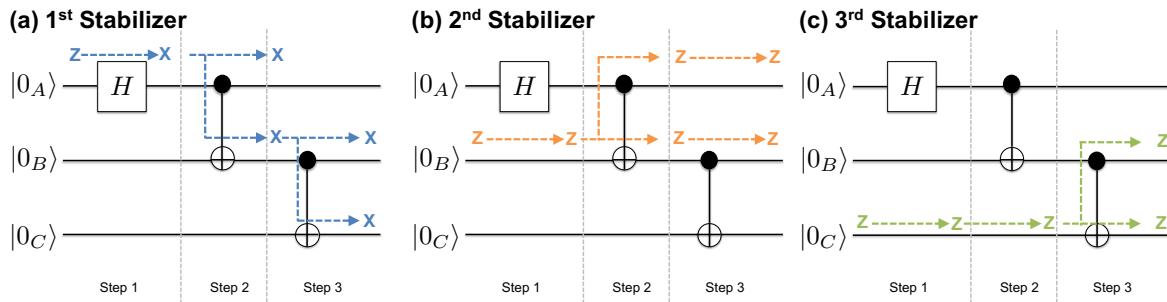


Figure 2.14: Propagation of stabilizers in the creation of a 3-qubit GHZ state. Stabilizer I is omitted in the figure for simplification. Stabilizer I will stay as I unless propagated from a different qubit through a multi-qubit gate operation.

## 2.8 Quantum networking

There are several important operations for accomplishing quantum networking. The following subsections explain some of the major operations in detail.

### 2.8.1 Quantum Teleportation

Using gate operations, it is possible to teleport quantum information from one place to another. This is not only limited to close distance but also for long distances. The simplest circuit implementation for quantum teleportation is as shown in Figure 2.15.

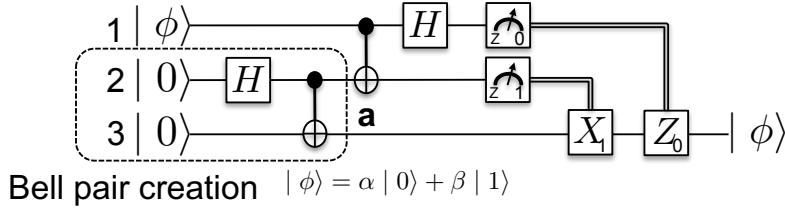


Figure 2.15: Simple quantum circuit of quantum teleportation

After the Bell pair creation at  $a$ , the two qubits may be separated physically. As shown in the circuit, each measurement comes with a classical feedforward operation to the residual qubit, which is essential for completing the teleportation of an arbitrary quantum state. Even though entangled particles always share physical properties regardless of the distance between them, the necessity of classical communication forbids the transmission of information from one place to another faster than the speed of light.

$$\begin{aligned}
 H_1 \Lambda_{1,2}(X) \Lambda_{2,3}(X) H_2 |0_3\rangle |0_2\rangle |\Phi_1\rangle &= H_1 \Lambda_{1,2}(X) \Lambda_{2,3}(X) H_2 |0_3 0_2\rangle \otimes \alpha |0_1\rangle + \beta |1_1\rangle \\
 &= H_1 \Lambda_{1,2}(X) \left( \frac{1}{\sqrt{2}} (|0_3 0_2\rangle + |1_2 1_3\rangle) \otimes \alpha |0_1\rangle + \beta |1_1\rangle \right) \\
 &= H_1 \left( \frac{1}{\sqrt{2}} (\alpha |0_1 0_2 0_3\rangle + \alpha |1_1 1_2 1_3\rangle + \beta |1_1 0_2 0_3\rangle + \beta |1_1 1_2 1_3\rangle) \right) \\
 &= \left( \frac{1}{2} (\alpha |0_1 0_2 0_3\rangle + \alpha |1_1 0_2 0_3\rangle + \alpha |0_1 1_2 1_3\rangle + \alpha |1_1 1_2 1_3\rangle \right. \\
 &\quad \left. - \beta |1_1 1_2 0_3\rangle + \beta |0_1 1_2 0_3\rangle - \beta |1_1 0_2 1_3\rangle + \beta |0_1 0_2 1_3\rangle) \right) \tag{2.93}
 \end{aligned}$$

Depending on the measurement results, byproduct operations are applied to the remaining qubit to complete the teleportation. For details, see Table 2.6. Each outcome can be found with equal probability of 25%.

Table 2.6: Byproduct operations to complete quantum teleportation

Measurement result	Output state	Byproduct operation
$0_1 0_2$	$\alpha  0_3\rangle + \beta  1_3\rangle$	I
$0_1 1_2$	$\alpha  1_3\rangle + \beta  0_3\rangle$	X
$1_1 0_2$	$\alpha  0_3\rangle - \beta  1_3\rangle$	Z
$1_1 1_2$	$\alpha  1_3\rangle - \beta  0_3\rangle$	XZ

Notice that when the measurement result of qubit 2 is 1, there is always a bit-flip error on the remaining qubit. Similarly, when the measurement result of qubit 1 is 1, Z gate is must be applied to qubit 3 as a byproduct to fix the phase. Thus, qubit 3's state can be manipulated beforehand to avoid any byproduct operation after the measurement as in Figure 2.16. The communication speed is still not faster than the speed of light as qubit 3, which is a part of the Bell pair, needs to be sent to another node to establish a long distance communication.

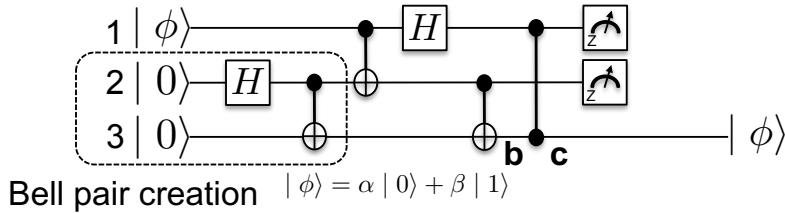


Figure 2.16: Simple quantum circuit of quantum teleportation without byproduct operations

If the qubits of the Bell pair are separated,  $b$  or  $c$  will require long-distance multi-qubit operations. The visualized model of quantum teleportation is shown in Figure 2.17.

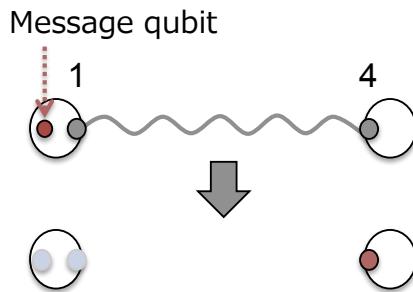


Figure 2.17: Visualized model of quantum teleportation. At the end of the operation, the red qubit has been recreated on the right.

## 2.8.2 Entanglement Swapping

The quantum teleportation technique introduced above is an example of transmitting one bit of quantum information from one place to another but networking often requires multi-hop communication. The simplest solution for such demand is to apply quantum teleportation hop-by-hop. Nevertheless, operating on the message qubit directly many times degrades the information. Another solution is the use of *entanglement swapping* [26], which is capable of lengthening the Bell pair to allow a direct teleportation of quantum information over multiple repeaters. As shown in Figure 2.18, entanglement swapping is based on the teleportation circuit that was introduced above and in Figure 2.15. In the below example, 2 Bell pairs are consumed to output 1 end-to-end Bell pair.

Subscripts of operators are identifiers for clarifying the interconnection between the measurement operator and its byproduct operator.

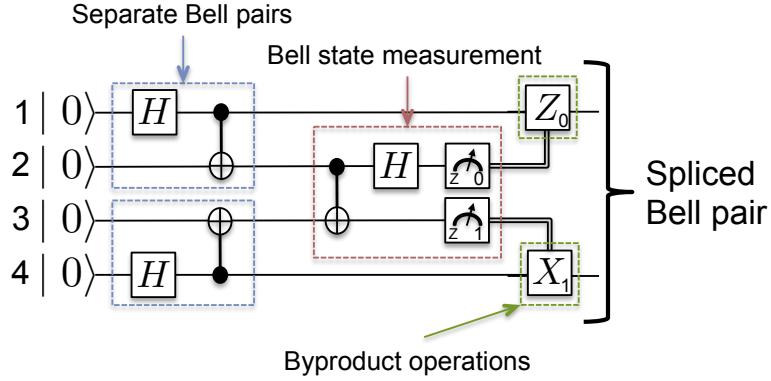


Figure 2.18: Simple quantum circuit of entanglement swapping

$$\begin{aligned}
 & H_2 \Lambda_{2,3}(X) \Lambda_{4,3}(X) \Lambda_{1,2}(X) H_4 H_1 |0_4\rangle |0_3\rangle |0_2\rangle |0_1\rangle \\
 &= H_2 \Lambda_{2,3}(X) \frac{1}{\sqrt{2}}(|0_1 0_2\rangle + |1_1 1_2\rangle) \otimes \frac{1}{\sqrt{2}}(|0_3 0_4\rangle + |1_3 1_4\rangle) \\
 &= H_2 \frac{1}{2}(|0_1 0_2 0_3 0_4\rangle + |0_1 0_2 1_3 1_4\rangle + |1_1 1_2 1_3 0_4\rangle + |1_1 1_2 0_3 1_4\rangle) \\
 &\quad \frac{1}{2\sqrt{2}}(|0_1 0_2 0_3 0_4\rangle + |0_1 1_2 0_3 0_4\rangle + |0_1 0_2 1_3 1_4\rangle + |0_1 1_2 1_3 1_4\rangle \\
 &\quad - |1_1 1_2 1_3 0_4\rangle + |1_1 0_2 1_3 0_4\rangle - |1_1 1_2 0_3 1_4\rangle + |1_1 0_2 0_3 1_4\rangle) \tag{2.94}
 \end{aligned}$$

Table 2.7: Byproduct operations to complete entanglement swapping

Measurement result	Output state	Byproduct operation
$0_2 0_3$	$\frac{1}{\sqrt{2}}( 0_1 0_4\rangle +  1_1 1_4\rangle)$	$I$
$0_2 1_3$	$\frac{1}{\sqrt{2}}( 0_1 1_4\rangle +  1_1 0_4\rangle)$	$X_4$
$1_2 0_3$	$\frac{1}{\sqrt{2}}( 0_1 0_4\rangle -  1_1 1_4\rangle)$	$Z_1$
$1_2 1_3$	$\frac{1}{\sqrt{2}}( 0_1 1_4\rangle -  1_1 0_4\rangle)$	$X_4 Z_1$

The byproduct operator, Z gate, can also be applied to qubit 4 instead of qubit 1. The visualized model is at Figure 2.19.

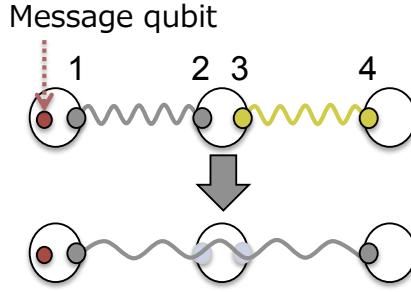


Figure 2.19: Visualized model of entanglement swapping

### 2.8.3 Entanglement Purification

Using two or more less-entangled mixed pairs shared among nodes, it is possible to create one pair with a higher entanglement. The easiest example of entanglement purification can be shown using 2 Bell pairs as in Figure 2.20.

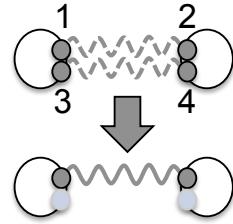


Figure 2.20: Visualized model of entanglement purification

In this example, two Bell pairs  $|\Phi_{1,2}^+\rangle$ ,  $|\Phi_{3,4}^+\rangle$  are shared between two nodes which are physically far apart, each holding half of the two pairs. In order to create and check the parity of two qubits in each node, both nodes locally perform a CNOT operation and measure one of the pairs with respect to the Z-basis, which destroys one Bell pair. If the measurement results agree,  $0_30_4$  or  $1_31_4$ , the remaining pair will have a higher fidelity than the original pairs given  $F_{input} > 50\%$ . On the other hand, the whole resource needs to be discarded if the measurement results do not agree.

When Bell pairs are both perfect and gates have no noise  $F_{input} = 1.00$  and  $F_{operation} = 1.00$ :

$$\begin{aligned}
 \Lambda_{2,4}(X)\Lambda_{1,3}(X)|\Phi_{1,2}^+\rangle|\Phi_{3,4}^+\rangle &= \Lambda_{2,4}(X)\Lambda_{1,3}(X)\frac{1}{\sqrt{2}}(|0_10_2\rangle)\otimes\frac{1}{\sqrt{2}}(|0_30_4\rangle) \\
 &= \Lambda_{2,4}(X)\Lambda_{1,3}(X)\frac{1}{2}(|0_10_20_30_4\rangle + |0_10_21_31_4\rangle + |1_11_20_30_4\rangle + |1_11_21_31_4\rangle) \\
 &= \Lambda_{2,4}(X)\frac{1}{2}(|0_10_20_30_4\rangle + |0_10_21_31_4\rangle + |1_11_21_30_4\rangle + |1_11_20_31_4\rangle) \\
 &= \frac{1}{2}(|0_10_20_30_4\rangle + |0_10_21_31_4\rangle + |1_11_21_31_4\rangle + |1_11_20_30_4\rangle) \\
 &= |\Phi_{1,2}^+\rangle|\Phi_{3,4}^+\rangle \quad (2.95)
 \end{aligned}$$

When either Bell pair is imperfect, a bit-flip error on  $|\Phi^+\rangle$  changes the state to  $|\Psi^+\rangle$ . Tab.2.10 is the summary of the behavior when  $F_{output} > F_{input}$  and  $F_{input} > 0.5$ .

Table 2.8: Entanglement swapping with noisy Bell pairs

Input state	Probability	Output state	Result
$ \Phi_{1,2}^+\rangle  \Phi_{3,4}^+\rangle$	$F^2$	$ \Phi_{1,2}^+\rangle$	True positive
$ \Phi_{1,2}^+\rangle  \Psi_{3,4}^+\rangle$	$F(1 - F)$	-	True negative
$ \Psi_{1,2}^+\rangle  \Phi_{3,4}^+\rangle$	$F(1 - F)$	-	False negative
$ \Psi_{1,2}^+\rangle  \Psi_{3,4}^+\rangle$	$(1 - F)^2$	$ \Psi_{1,2}^+\rangle$	False positive

With an assumption of two Bell pairs having the same fidelity, the probability of getting the same measurement results can be obtained by  $F_{input}^2 + (1 - F_{input})^2$ , while the probability of it actually being right is  $F_{input}^2$ . Therefore, The relation between the input fidelity and the output fidelity after performing entanglement purification can be derived as:

$$F_{output} = \frac{F_{input}^2}{F_{input}^2 + (1 - F_{input})^2} \quad (2.96)$$

The iteration of the above steps can be used to create an entangled pair of arbitrarily high purity  $F_{out} \rightarrow 1$ .

## 2.8.4 Quantum state tomography

Because measuring a qubit destroys the state, one cannot simply check the quality of a generated Bell pair. Instead, a common methodology is to prepare many of the same states, measure them along different axes, and estimate the average quality of the measured states by reconstructing the density matrix from the measurement outcomes. This process is called quantum tomography [27].

Given matrices  $\sigma_i$ , corresponding to our Pauli operators explained in subsection 2.4.1, as:

$$\sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.97)$$

$$\sigma_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (2.98)$$

$$\sigma_2 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad (2.99)$$

$$\sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad (2.100)$$

an arbitrary n-qubit density matrix can be rewritten using Stokes parameters ( $S$ ):

$$\rho = \frac{1}{2^n} \sum_{i_1, i_2, \dots, i_n=0}^3 S_{i_1, i_2, \dots, i_n} \sigma_{i_1} \otimes \sigma_{i_2} \otimes \dots \otimes \sigma_{i_n} \quad (2.101)$$

$$s.t. S_{i_1, i_2, \dots, i_n} = \text{Tr}[(\sigma_{i_1} \otimes \sigma_{i_2} \otimes \dots \otimes \sigma_{i_n}) \rho]. \quad (2.102)$$

We need a total of  $4^n$  Stokes parameters for an n-qubit system. Notice that each Stokes parameter requires the density matrix  $\rho$  to be known for the calculation. This is not accessible, unless for a theoretical calculation. However, each of the parameters corresponds to the outcome of a specific pair of projective measurements.

### Single qubit tomography using projective measurements

A single qubit density matrix, therefore, can be described as:

$$\rho = \frac{1}{2} \sum_{i=0}^3 S_i \sigma_i. \quad (2.103)$$

Stokes parameters for a single qubit tomography can be rewritten based on measurements as:

$$S_0 = P(|0\rangle) + P(|1\rangle) \quad (2.104)$$

$$S_1 = P\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right) - P\left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) \quad (2.105)$$

$$S_2 = P\left(\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)\right) - P\left(\frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)\right) \quad (2.106)$$

$$S_3 = P(|0\rangle) - P(|1\rangle), \quad (2.107)$$

where  $P(|\Psi\rangle)$  is the probability to observe outcome state  $|\Psi\rangle$ , which mathematically is  $P(|\psi\rangle) = \langle \Psi | \rho | \Psi \rangle = \text{Tr}[|\Psi\rangle \langle \Psi| \rho]$ . In the real world, those probabilities can be directly accessed from the accumulated measurement results. Notice that observing the state  $\{|0\rangle, |1\rangle\}$  requires a Z measurement. Similarly, observing  $\{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\}$  requires an X measurement, and  $\{\frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle), \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)\}$  requires a Y measurement. If one measured an unknown state along the Z axis 1000 times, in which 489 outcomes showed up as  $|0\rangle$  and 511 as  $|1\rangle$ , then  $P(|0\rangle) = 0.489$  and  $P(|1\rangle) = 0.511$ .  $S_0$  is, thus, always 1. The notation used is similar to optics, although measurement process is different.

### Two-qubit tomography using projective measurements

Similarly, a two-qubit density matrix can be described as:

$$\rho = \frac{1}{4} \sum_{i_1, i_2=0}^3 S_{i_1, i_2} \sigma_{i_1} \sigma_{i_2}. \quad (2.108)$$

Assuming that

$$H = |0\rangle \quad (2.109)$$

$$V = |1\rangle \quad (2.110)$$

$$D = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad (2.111)$$

$$A = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \quad (2.112)$$

$$R = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \quad (2.113)$$

$$L = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle), \quad (2.114)$$

then the 16 Stokes parameters can be expressed as:

$$S_{0,0} = 1 \quad (2.115)$$

$$S_{0,1} = P(DD) - P(DA) + P(AD) - P(AA) \quad (2.116)$$

$$S_{0,2} = P(RR) - P(LR) + P(RL) - P(LL) \quad (2.117)$$

$$S_{0,3} = P(HH) - P(HV) + P(VH) - P(VV) \quad (2.118)$$

$$S_{1,0} = P(DD) + P(DA) - P(AD) - P(AA) \quad (2.119)$$

$$S_{1,1} = P(DD) - P(DA) - P(AD) + P(AA) \quad (2.120)$$

$$S_{1,2} = P(DR) - P(DL) - P(AR) + P(AL) \quad (2.121)$$

$$S_{1,3} = P(DH) - P(DV) - P(AH) + P(AV) \quad (2.122)$$

$$S_{2,0} = P(RR) + P(LR) - P(RL) - P(LL) \quad (2.123)$$

$$S_{2,1} = P(RD) - P(RA) - P(LD) + P(LA) \quad (2.124)$$

$$S_{2,2} = P(RR) - P(RL) - P(LR) + P(LL) \quad (2.125)$$

$$S_{2,3} = P(RH) - P(RV) - P(LH) + P(LV) \quad (2.126)$$

$$S_{3,0} = P(HH) + P(HV) - P(VH) - P(VV) \quad (2.127)$$

$$S_{3,1} = P(HD) - P(HA) - P(VD) + P(VA) \quad (2.128)$$

$$S_{3,2} = P(HR) - P(HL) - P(VR) + P(VL) \quad (2.129)$$

$$S_{3,3} = P(HH) - P(HV) - P(VH) + P(VV). \quad (2.130)$$

### 2.8.5 Verification of quantum correlation (entangled states) via CHSH inequality

Whether qubits are entangled or not cannot be checked directly. In stead, the presence of an entanglement can be verified via a statistical experiment called the *CHSH inequality*, which is a similar technique to tomography. The classical CHSH inequality assumes two nodes performing measurements  $\alpha = A(a, \lambda)$ ,  $\alpha' = A(a', \lambda)$ , and  $\beta = B(b, \lambda)$ ,  $\beta' = B(b', \lambda)$  each with outcome  $\pm 1$ . A and B are functions that take a variable  $\{a, a'\}$  and  $\{b, b'\}$  accordingly, which are adjustable parameters representing its orientation of measurement. The variable

$\lambda$  is the hidden variable. The hidden variable is an unknown parameter determining the result of each of these experiments. Given  $p(\lambda)$  as the probability distribution determining the measurement outcome, the classical correlation between two measurements,  $E$ , can be gained as:

$$E(a, b) = E(a)E(b) = \int d\lambda p(\lambda)A(a, \lambda)B(b, \lambda) = \langle \alpha\beta \rangle. \quad (2.131)$$

Because each measurement outputs either +1 or -1,

$$\begin{cases} \alpha + \alpha' = A(a, \lambda) + A(a', \lambda) = 0 \\ \alpha - \alpha' = A(a, \lambda) - A(a', \lambda) = \pm 2, \quad \alpha, \alpha' \rightarrow \{1, -1\} \text{ or } \{-1, 1\}, \end{cases}, \quad (2.132)$$

$$\begin{cases} \alpha + \alpha' = A(a, \lambda) + A(a', \lambda) = \pm 2 \\ \alpha - \alpha' = A(a, \lambda) - A(a', \lambda) = 0, \quad \alpha, \alpha' \rightarrow \{-1, -1\} \text{ or } \{1, 1\} \end{cases} \quad (2.133)$$

in which case the following satisfies for a single measurement:

$$C \equiv (\alpha + \alpha')\beta + (\alpha - \alpha')\beta' = \pm 2. \quad (2.134)$$

The classical correlation, CHSH inequality, is:

$$| \langle \alpha\beta \rangle - \langle \alpha\beta' \rangle | + | \langle \alpha'\beta \rangle + \langle \alpha'\beta' \rangle | \leq 2. \quad (2.135)$$

This inequality, however, can be violated in quantum mechanics using entanglement. An example of an entangled state is:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \quad (2.136)$$

The degree of its violation depends on the measurement orientations, where the settings for accomplishing the maximum violation are:

$$\alpha = \frac{\pi}{2} \quad (2.137)$$

$$\alpha' = 0 \quad (2.138)$$

$$\beta = \frac{\pi}{4} \quad (2.139)$$

$$\beta' = \frac{-\pi}{4} \quad (2.140)$$

$$. \quad (2.141)$$

Here, each angle is adjusted over the ZX plane of the Bloch sphere (see subsection 2.2.2). Hence, the quantum correlation is:

$$E(a, b) = \langle \psi | \hat{\sigma}_\alpha \otimes \hat{\sigma}_\beta | \psi \rangle = \cos(\alpha - \beta) = \langle \alpha \beta \rangle. \quad (2.142)$$

Each term of the CHSH inequality will therefore be:

$$\langle \alpha \beta \rangle = \langle \alpha' \beta \rangle = -\langle \alpha \beta' \rangle = \langle \alpha' \beta' \rangle = \frac{1}{\sqrt{2}}. \quad (2.143)$$

Thus, the quantum correlation takes a value between:

$$2 < |\langle \alpha \beta \rangle - \langle \alpha \beta' \rangle| + |\langle \alpha' \beta \rangle + \langle \alpha' \beta' \rangle| \leq 2\sqrt{2}. \quad (2.144)$$

The noise on qubits, of course, decreases the calculated value. Because the probability of getting  $\pm 1$  measurement outcome depends on the measurement angles, we must choose the measurement angle at random. Classical correlation by pre-arranged photon states, therefore, can never violate the CHSH inequality. But the measurement basis must not be chosen before the photons are emitted. Those constraints that affect the validity of the experiment are known as the *loopholes*. Some of the major loopholes are summarized in the Wikipedia [28].

## 2.8.6 Quantum Repeaters

A quantum repeater network is a system of connected quantum repeater nodes (see Fig. 2.21). The abstract architecture of a simple repeater network consists of 6 important elements. The first element is the memory qubits, or sometimes called the stationary qubits. Memory qubits are physically fixed at each node, and used for storing quantum information. A quantum repeater network requires two types of channels. The first channel is called the quantum channel, which is used for transmitting quantum information. The other one is called the classical channel, which is used for transmitting ordinary classical packets. A memory qubit emits an entangled optical qubit through the quantum channel, towards the Bell State Analyzer (BSA). The BSA's role is to create an entangled state of memory qubits, by performing Bell measurements on received optical qubits (see Fig. 2.22). Entangled states shared between nodes are commonly, but not limited to,  $|\Phi^+\rangle$  Bell pairs.

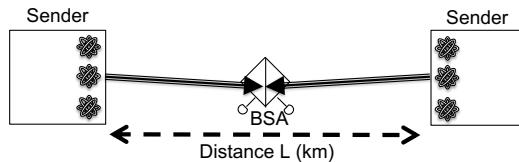


Figure 2.21: Architecture of a simple two-node quantum repeater network.

Nodes emit photons that are entangled with its source memory qubit. For each attempt, two photons arrive at the BSA, one from the left and another from the right. The polarizing

beamsplitter functions in a similar way as a polarizer, except it reflects the photon instead of blocking. When they arrive at the beamsplitter, they interfere in the middle. Detector outcomes are summarized in Tab. 2.9. As shown, when two detectors click, you cannot tell whether the two photons had states  $|0_L 0_R\rangle$  or  $|1_L 1_R\rangle$ , which in quantum mechanics, makes it a superposition state. The state of the photon immediately gets destroyed by the measurements, but preserves the entanglement of the source memory qubits.

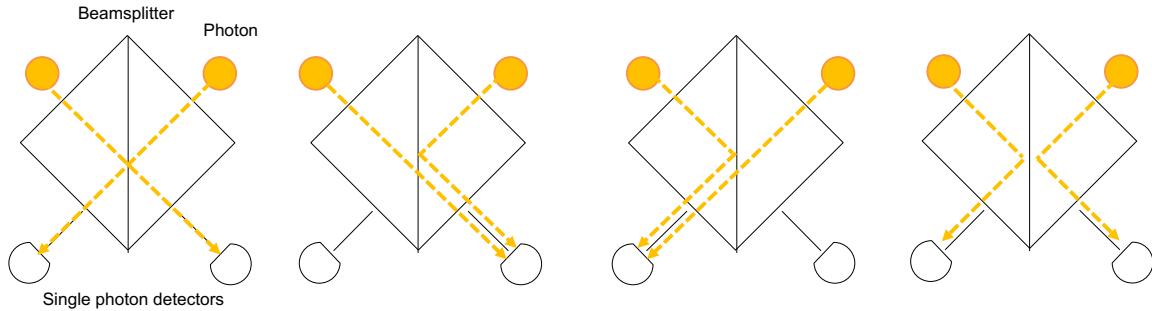


Figure 2.22: Bell measurement using a beam splitter and photon detectors.

Table 2.9: Bell measurement outcomes. The horizontal polarization  $|H\rangle$  may be used for representing  $|0\rangle$ , and  $|V\rangle$  for  $|1\rangle$ . The subscript  $L$  denotes the *Left photon*, and  $R$  denotes the *Right photon*.

Left detector count	Right detector count	State
True	False	$ 0_L\rangle  1_R\rangle$
False	True	$ 1_L\rangle  0_R\rangle$
True	True	$ 0_L\rangle  0_R\rangle \parallel  1_L\rangle  1_R\rangle$

## 2.8.7 Generations of Network Evolution

Muralidharan et al. categorized quantum repeaters, based on capabilities, into three generations [29]. The 1st generation quantum repeater network works based on purification and entanglement swapping, for detecting and discarding erroneous resources and to connect non-adjacent nodes. While this scheme is relatively simple and straightforward, its capability could strictly depend on the distance, mainly limited by classical latencies for receiving acknowledgements regarding purification and entanglement swapping. Its performance is also known to be limited by memory lifetime [30]. The 2nd generation utilizes encoded Bell pairs prepared between adjacent nodes, and performs quantum error correction [31–33]. Swapping is done at the logical level for Jiang’s architecture [31], via an extended surface for Fowler’s [32]. In the 3rd generation, quantum states are directly encoded to a block of physical qubits that will be sent through the channel. The receiver node can correct errors using the received physical qubits. The 3rd generation is very similar to the 2nd generation, but requires a very high success rate of photon reception through all of the physical elements of the channel. These two generations are semantically identical but the temporal behavior of the 3rd generation is more like classical packet forwarding networks.

## 2.9 Errors on quantum systems and its measure

### 2.9.1 Error

Quantum gates are inherently noisy, and may cause errors while operating on qubits.

#### Pauli errors

In general, there are two types of Pauli errors that should be taken into consideration, the bit-flip (X) error and the phase (Z) error – the Y error is a combination of both errors.

An X error, for example, can be seen as an undesired X gate applied to the system. For instance, assuming a single qubit ideal system in state  $|\psi_i\rangle = |0_A\rangle$ , an X error on the actual system ( $|\psi_a\rangle$ ) is:

$$|\psi_a\rangle = X_A. |\psi_i\rangle = X_A. |0_A\rangle = |1_A\rangle \quad (2.145)$$

There are also cases where an error does not impact the state at all. An example can be seen with the same ideal state,  $|\psi_i\rangle = |0_A\rangle$ , and a Z error on it.

$$|\psi_a\rangle = Z_A. |\psi_i\rangle = Z_A. |0_A\rangle = |0_A\rangle \quad (2.146)$$

When a particular error (or an intentional unitary operation) does not influence the state, the error is a *stabilizer* of the state (for details regarding stabilizers, see section 2.5.2). In this particular example, the Z operator is a stabilizer of state  $|0\rangle$ .

As mentioned above, the Y error is a combination of both X and Z error. Thus, a Y error on  $|\psi_i\rangle = |0_A\rangle$  is:

$$|\psi_a\rangle = Y_A. |\psi_i\rangle = Y_A. |0_A\rangle = |1_A\rangle. \quad (2.147)$$

Naturally, the Z error part will not be physically observable, and what's left on the output is solely an X error.

#### Propagation of Pauli errors

Equally important, errors may propagate through quantum circuits throughout the operation. The Hadamard gate converts bit-flip error to phase error, and phase error to bit-flip error.

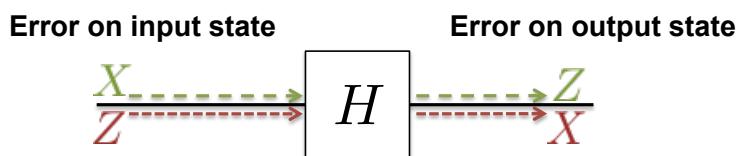


Figure 2.23: Error propagation of Hadamard gate

The CNOT operation will result in a propagation of the bit-flip error on the control qubit, ending up with bit-flip errors on both qubits. Similarly, a phase error on the target qubit will be transferred, after application of CNOT gate, to the control qubit.

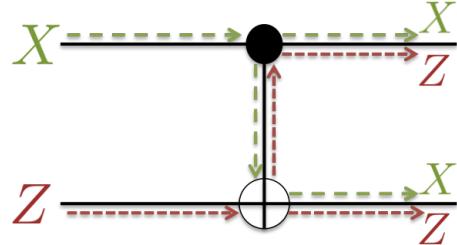


Figure 2.24: Error propagation of Controlled-Not gate

Similarly, the bit-flip error on the control qubit will result in a bit-flip error on the target qubit and a phase error on the control qubit after the CZ operation. Furthermore, a bit-flip error on the target qubit will result in a bit-flip error on the target qubit and a phase error on the control qubit.

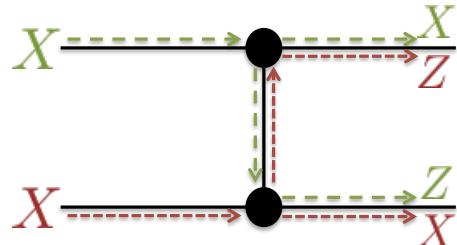


Figure 2.25: Error propagation of Controlled-Z gate

The measurement outcome depends on the quantum state, and thus a measurement error leads to an erroneous feedforward operation. Therefore measurement errors in MBQC, explained in section 2.10, also propagate to other qubits through byproduct operations.

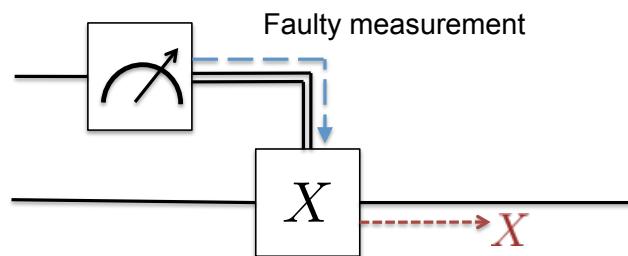


Figure 2.26: Error propagation of measurement

### Other unfixable errors

In general, Pauli errors can be detected and discarded through purification (see section 2.8.3), or corrected through error correction. However, there exists a type of errors that cannot be

fixed. The completely mixed state is of course an example of this. Other examples include qubit memory excitation/relaxation error.

A single qubit may hold an arbitrary quantum information  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ . The memory excitation error is a type of error where the state  $\alpha|0\rangle + \beta|1\rangle$  is forced to make a transition to  $|1\rangle$  state due to external energy absorbed from the environment. Similarly, the relaxation error happens when the qubit unintentionally loses energy, and is forced to move to the ground state ( $|0\rangle$  state).

### 2.9.2 Fidelity

The imperfection of a quantum state can be described by the *fidelity*. The fidelity is often defined as [34]:

$$F = \langle\psi|\rho_a|\psi\rangle = \text{Tr}[\rho_a\rho_i] \quad (2.148)$$

where  $0 \leq F \leq 1$ ,  $|\psi\rangle$  is the desired ideal state vector,  $\rho_i$  is the desired ideal density matrix and  $\rho_a$  is the density matrix of the actual state. With  $F = 1$ , the actual state is identical to the desired state. A single qubit system has a fidelity of 50% with a completely mixed state. Similarly, in  $n$ -qubit system, a completely mixed state has a fidelity of  $F = \frac{1}{2^n}$ .

The same fidelity, however, may imply different states. For instance, when the actual state ( $\rho_a$ ) is completely mixed and the assuming ideal state is  $|\Phi^+\rangle$  with a density matrix  $\rho_i$ , then the fidelity is:

$$\rho_a = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad (2.149)$$

$$\rho_i = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (2.150)$$

$$F = \text{Tr}[\rho_a\rho_i] = 1/2 = 0.5 \quad (2.151)$$

But the same fidelity can be observed when the state is 50%  $|\Phi^+\rangle$  and 50%  $|\Psi^+\rangle$ . The former example of  $F = 0.5$  is a case where none of the systems are actually entangled but completely mixed. The latter example with the same fidelity, however, is a case where all of the systems are entangled, but with a 50% chance of having an X error. Hence,  $F = 1 - P(\text{error rate})$  is only true under a system with only Pauli errors.

## 2.10 Measurement-based Quantum Computing

The Measurement-based Quantum Computing (MBQC) is the basis of Blind computing, which is an important application of the quantum Internet. MBQC is an alternative universal computation method based on single qubit measurements that was proposed by Raussendorf, Browne and Briegel in 2003 [35] – also known as one-way quantum computing. Unlike the

circuit model, the scheme of MBQC generally requires a two dimensional grid of qubits that are initialized as  $|+\rangle$  and entangled with all neighboring qubits using CZ gates, as a cluster state (see Figure 2.27). The initialized cluster state for MBQC is also called the *resource state*.

The following set of abilities sufficiently provide enough capabilities for universal computing. First is to be able to prepare qubits. Second is to be able to perform Hadamard gates to each of them. Third is to be able to perform CZ gates between neighboring qubits. And last is to be able to measure each of the qubit along an arbitrary axis.

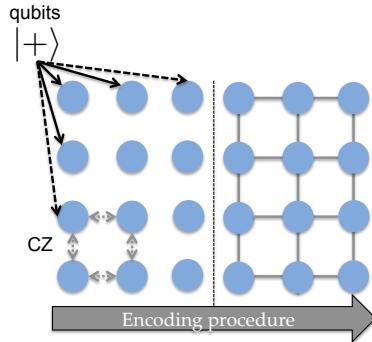


Figure 2.27: 2D resource state generation

The technique to map an unknown quantum state from one qubit to another is known as *one-bit teleportation* (see Figure 2.28) – introduced by Zhou, Leung and Chuang in 2000 [36].

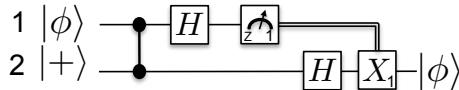


Figure 2.28: Quantum circuit for one-bit teleportation

The calculation using state-vector form is:

$$\begin{aligned}
 H_1 \Lambda_{1,2}(Z) |\phi_1\rangle |+2\rangle &= H_1 \Lambda_{1,2}(Z) \alpha |0\rangle + \beta |1\rangle \otimes |+2\rangle \\
 &= H_1 \alpha |0_1+2\rangle + \beta |1_1-2\rangle \\
 &= \alpha |+1+2\rangle + \beta |-1-2\rangle
 \end{aligned} \tag{2.152}$$

After the measurement operations, the residual state ends up in either output state in Table 2.10.

Table 2.10: Byproduct operation to complete one-bit teleportation

Measurement result	Output state	Byproduct operation
$0_1$	$\alpha  +\rangle + \beta  -\rangle$	$(H_2)I$
$1_1$	$\alpha  +\rangle - \beta  -\rangle$	$(H_2)X_2$

The X gate in this case behaves like a Z gate on  $|0\rangle, |1\rangle$  basis. The Hadamard gate is used to convert the state  $|+\rangle$  to  $|0\rangle$  and  $|-\rangle$  to  $|1\rangle$ .

Inserting a  $R_Z(\theta)$  operation to the one-bit teleportation circuit can simply be accomplished by adding the gate just before the CZ gate (shown Figure 2.29(a)). As  $R_Z(\theta)|\phi\rangle$  can be considered as state  $|\phi'\rangle$ , the circuit can be directly related to the circuit in Figure 2.28. However, as rotations about the Z-axis commute with the CZ operation, the circuit can be rewritten as Figure 2.29(b). The set of operations on qubit 1 after the CZ gate can be considered to be a single measurement operator in a particular basis, and the overall protocol can be simplified to a collection of CZ gates and measurement operations. Finally, the computation depends on the measurement basis, which specifies the  $\theta$ .

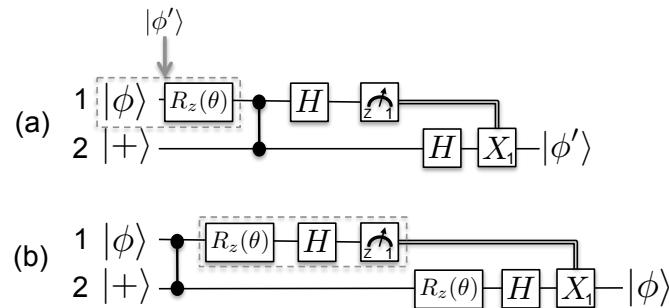


Figure 2.29: One-bit teleportation to Measurement-based Quantum Computing

An arbitrary single qubit unitary operation  $U$  can be decomposed into Euler angles:

$$U = He^{i\theta Z}e^{i\phi X}e^{i\xi Z} = He^{i\theta Z}He^{i\phi Z}He^{i\xi Z} \quad (2.153)$$

e.g., an X gate can also be described by two Hadamard gates and a Z gate  $X = HZH$ . Therefore, an arbitrary single qubit unitary operation can be represented by a sequence of one-bit teleportations.

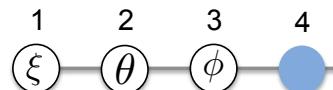


Figure 2.30: Measurement-based single qubit unitary operation

The realization of the CNOT operation can be accomplished by a 2-dimensional sequence of measurements as shown in Figure 2.31 below. Z-basis measurements are performed beforehand to omit unwanted qubits from the graph.

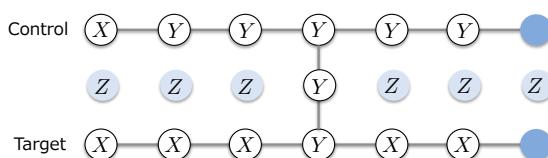


Figure 2.31: Measurement-based CNOT operation

Performing Pauli basis measurements transforms the graph states. Considering a 1D cluster state, the Z-basis measurement removes the measured qubit and disconnects the links, leaving two separate cluster states, or two independent qubits with a state  $|1\rangle$  respectively (see Figure 2.32(a)). The Y-basis measurement also removes the measured qubit but directly connects the neighbors up to the phase operations as a byproduct (see Figure 2.32(b)). Unlike the other Pauli basis measurements, the X-basis measurement transforms the linear graph into a non-linear graph. One X gate as an byproduct, and an additional Hadamard gate needs to be applied to a neighboring qubit of the measured target. While the target qubit of the byproduct X gate does not affect the overall state but fixes the phase, the target qubit of the Hadamard gate directly affects the graph as in Figure 2.32(c).

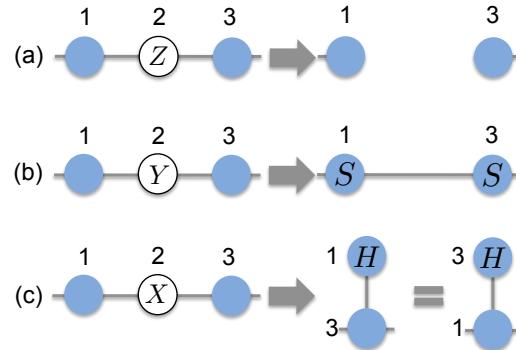


Figure 2.32: Topological transition by measurements on 1D cluster states

Similar to the 1D graph, performing measurements on a 2D cluster states affect the graph in a 2-dimensional manner. The Z-basis measurement performed on an arbitrary qubit will simply result in a new topology without the measured qubit and its links as in Figure 2.33(a). The Y-basis measurement also removes the measured qubit but leaves additional complementary links between the neighbors of the measured qubit (Figure 2.33(b)). The output topology after X-basis measurement differs according to the feedforward target, which is a neighbor qubit of the measurement target (filled with gray color in 2.33(c)). After the measurement, new complementary links are formed between the neighbors of the feedforward targeted qubit and the neighbors of the measured qubit, between the mutual neighbors of the feedforward targeted and the measured qubit, and between the feedforward targeted qubit and the neighbors of the measured qubit. Similar to other measurements, the measured qubit is removed and its links will be disconnected from other qubits.

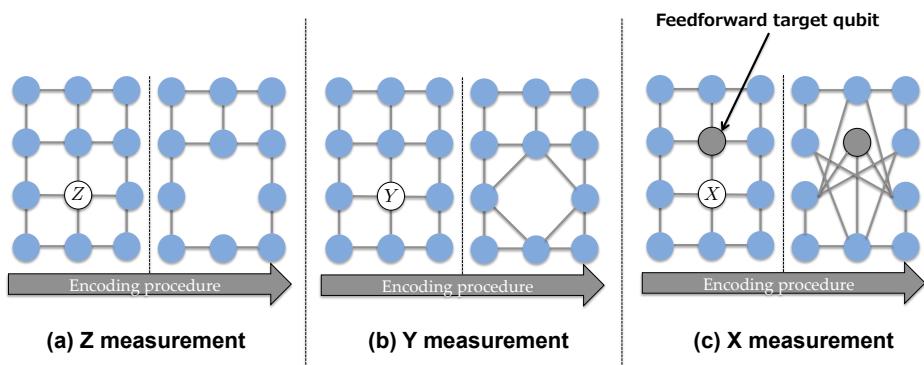


Figure 2.33: Topological transition by measurements on 2D cluster states

# Chapter 3

## Related Work

Quantum networking in general, which includes the thesis proposal in Section 5, strongly depends on the ability of experimentalists to successfully demonstrate the capability of establishing quantum communication using entangled states and distributing those entangled qubits across a long distance. Entangled states may be physically generated via photons directly or via stationary memories such as the NV diamond [37], trapped ion [38] and quantum dot [39, 40]. The experimental demonstration of quantum teleportation was accomplished in the late 1990s [23, 24].

### 3.1 Violation of a Bell inequality

One of the biggest questions that arise when dealing with quantum computing is, whether entanglement is real and is actually different from Newtonian physics based classical correlation. The CHSH inequality, which is one example of the Bell inequality, allows us to mathematically prove that quantum mechanics cannot be explained by a local hidden variable theory. In classical Newtonian physics, the correlations between outcomes of measurements on distant particles must satisfy an inequality  $|S| < 2$  ( $S$  is the correlation between measurement outcomes). Since quantum mechanics does not obey locality and realism, the same experiment on entangled particles violates the inequality and result in a different boundary  $2\sqrt{2} \approx 2.82843$  (see subsection 2.8.5 for details regarding CHSH inequality) - known as *Tsirelson's bound*.

In 2015, Poh et al. experimentally measured the correlation of maximally entangled photon polarization, and observed  $S = 2.82759 \pm 0.00051$ , which is very close to Tsirelson's bound  $S - 2\sqrt{2} = 0.00084 \pm 0.00051$  [41] and a tremendous advance in precision.

In the same year, Hensen et al. succeeded in demonstrating a loophole-free Bell inequality violation using NV electron spins separated by 1.3km [42]. Previously reported experimental results required extra assumptions which resulted in loopholes [43–46]. Using entangled electron spins of an estimated fidelity  $F = 0.92 \pm 0.03$ , 245 trials of direct CHSH-Bell inequality test have been performed, and observed  $S = 2.42 \pm 0.20$ .

Many other Bell inequality experiment related papers are summarized well in Wikipedia [47].

## 3.2 Long range entanglement distribution

In order to establish a long distance quantum communication, it is necessary to distribute the entangled qubits beforehand. In real networking, not only the fidelity but also the distribution rate plays a significant role.

In 2017, Stockill *et al.* succeeded in coupling two quantum dots located 2m apart. The observed average Bell state fidelity for  $|\Psi^+\rangle$  and  $|\Psi^-\rangle$  is  $F = 0.616 \pm 0.023$  with a high entanglement generation rate of 7.3kHz [48].

In the same year, China achieved the longest ever entanglement distribution using their quantum satellite, Micius, which was designed to produce two separate photons with entangled polarizations as  $|\Psi^+\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$  [49]. Pairs of photons are beamed down to earth under a pump power of 30 mW, with a rate of 5.9 million entangled pairs per second with a fidelity of  $0.907 \pm 0.007$ , and measured at ground stations separated by 1203km. Overall the experiment succeeded in distributing 1.1 entanglements per second on average across over 1203km distance with a fidelity  $0.869 \pm 0.085$ .

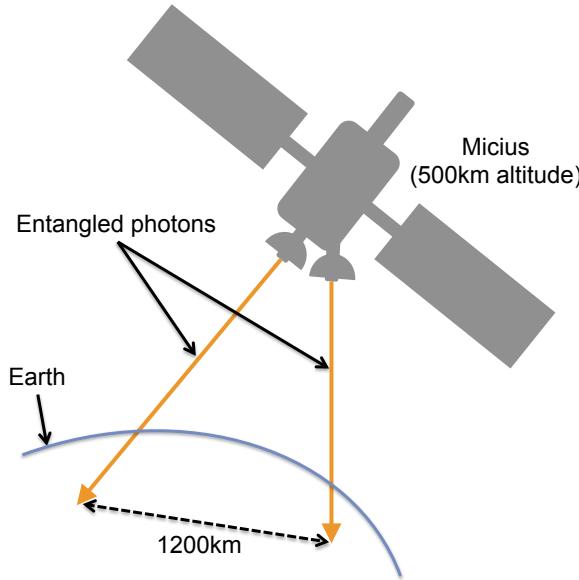


Figure 3.1: Chinese satellite beaming down entangled photons to earth

Note that the above experiments have no quantum memories used. Entangled qubits are immediately measured after reception.

## 3.3 Physical link architecture

Jones *et al.* introduced three different architectures for a repeater link; MeetInTheMiddle, SenderReceiver and MidpointSource [50].

### 3.3.1 MeetInTheMiddle

The pictorial model of the MeetInTheMiddle link is described in Figure 3.2. As shown, the MeetInTheMiddle model includes two quantum repeater nodes with a stand-alone Bell State

Analyzer (BSA) in between. The positioning of the BSA may be limited, for example, due to geographical constraints. Each node is connected via a quantum channel and a classical channel. In this model, both quantum nodes must transmit photons through the quantum channel, so that those photons reach at the BSA simultaneously for every attempt. The photon emission timing and its burst rate, therefore, must be synchronized between the nodes. For each attempt, the BSA must acknowledge the success/failure result to both nodes.

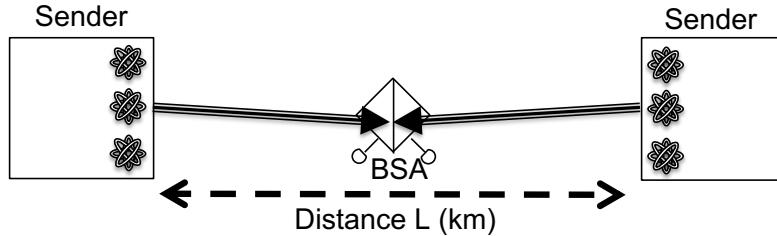


Figure 3.2: The MeetInTheMiddle model. This requires a stand-alone BSA node in between nodes.

### 3.3.2 SenderReceiver

The pictorial model of the SenderReceiver link is described in Figure 3.3. The Sender-Receiver model is very similar to the MeetInTheMiddle model, but with an internally installed BSA module at one endpoint (Receiver Node). The Sender node emits photons towards the Receiver node, and attempts to generate Bell pairs in a similar way as in the MeetInTheMiddle model. However, the Receiver node is capable of resetting memory qubits in real time at every attempt, because the success/failure results can be immediately referred locally. This is especially advantageous when the Receiver node's buffer size is smaller than that of the Sender node's. The installation cost is also less expensive compared to the MeetInTheMiddle model because we have one node fewer, but classical latency increases by a factor of two over a link with the same length  $L$ .

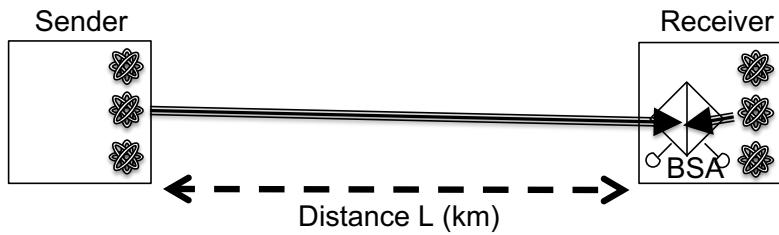


Figure 3.3: The SenderReceiver model. The BSA is installed inside one endpoint.

### 3.3.3 MidpointSource

The pictorial model of the MidpointSource link is described in Figure 3.4. This model requires one stand-alone Entangled Photon Pair Source (EPPS) node. The EPPS is a device

that is capable of generating entangled photon pairs, and forwards them to both nodes. Each node, therefore, must transmit photons from their local memory, so that its arrival time to the local BSA is synchronized to the arrival time of the photon from the EPPS. In order to entangle memory qubits node-to-node, local Bell measurements of both nodes must succeed upon the same photon pair sent from the EPPS.

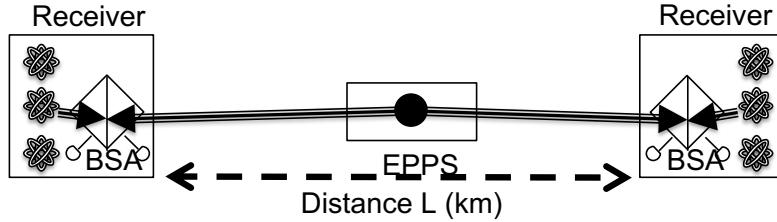


Figure 3.4: The MidpointSource model. The BSA is installed inside both nodes, with a stand-alone EPPS node in between.

The trial rate of entangling qubits is higher than the other two link architectures, but the success probability is lower.

### 3.4 Quantum data link protocol

Dahlberg et al. introduced a quantum data link protocol, and analyzed its performance over a detailed network simulation composed of a few qubits, executed on a supercomputer [51]. The required functionality for each layer in the network stack is summarized in Fig. 3.5, and the proposed architecture is shown in Fig. 3.6.

Application	
<b>Transport</b>	Qubit state transfer
<b>Network</b>	Multi-hop entanglement
<b>Link</b>	Robust entanglement gen.
<b>Physical</b>	Attempt entanglement gen.

Figure 3.5: Required functionality for each layer network stack as in [51].

The higher layer requests quantum state creation of a resource with certain parameters by communicating with the link layer. The Midpoint Heralding Protocol (MHP) periodically asks for triggering entanglement (via polling) to the link layer, and the link layer acknowledges the need with yes/no reply along with some other parameters. There are two types of requests; the *Create and Keep (K)* request or the *Create and Measure (M)* request. An example of included parameters in the *K* request may be the ID, sequence of gate operations, the type of generation and etc. The *GEN* message is transmitted to the BSA node, and the generation attempt's success/failure *REPLY* with its generation ID (which is used by the EGP

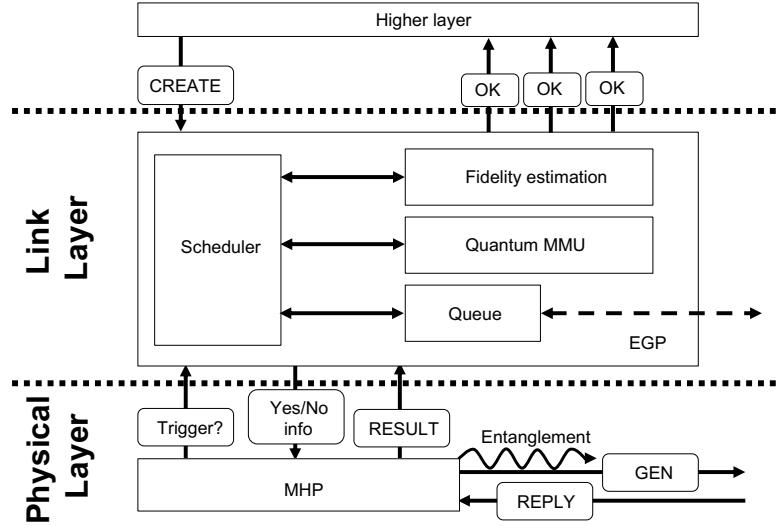


Figure 3.6: Layered architecture for quantum communication as in [51].

later) is acknowledged to nodes. This information will also be forwarded to the link layer. The  $K$  request specifies the measurement basis, and the qubit is measured along the basis immediately after the photon emission (even before receiving the REPLY message from the BSA node). The measurement result is forwarded to the EGP accordingly.

The Queue in the Entanglement Generation Protocol (EGP) is used for coordinating the MHP triggering timing. The Quantum Memory Management (QMM) tracks which physical qubit is used for which entanglement generation/storing. The Fidelity estimation unit (FEU), based on known hardware capabilities, estimates the minimal completion time for preparing an entangled resource with minimal fidelity  $F_{\min}$ . The minimal completion time is also used to reject requests from the higher layer based on the given timeout value. The FEU also performs test rounds for improving the estimation. The EGP scheduler manages which request in the queue should be served next.

### 3.5 Quantum link bootstrapping

Oka *et al.* discussed procedures for distributed quantum link bootstrapping based on link-level tomography (see Fig. 3.7(a)), and master-slave model based quantum link bootstrapping (see Fig. 3.7(b)), over a MidpointSource link [50]. The paper focuses on the accuracy of the reconstructed density matrix (reconstruction fidelity), based on the number of measured Bell pairs. The simulation assumes infinite ideal memories. As shown, both procedures assume the ability to raise the reconstruction fidelity up to a certain threshold.

Okas bachelor thesis [52] also considers recurrence purification for increasing the average resource fidelity, over a simulated noisy system with only Pauli errors. The flowchart of the quantum link bootstrapping with recurrence purification is shown in Fig. 3.8. The proposed protocol assumes the ability to raise the reconstruction fidelity and the resource fidelity up to a certain threshold.

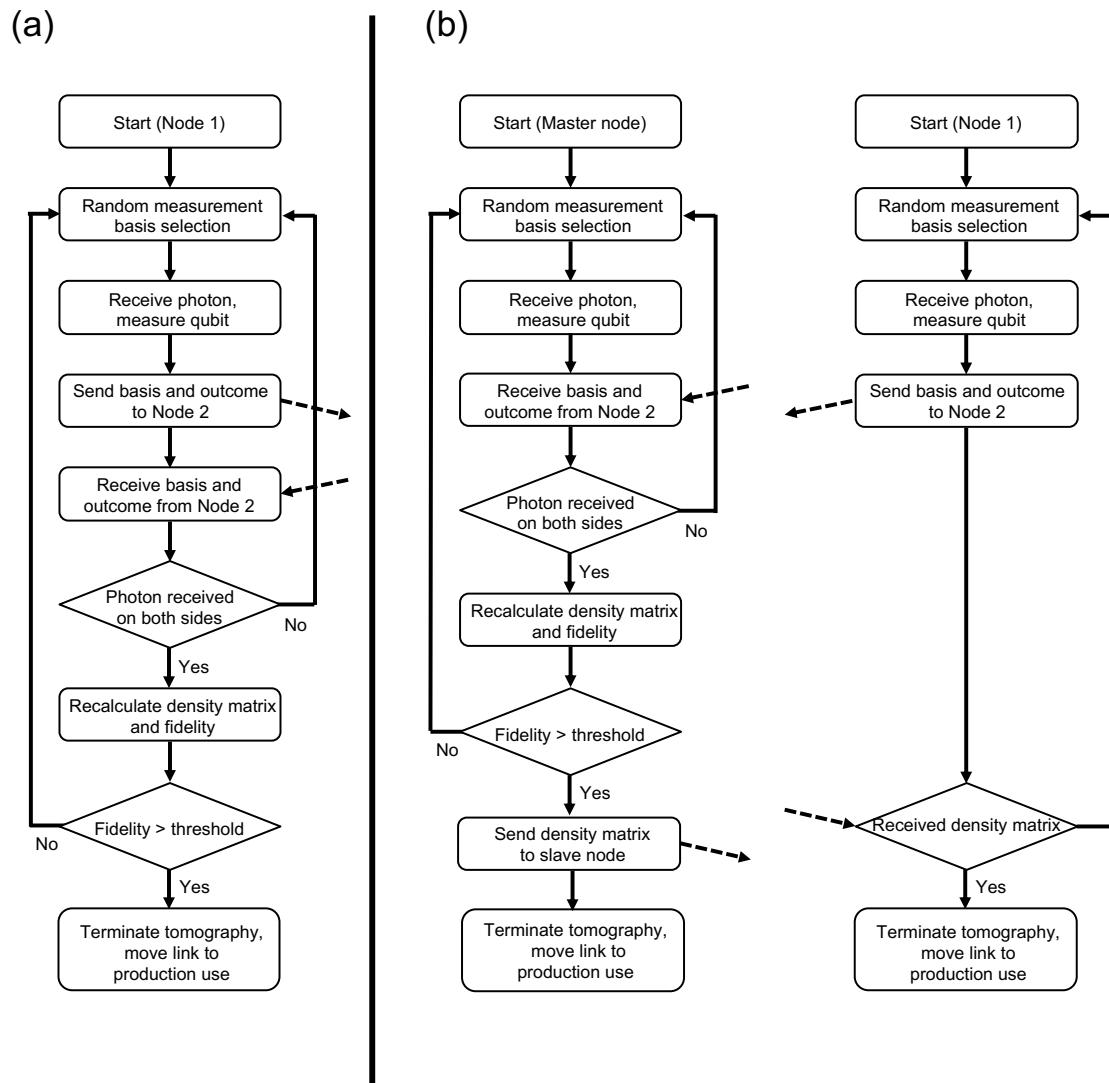


Figure 3.7: Procedures of quantum link bootstrapping. Here, the fidelity is the reconstruction fidelity. (a) Distributed model. (b) Master-Slave model.

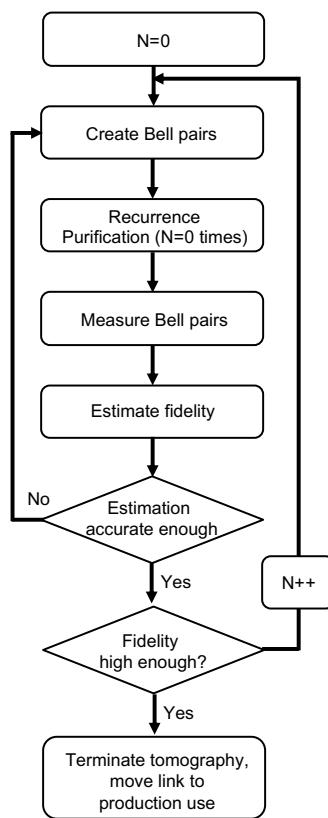


Figure 3.8: Procedure of quantum link bootstrapping with purification.

# Chapter 4

## Protocol Design

This chapter discusses the simulated protocols. The simulation design is made based on the following assumptions.

- Reliable, in-order timely delivery of classical messages.
- Paired classical and quantum channels installed between adjacent nodes.
- Perfectly synchronized clocks.
- Reliable nodes with no system failures.
- Nodes capable of determining the length of the quantum channels between neighboring nodes to sufficient accuracy.

### 4.1 Quantum data link protocol

A quantum repeater requires shared entangled resources between neighboring nodes. The role of the quantum data link protocol is to generate such resources continuously, yet autonomously. This thesis will focus on designing a data link protocol for two closely related repeater link architectures, the *MeetInTheMiddle* model and the *SenderReceiver* model (see section 3.3 for details regarding link architectures).

The core concept of the data link protocols introduced in [50] will be kept untouched. As in [50], this thesis does not elaborate how to synchronize clocks across a network, but will start with the coordination of photon emissions between adjacent nodes.

Supplemental components have been added to the protocol for the network simulation. Coordinating the arrival time of photons at the BSA via messaging between the two nodes with memory is difficult. Instead, the BSA node has been assigned to take responsibility in coordinating the entanglement generation. To start, an activated quantum node classically transmits a *Boot Up Notification* to all neighboring BSA nodes. For a *SenderReceiver* link, the sender transmits the message towards the receiver. Once received, the BSA node calculates the photon emission timings and the corresponding burst rate for neighboring nodes, based on the quantum channel lengths and its own single photon detector recovery time. Such information will be classically forwarded to the neighboring quantum nodes to start the entanglement generation. The quantum nodes also need to classically notify the BSA node

of the end of the burst. Reception triggers the BSA to recalculate the emission timing for the next round, and returns the information together with a list of *success/failure for transmission  $i$* , and node addresses specifying between which nodes the resources are shared. We buffer the success/failure results, and send them as a single packet to prevent overflowing the classical channel. Details regarding modules are also provided in section 5.1 and in section 5.2. The rule engine also operates qubit emissions for the quantum data link protocol, which is an operation for generating entangled resources (see Figure. 4.1).

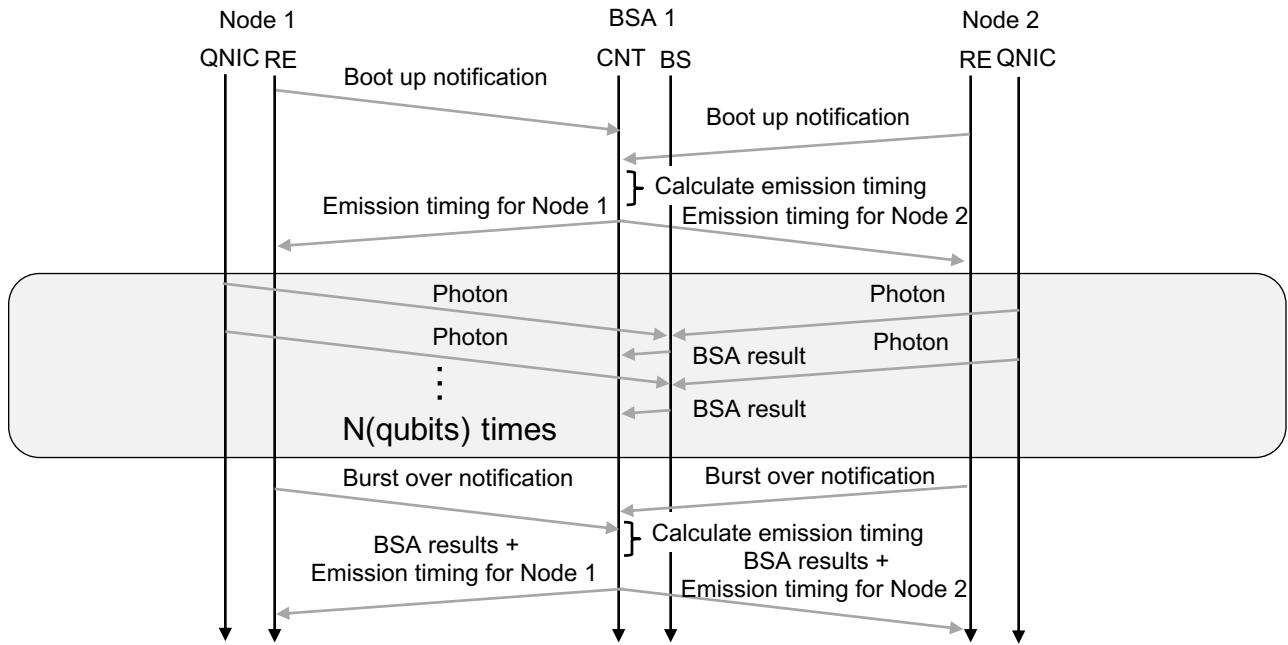


Figure 4.1: Flow chart of resource generation. This process is looped to continuously generate new resources. QNIC is Quantum Network Interface Card, RE is Rule Engine, CNT is Controller and BS is Beamsplitter. For details regarding modules, see section 5.1.

## 4.2 RuleSet-based communication protocol

Coordinated operation is a prerequisite functionality for quantum networking over both the link layer and the network layer [1]. When nodes perform a particular operation using pre-shared entangled resources, participants are assumed to perform the correct operation targeting the appropriate resources. Such consistency should not be achieved by exchanging messages with each other, especially over long distances due to the latency incurred. In this section, we introduce the concept of *RuleSet* for supporting quantum networking, which allows us to synchronize operations over a network with minimal classical message transmission. If a single connection follows a route involving  $n$  nodes, the source node requests the destination node to generate  $n$  RuleSets, and distributes them to all  $n$  nodes respectively (see Fig. 4.2) [53].

A RuleSet is an object consisting of one or more *Rules*, each holding a *Condition* and an *Action*. A Condition may have one or more *Clauses*, each of which is a conditional statement. An Action holds a list of operations that a node needs to perform in order to accomplish a

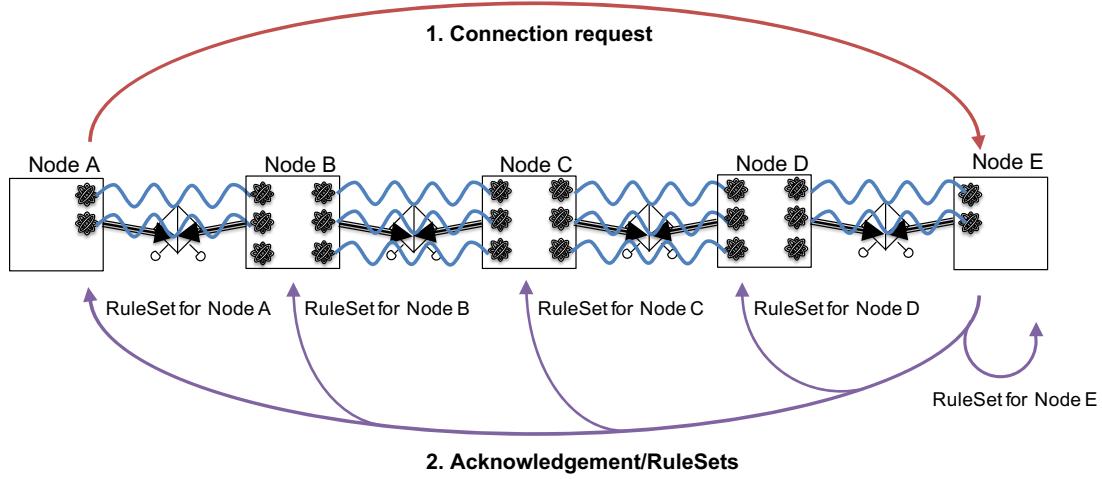


Figure 4.2: Node A generating and distributing RuleSets done at connection setup time. In this example, there are five nodes in total, which requires a total of five distinct RuleSets.

single task, which may be for example, entanglement swapping, purification, measurement for tomography, etc., and is invoked if and only if each Clause in the corresponding Condition is fulfilled. When the task involves at least one resource, the oldest available resource will be picked from the allocated set. An Action may also generate a message to another node, lock and reinitialize resources. In our current implementation, a RuleSet also has a *Termination Condition*, which is a simple counter for determining when to discard the RuleSet and discontinue the connection. This may also have multiple Clauses. Each Rule, also holds more than one *partner address*, identifying where the paired entangled qubit (or the paired node for a particular operation) is physically located. This is, of course, taken into consideration by the destination node generating the RuleSets for a particular connection.

RuleSets will be managed and executed by the RuleEngine, which is a software module installed in all quantum nodes that is responsible for interpreting the RuleSet instructions, and executing them in real time. The RuleEngine is event-driven, where the event may be a classical packet arrival, new resource allocation or RuleSet timeout. If a single RuleSet consists of multiple Rules, the RuleEngine refers to each Rule based on top-down strategy, invoking Actions from top to bottom sequentially. That is, if a node completes the Action in the first Rule on qubit A, if any, qubit A will be reassigned for the use of the second Rule in the same RuleSet. Therefore, newly generated resources are first assigned to the first Rule in the set, when the generated resource is shared between the partner node of the first Rule. They must complete the first rule before being reassigned to the second Rule. Timeout may be performed by tracking how well resources are upgraded to the upper Rules. This RuleSet level resource allocation can be directly translated into multiplexing schemes, such as buffer space multiplexing [54]. This top-down approach is also a simple solution to overcome the bias in knowledge regarding the resources across a network (see Fig. 4.3).

All RuleSets for the same connection share the same RuleSet identifier (*RuleSetID*). The RuleSet identifier is generated via a hash function using the time when the RuleSet was generated, the IP address of the node which generated the RuleSet, and a random number as a seed to avoid global conflicts. Rules are also indexed (*RuleID*) inside the RuleSet. Each Action holds a counter (*ActionIndex*), where the counter is incremented whenever the

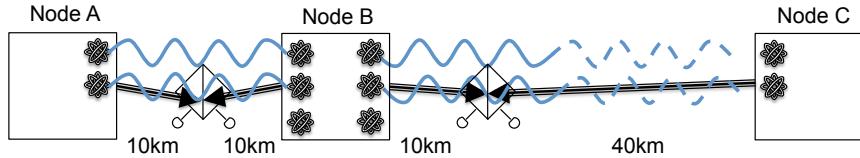


Figure 4.3: Bias in knowledge regarding the state of resources. Node C receives the success/failure acknowledgement later than Node A and B.

Action has been performed. These identifiers are used, for example, when a node needs to share the measurement result with another node for link-level tomography. Whenever an Action of a RuleSet requires a node to transmit a classical packet to another node, such as the measurement result, it encloses the identifiers in the packet, so that the receiving node can uniquely identify and pair its own measurement result with the received one accordingly. An example structure of a RuleSet composed of two Rules is shown in Fig. 4.4.

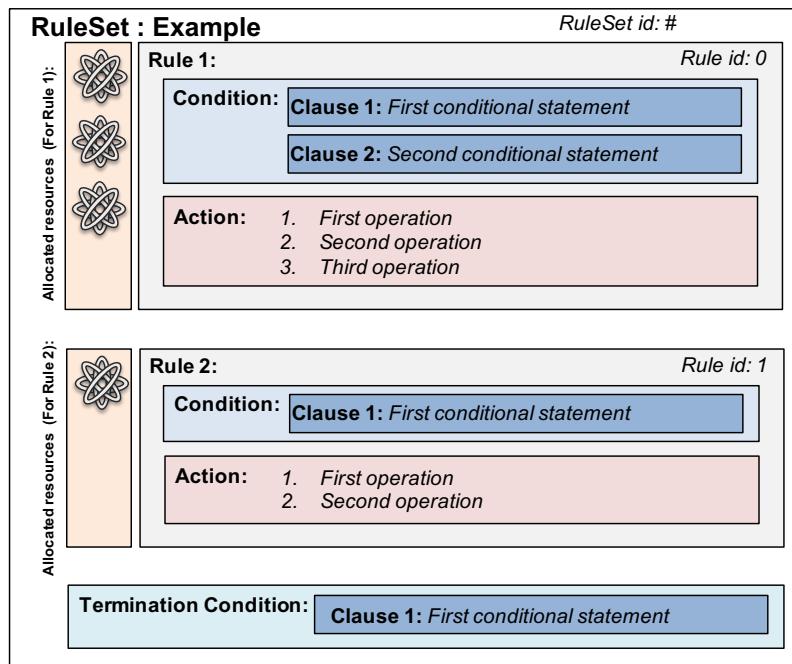


Figure 4.4: An example structure of a RuleSet. The resource allocated in the second Rule (represented by the atom symbol) has been already passed through the first.

When a RuleEngine executes an Action, the Action generally demands one or more entangled resources from its allocated list. Resources are always picked from the oldest, so that operations are performed consistently between the partner node(s). The flow chart of the RuleSet-based quantum link bootstrapping protocol is shown in Figugre 4.5.

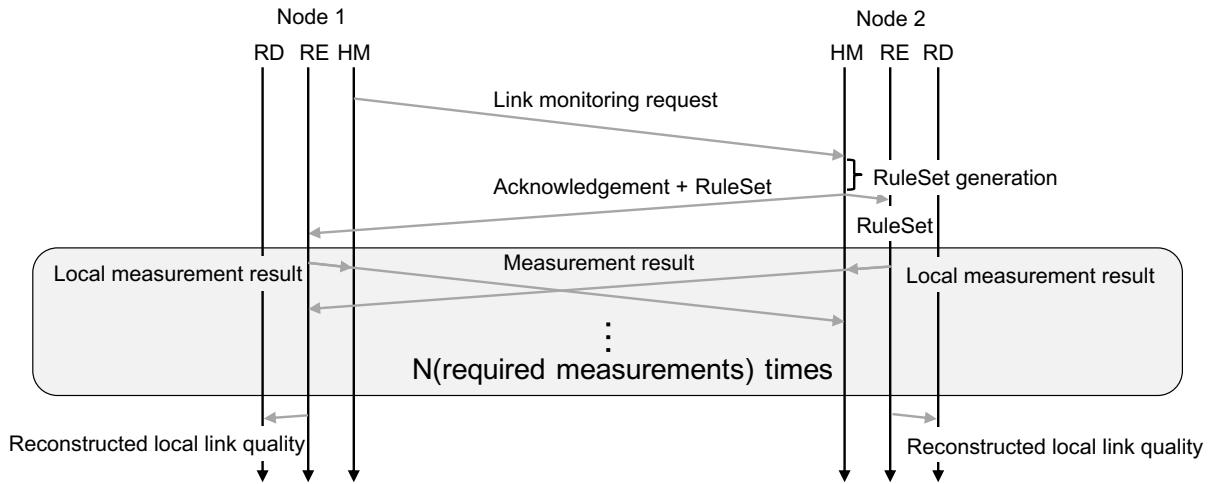


Figure 4.5: Flow chart of the hardware monitor working on link-level tomography. RD is Routing Daemon, HM is Hardware Monitor and RE is Rule Engine. For details regarding modules, see section 5.1.

### 4.3 Pseudocode of Rules

This subsection provides examples of Rules.

### 4.3.1 Single selection-Single error Purification (Ss-Sp) Rule

The Ss-Sp is the simplest method to detect either the X error or the Z error (see Fig. 6.3 in section 6.2 for the circuit).

## Condition

The Condition for the Ss-Sp Rule requires a single Clause, which is the *ResourceConditionClause* shown in Algorithm. 1. This Clause's role is to check the number of available resources, where *numRequired* = 2 in this case. A resource is *locked* when it should not be disturbed – e.g. waiting for purification result.

---

**Algorithm 1** ResourceConditionClause

---

**This Clause checks if enough resources are available for the corresponding Action.**

Input: resourceList  $\leftarrow$  List of allocated resources for the Rule. A resource may be locked when it is waiting for the classical packet to arrive, such as for purification.

Output: enoughResources  $\leftarrow$  A boolean value.

```
1: procedure RESOURCECONDITIONCLAUSE(resourceList)
2:   numRequired  $\leftarrow$  Number of required resources for the Action
3:   numFree = 0
4:   enoughResources = false
5:   for each resource in resourceList do
6:     if resource is not locked then
7:       numFree++
8:     end if
9:     if numFree  $\geq$  numRequired then
10:      enoughResources = true
11:      break
12:    end if
13:   end for
14:   return enoughResources
15: end procedure
```

---

**Action**

The Action for the Ss-Sp Rule simply picks two available resources, and performs an X purification. The purified resource will be locked until the node gets the paired measurement result from the other node.

---

**Algorithm 2** Ss-SpAction

**This Action performs an X error purification based on single selection. The measurement result is stored, and then sent to the partner node as a message.**

Input:  $\text{resourceList} \leftarrow$  List of allocated resources for the Rule.

Output:  $\text{msg} \leftarrow$  A message for another node.

**Require:**  $\text{ResourceConditionClause} == \text{true}$

```
1: procedure Ss-SPACTION(resourceList)
2:   self_addr  $\leftarrow$  Node address of this RuleSet owner.
3:   partner_addr  $\leftarrow$  Node address of tomography partner.
4:   resource1 = Select a free resource from resourceList
5:   resource2 = Select another free resource from resourceList
6:   resource2.CNOT(resource1) /*Target.CNOT(Controlled)*/
7:   basis = Z
8:   outcome = resource2.Measure(basis)
9:   removeResourceFromList(resourceList,resource2)
10:  lockResourceInList(resourceList,resource1)
11:  Data.result = {outcome, basis}
12:  Data.ruleSetId = this.RuleSetID
13:  Data.ruleId = this.RuleID
14:  Data.actionIndex = this.ActionIndex
15:  save(Data)
16:  msg.destination = partner_addr
17:  msg.source = self_addr
18:  msg.data = Data
19:  this.ActionIndex++
20:  return msg
21: end procedure
```

---

Each node will exchange its measurement outcome. The locked resource will only be unlocked and reassigned to the next Rule when the measurement outcomes coincide. Otherwise, purification failed and the locked resource is removed from the  $\text{resourceList}$  and returned to the pool of unused resources for reinitialization and reuse.

### 4.3.2 Double selection-Single error Purification (Ds-Sp) Rule

The Ds-Sp is a purification method similar to Ss-Sp but with double selection (see Fig. 6.5 in section 6.2 for the circuit) [55].

#### Condition

The only Clause required for the Ds-Sp Rule is also the *ResourceConditionClause* (see Algorithm. 1 above).

## Action

The Action for the Ds-Sp Rule picks three available resources, and performs an X purification based on double selection. The purified resource will be locked until the node gets the paired measurement result from the other node.

---

### Algorithm 3 Ds-SpAction

---

**This Action performs an X error purification based on double selection. The measurement result is stored, and then sent to the partner node as a message.**

Input: resourceList  $\leftarrow$  List of allocated resources for the Rule.

Output: msg  $\leftarrow$  A message for another node.

**Require:** ResourceConditionClause == true

```
1: procedure Ds-SPACTION(resourceList)
2:   self_addr  $\leftarrow$  Node address of this RuleSet owner.
3:   partner_addr  $\leftarrow$  Node address of tomography partner.
4:   resource1 = Select a free resource from resourceList
5:   resource2 = Select another free resource from resourceList
6:   resource3 = Select another free resource from resourceList
7:   resource2.CNOT(resource1) /*Target.CNOT(Controlled)*/
8:   resource2.CNOT(resource3) /*Target.CNOT(Controlled)*/
9:   resource3.H()
10:  basis = Z
11:  outcome1 = resource2.Measure(basis)
12:  outcome2 = resource3.Measure(basis)
13:  removeResourceFromList(resourceList,resource2)
14:  removeResourceFromList(resourceList,resource3)
15:  lockResourceInList(resourceList,resource1)
16:  Data.result = {{outcome1, basis},{outcome2, basis}}
17:  Data.ruleSetId = this.RuleSetID
18:  Data.ruleId = this.RuleID
19:  Data.actionIndex = this.ActionIndex
20:  save(Data)
21:  msg.destination = partner_addr
22:  msg.source = self_addr
23:  msg.data = Data
24:  this.ActionIndex++
25:  return msg
26: end procedure
```

---

Each node will exchange its measurement outcome. The locked resource will only be unlocked when the each of the paired measurement outcomes coincide. Otherwise, purification failed and the locked resource is removed from the *resourceList*.

### 4.3.3 Full-state link-level tomography Rule

#### Condition

The Condition for the tomography Rule requires two Clauses. One of them is the *Resource-ConditionClause* (Algorithm. 1), and another is the *MeasurementConditionClause* (Algorithm 4). The *MeasurementConditionClause* is a simple counter for checking the number of measured qubits.

---

#### Algorithm 4 MeasurementConditionClause

---

**This Clause checks if enough measurements has been performed.**

Input: none.

Output: enoughMeasurements  $\leftarrow$  A boolean value. True if enough measurements have been performed.

```
1: procedure MEASUREMENTCONDITIONCLAUSE( )
2:   numRequired  $\leftarrow$  Number of required measurements for tomography
3:   numCurrent = 0
4:   enoughMeasurements = false
5:   if numCurrent < numRequired then
6:     numCurrent++
7:     return enoughMeasurements=false
8:   else
9:     return enoughMeasurements=true
10:  end if
11:  return enoughMeasurements
12: end procedure
```

---

#### Action

The Action for the tomography selects a qubit, and performs a measurement on it based on a randomly selected basis.

---

**Algorithm 5** TomographyAction

---

**This Action performs measurement on an entangled qubit. The measurement result and the selected basis is stored, and sent to the partner node as a message.**

Input: resourceList  $\leftarrow$  List of allocated resources for the Rule.

Output: msg  $\leftarrow$  A message for another node.

**Require:** ResourceConditionClause == true && MeasurementConditionClause == true

```
1: procedure TOMOGRAPHYACTION(resourceList)
2:   self_addr  $\leftarrow$  Node address of this RuleSet owner.
3:   partner_addr  $\leftarrow$  Node address of tomography partner.
4:   resource = Select a free resource from resourceList
5:   basis = RandomBasisSelect({X,Y,Z})
6:   outcome = resource.Measure(basis)
7:   removeResourceFromList(resourceList,resource)
8:   Data.result = {outcome, basis}
9:   Data.ruleSetId = this.RuleSetID
10:  Data.ruleId = this.RuleID
11:  Data.actionIndex = this.ActionIndex
12:  save(Data)
13:  msg.destination = partner_addr
14:  msg.source = self_addr
15:  msg.data = Data
16:  this.ActionIndex++
17:  return msg
18: end procedure
```

---

Selected measurement basis and its outcome will be stored locally, and shared to the partner node.

# Chapter 5

## Quantum network simulator

The quantum network simulator implemented for this thesis is built on top of a publicly available discrete event network simulator named *OMNeT++* [56]. OMNeT++ provides users an easy framework for defining network topologies and system area topologies for nodes. Each software/hardware element is defined as a *module* in OMNeT++, and their functionalities must be programmed by the user. However, some of the basic functions, such as for running Dijkstra's algorithm (weighted or not), are provided as a standard function. Different classical packets can also be defined very easily as OMNeT++ messages. Simulations are runnable in CUI mode, or in GUI mode with animation.

Because OMNeT++ is a classical networking simulator, components for simulating the physics in quantum networking must be added. Basic mathematics, such as for measuring a qubit with a specific density matrix, have been accomplished though the use of *Eigen* package [57]. Physical entanglement of a qubit module has been tracked using a pointer (pointing to another qubit module), and distributed states are directly accessed and updated on demand.

The overall simulator size for this thesis is approximately 10,000 lines of code (see Appendix. A.1) exclusive of OMNeT++ and Eigen. The following subsections discuss the overall structure of the simulator and its flow. The simulator is designed to evaluate various protocols over a single network, and over the quantum Internet. The Quantum Internet will be a world-wide network interconnecting diverse quantum networks, both small scale and large scale [4–7]. These independent, interconnected networks are utilizing different technologies and managed by different organizations (see Fig. 5.1), known as Autonomous Systems in the classical Internet. The role of such a network is similar to the classical Internet: to provide to users a quantum information service between arbitrary nodes.

### 5.1 Design of a Quantum node architecture

Building a Quantum network simulator requires designing the hardware/software architecture of a quantum node, which includes end-nodes, quantum repeaters and quantum routers. The challenge is that designing the architecture requires understanding the required module functionalities and protocol specifications across different layers. In our work, we define a node with a single quantum network interface card (QNIC) as an *end-node*. Similar to the classical network interface card (NIC), the QNIC is connected to another node's QNIC via a

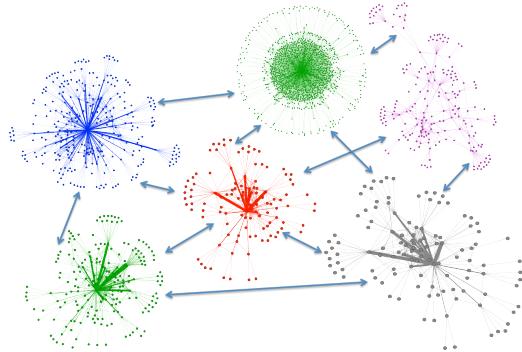


Figure 5.1: The Quantum Internet. Different networks are operated under different technologies. That includes the physical link architecture, data link protocol, routing protocol and any other necessary technology layered as in the OSI reference model [1]. Connecting those networks forms the Quantum Internet.

quantum channel. The end-node regularly entangles its own qubits with its neighbor's. An end-node also generates and transmits a classical connection setup request to another end-node through the classical channel. A node with two QNICs is a *quantum repeater*. The role of a quantum repeater is to forward classical packets, to create base-level entanglement over a link and to monitor and manage errors. The third type of nodes are the *quantum routers*, which have similar roles to quantum repeaters, but also takes responsibility for routing, for both classical and quantum information. A border quantum router, which is a router connected to another border quantum router of a different network, may have minimum trust between networks – a router may have no knowledge of the internals of other autonomous networks. Backbone routers also physically and logically link heterogenous networks. Below is the architecture of the simplest quantum node, which is the end-node (see Fig. 5.2) – a quantum repeater or a quantum router has the same architecture, except for the number of QNICs and its need to correctly route packets. Note that the required hardware/software still varies depending on the design decisions. The design is intended for enabling the RuleSet-based communication protocol.

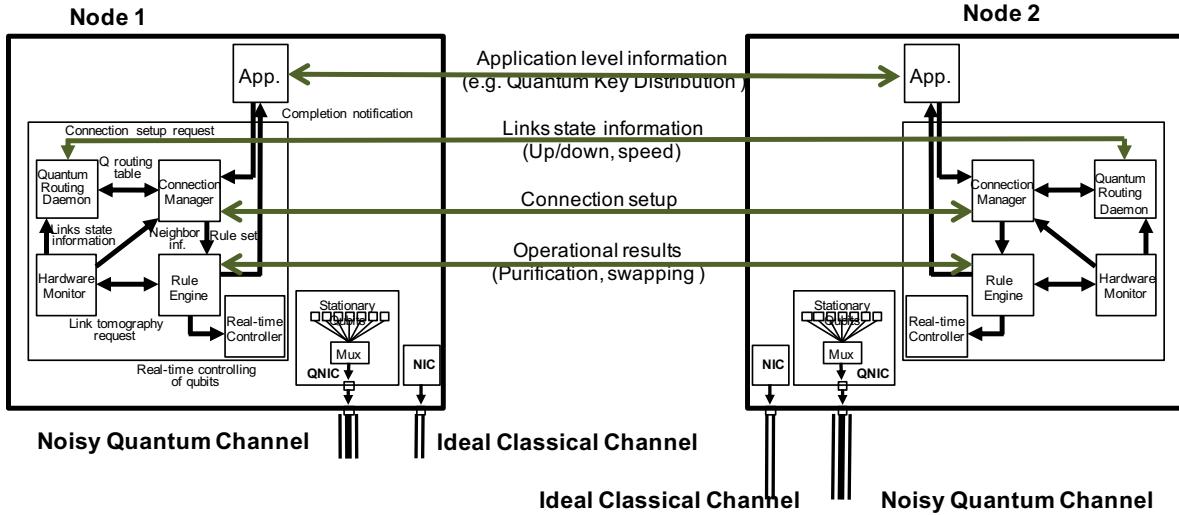


Figure 5.2: Quantum node architecture.

### 5.1.1 Network Interface Card (NIC)

A NIC is connected to the classical channel, and is capable of forwarding classical packets out to the channel.

**Assumption:** All NICs are connected to ideal classical channels – no error and infinite bandwidth and the same latency as corresponding QNICs.

### 5.1.2 Quantum Network Interface Card (QNIC)

A QNIC is connected to the quantum channel. This module contains multiple quantum memories, each capable of emitting one entangled photon out to the channel through the multiplexer (MUX).

**Assumption:** The simulation assumes the ability to perform any multi-qubit operations between qubits in any local QNICs. The error rate of a particular quantum gate can be set on a node-by-node basis but is fixed throughout the simulation. Gate times are assumed to be negligible.

### 5.1.3 Application module

The application generates the connection setup request to an arbitrary end-node, and passes the request to the connection manager. The connection setup request may include the required number of Bell-pairs and their minimum fidelity for the application. This module may directly be controlled by a user.

### 5.1.4 Hardware Monitor

The hardware monitor knows which local QNIC is connected to which neighboring node. An example of the table is shown below in Tab. 5.1.

Table 5.1: Example of a table keeping the pair of the local QNIC id and its connected neighbor address. Neighboring nodes of Node 1 are assumed to be Node 2, 3 and 4

Next hop	QNIC id
Node 2	QNIC 0
Node 3	QNIC 1
Node 4	QNIC 2

Its major role is to monitor and update the states of quantum links (up or down), by generating a request for checking the link quality based on tomography. When received a request, the hardware monitor also generates a RuleSet for the link monitoring. Based on the accumulated measurement results (basis and outcome), the hardware monitor reconstructs the density matrix based on a given algorithm (for details regarding the full-state reconstruction, see Sec. 2.8.4). The acquired link information may be stored by the node as in Tab. 5.2.

Table 5.2: Example of a table keeping the link fidelity and the link throughout information.

QNIC id	Fidelity	Throughput (Bell pairs per second)
QNIC 0	0.88	3312
QNIC 1	0.89	10011
QNIC 2	0.93	5504

The estimated link quality is then provided to the local routing daemon. The overall flow of the process has been provided in Figure .4.5 in section 4.2.

**Assumption:** The hardware monitor is assumed to be able to access to the exact length of the channels.

### 5.1.5 Routing Daemon

The routing daemon communicates with the neighbors to exchange the link state information received from the hardware monitor. The module accumulates all node/link information in a single network, and creates a quantum routing table based on a predefined algorithm (e.g. Dijkstra's algorithm [58]). This daemon, like its classical counterpart, can theoretically support a variety of routing protocols, whether link state, distance vector, or other. The BSA node is not visible in the routing system because the BSA can be considered as a part of the physical link. The message exchange timing diagram with three nodes is provided in Fig. 5.3.

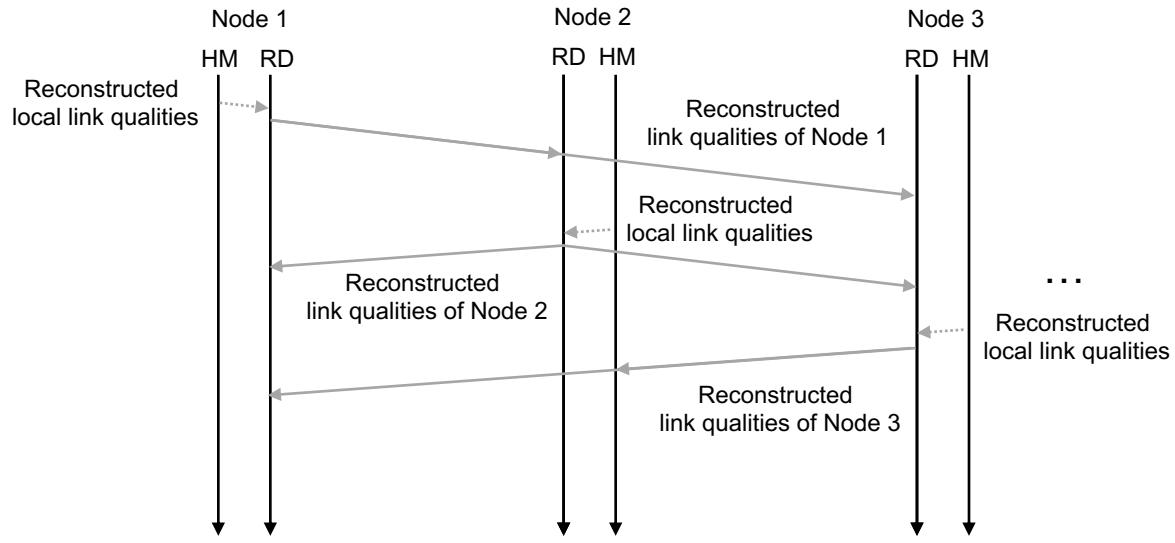


Figure 5.3: Message exchange of the routing daemon exchanging link qualities. RD is Routing Daemon and HM is Hardware Monitor. Link quality may include the achievable fidelity with its corresponding throughput.

A quantum routing table is distinct from the classical routing table, but the overall structure and use remains the same (see Tab. 5.3). The corresponding example topology is shown in Fig.5.4.

Table 5.3: Example quantum routing table of Node 1. A node only knows which neighbor to communicate with. Neighboring nodes of Node 1 are assumed to be Node 2, 3 and 4

Destination	Next hop
Node 2	Node 2
Node 3	Node 3
Node 4	Node 4
Node 5	Node 2
Node 6	Node 3

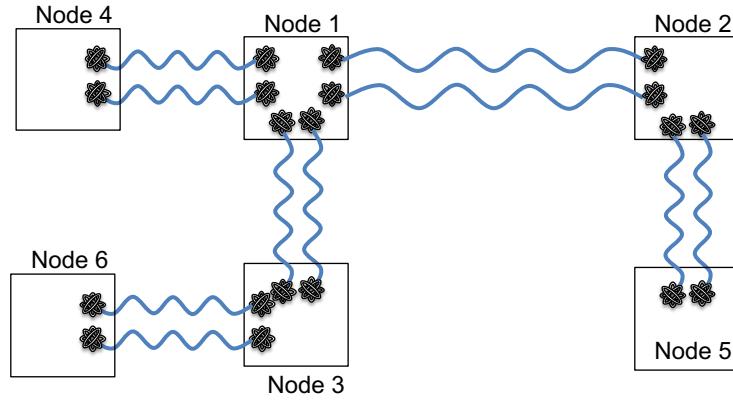


Figure 5.4: Example topology based on the routing table in Tab.5.3. Here, there is only one path available to each node from Node 1.

### 5.1.6 Connection Manager

The connection manager forwards the connection setup packet to the next quantum node by referring to the quantum routing Tab. The destination node creates the rule set for a particular connection, and distributes it back to all involving nodes (see Fig. 5.5). A RuleSet consists of multiple Rules, each with an Action and a Condition. The Action is a set of operations for accomplishing a single task, and the Condition tells when to execute it. For details, see Sec. 4.2.

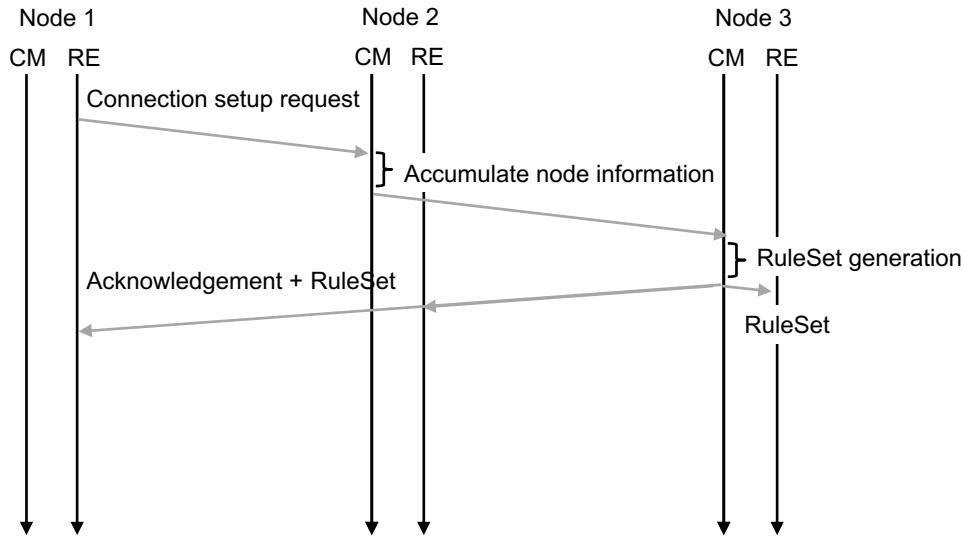


Figure 5.5: Flow chart of the connection setup including three nodes. RE is Rule Engine and CM is the Connection Manager.

### 5.1.7 Rule Engine

The rule engine is event-driven (e.g. classical packet arrival, new resource allocation or RuleSet timeout). It is responsible for interpreting and executing operations in real time through

the real-time controller based on the received RuleSet(s). The rule engine also executes the quantum data link protocol for generating new resources (see Fig. 4.1 in section 4.1). Those newly generated resources are assigned to the running RuleSets by the RuleEngine.

**Assumption:** Rule engines have perfectly synchronized clocks between nodes. Therefore, for example, each node is capable of transmitting a photon exactly at the desired timing.

### 5.1.8 Real-time Controller

The real-time controller has direct access to all local QNICs and their stationary qubits. Quantum operation, such as qubit measurement, is accomplished through this module<sup>1</sup>.

## 5.2 Design of a Bell State Analyzer (BSA) node/module architecture

Below is the architecture of a BSA node (see Fig. 5.6). The BSA is installed in between two nodes along the quantum channel for a MeetInTheMiddle link, or is directly installed at one endpoint's QNIC for a SenderReceiver link. This node's main role is to coordinate the resource generations of neighboring nodes.

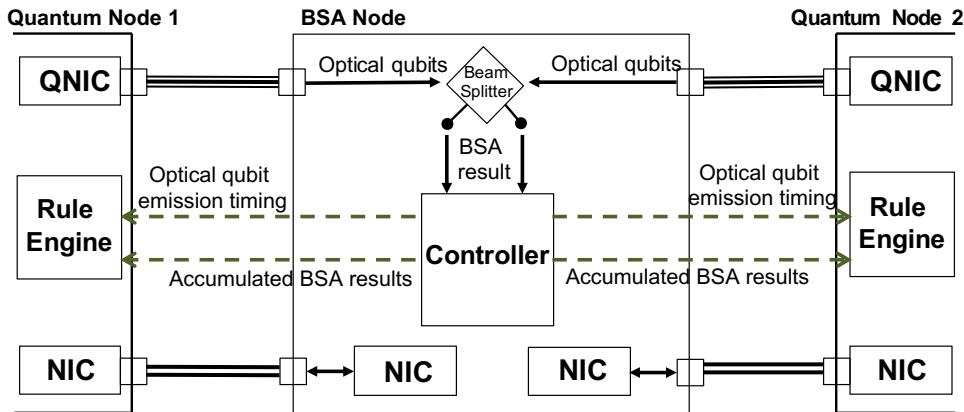


Figure 5.6: BSA node architecture. Classical communications, shown as dashed arrows, will be sent through the classical channel via NICs.

### 5.2.1 Beamsplitter & Photon detectors

Optical qubits arrive at the beamsplitter from QNICs via quantum channels. The beam splitter with its detectors' role is to perform Bell measurements on the received qubits to stochastically generate entangled resources. The two detectors' results will be forwarded to the controller module. The first step is to establish that photons can be made indistinguishable from the two nodes. This is confirmed using a Hong-Ou-Mandel test. Then hardware parameters are adjusted to erase the which-path information while detecting photons at both detectors.

<sup>1</sup>In the simulator, however, the Rule Engine directly executes the function of an Action.

## 5.2.2 Controller

The controller calculates the emission timings for both nodes, and also forwards the accumulated success/failure results.

**Assumption:** The controller also assumes that each node has the ability to emit photons at the exact timing provided.

## 5.3 Simulating errors

The error evolution is modeled based on Markov-Chain Monte-Carlo simulation.

### 5.3.1 Noisy memory errors

When a qubit is waiting for some time  $t$  since its initialization, we may observe an error on that qubit. The simulator uses a row vector to describe the present state of a qubit, which is one of the seven distinguishable states – no error, X error, Z error, Y error, excited, relaxed or completely mixed. For example, a Bell pair with no error as an initial input state can be described as in Eq. 5.1.

$$\vec{\pi}(0) = \begin{pmatrix} \text{Clean} & \text{Xerror} & \text{Zerror} & \text{Yerror} & \text{Excited} & \text{Relaxed} & \text{Mixed} \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (5.1)$$

Accordingly, the infinitesimal generator for our continuous-time Markov-Chain [59] we use for the memory error simulation also consists of seven states,

$$\bar{Q} = \begin{pmatrix} \text{Clean} & \text{Xerror} & \text{Zerror} & \text{Yerror} & \text{Excited} & \text{Relaxed} & \text{Mixed} \\ (1 - \Sigma_0) & P_X & P_Z & P_Y & P_E & P_R & 0 \\ P_X & (1 - \Sigma_1) & P_Y & P_Z & P_E & P_R & 0 \\ P_Z & P_Y & (1 - \Sigma_2) & P_X & P_E & P_R & 0 \\ P_Y & P_Z & P_X & (1 - \Sigma_3) & P_E & P_R & 0 \\ 0 & 0 & 0 & 0 & (1 - \Sigma_4) & P_R & 0 \\ 0 & 0 & 0 & 0 & P_E & (1 - \Sigma_5) & 0 \\ 0 & 0 & 0 & 0 & P_E & P_R & (1 - \Sigma_6) \end{pmatrix} \quad (5.2)$$

Element  $P_X$ ,  $P_Y$  and  $P_Z$  correspond to Pauli X, Y and Z error accordingly.  $P_E$  and  $P_R$  are memory excitation and relaxation errors, which are used to represent  $T_1$  time and the Boltzmann thermal distribution.  $\Sigma_i$  is the sum of every other element in row  $i$ . Mixed state occurs due to dark counts, or when the paired qubit gets excited/relaxed. The probability distribution of a qubit output state after time  $t$  can be found by:

$$\vec{\pi}(t) = \vec{\pi}(0)\bar{Q}^t. \quad (5.3)$$

Each element in the output vector  $\vec{\pi}(t)$  represents the probability for the qubit being in the corresponding state. The qubit state can be determined via a random selection based on the output probability distribution, and the density matrix is constructed on demand. This process is independently executed whenever a qubit is soon to be used by the simulator.

### 5.3.2 Noisy channels

The simulation of a noisy channel is done similarly to the memory error simulation, except the error rate will be per unit distance. Because each channel between nodes have a static length, we only have to exponentiate the matrix once for each channel at the beginning of the simulation. The infinitesimal generator describing the error model per unit distance for the quantum channel is:

$$\overline{Q} = \begin{pmatrix} \text{Clean} & \text{Xerror} & \text{Zerror} & \text{Yerror} & \text{Lost} \\ (1 - \Sigma_0) & P_X & P_Z & P_Y & P_L \\ P_X & (1 - \Sigma_1) & P_Y & P_Z & P_L \\ P_Z & P_Y & (1 - \Sigma_2) & P_X & P_L \\ P_Y & P_Z & P_X & (1 - \Sigma_3) & P_L \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (5.4)$$

Element  $P_X$ ,  $P_Y$ ,  $P_Z$  and  $P_L$  correspond to Pauli X, Y, Z, and photon loss error accordingly.  $\Sigma_i$  is the sum of every other element in row  $i$ . Given the input state  $\vec{\pi}(0)$ , the probability distribution of a qubit output state after traveling a channel with distance of  $d$  can be found by:

$$\vec{\pi}(d) = \vec{\pi}(0) \overline{Q}^d. \quad (5.5)$$

### 5.3.3 Bell State Analysis (BSA) error

There are cases where the BSA succeeds in entangling the memory qubits with errors on the consumed photons. The Fig. 5.7 is a case where one of the received photon included a Y error. In this case, if the BSA succeeds in entangling the memory qubits, the Y error will be reapplied as an error on the source memory qubit. Notice that there is no physical difference between  $Y_A |\Psi_{A,B}^+\rangle$  and  $Y_B |\Psi_{A,B}^+\rangle$ .

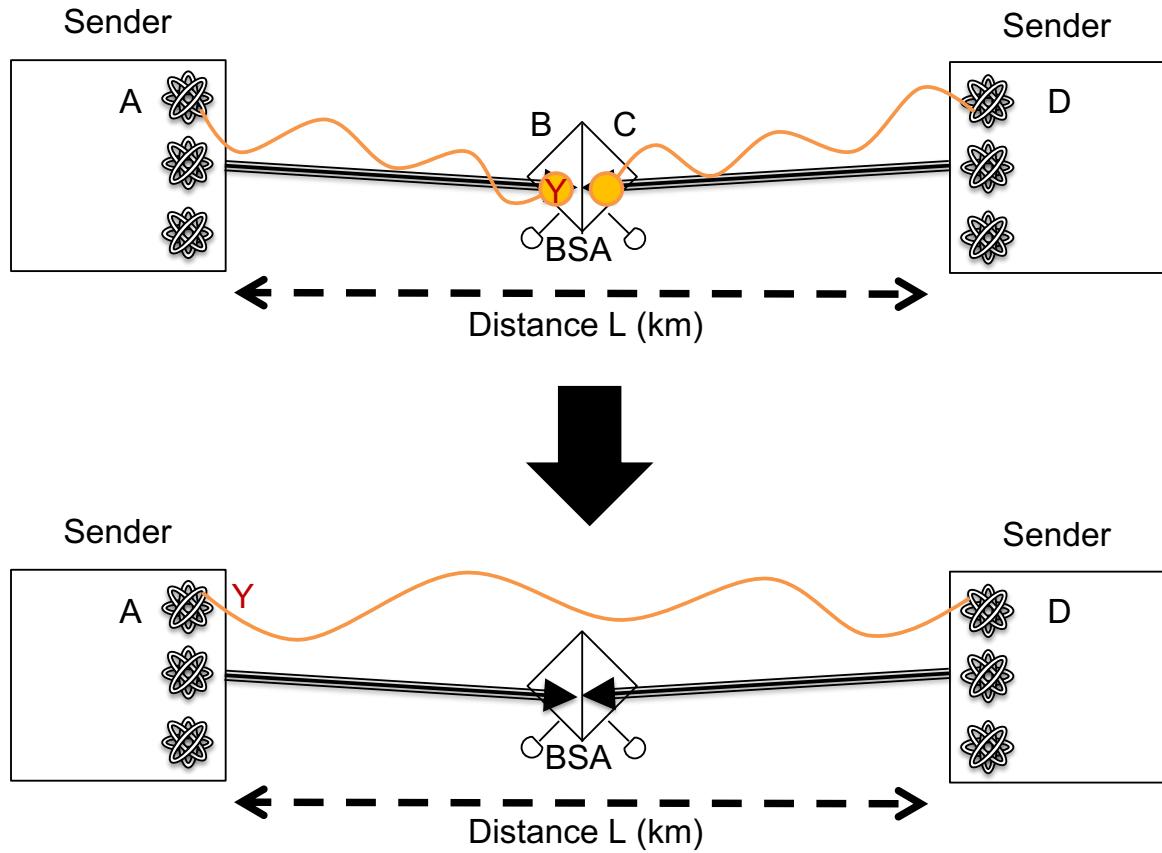


Figure 5.7: Bell measurement with a Y error on the received photon.

A photon may also be lost while traveling through the channel. Most of the cases, this error will be detected because less than 2 detector will click. However, depending on the detector darkcount rate, the system may mis-recognize qubits as entangled. In this case, both memory qubits will have either  $|0\rangle$  or  $|1\rangle$  classically.

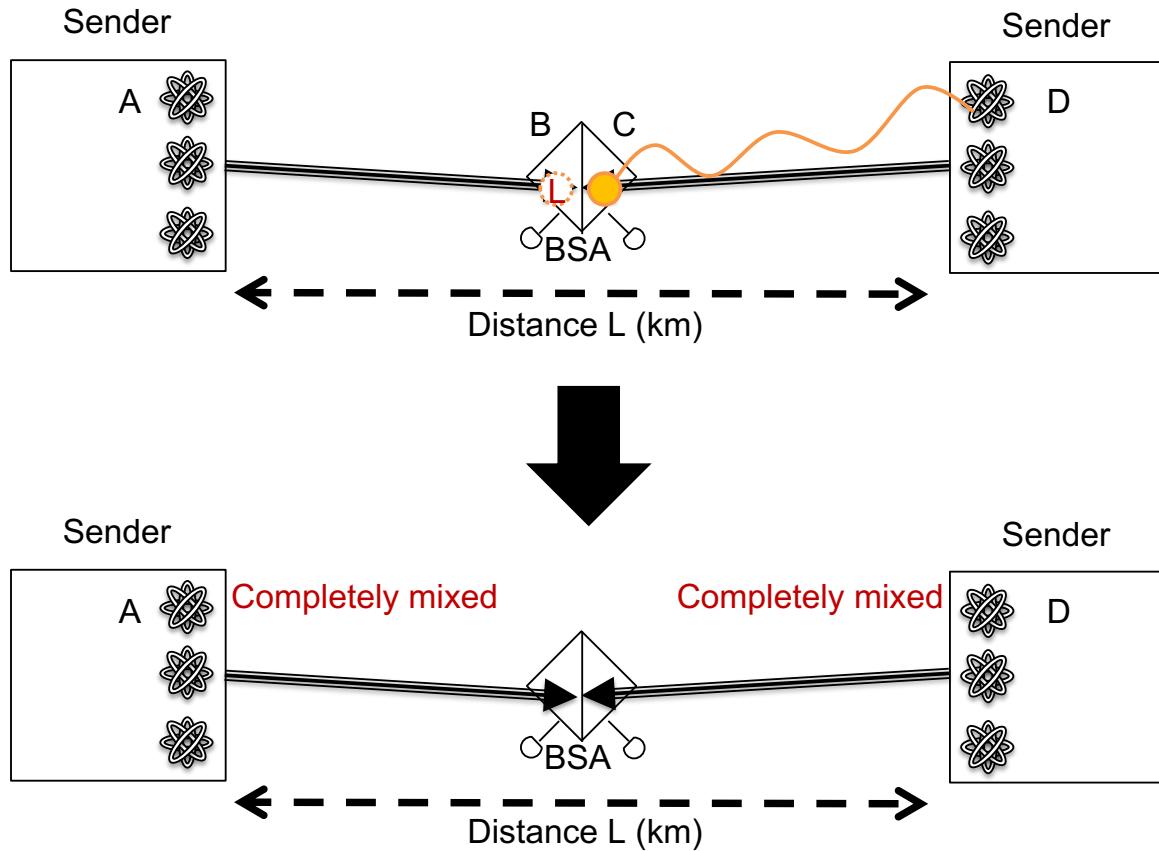


Figure 5.8: Bell measurement with a lost error on the received photon.

### 5.3.4 Noisy quantum gates

Quantum gates are modeled to stochastically generate one of the Pauli errors (X, Y or Z error) per each attempt based on a user-defined static probability distribution. To simulate the imperfection of a quantum circuit, the simulator classically tracks the error propagation throughout the circuit based on a standard Monte-Carlo simulation. Notice that errors on the input state will of course propagate through the circuit, but each gate may also generate a new error. For details regarding error propagations, see Sec. 2.9.1.

# Chapter 6

## Evaluation

In this chapter, we study the capability of a first generation quantum repeater link, with a limited amount of resources, and under noisy conditions generated by various hardware imperfections. Simulated error sources include quantum memories, quantum channels, quantum gates and single photon detector darkcounts. Our Markov-Chain Monte-Carlo simulation uses similar parameters as in [60] but with larger memory buffers (see Table 6.1 for details). We use the memoryless Markov-Chain in the simulation to dynamically model errors on memory qubits independently, based on a given lifetime  $T_1$  [61]. Other error sources, such as the gate error, are simulated via a random selection with static probabilities. In order to reduce the computation time, we only propagate Pauli errors through circuits as in [62,63]. The simulated purification circuit, therefore, is incapable of stochastically detecting excited/relaxed/completely mixed errors. Hence, our simulation generates a pessimistic output fidelity, and an optimistic output resource generation rate. To concretely identify the real impact of imperfect quantum systems, we assume ideal classical communication channels. Because classical communication latencies cover the majority of the simulated time, we assume negligible gate times. Furthermore, multi-qubit operations can also be performed between arbitrary qubits across QNICs within the same node. For each data point in the figures, a total of 25 simulations has been performed to quantify the average behavior.

Table 6.1: Default parameters used for all simulations (unless explicitly mentioned).

Key	Value
Fiber refractive index	1.44 [64]
Fiber Pauli error rate (per km)	0.03 <sup>1</sup>
Fiber photon loss rate (per km)	0.04501 <sup>2</sup> [64]
Memory Pauli error rate (per sec)	1/3 <sup>1</sup> [65]
Memory lifetime	50ms <sup>3</sup>
Emission probability into the zero phonon line	0.46 <sup>4</sup> [66]
Photon collection efficiency	0.49 <sup>4</sup> [67, 68]
Photon detector efficiency	0.8 [68]
Photon detector darkcount rate (per sec)	10 [68]
Photon detector recovery time	1ns [69]
Single-qubit gate error rate	0.0005 <sup>1</sup> [70]
Multi-qubit gate error rate	0.02 <sup>1</sup> [71]
Measurement error rate	0.05 <sup>1</sup> [72]
Number of memory qubits (per QNIC)	100

## 6.1 Performing simple tomography with different numbers of measurements

Performing quantum state tomography with more measurements typically results in a greater accuracy of the reconstructed density matrix. Although the required accuracy may depend on the demanded precision of a particular application, it is unlikely to perform an extremely large set of measurements solely for the reconstruction for purposes such as real time quantum channel monitoring. We first investigate how the number of measurements for the tomography,  $N_M$ , impacts the accuracy of the *reconstructed fidelity*,  $F_r$ , between adjacent nodes. Here,  $F_r = \text{Tr}[\rho_r \rho_i]$ , where  $\rho_i$  is the ideal density matrix of a Bell pair  $|\Phi^+\rangle$  and  $\rho_r$  is the density matrix reconstructed through the RuleSet-based quantum link-level tomography. This particular RuleSet consists of a single Rule composed of two Clauses and an Action. The first Clause, *MeasurementConditionClause*, tracks how many measurements have been performed. The second Clause, *ResourceConditionClause*, checks the availability of resources (see Algorithm 1 in section 4.3). The Action performs the measurement for the link-level tomography (see Algorithm 5 in section 4.3). Because each node picks a measurement basis at random from  $\{X, Y, Z\}$ , there are 9 detector settings in total used for the process. In this case, the Termination Condition also tracks the number of performed measurements, and stops the execution when the requirement is fulfilled.

<sup>1</sup>Including X, Y and Z error.

<sup>2</sup>Equivalent to 0.2dB/km.

<sup>3</sup>Memory lifetime can be, for example, up to the order to seconds [65] depending on the system. We use an artificial value of 50ms for the simulation. The ratio of excitation and relaxation probability is set to 100:1.

<sup>4</sup>Photon emission probability (memory-to-fiber) is the product of the emission probability into the zero phonon line and the collection efficiency.

The simulated result of a repeater network composed of two nodes based on a MeetInTheMiddle link, each 10km from the BSA node ( $L = 20\text{km}$ ), is shown in Fig. 6.1. Error bars represent one standard deviation uncertainty ( $\sigma$ ) from the average  $F_r$ . Max  $F_r$  and Min  $F_r$  are the highest and lowest fidelity observed within the 25 trials. Purple data points marked with cross between the minimum and the maximum are individual simulation results.

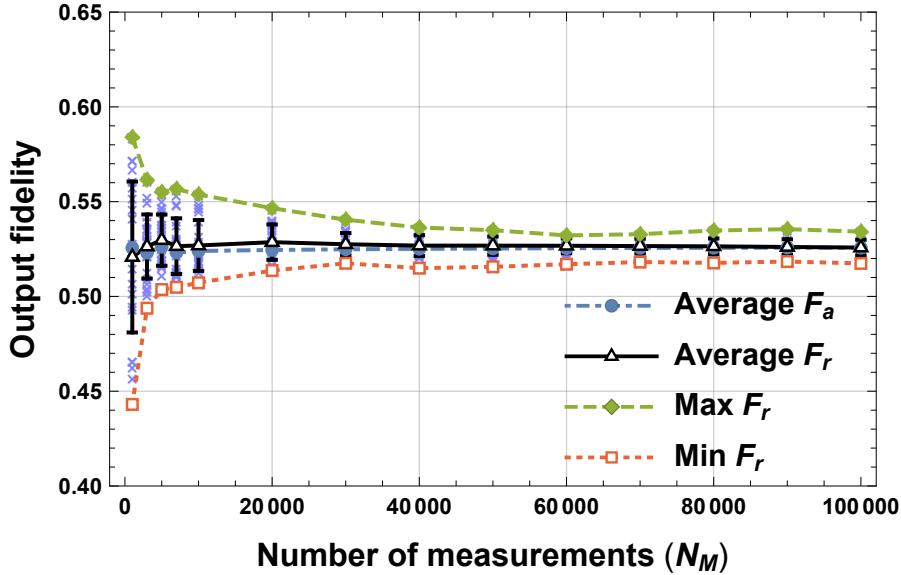


Figure 6.1: Impact of the number of measurements on the reconstructed fidelity error over the MeetInTheMiddle link.  $F_r$  is the reconstructed fidelity and  $F_a$  is the actual fidelity.

As in Fig. 6.1, the reconstructed fidelity has poor accuracy with only 1,000 measurements, and slightly underestimates the value simply due to the lack of samples –  $\sigma \approx 0.040$ . In order to achieve  $\sigma < 0.015$ , at least 7,000 measurement shots are required. A total of 20,000 measurements is capable of reconstructing the fidelity with a standard deviation less than 1%, and roughly converges to  $\sigma \approx 0.004$  when  $N_M > 60,000$ .

$F_a$  is the actual fidelity, accessible to us because this is a simulation, but of course hidden in the real world. The change in the accuracy of the reconstructed fidelity compared to the actual fidelity is shown in Fig. 6.2. The average difference between the reconstructed fidelity and the actual fidelity for a given  $N_M$  is

$$\overline{F}_{|r-a|} = \frac{1}{N} \sum_{i=1}^N |F_{r(i)} - F_{a(i)}|, \quad (6.1)$$

where  $N = 25$  is the total number of simulation trials.

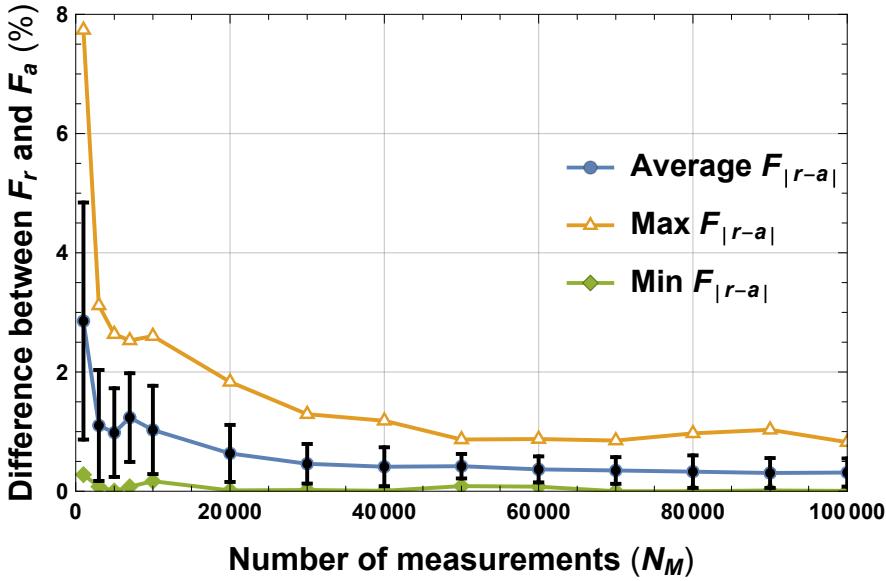


Figure 6.2: Change in the accuracy of the reconstructed fidelity relative to the actual fidelity.

The fidelity reconstruction process using only 1,000 measurement outcomes will result in about 3% error from the actual value on average. The accuracy improvement converges to  $\Delta F_{|r-a|} > 0.3\%$  when  $N_M \geq 40,000$ , in which case not much benefit can be gained from solely increasing the number of measurements.

The MeetInTheMiddle link and the SenderReceiver link have similar behavior regarding the fidelity, with sufficiently long memory lifetime relative to the idle time caused by classical latencies. Nevertheless, even over a single hop, the throughput strongly depends on the physical link architecture and the adopted data link protocol. Accumulating more measurements will increase the tomography time linearly, for both links, as the operation here consists only of measuring entangled qubits sequentially. The entanglement generation rate over the MeetInTheMiddle link is twice of what the SenderReceiver link achieves, simply because the BSA in the MeetInTheMiddle sits half-way through the link. With the given parameter settings, architecture and protocols, the MeetInTheMiddle link's Bell pair generation rate is around 6741/s (3368/s for the SenderReceiver link), when qubits are immediately measured when the system recognizes that they are entangled.

## 6.2 Performing tomography with single shot purification over different distances

Lengthening the channel between nodes increases the chance for optical qubits and memory qubits to experience errors. One way to tolerate such errors is to perform quantum purification.

The simplest purification scheme is the Single selection - Single error purification (Ss-Sp) [3]. As in Fig. 6.3, the Ss-Sp requires two Bell pairs along the channel to be purified – one Bell pair is consumed to detect the presence of an X error (or Z error) on the other Bell pair. The resource is only purified successfully when the measurement outcomes of the

consumed qubits (qubit C and qubit D in Fig. 6.3) coincide. Otherwise, both nodes discard their resources.

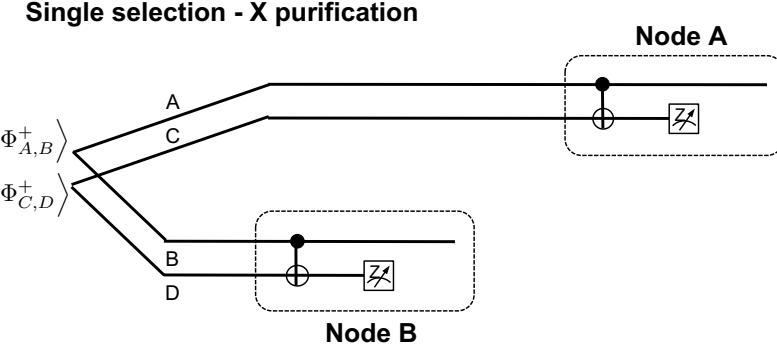


Figure 6.3: Single selection - Single error (X) purification (Ss-Sp). Consumes a single Bell pair  $|\Psi_{C,D}^+\rangle$  to detect the X error on  $|\Psi_{A,B}^+\rangle$ .

By extending Ss-Sp, we can also try to purify both errors (see Fig. 6.4). The Single selection - Double error purification (Ss-Dp) is similar to Ss-Sp, but requires another Bell pair to detect the Z error on the resource. However, because we perform the Z error purification after the X, the X error can propagate to the purified resource from the consumed Bell pair. In Fig. 6.3, the X error from qubit E and qubit F will propagate to qubit A and qubit B through the CNOT gate accordingly.

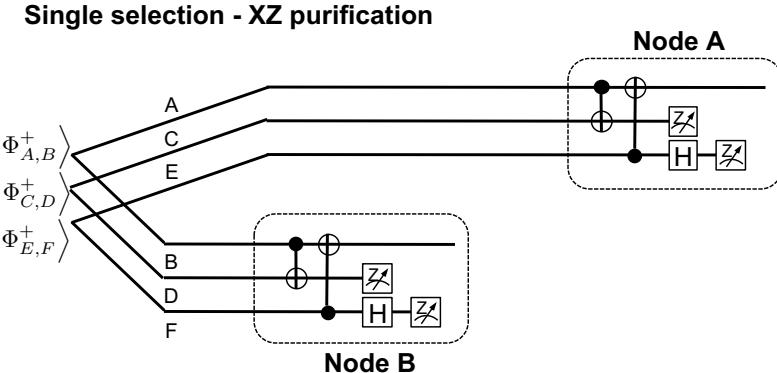


Figure 6.4: Single selection - Double error (XZ) purification (Ss-Dp). Consumes two Bell pairs,  $|\Psi_{C,D}^+\rangle$  and  $|\Psi_{E,F}^+\rangle$  to detect X error and Z error accordingly.

The double selection method introduced by Fujii and Yamamoto in 2009 [55] is a purification method utilizing an additional resource to improve the accuracy of post-selection by applying double verification. The Double selection - Single error purification (Ds-Sp) does the same job as Ss-Sp, but with an additional resource to detect the presence of a Z error on the consumed resource used for the X error purification (see Fig. 6.5). This minimizes the error propagation to the purified resource. Notice that any Z error that is originally present on the purified resource will not be detected.

**Double selection - X purification**

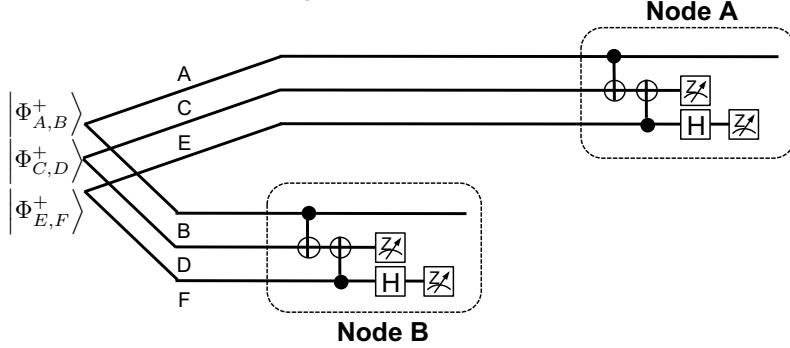


Figure 6.5: Double selection - Single error (X) purification (Ds-Sp). Consumes a single Bell pair  $|\Psi_{C,D}^+\rangle$  to detect X error on  $|\Psi_{A,B}^+\rangle$ , and another Bell pair  $|\Psi_{E,F}^+\rangle$  to avoid the Z error propagation from  $|\Psi_{C,D}^+\rangle$  to  $|\Psi_{A,B}^+\rangle$ .

The double selection scheme can also be applied to the double error purification circuit (Ds-Dp). The concept of this circuit is similar to Ds-Sp, except we perform the double verification process on both X and Z purification to the resource we keep. Thus, we can also minimize the error propagations that happen during the purification. For the circuit diagram, see Fig. 6.6.

**Double selection - XZ purification**

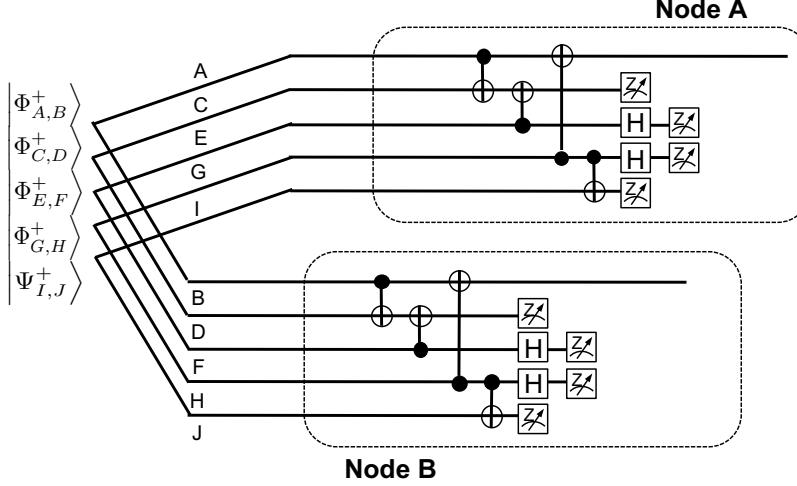


Figure 6.6: Double selection - Double error (XZ) purification (Ds-Dp). Consumes two Bell pairs,  $|\Psi_{C,D}^+\rangle$  and  $|\Psi_{G,H}^+\rangle$ , to detect X error and Z error on  $|\Psi_{A,B}^+\rangle$ . Each consumed Bell pair will also be verified through purification using  $|\Psi_{E,F}^+\rangle$  and  $|\Psi_{I,J}^+\rangle$  accordingly.

In this subsection, we describe simulated link-level tomography with 7000 measurements, with and without Rules that perform one of the purification methods above beforehand, over different distances. The simulated results of the fidelity reconstruction is shown in Fig. 6.7, and Fig. 6.8. The fidelity for SenderReceiver link is slightly worse than that of the MeetInTheMiddle link, but the overall behavior is approximately the same.

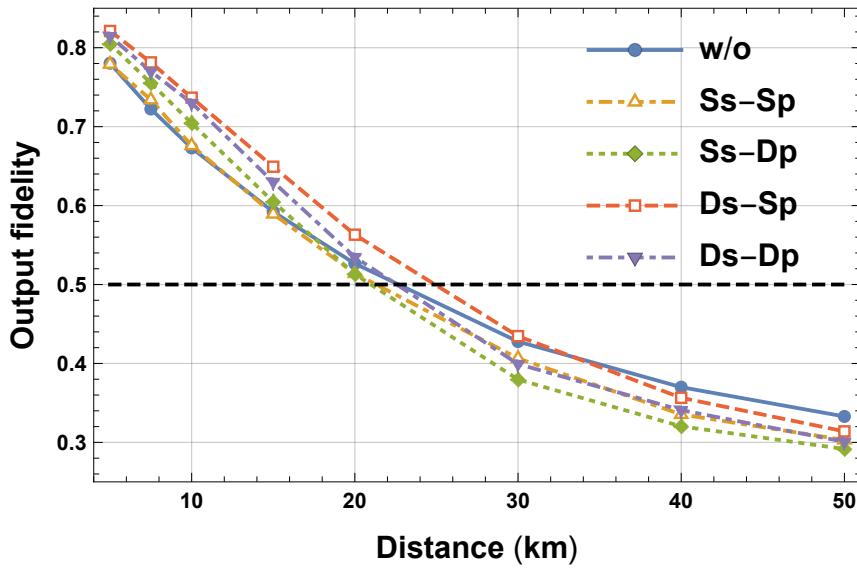


Figure 6.7: Impact of the MeetInTheMiddle channel length on the reconstructed fidelity with 7000 measurement outcomes w/o, with purification. The steep drop in the fidelity is due to the high error rate of 1% each X, Y and Z errors per km in the quantum channel chosen for this simulation.

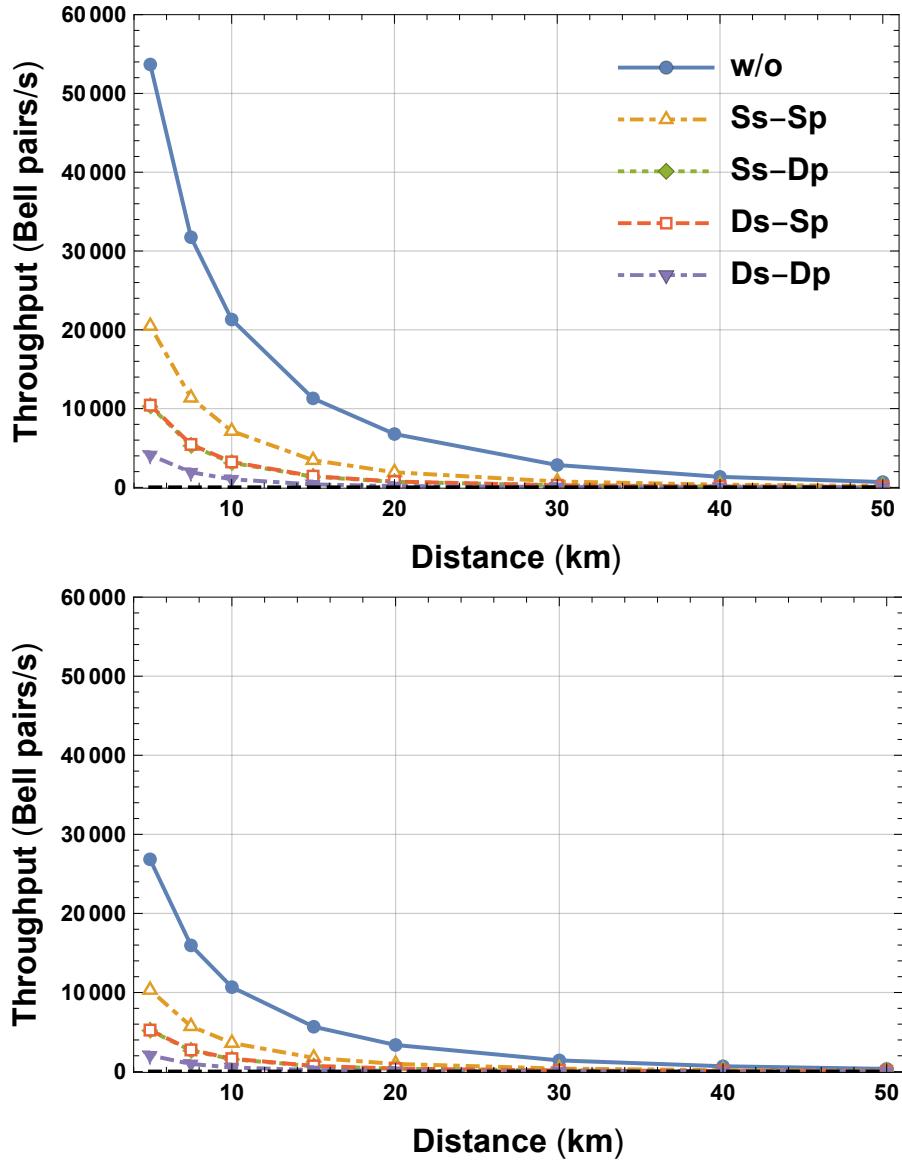


Figure 6.8: Impact of the channel distance on the channel throughput with and without performing purification. The throughput has been calculated as  $N_M/T$ , where  $T$  is the tomography process time to complete  $N_M$  measurements. (a) Throughput over the MeetInTheMiddle link. (b) Throughput over the SenderReceiver link.

As shown by the results in Fig. 6.7, performing Ss-Sp once slightly improves the fidelity compared to the case without, only over relatively shorter distances, but with a significant penalty on the throughput for both link architectures (see Fig. 6.8). With a total distance  $L = 10\text{km}$ , Ss-Sp suppresses about 5.6% of the X errors and 5.4% of Y errors, but the Z error rate increases approximately 8.8% due to the error propagation through the purification circuit. Overall, the fidelity improves roughly by 3%. While the fidelity did not change dramatically, the error distribution developed significantly different after performing the operation, which we can still take advantage of in the recurrence purification protocol (see Section 6.3). By implementing double selection to the same circuit (Ds-Sp), the output fi-

delity improved significantly – about 6.4% increase from the case without performing any purification. The second verification decreased the Z error rate by about 4.6% compared to the case of Ss-Sp with  $L = 10\text{km}$ . The Ss-Dp protocol purifies the X error first, and then the Z error afterwards. In this case, over  $L = 10\text{km}$  distance, the Z error rate decreases by 3.3% and the Y error rate by 5.7% – the X error rate increases by 4.3% due to propagation. Simply applying double selection to this (Ds-Dp) improved the fidelity by approximately 2.6% from Ss-Dp. Both purifications based on double selection are capable of purifying resources under longer distances, which is roughly 20km.

With the given hardware qualities in Table 6.1, Ds-Sp obtains the highest output fidelity. The output fidelity for Ds-Dp is lower than that of Ds-Sp, mainly because of larger  $KQ$  [73] with noisy gates, and the increase waiting time for resources, especially over long distances, where  $KQ$  is the product of the circuit depth ( $K$ ) and the number of qubits ( $Q$ ). Under this scenario, because Ds-Sp only requires three resources at once, the throughput is also higher than that of Ds-Dp.

### 6.3 Bootstrapping using Ruleset-based recurrence purification and tomography

Network operations, such as routing, often require knowledge regarding the links – e.g. fidelity, throughput, etc. Therefore, the channel characteristics to be shared need to be acquired beforehand via quantum link bootstrapping and characterization. In quantum networking, we commonly focus on optimizing the link fidelity for realizing a robust connection, where the goal may be achieved by performing nested purifications. The recurrence protocol [74], performs quantum purification on top of pre-purified resources to effectively improve the fidelity – the required number of resources for the operation, on the other hand, increases exponentially in the number of purification rounds  $N_p$ , which alone linearly affects the total idle time due to classical communication latencies.

In this subsection, we describe our simulation of quantum link bootstrapping to quantify the achievable link fidelity and its corresponding throughput, using RuleSet-supported tomography with recurrence purification based on one of the four circuits shown in subsection 6.2; recurrent single selection single error purification (RSs-Sp), recurrent single selection double error purification (RSs-Dp), recurrent double selection single error purification (RDs-Sp) and recurrent double selection double error purification (RDs-Dp). Each round of purification is a separate Rule in the same RuleSet. In the given example of a two round RSs-Sp circuit shown in Fig. 6.9, the first Rule performs X purifications to resources. In the second Rule, we use those purified resources to perform another Z error purification. Each round, we alternate the X purification and the Z purification to suppress the error propagation. For double error purifications, we alternate the XZ purification and the ZX purification. With probabilistic generation of Bell pairs and scarce memory qubits, operations can stall in the middle, in which case the Rule waits for new resources to arrive. Using RuleSets, the whole task is completed autonomously over a network. The simulation continues reconstructing the density matrix every round, and terminates the process when the same operation results in a lower reconstructed fidelity, or due to timeout. For practical reasons, each RuleSet timeout is set to two minutes.

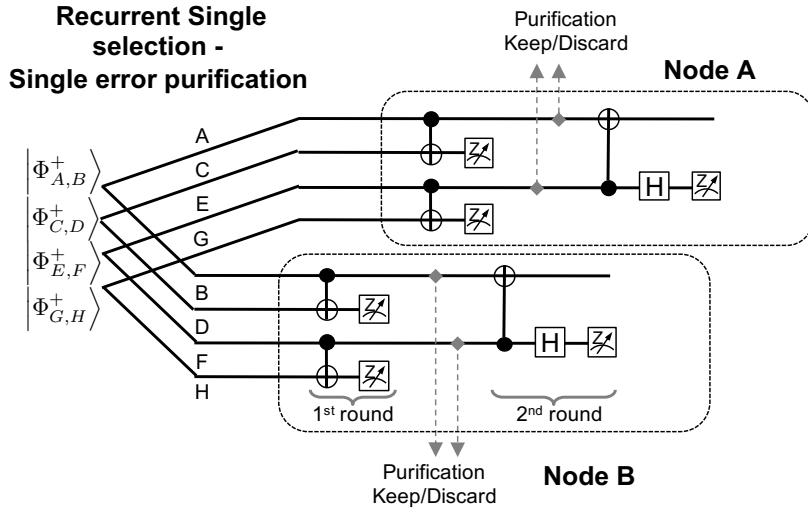


Figure 6.9: Recurrence purification protocol based on single selection X purification and Z purification. Each round of purification is a Rule in a Ruleset.

We assume perfect photon emissions from memories to fiber, where in the real world, similar effects may be observed by increasing the number of memory qubits. CNOT gates are also assumed to be ideal to solidify the real merit of recurrence protocols. We first focus on a repeater network with a relatively short channel, where the total length is set to 10km. Secondly, we perform similar simulations over a longer distance,  $L = 20km$ . For the MeetInTheMiddle link, each node is the same distance away from the BSA node. The flowchart of the protocol is provided below in Fig. 6.10. The stored fidelity and throughput by the selected recurrence purification method may be referred by other operations.

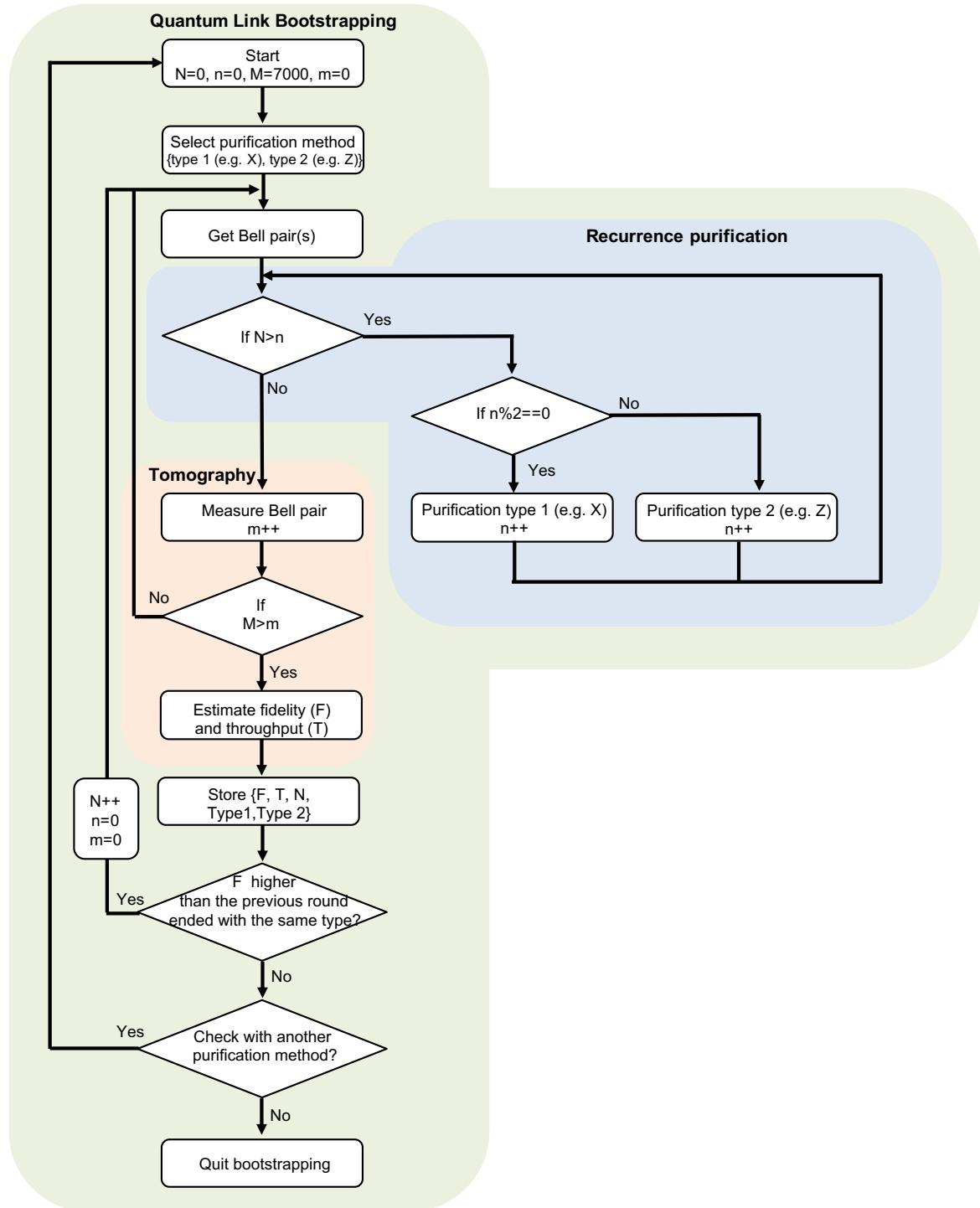


Figure 6.10: Flowchart of the quantum link bootstrapping. The termination process due to timeout is omitted from the chart. We need  $1 - F(\rho_{\text{reconstructed}}, \rho_{\text{real}}) \ll 1 - F(\rho_{\text{real}}, \rho_{\text{ideal}})$  in order to make high-quality decisions about next operations in real time.

## Closely spaced repeater nodes

We start by investigating how recurrent purification benefits over a link between closely located repeater nodes,  $L = 10\text{km}$ . The simulated result over the MeetInTheMiddle link is shown in Fig. 6.11. Notice that each protocol has a different number of purification rounds performed.

As in the figure, the given system generates resources with an average fidelity  $F_r \approx 0.675$ , but the RDs-Dp is capable of bringing up the quality up to  $F_r \approx 0.840$  through a two round purification, which is the fastest in terms of required rounds. The RSs-Sp, on the other hand, starts with the slightest fidelity improvement since applying X purification inherently results in a significant penalty to the Z error rate, as we discussed in section 6.2. While this looks operationally impractical, by alternating X and Z purification multiple rounds, we can purify resources while minimizing error propagations – this appears as the staircase-like curve in Fig. 6.11(a). RSs-Sp gradually improves the fidelity as we perform more purifications, and produces resources with optimal average fidelity,  $F_r \approx 0.865$  ( $F_r \approx 0.852$  for the SenderReceiver link) at  $N_p = 6$ . About 3.1% of the errors are X errors, 2.7% are Y errors and 3.4% are Z errors. 5.6% of the outputs consists of unfixable errors, which can mainly be avoided by improving the memory lifetime, shortening the distance between nodes or adjusting the algorithm and the goal. RDs-Sp is also capable of producing high fidelity resources,  $F_r \approx 0.860$ , with more rounds of purifications than RDs-Dp but fewer than RSs-Sp. The fidelity starts declining for all protocols in the end, due to critical waiting time of memories relative to its own lifetime. The reconstructed fidelity also gets overestimated, because the randomness of measuring unentangled qubits due to errors, stochastically contributes to the fidelity estimation.

The number of rounds does not necessarily determine how fast the protocol processes resources, especially with a limited number of qubits. As shown in Fig. 6.11, the RDs-Dp protocol over the MeetInTheMiddle link reaches its highest fidelity with a throughput of roughly 1565/s, completing 7000 purified Bell pair measurements in around 37.1 seconds. The RSs-Sp protocol produces 1106 resources per second with the local optimal fidelity, but also achieves similar fidelity as RDs-Dp,  $F_r \approx 0.848$ , at  $N_p = 4$ , which obtains a higher generation rate of 5923/s. The RDs-Sp protocol and the RSs-Dp protocol have a throughput of 816/s and 520/s respectively. Although both protocols consume the same amount of resources per purification, the RSs-Dp took longer in total because at each step, RSs-Dp faces higher error propagation probability, resulting in greater purification failure rate. As provided in Fig. 6.11(e) and Fig. 6.11(f), the throughput over the SenderReceiver link is approximately half of what the MeetInTheMiddle link achieves. In this case, RSs-Dp shows no advantage over RDs-Sp.

The quality of a link can also be increased by installing larger sets of memory qubits. As in Fig. 6.12, with 700 memory qubits, a six-round RSs-Sp can generate resources with  $F_r \approx 0.879$ . Compared to the case with only 100 qubits, the fidelity at  $N_p = 4$  also increases around by 0.6%. For  $N_p < 4$ , the fidelity stays unchanged because having 100 qubits is more than sufficient for a four-round RSs-Sp – the throughput will, however, improve. Notice that a linear increase in memory buffer size will not provide an effective solution to extend the performable rounds. Over a 10km MeetInTheMiddle link, owning 700 memory qubits is still not enough for an eight-round RSs-Sp.

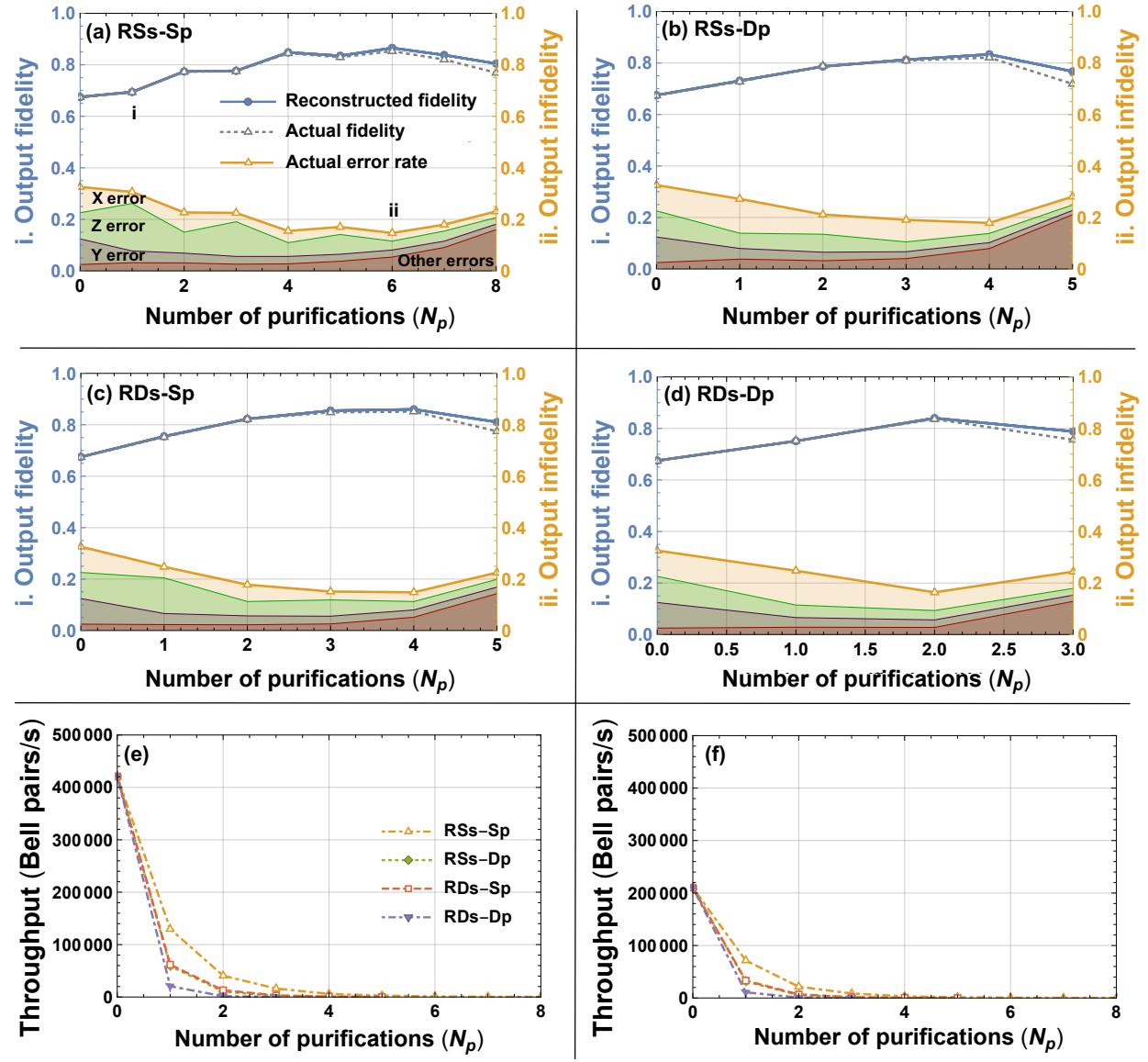


Figure 6.11: Reconstructed fidelity with actual fidelity and actual error rates, and estimated throughput of protocols over  $L = 10\text{km}$  MeetInTheMiddle link. "Other errors" include memory excitation/relaxation error, and completely mixed error due to photon detector dark counts. (a) Simulation result of the RSs-Sp protocol. (b) Simulation result of the RSs-Dp protocol. (c) Simulation result of the RDs-Sp protocol. (d) Simulation result of the RDs-Dp protocol. (e) Protocol throughputs over the MeetInTheMiddle link. (e) Protocol throughputs over the SenderReceiver link.

### Distantly spaced repeater nodes

Errors on qubits have more time to develop when two repeater nodes are located farther away. Hence, how well errors are detected at each round, especially with noisy inputs, becomes an important factor for an effective recurrence purification protocol. The simulation results over

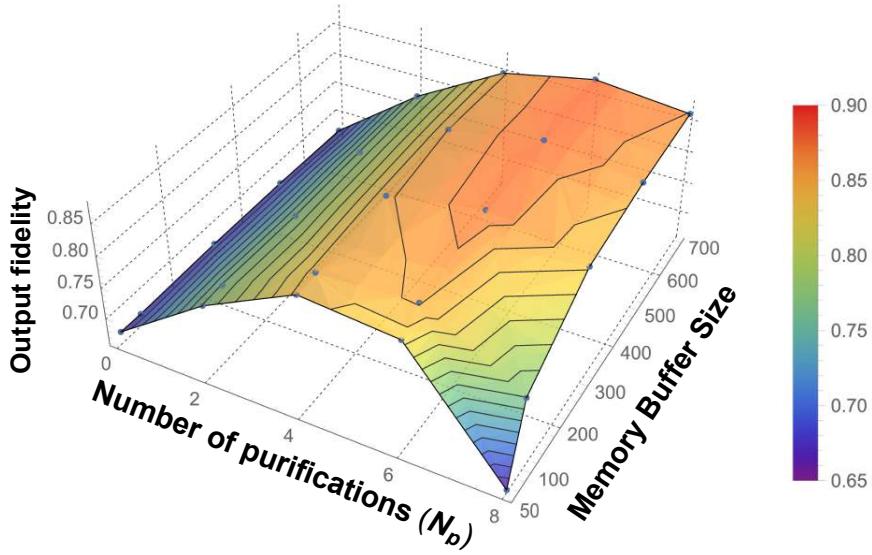


Figure 6.12: Impact of the memory buffer size to the maximum fidelity purified through the RSs-Sp protocol over a 10km MeetInTheMiddle link.

a repeater network with channel length  $L = 20km$  are is provided in Fig. 6.13.

Unlike the case of Fig. 6.11, neither RSs-Sp nor RSs-Dp improves the fidelity effectively, regardless of the number of purification rounds. As shown in the corresponding error distributions, performing purifications with the given input resources, in such a way done by RSs-Sp and RSs-Dp, ends up failing because the loss lead by error propagations and memory errors is greater than the gain. Resources used for RSs-Sp suffer more from memory errors as they go through more rounds of purifications, resulting in longer average idle time per resource. The other two protocols implemented with double selection, however, are capable of effectively purifying resources. The RDs-Sp protocol produces the highest fidelity among all, with an output value  $F_r \approx 0.667$  and a rate of 325/s (168/s with  $F_r \approx 0.651$  for the Sender-Receiver link) at  $N_p = 3$ , whereas RDs-Dp also produces similar quality,  $F_r \approx 0.637$ , or  $F_r \approx 0.626$  for the Sender-Receiver link, within two rounds. The corresponding throughput is 116/s for the MeetInTheMiddle link, and 60/s for the Sender-Receiver link.

Under this scenario, consuming another Bell pair via the double selection shows advantage in terms of fidelity. However, from the previous discussion, we know that RSs-Sp generates resources with higher fidelity and higher rate, given slightly better average input resources and sufficiently long-lived memories relative to its average idle time. Thus, for longer distances, we can also perform double selection-based purification at the beginning to raise the fidelity at a certain level, then change to single selection purification afterwards to maximize the fidelity. Below in Fig. 6.14 are the simulation results of two cases; *Case A* that switches to RSs-Sp after performing a single round of RDs-Sp, and *Case B* that switches to RSs-Sp after performing two rounds of RDs-Sp.

Whether this method is beneficial or not very much depends on the situation, and the switching timing as shown. Under this scenario, *Case A* produces higher fidelity,  $F_r = 0.680$ , with a higher rate of 396 Bell pairs per second, relative to the optimal case of RDs-Sp in Fig. 6.13. *Case B* results in worsening the fidelity to  $F_r = 0.666$ , and the throughput to 218/s. The total bootstrapping time for completing the RSs-Sp, RSs-Dp, RDs-Sp, RDs-Dp,

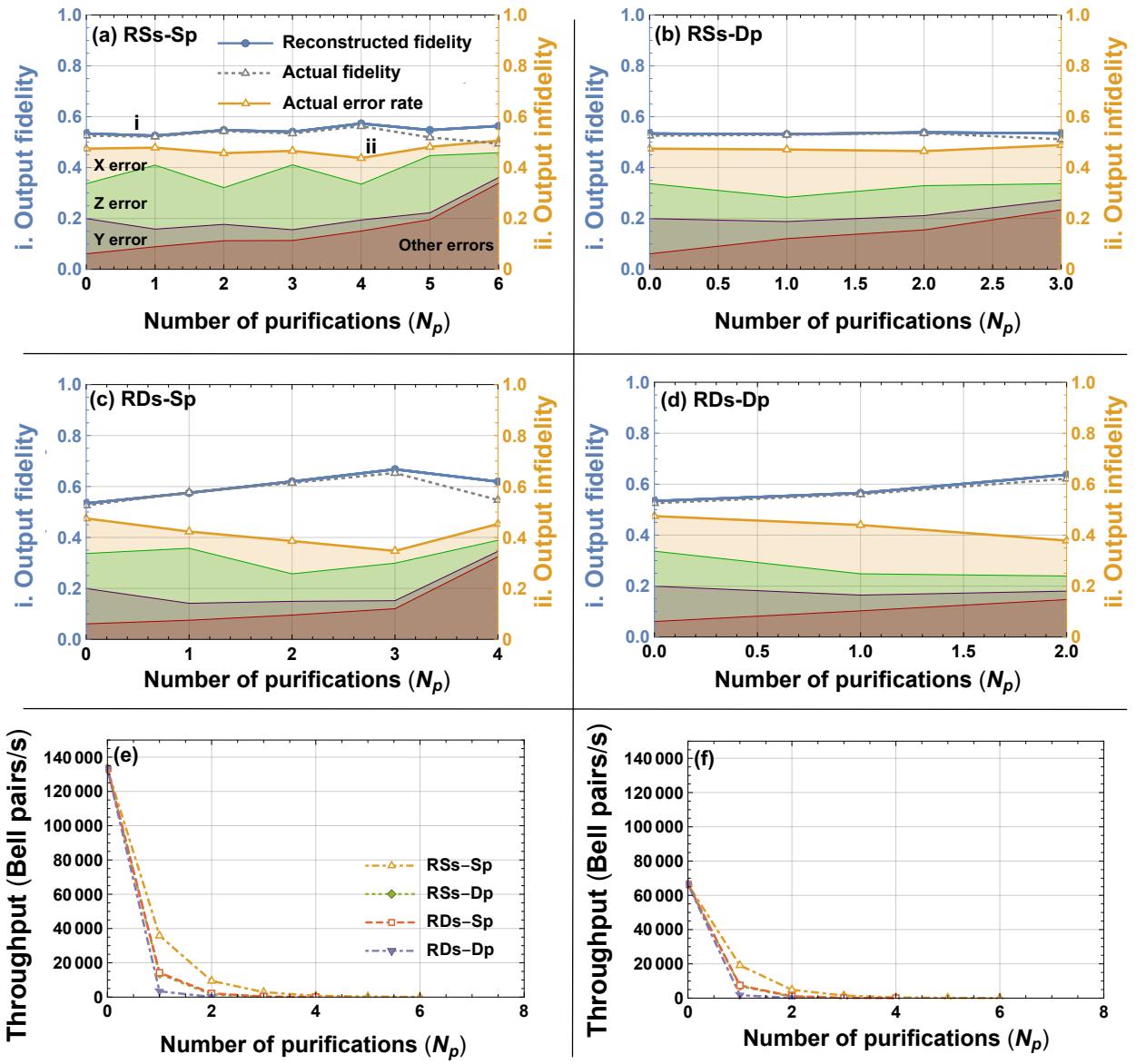


Figure 6.13: Reconstructed fidelity with actual fidelity and actual error rates, and estimated throughput of protocols over  $L = 20\text{km}$  MeetInTheMiddle link. "Other errors" include memory excitation/relaxation error, and completely mixed error due to photon detector dark counts. (a) Simulation result of the RSs-Sp protocol. (b) Simulation result of the RSs-Dp protocol. (c) Simulation result of the RDs-Sp protocol. (d) Simulation result of the RDs-Dp protocol. (e) Protocol throughputs over the MeetInTheMiddle link. (f) Protocol throughputs over the SenderReceiver link.

Case A and Case B with a single trial was 561 seconds, which is approximately 9.4 minutes.

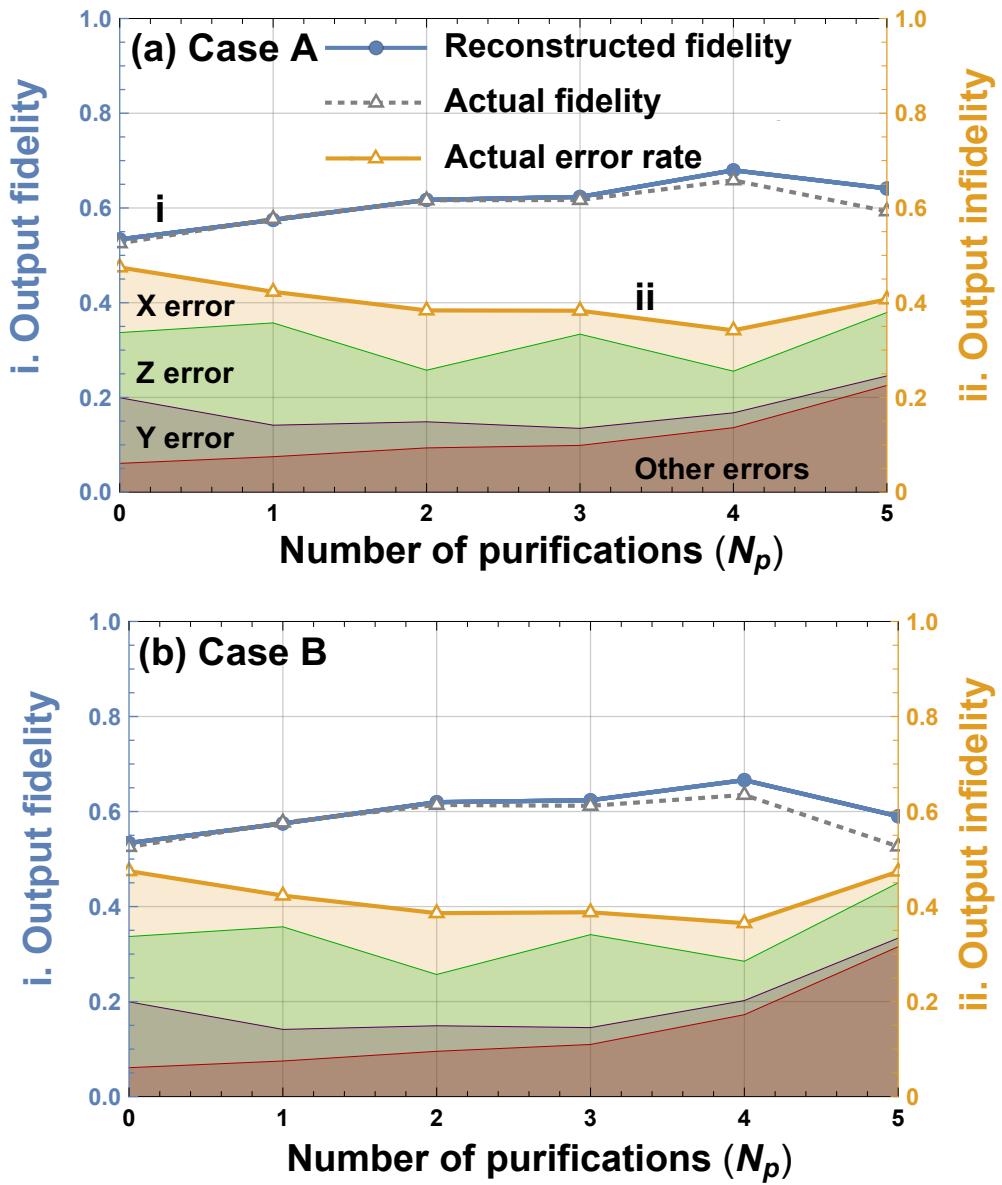


Figure 6.14: Switching RDs-Sp to RSs-Sp over  $L = 20\text{km}$ . a) Pattern 1 consists of one round RDs-Sp and four rounds of RSs-Sp. Purification methodology switches at  $N_p = 1$ . b) Pattern 2 consists of two rounds RDs-Sp and three rounds of RSs-Sp. Purification methodology switches at  $N_p = 2$ .

# Chapter 7

## Conclusion

This thesis has introduced the RuleSet-based communication protocol, which is used to autonomously support coordinated decision making in quantum operations over a network. The RuleSet architecture is not only useful for the coordination, but also allows us to dynamically generate suitable operations based on link qualities, dynamically assign resources and manage network processes based on distinguishable connections. The technology is adapted to conduct quantum link bootstrapping in a network simulation. In total, four different recurrent purification schemes were simulated with the goal to quantifying the achievable link fidelity with its corresponding throughput via link-level tomography, using Markov-Chain Monte-Carlo simulation with a noisy quantum repeater system. Such information may be distributed across the network, and used for different purposes such as quantum routing.

The RSs-Sp requires more purification rounds to optimize the fidelity compared to other schemes, which results in a longer average idle time for purified resources. Given 100 link memory qubits each with a 50ms lifetime, we found that RSs-Sp is still stronger than other simulated schemes in terms of the achievable fidelity and the throughput, when the distance between nodes is kept short ( $L \approx 10\text{km}$ ). In contrast, because the system is noisier with longer channels ( $L \approx 20\text{km}$ ), RSs-Sp alone may be incapable of purifying the link, as errors can evolve faster than the purification, in which case schemes such as the double selection becomes more preferable. The simulations show that the bootstrapping process, therefore, must be able to search and choose the most effective purification method for that specific channel. Changing from double selection to single selection in the middle indeed pushed the link limit, but finding the optimal changeover timing may add an additional complexity to the process. Other methods that detect errors with a higher probability requiring fewer rounds, such as RDs-Dp, may be more advantageous when a larger set of memory qubits is available for each link, relative to its demand.

While this work focuses on link fidelity optimization, connecting source to destination multi-hops away also requires an adequately high generation rate, or possibly a synchronized generation timing of Bell pairs. Such an optimization problem of the fidelity-throughput tradeoff remains an open question.

### Future work

A simulation of multi-hop quantum communication remains a future work. Although not finished, OMNeT++ is capable of simulating a multi-hop quantum communication, and much preparatory work has been done for that. Optimizing a link between adjacent nodes

through bootstrapping, and optimizing multiple links for connecting non-adjacent nodes are two different tasks. An interesting topic would be to investigate the minimum throughput required for connecting  $n$  resources. Discussion regarding the quantification of costs for quantum channels, required for routing, also remains an important open question. The current version of the RuleSet-based communication protocol is also not flexible enough to accomplish a particular set of tasks, such as performing purification through entanglement pumping - the current RuleSet does not have a shared resource list. Improving the protocol also remains an important future work.

# Acknowledgements

First and foremost, I would like to express my deepest gratitude to Professor Rodney Van Meter at Keio University for his long term support through out my Bachelor's and Master's program. I truly enjoyed working with you on research projects. I have had countless amazing opportunities at the AQUA lab to learn, which I could have never gained without you. When I joined AQUA, I never imagined myself presenting my own research in conferences, and publishing them as a paper. It is, without doubt, the most valuable and irreplaceable experiences in my life.

I especially thank professor Jun Murai at Keio University for his kindness and for providing me all kinds of opportunities. Working as your SA/TA was tough, but I am glad that I was capable of working with you throughout my university life. I was lucky to have a lot of time to spend with you periodically inside and outside the campus.

I also acknowledge Project Research Associate Takahiko Satoh at Keio University for being always a supportive supervisor, and spending me a large fraction of your time without hesitation to share your knowledge. I enjoyed talking with you, and I am fortunate to have you in the AQUA lab throughout my master's program.

I would also like to thank all the members of the AQUA lab. Shin Nishio, Ryosuke Sato, Yasuhiro Okura, Shigetora Miyashita, Nozomi Tanetani, Sayyor Yusupov and Makoto Nakai, which I enjoyed frequently hanging out (bouldering, karaoke and eating) outside the campus. I'm glad that I had you all. I also thank all the new comers, Shuta Kobayashi, Liu Sitong, Yakabe Mahiro, Shu Toraketu and Nobuhide Hiranuma. I can already imagine you all producing great research results in the future!!!

I also thank to those that used to be a member of the AQUA lab. First, I have to thank Shota Nagayama for helping me out throughout my Bachelor's program, and for working with me as a SA/TA in many classes. I shall thank Takafumi Oka, who is a good friend of mine. You are very kind for always picking a place somewhere in between where I live and where you live, when we hang out! I also thank Shinnosuke Ozawa for whom I enjoyed working in the AQUA lab for a long time. I'm glad that I had you as a *kohai* in AQUA.

I am grateful to all members in Murai, Kusumoto, Nakamura, Takashio, Uehara, Van Meter, Mitsugi, Nakazawa Laboratories for all the supports.

I must thank Nazmi Tanaka and Yuji Yamada for always being one of my best friends for more than 13 years. I am grateful to have someone like you, and have never stopped enjoying to play games, watch movies, and do all other stuffs together. I must also thank Kaho Yogi for always being closest to me. Let's stay fit, and try to live together as long as possible! I also thank Kennedy Ho, who always comes to Shonandai all the way from Tokyo just to hang out with me. I truly enjoyed (and I'm still enjoying) spending time with you in the weekends, but please stop complaining how I never go out to Tokyo to meet you. I did

once!!! Good for you I'm moving out from Shonandai! I thank Peter Tien for frequently sending me messages, and sometimes advertising me your own company for no reason at all. For about 3 years, I have been struggling to throw away the junk (or present, I shall say) you gave me for birthday. Allow me to throw it away, and stop saying it is the symbol of our friendship. I thank the *AES family members*, Tomas Svitil, Petr Svitil, Jeremy Shih, Karanvir Chauhan, Sangwon Lee and Joshua Woods. Lucky me to have you all to chit chat, and to spend time outside (and sometimes inside) Japan!

# Bibliography

- [1] Hubert Zimmermann. OSI reference model—the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications*, 28(4):425–432, 1980.
- [2] Takaaki Matsuo, Clément Durand, and Rodney Van Meter. Quantum link bootstrapping using a ruleset-based communication protocol. *arXiv e-prints*, page arXiv:1904.08605, Apr 2019.
- [3] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller. Quantum repeaters: The role of imperfect local operations in quantum communication. *Phys. Rev. Lett.*, 81:5932–5935, Dec 1998.
- [4] H. J. Kimble. The quantum internet. *Nature*, 453:1023 EP –, 06 2008.
- [5] Rodney Van Meter. *Quantum Networking*. Wiley-IEEE Press, 1st edition, 2014.
- [6] Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412), 2018.
- [7] Wojciech Kozlowski. Architectural Principles for a Quantum Internet. Internet-Draft draft-irtf-qirg-principles-00, Internet Engineering Task Force, March 2019. Work in Progress. Expires: September 10, 2019.
- [8] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7 – 11, 2014. Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84.
- [9] Charles H. Bennett, Gilles Brassard, and N. David Mermin. Quantum cryptography without Bell’s theorem. *Phys. Rev. Lett.*, 68:557–559, Feb 1992.
- [10] Artur K. Ekert. Quantum cryptography based on Bell’s theorem. *Phys. Rev. Lett.*, 67:661–663, Aug 1991.
- [11] Masahiro Takeoka, Saikat Guha, and Mark M. Wilde. Fundamental rate-loss tradeoff for optical quantum key distribution. *Nature Communications*, 5:5235 EP –, 10 2014.
- [12] J. S. Bell. On the einstein podolsky rosen paradox. *Physics Physique Fizika*, 1:195–200, Nov 1964.

- [13] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Measurement-based and universal blind quantum computation. In Alessandro Aldini, Marco Bernardo, Alessandra Di Pierro, and Herbert Wiklicky, editors, *Formal Methods for Quantitative Aspects of Programming Languages: 10th International School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM 2010, Bertinoro, Italy, June 21-26, 2010, Advanced Lectures*, pages 43–86. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
- [14] Chia-Hung Chien, Rodney Van Meter, and Sy-Yen Kuo. Fault-tolerant operations for universal blind quantum computation. *J. Emerg. Technol. Comput. Syst.*, 12(1):9:1–9:26, August 2015.
- [15] Richard Jozsa, Daniel S. Abrams, Jonathan P. Dowling, and Colin P. Williams. Quantum clock synchronization based on shared prior entanglement. *Phys. Rev. Lett.*, 85:2010–2013, Aug 2000.
- [16] Isaac L. Chuang. Quantum algorithm for distributed clock synchronization. *Phys. Rev. Lett.*, 85:2006–2009, Aug 2000.
- [17] Richard Feynman. Simulating physics with computers. *SIAM journal on Computing*, Vol. 26, pp. 1484–1509, 1982.
- [18] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 400(1818):97–117, 1985.
- [19] C. H. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, Vol. 11, pp. 175–179, 1984.
- [20] C. H. Bennett and G. Brassard. The dawn of a new era for quantum cryptography: The experimental prototype is working! *Sigact News*, Vol. 20, no. 4, pp. 78–82, 1989.
- [21] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Vol. 35, pp. 124–134, 1994.
- [22] Charles H. Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K. Wootters. Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Phys. Rev. Lett.*, 70:1895–1899, Mar 1993.
- [23] Dik Bouwmeester, Jian-Wei Pan, Klaus Mattle, Manfred Eibl, Harald Weinfurter, and Zeilinger Anton. Experimental quantum teleportation. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 356(1743):1733–1737, 1998.
- [24] A. Furusawa, J. L. Sørensen, S. L. Braunstein, C. A. Fuchs, H. J. Kimble, and E. S. Polzik. Unconditional quantum teleportation. *Science*, 282(5389):706–709, 1998.

- [25] W. Dür, H.-J. Briegel, J. I. Cirac, and P. Zoller. Quantum repeaters based on entanglement purification. *Phys. Rev. A*, 59:169–181, Jan 1999.
- [26] M. A. Horne M. Zukowski, A. Zeilinger and A. K. Ekert. Event-ready-detectors Bell experiment via entanglement swapping. *Phys. Rev. Lett.* 71, 4298, 1993.
- [27] J. B. Altepeter, E. R. Jeffrey, and P. G. Kwiat. Photonic State Tomography. *Advances in Atomic Molecular and Optical Physics*, 52:105–159, 2005.
- [28] Wikipedia. Loopholes in bell test experiments, Jul 2019.
- [29] Sreraman Muralidharan, Linshu Li, Jungsang Kim, Norbert Lütkenhaus, Mikhail D. Lukin, and Liang Jiang. Optimal architectures for long distance quantum communication. *Scientific Reports*, 6:20463 EP –, 02 2016.
- [30] L. Hartmann, B. Kraus, H.-J. Briegel, and W. Dür. On the role of memory errors in quantum repeaters. *Phys. Rev. A*, 75:032310, 2007.
- [31] Liang Jiang, J. M. Taylor, Kae Nemoto, W. J. Munro, Rodney Van Meter, and M. D. Lukin. Quantum repeater with encoding. *Phys. Rev. A*, 79:032325, Mar 2009.
- [32] Austin G. Fowler, David S. Wang, Charles D. Hill, Thaddeus D. Ladd, Rodney Van Meter, and Lloyd C. L. Hollenberg. Surface code quantum communication. *Phys. Rev. Lett.*, 104:180503, May 2010.
- [33] Ying Li, Sean D Barrett, Thomas M Stace, and Simon C Benjamin. Long range failure-tolerant entanglement distribution. *New Journal of Physics*, 15(2):023012, 2013.
- [34] Richard Jozsa. Fidelity for mixed quantum states. *Journal of Modern Optics*, vol. 41, Issue 12, p.2315-2323, 1994.
- [35] Daniel E. Browne Robert Raussendorf and Hans.J. Briegel. Measurement-based quantum computation with cluster states. *Phys. Rev. A*, Vol. 68, 2003.
- [36] Xinlan Zhou, Debbie W. Leung, and Isaac L. Chuang. Methodology for quantum logic gate constructions. *Phys. Rev. A*, 62,052316, 2000.
- [37] R. Hanson, F. M. Mendoza, R. J. Epstein, and D. D. Awschalom. Polarization and readout of coupled single spins in diamond. *Phys. Rev. Lett.*, 97:087601, Aug 2006.
- [38] D Leibfried, R Blatt, C Monroe, and D Wineland. Quantum dynamics of single trapped ions. *Reviews of Modern Physics*, 75, 05 2003.
- [39] R. Hanson, L. P. Kouwenhoven, J. R. Petta, S. Tarucha, and L. M. K. Vandersypen. Spins in few-electron quantum dots. *Rev. Mod. Phys.*, 79:1217–1265, Oct 2007.
- [40] J. M. Elzerman, R. Hanson, L. H. Willems van Beveren, B. Witkamp, L. M. K. Vandersypen, and L. P. Kouwenhoven. Single-shot read-out of an individual electron spin in a quantum dot. *Nature*, 430(6998):431–435, 2004.

- [41] Hou Shun Poh, Siddarth K. Joshi, Alessandro Cerè, Adán Cabello, and Christian Kurtsiefer. Approaching Tsirelson's bound in a photon pair experiment. *Phys. Rev. Lett.*, 115:180408, Oct 2015.
- [42] B. Hensen, H. Bernien, A. E. Dréau, A. Reiserer, N. Kalb, M. S. Blok, J. Ruitenberg, R. F. L. Vermeulen, R. N. Schouten, C. Abellán, W. Amaya, V. Pruneri, M. W. Mitchell, M. Markham, D. J. Twitchen, D. Elkouss, S. Wehner, T. H. Taminiau, and R. Hanson. Loophole-free Bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526:682–686, 2015.
- [43] N. Brunner, D. Cavalcanti, S. Pironio, V. Scarani, and S. Wehner. Bell nonlocality. *Reviews of Modern Physics*, 86:419–478, April 2014.
- [44] Anupam Garg and N. D. Mermin. Detector inefficiencies in the Einstein-Podolsky-Rosen experiment. *Phys. Rev. D*, 35:3831–3835, Jun 1987.
- [45] Philippe H. Eberhard. Background level and counter efficiencies required for a loophole-free einstein-podolsky-rosen experiment. *Phys. Rev. A*, 47:R747–R750, Feb 1993.
- [46] J. Barrett, D. Collins, L. Hardy, A. Kent, and S. Popescu. Quantum nonlocality, Bell inequalities, and the memory loophole. *Phys. Rev. A*, 66(4):042111, Oct 2002.
- [47] Wikipedia. Bell test experiments, Jul 2017.
- [48] R. Stockill, M. J. Stanley, L. Huthmacher, E. Clarke, M. Hugues, A. J. Miller, C. Matthiesen, C. Le Gall, and M. Atatüre. Phase-tuned entangled state generation between distant spin qubits. *ArXiv e-prints*, February 2017.
- [49] Juan Yin, Yuan Cao, Yu-Huai Li, Sheng-Kai Liao, Liang Zhang, Ji-Gang Ren, Wen-Qi Cai, Wei-Yue Liu, Bo Li, Hui Dai, Guang-Bing Li, Qi-Ming Lu, Yun-Hong Gong, Yu Xu, Shuang-Lin Li, Feng-Zhi Li, Ya-Yun Yin, Zi-Qing Jiang, Ming Li, Jian-Jun Jia, Ge Ren, Dong He, Yi-Lin Zhou, Xiao-Xiang Zhang, Na Wang, Xiang Chang, Zhen-Cai Zhu, Nai-Le Liu, Yu-Ao Chen, Chao-Yang Lu, Rong Shu, Cheng-Zhi Peng, Jian-Yu Wang, and Jian-Wei Pan. Satellite-based entanglement distribution over 1200 kilometers. *Science*, Vol.356, Issue 6343, pp. 1140-114, 2017.
- [50] C. Jones, D. Kim, M. T. Rakher, P. G. Kwiat, and T. D. Ladd. Design and analysis of communication protocols for quantum repeater networks. *New Journal of Physics*, 18(8):083015, August 2016.
- [51] Axel Dahlberg, Matthew Skrzypczyk, Tim Coopmans, Leon Wubben, Filip Rozpędek, Matteo Pompili, Arian Stolk, Przemysław Pawełczak, Robert Knegjens, Julio de Oliveira Filho, Ronald Hanson, and Stephanie Wehner. A Link Layer Protocol for Quantum Networks. *arXiv e-prints*, page arXiv:1903.09778, Mar 2019.
- [52] Oka Takafumi. Optimized Fidelity Estimation in Purification for the Fastest Bootstrap of a Quantum Link, Keio University, Bachelor's thesis, 2017.

- [53] Rodney Van Meter and Takaaki Matsuo. Connection Setup in a Quantum Network. Internet-Draft `draft-van-meter-qirg-quantum-connection-setup-00`, Internet Engineering Task Force, March 2019. Work in Progress. Expires: September 12, 2019.
- [54] Luciano Aparicio and Rodney Van Meter. Multiplexing schemes for quantum repeater networks. *Proceedings of SPIE - The International Society for Optical Engineering*, 09 2011.
- [55] Keisuke Fujii and Katsuji Yamamoto. Entanglement purification with double selection. *Phys. Rev. A*, 80(4):042308, Oct 2009.
- [56] András Varga and Rudolf Hornig. An overview of the OMNeT++ simulation environment. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
- [57] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. <http://eigen.tuxfamily.org>, 2010.
- [58] Rodney Van Meter, Takahiko Satoh, Thaddeus D. Ladd, William J. Munro, and Kae Nemoto. Path selection for quantum repeater networks. *Networking Science*, 3(1):82–95, Dec 2013.
- [59] L. Kleinrock. *Queueing systems. volume I, Theory*. New York: Wiley, 1974.
- [60] Filip Rozpędek, Raja Yehia, Kenneth Goodenough, Maximilian Ruf, Peter C. Humphreys, Ronald Hanson, Stephanie Wehner, and David Elkouss. Near-term quantum repeater experiments with NV centers: overcoming the limitations of direct transmission. *arXiv e-prints*, page arXiv:1809.00364, Sep 2018.
- [61] Rodney Van Meter, Takaaki Matsuo, Clement Durand, and Takahiko Satoh. Markov error models for scalable quantum internet simulation. In preparation.
- [62] Takahiko Satoh, François Le Gall, and Hiroshi Imai. Quantum network coding for quantum repeaters. *Phys. Rev. A*, 86:032331, Sep 2012.
- [63] Takaaki Matsuo, Takahiko Satoh, Shota Nagayama, and Rodney Van Meter. Analysis of measurement-based quantum network coding over repeater networks under noisy conditions. *Phys. Rev. A*, 97:062328, Jun 2018.
- [64] R Paschotta.
- [65] P. C. Maurer, G. Kucsko, C. Latta, L. Jiang, N. Y. Yao, S. D. Bennett, F. Pastawski, D. Hunger, N. Chisholm, M. Markham, D. J. Twitchen, J. I. Cirac, and M. D. Lukin. Room-temperature quantum bit memory exceeding one second. *Science*, 336(6086):1283–1286, 2012.
- [66] Daniel Riedel, Immo Söllner, Brendan J. Shields, Sebastian Starosielec, Patrick Appel, Elke Neu, Patrick Maletinsky, and Richard J. Warburton. Deterministic enhancement of coherent photon generation from a nitrogen-vacancy center in ultrapure diamond. *Phys. Rev. X*, 7:031040, Sep 2017.

- [67] B. Hensen, H. Bernien, A. E. Dréau, A. Reiserer, N. Kalb, M. S. Blok, J. Ruitenberg, R. F. L. Vermeulen, R. N. Schouten, C. Abellán, W. Amaya, V. Pruneri, M. W. Mitchell, M. Markham, D. J. Twitchen, D. Elkouss, S. Wehner, T. H. Taminiau, and R. Hanson. Loophole-free Bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526:682 EP –, 10 2015.
- [68] Stefan Bogdanović, Madelaine S. Z. Liddy, Suzanne B. van Dam, Lisanne C. Coenen, Thomas Fink, Marko Lončar, and Ronald Hanson. Robust nano-fabrication of an integrated platform for spin control in a tunable microcavity. *APL Photonics*, 2(12):126101, 2017.
- [69] Chandra Natarajan, Lijian Zhang, Hendrik Coldenstrodt-Ronge, Gaia Donati, S Dorenbos, V Zwiller, Ian Walmsley, and Robert Hadfield. Quantum detector tomography of a time-multiplexed superconducting nanowire single-photon detector at telecom wavelengths. *Optics express*, 21:893–902, 01 2013.
- [70] Andreas Reiserer, Norbert Kalb, Machiel S. Blok, Koen J. M. van Bemmelen, Tim H. Taminiau, Ronald Hanson, Daniel J. Twitchen, and Matthew Markham. Robust quantum-network memory using decoherence-protected subspaces of nuclear spins. *Phys. Rev. X*, 6:021040, Jun 2016.
- [71] Peter C. Humphreys, Norbert Kalb, Jaco P. J. Morits, Raymond N. Schouten, Raymond F. L. Vermeulen, Daniel J. Twitchen, Matthew Markham, and Ronald Hanson. Deterministic delivery of remote entanglement on a quantum network. *Nature*, 558(7709):268–273, 2018.
- [72] N. Kalb, A. A. Reiserer, P. C. Humphreys, J. J. W. Bakermans, S. J. Kamerling, N. H. Nickerson, S. C. Benjamin, D. J. Twitchen, M. Markham, and R. Hanson. Entanglement distillation between solid-state quantum network nodes. *Science*, 356(6341):928–932, 2017.
- [73] A. M. Steane. Overhead and noise threshold of fault-tolerant quantum error correction. *pra*, 68(4):042322, Oct 2003.
- [74] W. Dür and H. J. Briegel. Entanglement purification and quantum error correction. *Reports on Progress in Physics*, 70:1381–1424, Aug 2007.

# Appendix A

## Other details regarding the Quantum networking simulator

### A.1 Simulator size

The Network Description File (NED) file specifies the connectivities of nodes and modules, and initialize parameters. A value may be assigned to a parameter simultaneously.

The length of *.ned* files, codes for describing the module connectivity within a single node, are as shown below. Length includes empty lines.

- 138 `QNode.ned`:  
Defines the module connectivity within a single quantum node. Modules in a quantum node include the Quantum Routing Software (QRSA) module, the classical routing software module, QNIC module, NIC module and the application module.
- 76 `classical_softwares.ned`:  
Defines classical software modules, including the queue module, the application module and the routing module.
- 184 `qnic.ned`:  
Defines the module connectivity within a single QNIC. Modules in a QNIC include the stationary qubit module, the multiplexer module and the internal BSA module.
- 141 `qrsa.ned`:  
Defines modules and their connectivity within a single Quantum Routing Software (QRSA) module. Modules include the routing daemon module, the hardware monitor module, the real time controller module, the rule engine module and the connection manager module.
- 29 `channels.ned`:  
Defines channel modules. Channels include the quantum channel, and the classical channel.
- 180 `HoM.ned`:  
Defines the BSA module, and the internal BSA module for a MeetInTheMiddle link and for a SenderReceiver link accordingly.

- 160 `EPPS.ned`:  
Defines the SPDC module for a MidpointSource link.

The length of an example `.ned` file, codes for describing the node connectivity of a network, is as shown below. The length strongly depends on the number of simulated networks (including those used for testing).

- 1108 `topology_realistic_layer2_experiments.ned`:  
Defines one or more network for a simulation. A network is also a module that is composed of predefined nodes such as the QNode.

The following `.cc` files specifies the behavior (functions) of modules defined in the NED files. The total length of the files is 6835 lines.

- 191 `Application.cc`
- 388 `BellStateAnalyzer.cc`
- 266 `ConnectionManager.cc`
- 88 `EntangledPhotonPairSource.cc`
- 58 `EntangledPhotonPairSource.h`
- 936 `HardwareMonitor.cc`
- 148 `HardwareMonitor.h`
- 442 `HoM_Controller.cc`
- 97 `HoM_Controller.h`
- 47 `QNIC.h`
- 117 `QNIC_photonic_switch.cc`
- 53 `QUBIT.h`
- 160 `Queue.cc`
- 73 `RealTimeController.cc`
- 43 `RealTimeController.h`
- 24 `ResourceManager.cc`
- 38 `ResourceManager.h`
- 181 `Router.cc`
- 114 `RoutingDaemon.cc`

- 44 RoutingDaemon.h
- 1194 RuleEngine.cc
- 169 RuleEngine.h
- 247 SPDC\_Controller.cc
- 1332 stationaryQubit.cc
- 348 stationaryQubit.h

A RuleSet is an object, which holds at least one Rule object. Each Rule object holds an Action object, and a Condition object that consist of one or more Clause object. Below are lengths of files defining the structure and the functions of a RuleSet, Rule, Action, Condition and Clause.

- 686 Action.cc
- 436 Action.h
- 136 Clause.cc
- 151 Clause.h
- 66 Condition.cc
- 37 Condition.h
- 93 Rule.cc
- 70 Rule.h
- 20 RuleSet.cc
- 56 RuleSet.h
- 129 tools.h

The *.ini* files are used to initialize the simulation, and assigns values to all variables. Variables may be the channel cost, channel length, address, number of qubits installed in a particular QNIC and others. Values may also be assigned directly in the *.ned* file as a static value. Notice that each simulation will require one *.ini* definition. Some of the used files and their lengths are shown below.

- 253 Fixed10\_No\_X\_XZ\_XXZPuri\_Low\_memErr\_.ini
- 288 Fixed10\_No\_X\_XZ\_XXZ\_DSPuri\_Low\_memErr\_.ini
- 250 Fixed15\_No\_X\_XZ\_XXZPuri\_Low\_memErr.ini

Simulated packet structures are defined in the *.msg* files.

- 217 classical\_messages.msg
- 70 PhotonicQubit.msg

## A.2 How to build and run the simulator

The simulator is known to work with OmNET++ 5.4.1 and an external C++ library, Eigen 3.3.90. This thesis ran the simulator over Ubuntu 16.04.6 LTS using gcc 5.4.0. Building the simulation is done automatically when you hit the run button via the OMNeT++ GUI interface. Otherwise, use the `opp_makemake` command and type `opp_makemake -f --deep -O out -I/path_to_eigen/eigen3 -I. .`. Once this is done, type `make`. This will output an executable file `quisp`.

### A.2.1 Run-time configurable parameters

To run a simulation, you need to choose which simulation to run via specifying the `.ini` file for execution - all `.ini` files are stored in the `network` directory. Suppose you want to run the `Linear_Single_MIM` simulation, type `.. ./quisp -m -u Cmdenv -n .. omnetpp.ini -c Linear_Single_MIM` inside the `network` directory. All parameters values must be pre-specified in the corresponding `.ini` file. The current implementation supports the following parameters. Those parameters can also be specified per module (e.g. `qubit`). Details are also provided in the official OMNeT++ documentation.

```
**.Hgate_error_rate // Hadamard gate error rate. Scale from 0
to 1.

**.Hgate_X_error_ratio // The error rate will be divided in to
three Pauli errors as in X:Y:Z, with P(X)+P(Y)+P(Z) = Hgate-
error_rate

**.Hgate_Y_error_ratio

**.Hgate_Z_error_ratio

**.Measurement_error_rate // Measurement gate error rate

**.Measurement_X_error_ratio

**.Measurement_Y_error_ratio

**.Measurement_Z_error_ratio

**.Xgate_error_rate // X gate error rate

**.Xgate_X_error_ratio

**.Xgate_Y_error_ratio

**.Xgate_Z_error_ratio

**.Zgate_error_rate // Z gate error rate

**.Zgate_X_error_ratio
```

```

**.Zgate_Y_error_ratio
**.Zgate_Z_error_ratio
**.CNOTgate_error_rate // CNOT gate error rate
**.CNOTgate_IZ_error_ratio
**.CNOTgate_ZI_error_ratio
**.CNOTgate_ZZ_error_ratio
**.CNOTgate_IX_error_ratio
**.CNOTgate_XI_error_ratio
**.CNOTgate_XX_error_ratio
**.CNOTgate_IY_error_ratio
**.CNOTgate_YI_error_ratio
**.CNOTgate_YY_error_ratio
**.memory_X_error_rate // Memory X error rate per microsec.
**.memory_Y_error_rate // Memory Y error rate per microsec.
**.memory_Z_error_rate // Memory Z error rate per microsec.
**.memory_energy_excitation_rate // Memory excitation error rate
**.memory_energy_relaxation_rate // Memory relaxation error rate
**.memory_completely_mixed_rate // Memory completely mixed error rate
**.channel_Loss_error_rate //Channel photon loss error rate per km.
**.channel_X_error_rate //Channel X error rate per km.
**.channel_Z_error_rate //Channel Z error rate per km.
**.channel_Y_error_rate //Channel Y error rate per km.
**.internal_hom_photon_detection_per_sec //Internal BSA detection rate. This will determine the photon burst rate over the SenderReceiver link.
**.internal_hom_darkcount_probability //dark count probability per photon detector

```

```

**.hom_photon_detection_per_sec //BSA detection rate. This will
determine the photon burst rate over the MeetInTheMiddle link.

**.hom_darkcount_probability //dark count probability per pho-
ton detector

**.num_measure //Number of qubits to be measured for the boot-
strapping

**.emission_success_probability //Photon emission probability
from the qubit to optical fiber. If the emission fails, the
error is treated the same way as photon loss in fiber.

**.buffers //Number of qubits per QNIC

**.tomography_output_filename //Output file name

**.link_tomography //A boolean identifying whether to perform
link bootstrapping in the simulation. This may be turned off
when simulating multi-hop quantum communication with prede-
fined quantum link costs.

**.initial_purification //Integer identifying the number of pu-
rifications for the link bootstrapping.

**.Purification_type //Integer (ID) identifying the type of pu-
rification for link bootstrapping.

**.Initial_notification_timing_buffer // Extra buffer time for
starting the entanglement generation (used when RuleSets need
to be distributed before the entanglement generation)

```

See the official OMNeT++ documentation for more details regarding building and running a simulation.

### A.2.2 Output file format

In the .ini file, you have to specify the output file name for storing simulation data of the quantum link bootstrapping experiment: `**.tomography_output_filename = "Output_File_Name"`. Once the simulation is done, it outputs two files: one as `Output_File_Name_dm` and another as `Output_File_Name`. Those output files will be created in the `network` directory. The `Output_File_Name_dm` is a raw ASCII text file with the reconstructed density matrix between all connected nodes. An example output of the `Output_File_Name_dm` file is:

```

EndNode1[0]<-->Repeater1[0] -->
REAL
0.245333 0.00546737 0.00436971 0.239046
0.00546737 0.250667 0.260954 0.0115033

```

```

0.00436971 0.260954 0.253333 0.0104056
0.239046 0.0115033 0.0104056 0.250667
IMAGINARY
0 0.00795211 0.00799364 -0.00275699
-0.00795211 0 0.0163757 -0.00541632
-0.00799364 -0.0163757 0 0.00106851
0.00275699 0.00541632 -0.00106851 0

```

The first line specifies which link the reconstructed density matrix corresponds to. In this case, the density matrix is reconstructed using resources between EndNode1[0] and Repeater1[0] (which are, of course, neighbors). The real part and the imaginary part of the density matrix is outputted separately as shown.

The `Output_File_Name` file contains the important simulation data of related to the quantum link bootstrapping. An example output of the `Output_File_Name` file is:

```

EndNode1[0]<->QuantumChannel{cost = 602856; distance = 5km; fidelity = 0.487046; bellpair_per_sec = 699.271; tomography_time = 0.010420677295; tomography_measurements = 7000; actualmeas = 7000; GOD_clean_pair_total = 3525; GOD_X_pair_total = 3475; GOD_Y_pair_total = 0; GOD_Z_pair_total = 0; }<->Repeater1[0]; F = 0.487046; X = 0.512954; Z = 0.00895361; Y = -0.00895361

```

The line `EndNode1[0]<->QuantumChannel{ *** }<->Repeater1[0];` is directly readable by OMNeT++ as a topology (node connectivity) definition in the `.ned` file. The `cost` is the cost of the quantum channel. This is currently not in use, but needs to be defined properly later for routing. The `distance` is the length of the quantum channel. The `fidelity` is the reconstructed fidelity, and the `bellpair_per_sec` is its corresponding throughput calculated as measured Bell pairs per second. The `tomography_time` shows how much simulated time it took for completing the link bootstrapping from the start of the simulation. The `tomography_measurements` is the expected number of measured resources, where `actualmeas` was used for debugging to identify how many of them were actually measured. The `GOD_clean_pair_total` is the total amount of measured resources with no error. The `GOD_X_pair_total`, `GOD_Y_pair_total`, `GOD_Z_pair_total` are the total amount of measured resources with a Pauli error, X, Y and Z accordingly. Those `GOD` parameters are not accessible in the real world, but is there for practical reasons (e.g. for evaluation). The `F,X,Y,Z` correspond to the reconstructed fidelity (which is actually redundant because it is technically the same as `fidelity`), and the reconstructed Pauli error rates. As you may have noticed by now, this output needs to be cleaned up a little bit more in the future.

## A.3 How to extend the simulator

### A.3.1 Adding a new Action and a new Clause

The Action already has an abstract class defined as `class Action`, which serves as an interface. This class provides a common function named `virtual cPacket* run(`

`cModule *re ) = 0;`. As shown, the function may return a `cPacket`, which is a classical packet (e.g. for sharing measurement results) that will be passed to the `RuleEngine`, and takes the `cModule RuleEngine` as an argument because it may require some of the functions belonging to the `RuleEngine` (e.g. freeing resources). If you wish to add a new Action to the simulation, you need to write a constructor for the new Action (which extends the `Action` class) in the `Action.h` file. The behavior of the new Action must be written inside the overriding `run()` function in the `Action.c` file. The Action currently only supports operations requiring a single list of allocated resources, which is physically located at a single QNIC. Therefore, when you need to simulate entanglement swapping, you have to rewrite and extend the code so that it supports two lists of allocated resources for each local QNIC.

The Clause works similarly to the Action. The interface is provided as `class Clause` in the `Clause.h` file, with common functions `virtual bool check( std::map< int, stationaryQubit* > ) const = 0;` and `virtual bool checkTerminate( std::map< int, stationaryQubit* > ) const = 0;`. The function `check()` is used for checking whether the clause satisfies to invoke the corresponding Action, and the `checkTerminate( std::map< int, stationaryQubit* > )` checks when to terminate and delete the whole RuleSet. Those functions only return a boolean for identifying whether the clause has satisfied or not. The argument, `std::map< int, stationaryQubit* >`, is a list of resources allocated for the Rule.

### A.3.2 Adding a new quantum gate

Quantum gate operations in the simulation track Pauli error propagations, instead of doing the actual matrix calculation (except for measurement gates used for tomography). A gate may also be noisy, where the error is stochastically applied to the qubit based on a random selection with a given probability distribution. All gate operations are defined inside the `stationaryQubit.h` and `stationaryQubit.cc`. If necessary, you can also add gate operations that actually do the math using vectors and matrices. In that case, you need to correctly update and manage the distributed state. Those function should look similar to the measurement function `measurement_outcome stationaryQubit::measure_density_independent()`.

### A.3.3 How to add a new protocol (classical packet)

The packet must be defined in the `classical_messages.msg` file as: `packet ExampleProtocol extends header{ *** }.` The header holds three basic parameters, which are `int srcAddr; //Source node's address, int destAddr; //Destination node's address and int hopCount; // Hop count for time-to-live`, which is currently not implemented. OMNeT++, by default, distinguishes protocols by trying to cast the received packet as `if(dynamic_cast<ExampleProtocol *>(msg) != nullptr)`, where `msg` is the received message/packet. In the current implementation, the `Router.cc` must recognize the protocol because every classical packet from another node goes through this module.

## A.4 Current state of the simulator

The simulator meets the primary goals establishing for this project. It is also a platform for future experiments. The list below should help those who wish to extend the simulator. The simulator has been developed while prototyping simultaneously. In reading the code, you may find some parameters in modules have been used for debugging purposes, or are already unused.

- The current simulator is capable setting a dynamic size of memory buffer per QNIC. Each memory will transmit one photon, unless it is reinitialized after some operation.
- Entangled qubits have a pointer pointing to each other.
- The RuleSet is programmable, but Actions other than purifications (Ss-Sp, Ds-Sp, Ss-Dp, Ds-Dp) and measurements (based on error propagation or based on density matrix) are not implemented in the current version – e.g. entanglement swapping.
- The current Rule assumes a single allocated resource list. We may need two for operations requiring two resources from two different local QNICs.
- The RuleEngine assigns all resources to the first RuleSet stored because my thesis assumes a single RuleSet per link (like circuit switching). You may improve this to simulate multiplexing schemes.
- The real-time controller has not yet been implemented. Operations are directly called by the RuleEngine through a function in the RuleSet.
- The hardware monitor is capable of generating a routing table for both classical and quantum based on Dijkstra's algorithm. This, however, assumes a single simulated network. To simulate quantum Internet, you need to make each node recognize its own network, and implement other routing protocols to connect those (e.g. BGP).
- For a multi-hop quantum communication, the connection manager can break down the path to decide where to perform entanglement swapping. This, however, does not consider the link quality at all.
- The output file of the link bootstrapping simulation does include a *cost* parameter, but the calculation process isn't carefully designed yet. Once this is taken care, the value is used by the hardware monitor to create the routing table.

### A.4.1 Limitations

First of all, the developed simulator only tracks Pauli error propagations through gates. This will simplify the calculation process, but approximate the output fidelity to a pessimistic value because non-Pauli errors will be ignored – the throughput, therefore, will be an optimistic value.

The simulation assumes a Bell pair with exactly two qubits entangled across arbitrary nodes. You can extend the simulator to make it recognize some other types of entangled states, but making this dynamic is very hard.

## A.4.2 Example run

An example configuration is shown below.

```
[General]
**.Hgate_error_rate = 1/2000
**.Hgate_X_error_ratio = 0
**.Hgate_Y_error_ratio = 0
**.Hgate_Z_error_ratio = 0

**.Measurement_error_rate = 1/2000
**.Measurement_X_error_ratio = 1
**.Measurement_Y_error_ratio = 1
**.Measurement_Z_error_ratio = 1

**.Xgate_error_rate = 1/2000
**.Xgate_X_error_ratio = 0
**.Xgate_Y_error_ratio = 0
**.Xgate_Z_error_ratio = 0

**.Zgate_error_rate = 1/2000
**.Zgate_X_error_ratio = 0
**.Zgate_Y_error_ratio = 0
**.Zgate_Z_error_ratio = 0

#Error on Target, Error on Controlled
**.CNOTgate_error_rate = 1/2000
**.CNOTgate_IZ_error_ratio = 1
**.CNOTgate_ZI_error_ratio = 1
**.CNOTgate_ZZ_error_ratio = 1
**.CNOTgate_IX_error_ratio = 1
**.CNOTgate_XI_error_ratio = 1
**.CNOTgate_XX_error_ratio = 1
**.CNOTgate_IY_error_ratio = 1
**.CNOTgate_YI_error_ratio = 1
**.CNOTgate_YY_error_ratio = 1

**.memory_X_error_rate = 1.1111111e-7 #ratio. Not the error rate!!
**.memory_Y_error_rate = 1.1111111e-7 #ratio. Not the error rate!!
**.memory_Z_error_rate = 1.1111111e-7 #ratio. Not the error rate!!
By default the ratio is 1:1:1
**.memory_energy_excitation_rate = 0.000198
**.memory_energy_relaxation_rate = 0.00000198
**.memory_completely_mixed_rate = 0

**.Initial_notification_timing_buffer = 10 s #when to start the
BSA timing notification.
```

```

[Config Example_run_Ss-Sp]
network= Realistic_Layer2_Simple_MIM_MM_5km
seed-set = $0
**.num_measure = 7000
**.buffers = 100
**.tomography_output_filename = "Example_run_Ss-Sp_5km"
**.emission_success_probability = 0.46*0.49

# Error on optical qubit in a channel
**.channel_Loss_error_rate = 0.04500741397 # per km
**.channel_X_error_rate = 0.01
**.channel_Z_error_rate = 0.01
**.channel_Y_error_rate = 0.01

# Internal HoM in QNIC
**.internal_hom_photon_detection_per_sec = 1000000000
**.internal_hom_darkcount_probability = 10e-8 #10/s

#Stand-alone HoM in the network
**.hom_photon_detection_per_sec = 1000000000
**.hom_darkcount_probability = 10e-8

**.link_tomography = true # true activates bootstrapping.
**.initial_purification = 1 # Number of purification to be performed before tomography. Since it is once, only Ss-Sp X purification will be performed.
**.Purification_type = 3003 # ID=3003 refers to Ss-Sp. Switches Ss-Sp X purification and Ss-Sp Z purification every attempt.

```

This simulation will run quantum link bootstrapping, based on Ss-Sp with only  $n = 1$  round. The output, therefore, will only purify the X error before performing tomography. The network topology used for the simulation is shown in Fig. A.1. This particular simulation took about 3.5 minutes over a server with OS Ubuntu 16.04.6 LTS including 24 processors, with a maximum total memory size of 491520 MB. The *processor 0*'s specification is summarized in Tab.A.1.

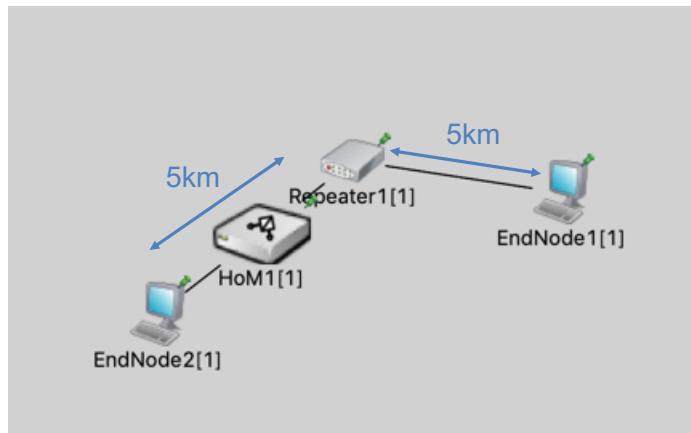


Figure A.1: Network topology defined as Realistic\_Layer2\_Simple\_MIM\_MM\_5km.

Table A.1: Major Processor 0 specification.

processor	0
vendor_id	GenuineIntel
cpu family	6
model	85
model name	Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz
stepping	4
microcode	0x2000043
cpu MHz	2999.999
cache size	25344 KB
physical id	0
siblings	1
core id	0
cpu cores	1
apicid	0
initial apicid	0
fpu	yes
fpu_exception	yes
cpuid level	22
wp	yes
bugs	cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf
bogomips	5999.99
clflush size	64
cache_alignment	64
address sizes	43 bits physical, 48 bits virtual

The output Example\_run\_Ss-Sp\_5km is:

```
EndNode1[0]<->QuantumChannel{cost=209924; distance=5km;
fidelity=0.850604; bellpair_per_sec=658.389;
```

```

tomography_time=10.63201768213; tomography_measurements=7000;
actualmeas=7000; GOD_clean_pair_total=5996; GOD_X_pair_to-
tal=44; GOD_Y_pair_total=30; GOD_Z_pair_total=691;
} <-> Repeater1[0]; F=0.850604; X=0.0292881; Z=0.110106;
Y=0.0100021

EndNode2[0] <-> QuantumChannel{cost=202156; distance=2.5km;
fidelity=0.854019; bellpair_per_sec=678.232;
tomography_time=10.320951596033; tomography_measurements=7000;
actualmeas=7000; GOD_clean_pair_total=6025; GOD_X_pair_to-
tal=36; GOD_Y_pair_total=31; GOD_Z_pair_total=687;
} <-> HoM1[0]; F=0.854019; X=0.0185928; Z=0.11239;
Y=0.0149989

Repeater1[0] <-> QuantumChannel{cost=209924; distance=5km;
fidelity=0.850604; bellpair_per_sec=658.389;
tomography_time=10.63201768213; tomography_measurements=7000;
actualmeas=7000; GOD_clean_pair_total=5996; GOD_X_pair_to-
tal=44; GOD_Y_pair_total=30; GOD_Z_pair_total=691;
} <-> EndNode1[0]; F=0.850604; X=0.0292881; Z=0.110106;
Y=0.0100021

Repeater1[0] <-> QuantumChannel{cost=202156; distance=2.5km;
fidelity=0.854019; bellpair_per_sec=678.232;
tomography_time=10.320951596033; tomography_measurements=7000;
actualmeas=7000; GOD_clean_pair_total=6025; GOD_X_pair_to-
tal=36; GOD_Y_pair_total=31; GOD_Z_pair_total=687;
} <-> HoM1[0]; F=0.854019; X=0.0185928; Z=0.11239;
Y=0.0149989

```

The output Example\_run\_Ss-Sp\_5km\_dm is:

```

EndNode1[0] <--> Repeater1[0]
REAL
0.462611 0.00723047 0.0170436 0.370249
0.00723047 0.0202788 0.00964299 -0.00152404
0.0170436 0.00964299 0.0190114 0.00154146
0.370249 -0.00152404 0.00154146 0.498099
IMAGINARY
0 -0.0146235 0.0182761 -0.00885229
0.0146235 0 0.0129395 -0.00312463
-0.0182761 -0.0129395 0 0.00553257
0.00885229 0.00312463 -0.00553257 0

EndNode2[0] <--> HoM1[0]
REAL
0.488372 0.0162139 0.00510361 0.370815

```

```

0.0162139 0.0142119 0.00179692 0.00827218
0.00510361 0.00179692 0.0193798 0.00990076
0.370815 0.00827218 0.00990076 0.478036
IMAGINARY
0 0.000422923 3.21287e-05 -0.0176171
-0.000422923 0 -0.00498175 -0.00506991
-3.21287e-05 0.00498175 0 -0.00294181
0.0176171 0.00506991 0.00294181 0

Repeater1[0]<-->EndNode1[0]
REAL
0.462611 0.0170436 0.00723047 0.370249
0.0170436 0.0190114 0.00964299 0.00154146
0.00723047 0.00964299 0.0202788 -0.00152404
0.370249 0.00154146 -0.00152404 0.498099
IMAGINARY
0 0.0182761 -0.0146235 -0.00885229
-0.0182761 0 -0.0129395 0.00553257
0.0146235 0.0129395 0 -0.00312463
0.00885229 -0.00553257 0.00312463 0

Repeater1[0]<-->HoM1[0]
REAL
0.488372 0.00510361 0.0162139 0.370815
0.00510361 0.0193798 0.00179692 0.00990076
0.0162139 0.00179692 0.0142119 0.00827218
0.370815 0.00990076 0.00827218 0.478036
IMAGINARY
0 3.21287e-05 0.000422923 -0.0176171
-3.21287e-05 0 0.00498175 -0.00294181
-0.000422923 -0.00498175 0 -0.00506991
0.0176171 0.00294181 0.00506991 0

```

As shown in the output, the reconstructed fidelity is about  $F_r = 0.85$ . As the simulation only performs X purification, the X error rate is lower than the Z error rate. The actual process time for the link bootstrapping in this case is `tomography_time` (in the `Example_run_Ss-Sp_5km`) - `Initial_notification_timing_buffer` (in the configuration). To check the impact of purification, you can run the same simulation with a different `initial_purification` value.