Master's Thesis (Academic Year 2023)

Internetworking for RuleSet-based Quantum Repeater Networks

A thesis presented for the degree of
Master of Media and Governance

Keio University
Graduate School of Media and Governance

Kentaro Teramoto

Abstract of Master' s Thesis Academic Year 2023

Internetworking for RuleSet-based Quantum Repeater Networks

A quantum repeater network allows the transfer of an arbitrary quantum state over a long distance via quantum teleportation, which consumes an entangled state known as a Bell pair between sender and receiver. The quantum repeater network allows quantum nodes to properly perform entanglement swapping and purification to generate an end-to-end Bell pair between the sender and the receiver and provide it to the application on the two nodes. Quantum repeater networks enable applications such as distributed quantum computation, quantum sensing, and quantum cryptographic communications. The Quantum Internet will be a scalable and global infrastructure that will further expand the potential of quantum networking around the world and bring the benefits of quantum computing to the world. To realize a Quantum Internet that interconnects quantum networks, we need a protocol and architecture for internetworking that advances toward a scalable Quantum Internet. To date, only an abstract concept has been proposed for internetworking, with no implementation and many design decisions left pending. In this research, we propose two distinct approaches for recursive quantum network structure, namely link- and node-recursion, and apply them to RuleSet-based quantum networks and implement the QRNA with the link-recursion on a simulator to analyze their characteristics.

Keywords: 1. Quantum Internet, 2. Quantum Networking, 3. Quantum Repeater, 4. RuleSet-based Protocol,

Keio University
Graduate School of Media and Governance

Kentaro Teramoto

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Expectations for quantum computers have been growing in recent years. Quantum computers are built on principles of quantum mechanics, which differ greatly from those of classical computers. Quantum computers are expected to offer algorithms that require less computational complexity than classical computers [1], information-theoretically secure cryptography [2, 3], and quantum sensing [4].

In order to maximize the performance of such quantum computers, many proposals have been made in recent years regarding quantum networks and the quantum Internet [5].

There have been many studies and proposals for the realization of a quantum Internet, especially for a single quantum network, from low layers, such as experiments on physical systems, to high layers, such as protocol stacks and architectures[6].

Connecting quantum networks to each other is essential to create a global quantum Internet. A single quantum network will not be able to manage the large number of nodes that will exist around the world in the future. In addition, if a quantum network is created that spans multiple countries or regions, it cannot be managed by a single organization, but will likely be managed by multiple organizations. The quantum computers that will be connected to the quantum network in the future will be owned by several different organizations. Also, since there are various physical systems and schemes for quantum networks, they may want to connect them as different networks. The interconnection of quantum networks is essential to the realization of the quantum Internet. On the other hand, there is still little research on how to connect quantum networks together as a quantum Internet [7, 8].

However, internetworking in quantum networks and its architecture have a significant impact on the design of the protocols that operate within a quantum network. When connecting quantum networks that operate with different mechanisms, internetworking is the junction point, and its design determines the scalability of each quantum network. Therefore, research on internetworking should be conducted si-

multaneously with research on quantum network protocols.

Therefore, in this thesis, we design a more specific quantum recursive network architecture (QRNA), which has been proposed as an architecture for the quantum Internet [9], apply it to RuleSet-based protocol [10], and implement it on a quantum network simulator.

1.2 Research Contribution

The major contributions of this research are the analysis of two important recursion types for the design of quantum network architectures and the implementation of the first internetworking for quantum networks on a simulator by applying one of the recursion types.

The two recursion types were not significant enough to be taken into account, although similar structures could appear in the classical Internet. However, in the design of the quantum Internet, it is a factor that significantly impacts the performance and design of each protocol.

We have shown that the internetworking connecting quantum networks can be implemented on a simulator as a concrete protocol. This is a first step toward simulating a more complex quantum Internet. This implementation is available as open-source software.

These contributions will be beneficial to those involved in the design and implementation of protocols and architectures for the quantum Internet. The difference between the two types of recursion makes the structure of subnetworks in quantum networks clearer and allows for the search for suitable architectures for quantum networks that are not rehashes of classical networks. The implementation of internetworking on the first simulator will serve as a stepping stone for developers of different quantum network simulators to implement internetworking.

1.3 Thesis Structure

This thesis is structured as follows. Chapter 2 provides the necessary preliminaries for this thesis, describing the basics of quantum computers, quantum networks, and the quantum Internet. It also describes the architecture of the quantum Internet that has already been proposed. Chapter 3 describes the problem addressed in this thesis. In Chapter 4, we list design considerations for internetworking for RuleSet-based networks and discuss what kind of design should be used to achieve this. Chapter 5 describes the implementation of the protocol designed based on the discussion on the simulator in Chapter 4. In Chapter 6, we evaluate the designed protocol and implementation. Chapter 7 provides the conclusion.

Chapter 2

Preliminaries

This chapter describes the knowledge of quantum information and quantum networks, a prerequisite for this thesis. In the following sections, we will begin with a basic description of qubits, their counterintuitive nature, and how quantum networks work with qubits.

2.1 Qubits

The bit, the basic unit of information in a classical digital computer, is called a quantum bit (qubit) in a quantum computer. A qubit can represent a quantum state that is fundamentally different from that of a classical digital bit, but we will start by looking at the similarities with the classical digital bit. Then we will look at the unique properties of quantum.

A qubit can store the states $|0\rangle$ and $|1\rangle$. This can be thought of as corresponding to 0 and 1 in the classical bit, and to build a computer using a qubit, we need at least a way to know the state of the qubit (measurement) and a way to change the state of the qubit (gate operation).

For example, in classical computers, a bit is physically represented by a level of electrical voltage. The state of a bit can be determined by whether the voltage is higher or lower than a specified value. The state of a bit can also be changed by connecting elements called gates: a NOT gate takes one bit as an input and outputs the inverted bit; an XOR gate takes two bits as inputs and outputs 1 only if one of the inputs is 1, and 0 otherwise. Classical computers are built on such operations on bits.

On the other hand, in a quantum computer, the state of a qubit can also be represented as part of an actual physical phenomenon. Physical phenomena that can represent qubits include the polarization of photons, charged particles trapped in electromagnetic fields, superconducting circuits, etc. The method of detecting the state of these qubits is called **measurement**, and it is particularly unique in that it can affect the state of the qubit. The NOT gate in a classical computer is called an

X-gate in a quantum computer, which inverts $|0\rangle$ and $|1\rangle$.

2.1.1 Quantum State

Quantum states can generally be expressed in the form of a complex vector. For example, a single qubit state $|\psi\rangle$ has the form

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,. \tag{2.1.1}$$

Here, the $|0\rangle$ and $|1\rangle$ are called Dirac's braket notation. The character in the ket symbol ($|\rangle$) means the name of the column vector

$$|0\rangle \equiv \begin{pmatrix}1\\0\end{pmatrix}, \qquad |1\rangle \equiv \begin{pmatrix}0\\1\end{pmatrix}. \tag{2.1.2}$$

And also, the bra symbol ($\langle|$) represents a Hermitian conjugate of the corresponding ket vector

$$\langle 0| \equiv |0\rangle^\dagger \equiv \begin{pmatrix}1 & 0\end{pmatrix}, \qquad \langle 1| \equiv |1\rangle^\dagger \equiv \begin{pmatrix}0 & 1\end{pmatrix}. \tag{2.1.3}$$

Here, $\langle\psi| = |\psi\rangle^\dagger = (\psi_1^*, \psi_2^*, \ldots, \psi_n^*)$. The $\dagger$ is the symbol for the Hermitian conjugate. Also, $\langle\phi|\psi\rangle$ denotes the inner product of vectors.

The set of these two vectors $|0\rangle, |1\rangle$ is called the Z basis or the computational basis. Other commonly used states are $|+\rangle$ and $|-\rangle$. The set of these two is called the X basis or the Hadamard basis. These represent vectors

$$|+\rangle \equiv \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \qquad |-\rangle \equiv \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \tag{2.1.4}$$

We can write down Dirac's braket notation into single-column vectors

$$\begin{aligned} |\psi\rangle &= \alpha\,|0\rangle + \beta\,|1\rangle \\ &= \alpha \cdot \begin{pmatrix}1\\0\end{pmatrix} + \beta \cdot \begin{pmatrix}0\\1\end{pmatrix} \\ &= \begin{pmatrix}\alpha\\0\end{pmatrix} + \begin{pmatrix}0\\\beta\end{pmatrix} \\ &= \begin{pmatrix}\alpha\\\beta\end{pmatrix}, \end{aligned} \tag{2.1.5}$$

where α and β are complex numbers and satisfy the equation

$$|\alpha|^2 + |\beta|^2 = 1.$$

In this case, the states that a qubit can be in are not just either $|0\rangle$ or $|1\rangle$. For example, consider the state

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}. \tag{2.1.6}$$

This state has $|0\rangle$ and $|1\rangle$ at the same time. This state, represented by a linear combination of vectors, is called a **superposition state**. This superposition state cannot exist in a classical bit. In this superposition state, the $|0\rangle$ state and the $|1\rangle$ state exist simultaneously. In quantum computation and quantum networks, this state will be skillfully exploited.

2.1.2 Measurement

Measurement is a vital topic in quantum computation because it changes the quantum state of a qubit when a measurement is performed to retrieve information about it. This property is a major difference from classical information, and while it is useful from a security perspective, it is also a nuisance that makes the operation of a qubit difficult.

When a quantum state is measured under a certain basis, a corresponding measurement is obtained. Strangely enough, however, these measurements are probabilistic, and different measurement outcomes can be obtained even if the same quantum state is measured. The probability of obtaining the measurement $i = 0, 1$ is expressed as $|\langle e_i|\psi\rangle|^2$ using the orthonormal basis $\{|e_0\rangle, |e_1\rangle\}$ and the quantum state $|\psi\rangle$. Let's consider what measurements can be obtained from the state of the equation 2.1.6. If the basis is $\{|0\rangle, |1\rangle\}$, called the **computational basis**, we obtain the following probabilities, respectively.

$$|\langle 0|\psi\rangle|^2 = \left|\frac{1}{\sqrt{2}}\langle 0|0\rangle + \frac{1}{\sqrt{2}}\langle 0|1\rangle\right|^2 = \left|\frac{1}{\sqrt{2}}\cdot 1 + \frac{1}{\sqrt{2}}\cdot 0\right|^2 = \frac{1}{2} \tag{2.1.7}$$

$$|\langle 1|\psi\rangle|^2 = \left|\frac{1}{\sqrt{2}}\langle 1|0\rangle + \frac{1}{\sqrt{2}}\langle 1|1\rangle\right|^2 = \left|\frac{1}{\sqrt{2}}\cdot 0 + \frac{1}{\sqrt{2}}\cdot 1\right|^2 = \frac{1}{2} \tag{2.1.8}$$

Thus, the state vector representation of the quantum state has a value that affects the probability distribution of the measurements. However, there is no one-to-one correspondence between the probability distribution of measurements and the vector representation of quantum states. For example, consider the states $|\psi\rangle = |1\rangle$ and $|\phi\rangle = i|1\rangle$. In the case of the computational basis, both would have a measurement corresponding to $|1\rangle$ with probability 1. Thus, several different state vectors may have the same measurement probability distribution. The case in which different state vectors $|\psi\rangle, |\phi\rangle$ are considered as the same quantum state is denoted as follows.

$$|\psi\rangle = c|\phi\rangle \quad (|c| = 1, c \in \mathbb{C}) \tag{2.1.9}$$

2.1.3 Density Matrix

The density matrix is more convenient for dealing with general quantum states than the state vector. If there exists a set of quantum states $\{|\psi_i\rangle\}$ and a quantum state $|\psi_i\rangle$ can be obtained with probability p_i, the density matrix ρ is defined as

$$\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i| . \tag{2.1.10}$$

For example, when a qubit is in the state $|+\rangle$, we obtain $|+\rangle$ with probability 1, so the density matrix ρ can be described as follows

$$\rho = 1 \cdot |+\rangle \langle +| = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} . \tag{2.1.11}$$

On the other hand, if we measure this state in the computational basis and lose the result without knowing it, the state should be $|0\rangle$ or $|1\rangle$ with probability $\frac{1}{2}$. In this case, the density matrix ρ is described as

$$\rho = \frac{1}{2} |0\rangle \langle 0| + \frac{1}{2} |1\rangle \langle 1| = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} . \tag{2.1.12}$$

Thus, a clear distinction is made between quantum superposition states and statistically mixed states. If one quantum state $|\psi\rangle$ is known to be a single state, it is called a pure state. Conversely, if several pure states are mixed together, it is called a mixed state.

2.1.4 Fidelity

Fidelity is often used as a benchmark for how similar two quantum states are. The fidelity F between a mixed state ρ and a pure state $|\psi\rangle$ is described as follows

$$F(\rho, |\psi\rangle \langle\psi|) = \langle\psi| \rho |\psi\rangle . \tag{2.1.13}$$

The fidelity F is 1 if the two states are the same and 0 if the two states are orthogonal. For this reason, fidelity often appears as a key performance metric in the context of quantum computation and quantum networks.

2.1.5 Decoherence

Quantum-specific information on a qubit, such as superposition state and phase, is lost over time under the influence of the external environment. This is called **decoherence** and is one of the factors that makes quantum computation difficult.

T_1 and T_2 called the coherence time, are important indicators of the performance of a qubit. The T_1 is called the energy relaxation time and refers to the time it takes for the state $|1\rangle$ to become $|0\rangle$. The T_2 called dephasing time represents the time

until the superposition state is lost. The coherence time varies greatly depending on the physical system that implements the qubit.

2.1.6 Single Qubit Gates

In real physical systems, the state of a qubit is manipulated by passing a photon through a crystal or shooting a laser at an ion. Such manipulation of the state of a qubit is expressed by multiplying a matrix. In the same way as a classical logic circuit, a series of operations on a qubit is represented as a circuit, and each operation is called a gate. The representation in terms of quantum circuits is commonly used for describing quantum algorithms, debugging quantum programming languages, and so on. Fig. 2.1 shows the symbols used for some single-qubit gates.

(a) X gate (b) Y gate (c) Z gate (d) H gate

Figure 2.1: Quantum Gates in the Circuit Notation

Identity Matrix and Pauli Matrices

The unitary square matrix of order 2 represents operations on a single qubit. The Pauli matrices are three pairs of second-order square matrices, often represented as $\sigma_{1,2,3}$ or $\sigma_{x,y,z}$. Pauli matrices are also called X-gate, Y-gate, and Z-gate, respectively.

$$\mathbb{I} \equiv \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{2.1.14}$$

$$\sigma_x = \sigma_1 = X \equiv \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.1.15}$$

$$\sigma_y = \sigma_2 = Y \equiv \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{2.1.16}$$

$$\sigma_z = \sigma_3 = Z \equiv \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.1.17}$$

Let's look at the application of these matrices to a qubit: the X-gate inverts $|0\rangle$ and $|1\rangle$. The Z gate has no effect on $|0\rangle$, while it inverts the phase for $|1\rangle$. This is why the X-gate is called bit-flip, and the Z-gate is called phase-flip.

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{2.1.18}$$

$$X|1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \tag{2.1.19}$$

$$Z|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \tag{2.1.20}$$

$$Z|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -|1\rangle \tag{2.1.21}$$

$$Z(-|1\rangle) = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{2.1.22}$$

Hadamard Gate

The Hadamard gate, or H-gate for short, is a frequently used gate because it can be applied to $|0\rangle$ to create $|+\rangle$.

$$H \equiv \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.1.23}$$

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = |+\rangle \tag{2.1.24}$$

2.1.7 Time Evolution

The states possessed by a qubit generally change over time. This state change is represented by a unitary matrix U. This state transition is represented by a unitary matrix U, where the original state is $|\psi\rangle$ and the changed state is $|\psi\rangle'$, in the following form.

$$|\psi\rangle' = U|\psi\rangle \tag{2.1.25}$$

If $|\psi\rangle$ is a unit vector of $\mathbb{C}^2$, then $|\psi\rangle'$ applied to it with a unitary matrix is also a unit vector of $\mathbb{C}^2$.

On the other hand, there is a state change by measurement that differs from gate operation and time evolution. The change of state by measurement is not reversible like the gate operation and time evolution introduced so far. For example, in projective measurement, a physical quantity is measured under the quantum state $|\psi\rangle$, and the state changes according to the observed eigenvalues. This is an irreversible change.

2.1.8 Multi-Qubit System

So far, we have dealt with a single qubit, but from now on, we will consider multiple qubits as well. First, consider the state of a single qubit $|\psi\rangle, |\phi\rangle \in \mathbb{C}^2$. If we consider these two qubits together as a single system, we can describe it by a tensor product as follows.

$$|\phi\rangle \otimes |\psi\rangle \in \mathbb{C}^2 \otimes \mathbb{C}^2 = \mathbb{C}^4 \tag{2.1.26}$$

The state vector representing this two-qubit state is a unit vector of $\mathbb{C}^4$ and consists of four elements. The measurement of the two-qubit state is represented by an orthonormal basis of $\mathbb{C}^4$. The gate operation for 2 qubits is represented by a unitary matrix of 4×4 instead of a unitary matrix of 2×2 as in the case of 1 qubit.

The following expressions are often used to resemble classical bits. These consist of the orthonormal basis, which is the computational basis in two-qubit systems. The first and second digits in the ket correspond to the first and second qubits, respectively.

$$|00\rangle \equiv \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |01\rangle \equiv \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad |10\rangle \equiv \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad |11\rangle \equiv \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{2.1.27}$$

The computational basis in the n-qubit system can be expressed in the orthonormal basis of $(\mathbb{C}^2)^{\otimes n}$ as follows

$$\{|i_1\rangle \otimes |i_2\rangle \otimes \cdots |i_n\rangle : i = 0, 1\} \tag{2.1.28}$$

2.2 Quantum Entanglement

Let's look again at the two-qubit state as a composition of two single-qubit $|\psi\rangle = (\alpha_1, \beta_1)^\top$ and $|\phi\rangle = (\alpha_2, \beta_2)^\top$.

$$|\psi\rangle \otimes |\phi\rangle = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} \otimes \begin{pmatrix} \alpha_2 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \alpha_1\alpha 2 \\ \alpha_1\beta 2 \\ \beta_1\alpha 2 \\ \beta_1\beta 2 \end{pmatrix} \tag{2.2.1}$$

In this case, the two-qubit state is represented in the form of a product of two one-qubit states and is therefore called a **product state**. On the other hand, for example, the following state cannot be written down as a tensor product of single qubits.

$$|\psi\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{2.2.2}$$

In a system of two or more qubits, a quantum state that cannot be represented by the tensor product of single qubits is called a **quantum entangled state**. Quantum entangled states are one of the most important concepts in quantum computation and quantum networks and have various characteristics.

The equation (2.2.2) is a superposition state of $|00\rangle$ and $|11\rangle$. When one qubit is measured under the computational basis, this two-qubit state changes irreversibly to either $|00\rangle$ or $|11\rangle$. As a result, the same measurement result can be obtained for the other qubit. This is reproduced even if the two qubits are far apart. Note that it is not the case that the qubit had already changed to one of the states before the measurement.

2.2.1 Two Qubits Gates

Gates operating on two qubits are very important for the transfer of information between qubits and for quantum entanglement generation. The most basic of these gates, the CNOT gate and the CZ gate, will be introduced, and the two-qubit gate is generally represented by a unitary matrix of 4×4 as follows:

$$CNOT \equiv \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.2.3}$$

$$CZ \equiv \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{2.2.4}$$

(a) CNOT(CX) gate (b) CZ gate

Figure 2.2: Two Qubit Gates

Fig. 2.2 shows the symbols for the two gates: the CNOT gate has the first qubit as the control qubit and the second qubit as the target qubit. In other words, the

X gate is applied to the second qubit only when the first qubit is 1. In the circuit notation, the control qubit is represented by a black dot and the target qubit by a crossed circle.

Note that the CNOT gate may be represented by a matrix of a different form depending on the placement of the control and target qubits.

In the CZ gate, both qubits are represented by black dots as control qubits. This representation is made because the operation is symmetric.

2.2.2 No-Cloning Theorem

One property of quantum and classical that is very different is called the no-cloning theorem [11, 12], which states that an arbitrary qubit cannot be copied to another qubit. This is shown by the fact that given a one-qubit state $|\phi\rangle, |\psi\rangle \in \mathbb{C}^2$, there is no 4×4 unitary matrix U such that

$$U |\phi\rangle_1 |\psi\rangle_2 = |\phi\rangle_1 |\phi\rangle_2 \tag{2.2.5}$$

However, if one knows how to prepare that qubit, it is possible to duplicate it by the operation. Another way to move rather than copy any arbitrary quantum state is known as quantum teleportation.

2.2.3 Bell States

In addition to the computational basis, there are four other important states in a two-qubit system, called the Bell basis or Bell states. These four Bell states can be converted to each other simply by applying a unitary matrix to one of the qubits. They are also a kind of maximum entanglement state.

$$|\Phi^+\rangle \equiv \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{2.2.6}$$

$$|\Phi^-\rangle \equiv \frac{|00\rangle - |11\rangle}{\sqrt{2}} \tag{2.2.7}$$

$$|\Psi^+\rangle \equiv \frac{|01\rangle + |10\rangle}{\sqrt{2}} \tag{2.2.8}$$

$$|\Psi^-\rangle \equiv \frac{|01\rangle - |10\rangle}{\sqrt{2}} \tag{2.2.9}$$

2.2.4 Quantum Teleportation

A quantum entanglement is a special state composed of multiple qubits. In particular, the maximum quantum entanglement state consisting of two qubits, called the Bell pair, is essential for quantum teleportation.

Let us assume that we want to send the quantum state that Alice has to Bob as shown in Fig. 2.3. Let $|\psi\rangle_D$ be the quantum state Alice want to send and $|\Phi^+\rangle_{AB}$ be the Bell pair shared between the two. We will now transfer Alice's quantum state to Bob's qubit by local quantum operation and classical communication.

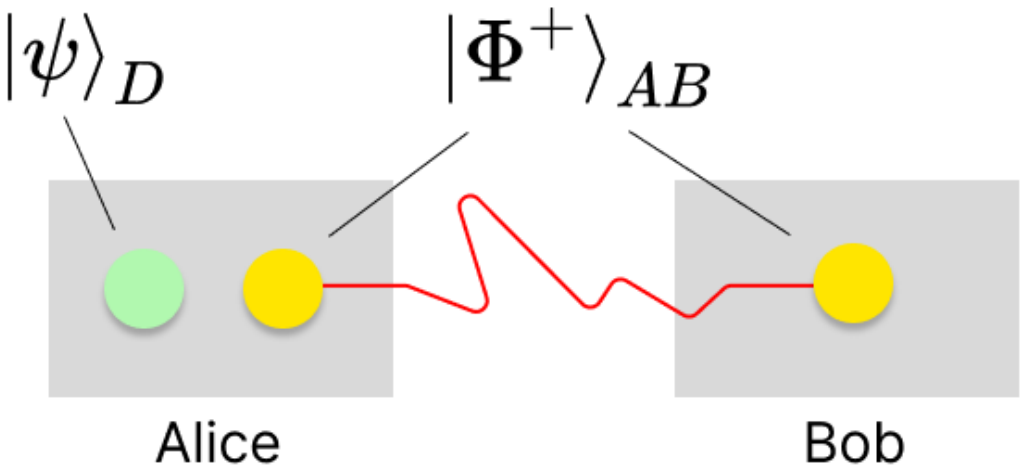

Figure 2.3: Quantum Teleportation Setting

First, a CNOT gate is applied to Alice's two qubits, then an H-gate is applied to the control qubit, and both of Alice's qubits are measured. Next, the measurement results are sent to Bob via classical communication. Bob then applies the X-gate and Z-gate to the qubits according to the measurement results he receives. This procedure allows Bob to obtain the quantum state that Alice initially had.

Fig. 2.4 shows the quantum teleportation procedure as a quantum circuit. In qubits A and B, the H-gate and CNOT gate are first applied to create a Bell pair state $|\Phi^+\rangle$. The next process for Alice's two qubits is to perform a Bell state measurement (BSM) and pass the results to Bob using classical communication. Bob applies the X-gate and Z-gate according to the measurement results received.

Let $|\psi\rangle_D = \alpha|0\rangle_D + \beta|1\rangle_D$ be the qubit Alice wants to send, the state of the three qubits can be expressed as follows.

$$(\alpha|0\rangle_D + \beta|1\rangle_D)|\Phi^+\rangle_{AB} \tag{2.2.10}$$

$$= \frac{\alpha}{\sqrt{2}}|0\rangle_D(|00\rangle_{AB} + |11\rangle_{AB}) + \frac{\beta}{\sqrt{2}}|1\rangle_D(|00\rangle_{AB} + |11\rangle_{AB}) \tag{2.2.11}$$

$$= \frac{\alpha}{\sqrt{2}}(|0\rangle_D|00\rangle_{AB} + |0\rangle_D|11\rangle_{AB}) + \frac{\beta}{\sqrt{2}}(|1\rangle_D|00\rangle_{AB} + |1\rangle_D|11\rangle_{AB}) \tag{2.2.12}$$

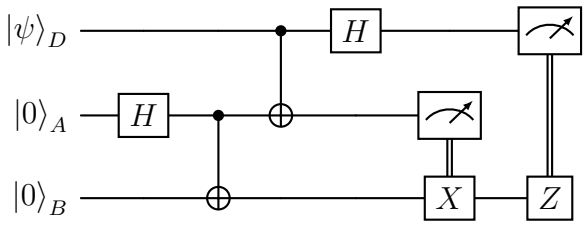

Figure 2.4: Circuit Diagram of Quantum Teleportation

Applying the CNOT gate to the two qubits that Alice has, we obtain the following.

$$\frac{1}{\sqrt{2}}\left(\alpha\left|0\right\rangle_D\left|00\right\rangle_{AB}+\alpha\left|1\right\rangle_D\left|01\right\rangle_{AB}+\beta\left|1\right\rangle_D\left|10\right\rangle_{AB}+\beta\left|1\right\rangle_D\left|01\right\rangle_{AB}\right) \tag{2.2.13}$$

Furthermore, the H-gate is applied as follows.

$$\begin{aligned}&\frac{\alpha}{\sqrt{2}}\frac{\left|0\right\rangle_D+\left|1\right\rangle_D}{\sqrt{2}}\left|00\right\rangle_{AB}+\frac{\alpha}{\sqrt{2}}\frac{\left|0\right\rangle_D+\left|1\right\rangle_D}{\sqrt{2}}\left|11\right\rangle_{AB}+\\&\frac{\beta}{\sqrt{2}}\frac{\left|0\right\rangle_D-\left|1\right\rangle_D}{\sqrt{2}}\left|10\right\rangle_{AB}+\frac{\beta}{\sqrt{2}}\frac{\left|0\right\rangle_D-\left|1\right\rangle_D}{\sqrt{2}}\left|01\right\rangle_{AB}\end{aligned} \tag{2.2.14}$$

$$\begin{aligned}=\frac{1}{2}&\left(\alpha\left|000\right\rangle+\alpha\left|100\right\rangle+\alpha\left|011\right\rangle+\alpha\left|111\right\rangle+\right.\\&\left.\beta\left|010\right\rangle-\beta\left|110\right\rangle+\beta\left|001\right\rangle-\beta\left|101\right\rangle\right)\end{aligned} \tag{2.2.15}$$

If we reorganize this again into Alice's qubit and Bob's qubit, we get the following

$$\begin{aligned}\frac{1}{2}\{&\left|00\right\rangle_{DA}\left(\alpha\left|0\right\rangle_B+\beta\left|1\right\rangle_B\right)+\left|01\right\rangle_{DA}\left(\alpha\left|1\right\rangle_B+\beta\left|0\right\rangle_B\right)+\\&\left|10\right\rangle_{DA}\left(\alpha\left|0\right\rangle_B-\beta\left|1\right\rangle_B\right)+\left|11\right\rangle_{DA}\left(\alpha\left|1\right\rangle_B-\beta\left|0\right\rangle_B\right)\}\end{aligned} \tag{2.2.16}$$

By measuring the state of Alice's qubit DA, we know that Bob's qubit B can be in any of four states. If Alice's measurement result is $|00\rangle$, Bob already has $\alpha|0\rangle+\beta|1\rangle$ states. If the measurement result is $|01\rangle$, we need to apply an X-gate to invert $|0\rangle$ and $|1\rangle$. If the measurement result is $|10\rangle$, the Z-gate should be applied because the phases are different. If the measurement result is $|11\rangle$, both phase-flip and bit-flip are occurring, so X-gate and Z-gate are applied. In this way, arbitrary qubits can be transferred.

Quantum teleportation is a powerful technique in that it can send an arbitrary quantum state between two parties as long as there is a high-fidelity Bell pair between them, regardless of the quality of the transmission channel. If one only needs

to send a quantum state, one can encode the quantum state in a photon and send it through an optical fiber, for example. Optical fibers cannot transmit signals with 100% efficiency. In classical communication, long-distance transmission is achieved by installing repeaters in the optical fiber and amplifying the attenuated signal. However, since quantum states are lost by amplification, the same method as in classical communication cannot be used. Bell pairs are generated in advance using optical fibers, and quantum states are sent by quantum teleportation using these Bell pairs. Unlike important quantum states, a Bell pair can be re-created many times, so failures are allowed. Once a high-fidelity Bell pair is successfully generated, quantum teleportation can be performed using it, enabling quantum state transmission independent of the quality of the optical fiber.

2.2.5 Entanglement Swapping

In order to achieve quantum teleportation over longer distances, a technique called entanglement swapping must be used to generate long-range Bell pairs. To put it differently, entanglement swapping can create a longer Bell pair by joining two Bell pairs together. Charlie's, and then the entanglement swapping creates a long-distance Bell pair between Alice and Bob.

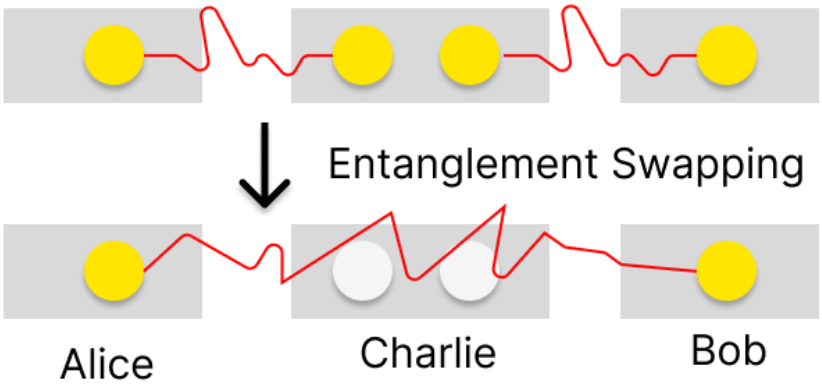

Figure 2.5: Entanglement Swapping

Fig. 2.6 is an entanglement swapping procedure. Let A be Alice's qubit, B be Bob's qubit, and C_1, C_2 be Charlie's qubits, respectively. In the circuit, the first step is to create a Bell pair between AC_1 and BC_2. Next, BSM is performed on Charlie's two qubits and the results are sent to Bob. Bob decides whether to apply the X-gate or the Z-gate according to the measurement results he receives. In this way, a Bell pair can be created between Alice and Bob.

From a different perspective, this circuit is equivalent to sending one of the qubits C_1 that consists of the Bell pair to B by quantum teleportation.

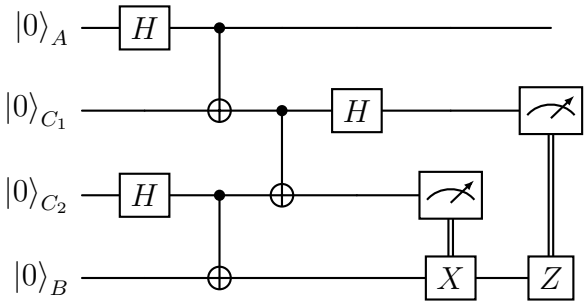

Figure 2.6: Entanglement Swapping Circuit

Entanglement swapping can thus extend the length of a Bell pair and repeated entanglement swapping can generate Bell pairs over longer distances. The generation of long-range Bell pairs extends the distance over which quantum teleportation is possible, allowing arbitrary quantum states to be sent around the world. For this reason, the generation and distribution of Bell pairs is one of the fundamental services of the quantum Internet.

2.2.6 Entanglement Purification

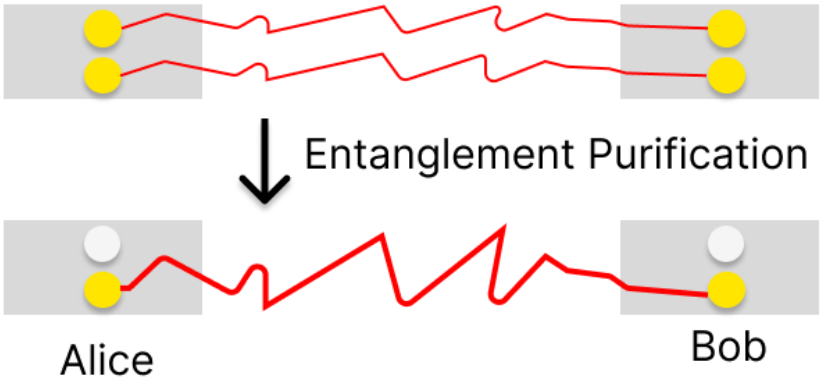

Figure 2.7: Entanglement Purification

Quantum entanglement purification is a method that consumes two Bell pairs to probabilistically generate a Bell pair with higher fidelity [13]. This is an important technique because the fidelity of the Bell pair used for quantum teleportation determines the quality of the data qubits after teleportation.

On the other hand, since entanglement purification is a stochastic operation, it may fail. In such a case, it is necessary to discard the Bell pair and generate the Bell pair again.

2.3 Applications

Quantum networks, which use quantum entanglement to connect multiple quantum computers, enable applications that are impossible to realize on classical networks. Various applications to be built on quantum networks have already been proposed and physically experimented with.

While the current security of the classical Internet is based on the computational security of classical cryptography, quantum cryptography enables classical communications with information-theoretic security [2, 3].

Blind quantum computation is a method of using a remotely located quantum computer while keeping the contents of the computation secret [14, 15]. The client can delegate the quantum computation while keeping the input and output completely unknown to the server, and can even detect server cheating.

Distributed quantum computation uses a quantum network to connect many quantum computers, enabling quantum computation with more qubits [16]. The number of qubits that can be included in a single quantum computer is currently in the hundreds, but more qubits are needed to perform practical quantum computations [17]. Distributed quantum computation is one of the leading approaches to unlocking the power of further quantum computation.

Quantum sensing uses its quantum properties to obtain measurements of physical quantities with a precision that cannot be achieved with classical [18]. It has also been proposed to further improve accuracy by using distant quantum entanglement [4, 19, 20].

2.4 Quantum Network

A quantum network is a network of quantum computers connected to each other by links. The quantum network enables quantum computers to exchange and use quantum states for quantum computation.

In quantum computation, there is a restriction called the no-cloning theorem 2.2.2, which means that arbitrary quantum states cannot be copied. Therefore, data cannot be transferred in the same store-and-forward, retry on loss way as in a classical computer. Therefore, quantum teleportation must be used to transfer valuable quantum data.

2.4.1 Quantum Node

A quantum node consists of an interface to the outside world to form a quantum link, a mechanism to generate a Bell pair in a quantum link, and a classical controller to manage and use the generated Bell pair. Fig. 2.8 shows the icons proposed in Ref. [8]. In this thesis, these icons are used to draw a diagram of the network.

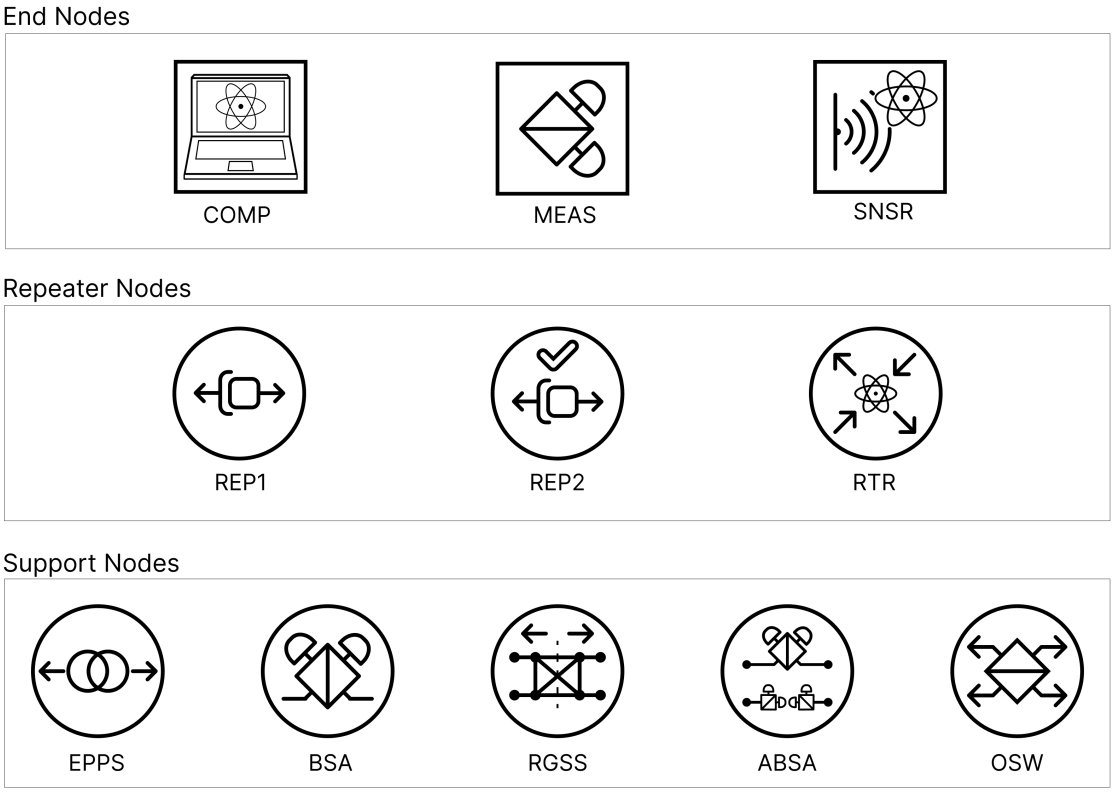

Figure 2.8: Quantum Network Component Icons proposed in Ref. [8]

End Nodes: End Nodes are nodes that can run quantum applications. This node initiates a connection by making a request to generate a Bell pair if necessary. Alternatively, it receives the request and establishes a connection. Finally, the generated end-to-end Bell pair is used by the quantum application running at this node.

COMP: COMP is a type of end node that can perform quantum computation, such as blind quantum computation.

MEAS: MEAS is an end node that only measures the generated quantum entanglement, unlike COMP, and is used for quantum key distribution, for example.

SNSR: SNSR is an end node for quantum sensing, which, unlike MEAS, is used to probe external objects using the quantum entanglement generated.

Repeater Nodes: A repeater is a node that connects two or more nodes; the role of the repeater is to extend quantum entanglement by entanglement swapping rather than amplifying signals as in classical networks.

REP1: REP1 is a 1G [21] repeater connected to two nodes, with entanglement swapping to extend quantum entanglement and entanglement purification for error management.

REP2: REP2 is a 2G [21] repeater and connects to two nodes; in addition to the functions of REP1, it can handle logical qubits with error correction applied.

RTR: A router functions similarly to REP1 and REP2 but differs in that it can connect to three or more nodes. A router placed at the border of a network is called a gateway or gateway router.

Support Nodes: support nodes are basically located along the links between the repeater node and end node introduced so far and support the generation of quantum entanglement.

EPPS: An entangled photon pair source is connected to two end nodes or repeater nodes and distributes two photons in a quantum entangled state to each node.

BSA: The Bell state analyzer is connected to two end nodes or repeater nodes and performs a Bell-state measurement on two photons sent from both nodes. By returning the measurement results to the two end nodes, a Bell pair can be generated between the memory qubits of the two nodes.

RGSS: repeater graph state source is used to generate many-body quantum entanglement [22, 23].

ABSA: advanced Bell state analyzer is a feature-rich BSA node used for repeater graph state [23]

OSW: optical switch is a support node for switching the destination link without measuring photons.

2.4.2 Quantum Link

A quantum link connects two quantum nodes to form a network. The most important role of the quantum link is to assist in the generation of a link Bell pair between the nodes at both ends of the connection. Protocols and architectures have been proposed

for quantum links as well [10, 24, 25]. Quantum links include the placement of support nodes such as BSA and EPPS. The implementation of quantum links is also deeply dependent on the capabilities and schemes of the nodes connected to both ends of the link.

Experiments have already been done to generate quantum entanglement as a quantum link, sending photons encoded with quantum states through telecom-wavelength optical fibers and free space [26, 27].

2.4.3 Routing

Routing is the operation of choosing a path. In a quantum network, this means choosing which nodes and links to go through to generate long-distance quantum entanglement with the desired node. A number of routing algorithms for quantum networks have been proposed [28, 29, 30].

Various routing protocols have been proposed for classical networks, but it is difficult to apply them directly to quantum networks. In order to generate long-distance quantum entanglement, there are many factors to be considered, such as the time required to generate the quantum entanglement at the link, the generation probability, its fidelity, the time required for entanglement swapping, and the success probability, etc. Moreover, the decoherence of a qubit occurs as time passes.

2.5 Quantum Network Simulators

Various quantum network simulators have been developed to investigate the characteristics of quantum networks. Simulators model physical phenomena and can be run on commercially available classical computers. This allows the behavior of quantum networks to be investigated without the need for expensive facilities for physical experiments. Even if it is not yet possible with current technology, it is possible to investigate its behavior and properties by putting it into code on a simulator. Furthermore, even events that are difficult to measure in actual experiments can be learned as a result of following the implemented physical model.

NetSquid [31] is a quantum network simulator with detailed physical models available, covering a wide range from the physical layer to the application layer. SimulaQron [32] is a quantum Internet simulator designed to support the development of applications running on top of the Internet; SimulaQron focuses on the application layer rather than the physical layer and allows applications to be written in Python. Also currently SimulaQuron integrates NetQASM [33], a set of low-level instructions for writing quantum network applications. SQUANCH [34] is a simulator for distributed quantum computation based on quantum networks. SimQN [35] is a generically designed quantum network simulator on which various network architectures can be implemented. QuNetSim [36] is a quantum network simulator with many existing quantum network protocols and applications implemented on it that

can be easily tried. In addition, QuNetSim has a high degree of flexibility regarding routing, making it suitable for verifying routing protocols. QuISP [37] is a simulator for large-scale quantum networks that implements a RuleSet-based protocol, which will be introduced in the next section. SeQUeNCe [38] is a simulator based on the RuleSet-based protocol and focuses on the correctness of the physical layer rather than large networks.

2.6 RuleSet-based Quantum Repeater Networks

The RuleSet-based protocol is a protocol for quantum networks proposed in Ref. [10, 39]. A RuleSet-based quantum repeater network uses this protocol to distribute Bell pairs. The protocol sits on top of the link layer and aims to use the Bell pairs generated by the link layer to generate Bell pairs with high fidelity over long distances. The protocol is designed to minimize synchronous classical communication between nodes and to allow each node to operate independently in a distributed manner and generate globally useful results.

As shown in Fig. 2.9, there are three main phases in the RuleSet-based protocol. One is the connection setup phase, in which a request gathers path information, and the Responder generates and distributes RuleSets to each node. The second is the RuleSet execution phase, in which the nodes execute the given RuleSet and generate an end-to-end Bell pair between the EndNodes. Finally, there is the teardown phase in which RuleSets are removed and the allocated resources, such as qubits, are released.

2.6.1 Two-pass Connection Setup

The two-pass connection setup distributes RuleSets for Bell pair generation. Fig. 2.10, EndNode A wants to generate a Bell pair between EndNode A and EndNode E. We refer to EndNode A and EndNode E as Initiator and Responder, respectively. This is because, unlike classical communication, the primary communication resource, the Bell pair, is symmetric, so calling them sender and receiver would cause confusion. In the first pass, the initiator generates a request and sends it to the next node, repeater B. Repeater B adds the link information between A and B to the received request and sends it to the next repeater, C. In this way, the request collects information on each link, and finally, EndNode E receives the request that has the path information. EndNode E, the Responder, decides how each node should behave considering the information of each link and generates RuleSets, and distributes the generated RuleSets to all nodes participating in the communication.

The Initiator generates a connection setup request to initiate the connection to create Bell pairs between the Initiator and the Responder. The request contains the addresses of the Initiator and the Responder and the quantum state to be generated between the Initiator and the Responder, like the number of the Bell pairs.

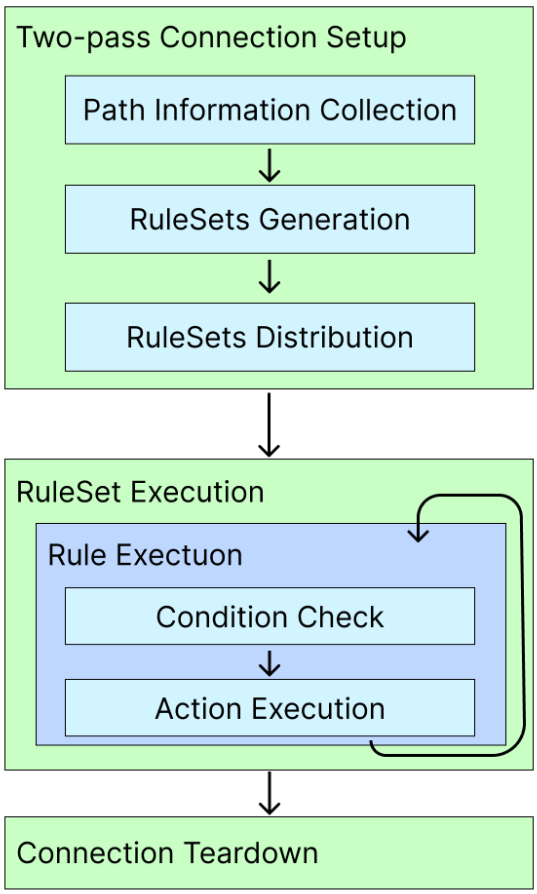

Figure 2.9: Three main phases in RuleSet-based protocol.

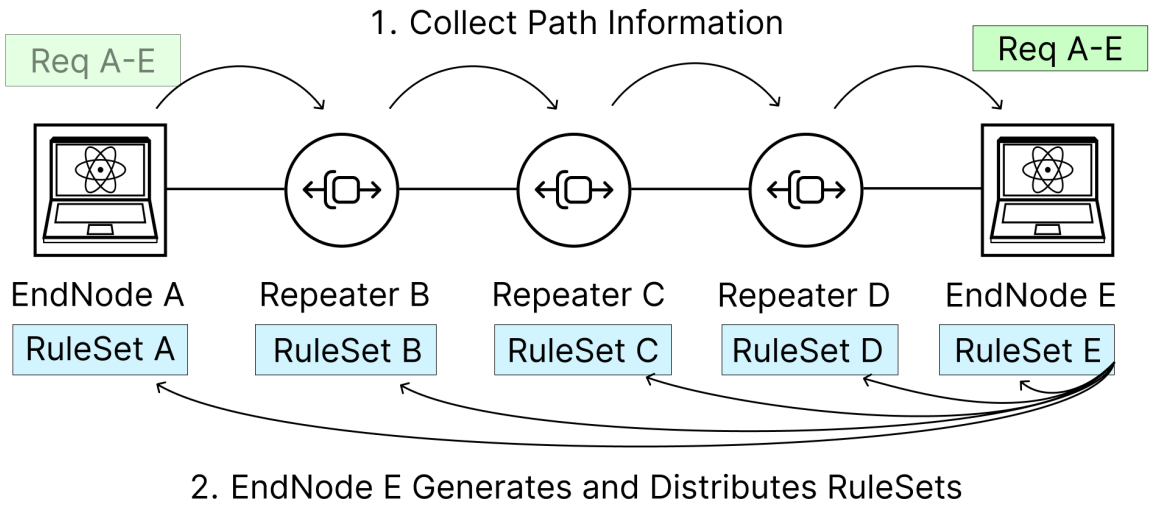

Figure 2.10: Two-pass connection setup workflow. EndNode A is an initiator, and EndNode E is a responder. Connection Setup Request packet collects path information, and the responder generates RuleSets for nodes in the path.

The most important role of the Responder is to generate RuleSets. The Responder receives connection setup requests, generates RuleSets, and distributes them to the nodes in the path. The connection setup request contains information about the nodes in the path. Based on this information, the Responder decides in which order to perform the entanglement swapping and where and in which Bell pair the purification should be performed, and generates RuleSets. All nodes in a path, including the Initiator and the Responder that make up the connection, decide their behavior based on the RuleSets generated at the Responder.

The paths and methods for distributing RuleSets do not matter because the topology of the quantum links in this network does not necessarily match the topology of the classical links. Therefore, in the figure, the arrows indicating the distribution of RuleSets are connected to each node. On the other hand, the first pass requires the request to relay each node in turn.

After distributing RuleSets to each node, each node operates according to the received RuleSets, creates Bell pairs between links, performs entanglement swapping at the appropriate time, and finally creates end-to-end Bell pairs as many times as necessary. The RuleSets are executed autonomously and distributed, and end-to-end Bell pairs can be efficiently generated with a minimum of classical communication.

Various parameters have been considered for the quantum state to be generated between the Initiator and Responder, including the fidelity of the Bell pair, throughput, and number of Bell pairs. This is currently under discussion, and appropriate

parameters will be determined by future proposals for quantum network use cases, etc. The design of the parameters included in the request may have a significant impact on the ease of creating applications and generating RuleSets in the Responder. However, since this is not the central topic of this study, we will only deal with the number of Bell pairs as a quantum state parameter to be included in the request on our simulator.

Why Two-pass Connection Setup?

The number of classical communications between each node can be reduced by generating RuleSets in a single location. For example, if the number of links constituting a connection is 3 or more, it is non-trivial to determine in which node and in which order the entanglement swapping should be performed. However, this order has been found to have a significant impact on the fidelity and throughput of the end-to-end Bell pair . Therefore, it is necessary to determine the nodes that make up the connection at the time the connection is requested, and to determine the order of entanglement swapping based on the performance of each node and link. If each node tries to determine the order of entanglement swapping in an autonomous and decentralized manner through classical communication, the number of classical communications will increase. Another possible strategy is to swap from the point where a link bell pair is generated, but this will lead to incorrect swapping for long connections. The two-pass connection setup solves these problems.

2.6.2 RuleSet

A RuleSet is a restricted program that can be distributed so that each node on a quantum network can operate in an autonomous decentralized manner. Each node on the path can generate a long-distance Bell pair simply by operating according to a given RuleSet.

Since the RuleSet is generated in response to connection requests, it has the flexibility to adapt to different situations. For example, the Responder can specify the order of swapping for optimal fidelity or Bell pairs generation rate or apply purification if they can expect the low fidelity Bell pair in the link depending on the characteristics of nodes or link in the path. The RuleSet and two-pass connection setup mechanisms force this planning to be done in advance rather than when the actual creation of the bell pair.

Fig. 2.11 depicts the structure of a RuleSet. A RuleSet consists of one or more Rules and a termination condition, and a Rule consists of a Condition and an Action. Clauses are defined as Condition Clauses and Action Clauses, respectively, which define concrete conditions, actual quantum operations, classical operations, and so on. The combination of these Clauses defines the behaviors of the Condition and Action.

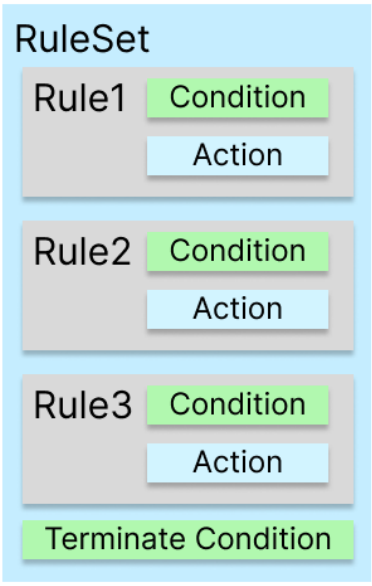

Figure 2.11: Structure of a RuleSet. In this case, the RuleSet contains three Rules. Each Rule has one Condition and Action. Responder generates and distributes RuleSet for each node in the path.

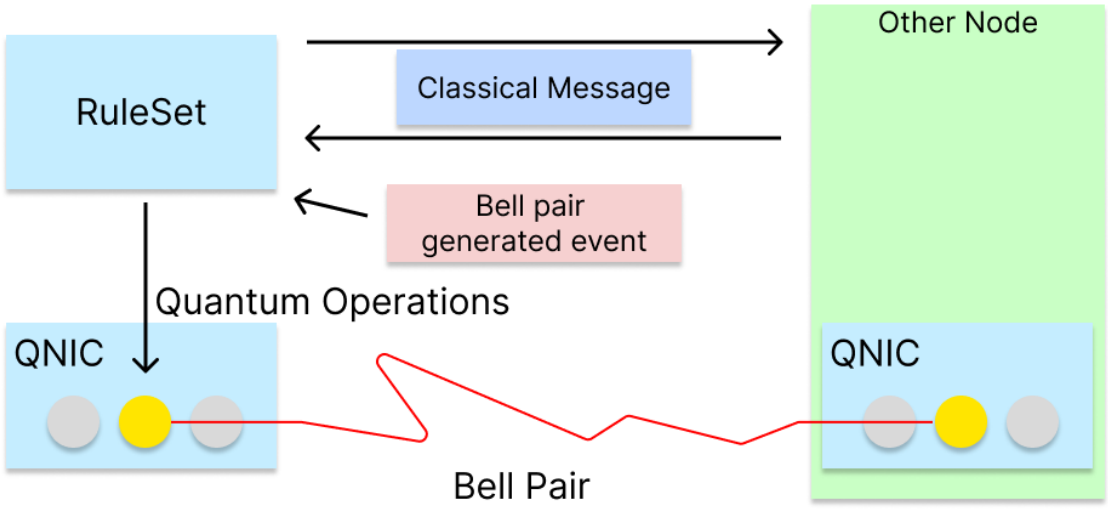

Figure 2.12: RuleSet interacts with classical messages and qubits.

RuleSet is designed as event-driven, and all operations are triggered by receiving an event. Fig. 2.12 shows how a RuleSet interacts with its components. The distributed RuleSet is executed upon receiving a classical message from another node and a Bell pair generated event from the node itself; the RuleSet can perform quantum operations on qubit or send classical messages to another node, depending on the Actions in the RuleSet. These operations can be used to construct algorithms such as entanglement swapping and purification.

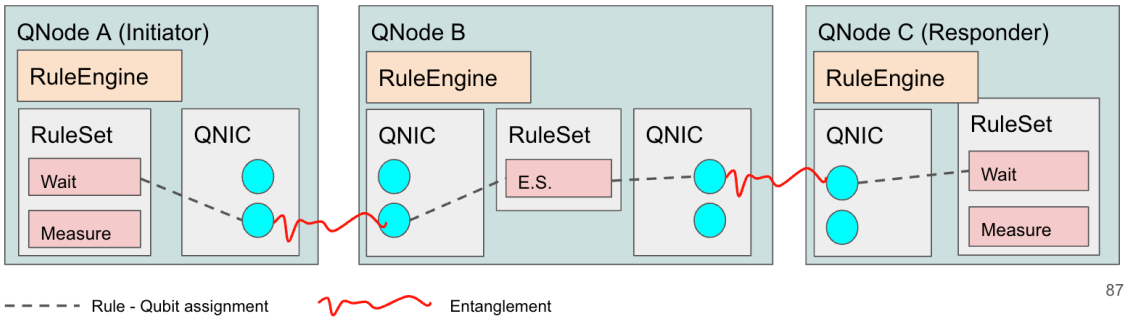

Figure 2.13: Each node received the RuleSet, and the link Bell pairs are generated. The Bell pairs are assigned to the Rules by RuleEngine. The RuleSets contain one or two Rules. Skyblue circles represent qubits.

Let's see the typical RuleSets example for the case of entanglement swapping. Fig. 2.13 shows that the RuleSet is distributed to each node, a link Bell pair between AB and BC is generated, and it is assigned to a Rule.

First, the RuleSet of node B receives a Bell pair generation event when a link Bell pair is generated. The RuleSet of node B checks the Condition of the Rule, and if the condition is met, it executes the Action associated with the Rule. Here, the two conditions for entanglement swapping are the existence of a Bell pair with node A and the existence of a Bell pair with node B.

Since the Condition has been fulfilled at node B, the next Action is performed. Here, in order to perform entanglement swapping, node B performs a Bell state measurement and sends the measurement results to node A and node C as a classical message. This classical message contains the RuleSet ID, so that even if there are multiple RuleSets in a node, they will be handled appropriately.

Nodes A and C receive the classical message and check the RuleSet at each node. If the classical message meets the Rule's Condition, the associated Action is executed. Here, if the classical message is the result of the previous Bell state measurement, the Condition is fulfilled, the quantum gate is applied to the local qubit according to the content of the message, and the node obtains the expected Bell pair.

Finally, a Bell pair is generated between nodes A and C, and each node notifies itself of the new Bell pair generation event. In this case, since it is already a Bell pair between AC, it is handled by a different Rule. In this figure, there is a measurement

Rule whose Condition is to check for a Bell pair between AC, and its Action is to perform the measurement.

In this way, RuleSets can operate autonomously and in a decentralized manner. Even connections consisting of many nodes can be performed in this iterative manner. The order of the entanglement swapping can be controlled by the RuleSet configuration, and purification can be performed as needed.

2.7 Quantum Internet

Quantum Internet [5, 40] is a quantum communication network that will spread worldwide. The quantum Internet is expected to be a global infrastructure that provides a Bell pair between any two points and allows people around the world to benefit from quantum computing. Currently, the expected benefits of the existence of the quantum Internet include the possibility of information-theoretically secure communication and computation, the ability to connect quantum computers to each other and use them as a larger quantum computing resource, and quantum sensor networks.

The quantum Internet will not replace the classical Internet we use today. As we have already mentioned, the quantum Internet cannot be realized without the classical Internet. Whether quantum teleportation or entanglement swapping, quantum communication would not exist without classical communication. Therefore, a quantum network can coexist with a classical network.

However, quantum communication has properties that are very different from those of classical communication, and simply applying existing protocols and architectures for classical networks to quantum networks will not work. In addition, the classical Internet has not always been implemented and operated in an optimal way, and some aspects of its history have not been designed in a sophisticated manner.

Reviewing the many design decisions and improvement attempts in the classical Internet, the design of the quantum Internet needs to be done without wasting the lessons learned from them. One of the lessons to be learned from the classical Internet is that it is difficult to fundamentally change and update a technology once it has become widespread.

2.8 Internetworking

Creating a single giant quantum network is difficult when providing quantum communications worldwide. This is because managing paths and allocating and scheduling resources required for each communication is difficult to scale. Therefore, in the classical Internet, networks are connected to each other to reduce the size that must be managed. A method to realize the quantum Internet by applying this idea to quantum networks has already been proposed.

Internetworking is essential for building a global quantum Internet and is a technique for connecting quantum networks to each other. This technique defines the boundaries of the quantum network and hides the details within the sub-networks from the global network. The load on the nodes of the quantum Internet is then distributed since the higher-level network no longer needs to manage resources and paths within the sub-network, making it possible to realize a large-scale quantum Internet.

Internetworking is also important for connecting different networks and handling heterogeneous networks. This requires that the protocols in the inter-layer are sufficiently abstract.

2.9 Recursive Network Architecture

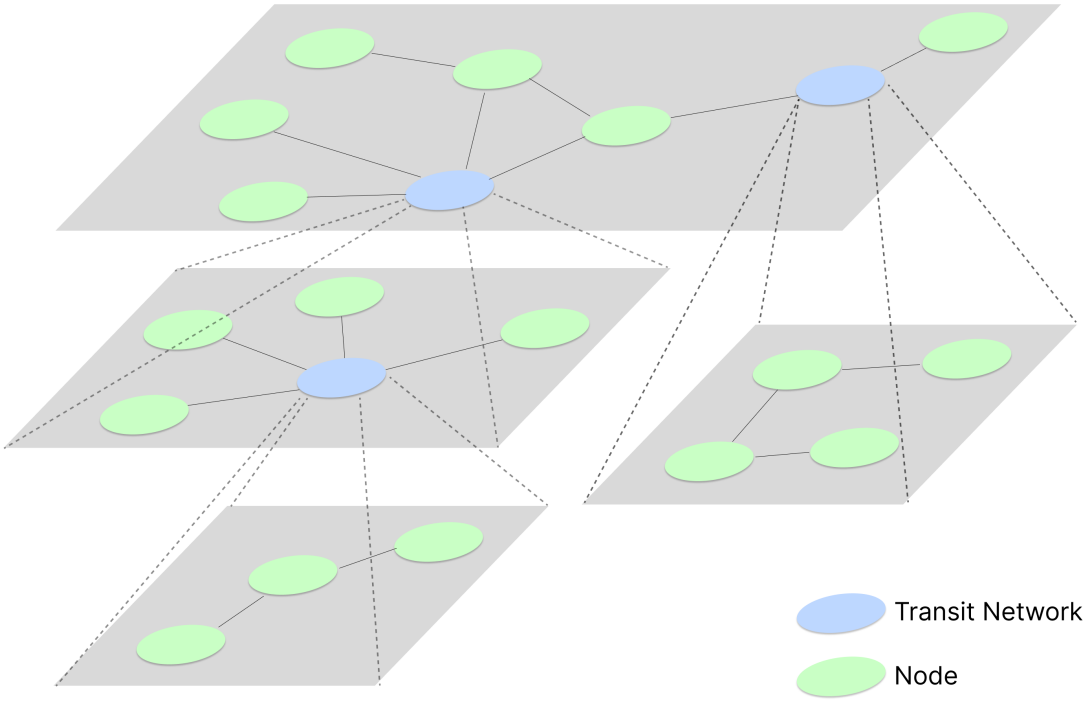

Figure 2.14: Recursive network architecture allows embedding a subnetwork as a node in a network recursively. This embedding leads to the new layer and hides its detail from the higher layer.

Recursive network architecture (RNA) [41] is a completely new architecture for the classical Internet. As shown in Fig. 2.14, the RNA embeds a subnetwork within the network, leading to a new layer. It is a mechanism that provides necessary services based on requests dynamically.

The recursive network structure allows subnetworks to be created as needed, providing a flexible and scalable network. In practice, during the course of operation, the network grows organically as users demand, so it is not convenient if the number of embedding as a subnetwork is limited as an architecture. Maximum scalability can be achieved by designing the subnetwork to be configured as many times as technically possible.

The **metaprotocol** is one of the core ideas of RNA, which eliminates the reimplementation of the same idea at each layer. Many current protocols reimplement numerous features such as encryption, policy, and retransmission control. A metaprotocol allows for the reuse of each function by dynamically constructing a protocol layer that collects the necessary functions.

The relationship between networks and protocol layers is slightly different in RNA than in the existing TCP/IP protocol stack concept, where each network constitutes a single layer. Each time a packet enters a sub-network, the layer goes one level deeper, and information to be processed in that layer is added to the packet.

Layers for a recursive network, containing metaprotocols, can be considered as a single layer. On the other hand, it is also possible to connect networks that operate with different mechanisms, in which case the functions of each layer of the network will operate. When looking at situations where each layer provides a different function, it should not be considered a single layer.

The TCP/IP model and the OSI reference model in the classical Internet and RNA differ significantly in that each protocol layer is static or dynamic. This difference is misleading when expressed as a protocol stack. In the case of a stack containing dynamic protocols, i.e., RNA, the number and behavior of the recursive network layers depend on the route through which the connection is formed. The behavior of each of the recursive network layers changes depending on the network traversed, and the order of the protocol layers changes depending on which network is traversed and in which order. It is necessary to choose how to represent the recursive network layers according to what you want to express by the protocol stack.

Another example that uses a recursive network structure is the Recursive Inter-Network Architecture (RINA) [42]. RINA is designed under the philosophy that all communication is Interprocess communication (IPC).

2.10 Quantum Recursive Network Architecture

Quantum recursive network architecture (QRNA) [9, 8] is an architecture that introduces RNA to quantum networks. While RNA was an architecture and protocol for classical data exchange, QRNA deals with an architecture and protocol for the generation of quantum entanglement involving two or more qubits. In this study, we implemented the QRNA on a quantum network simulator.

QRNA focuses on the recursive structure of the network and the fact that the layers are nested again and again. This recursive structure, similar to RNA, allows

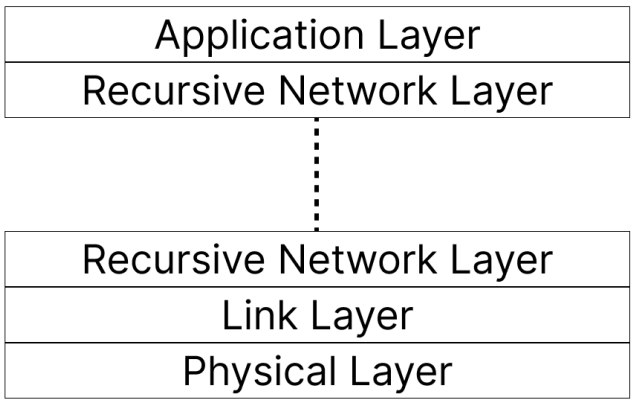

Figure 2.15: QRNA Protocol Stack. The dotted line shows that more than one recursive network layer exists.

the network to be partitioned as needed without the development of new protocols, increasing the scalability of the quantum network.

The idea of applying QRNA to the aforementioned RuleSet-based protocol has been proposed by Van Meter *et al.* [8].

Fig. 2.15 shows the protocol stack in QRNA and how each layer is stacked. There is more than one recursive network layer; the number of recursive network layers depends on which network a given request goes through and at which point the protocol stack is observed. This is because the recursive network layers are determined dynamically when establishing a connection.

From the protocol stack perspective, the flow of end-to-end Bell pair generation is described as follows. Bell pairs between links generated by the physical layer and the link layer are provided to the underlying recursive network layer, which uses entanglement swapping to generate longer Bell pairs from the link Bell pairs. The generated Bell pair is passed to the upper recursive network layer, where it is used to generate a longer Bell pair, and the process is repeated to generate an end-to-end Bell pair at the top-level recursive network layer, The recursive network layer is then passed to the application layer.

Chapter 3

Problem Definition and System Requirements

3.1 Problem Definition

There have already been proposals for internetworking protocols and architectures for interconnecting quantum networks [9, 8, 43], but none have been implemented on simulators or in a concrete form, such as a detailed protocol definition. QRNA is one of the architectures for internetworking, but it has not been proposed in a concrete form either.

The method of interconnecting quantum networks is an important topic for the realization of the quantum Internet. This method will affect the design of the protocols for the quantum networks that make up the quantum Internet. It also affects the extensibility of the quantum Internet in terms of functionality.

The problem addressed in this thesis is that since QRNA has not been actually implemented, it is not known whether QRNA is a suitable architecture for the quantum Internet in the first place. Therefore, we will identify design issues, implement QRNA on a simulator, and examine whether the implemented protocol is suitable for the quantum Internet.

In this thesis, QRNA is tested against a RuleSet-based quantum network. Ideally, various types of quantum network protocols should be applied to QRNA to verify each other. However, since only RuleSet-based protocols have been implemented as open-source simulators, this thesis focuses on the combination of RuleSet-based protocols and QRNA.

When considering the quantum Internet as the world's only global network, it is important to support future applications, allow for an increasing number of quantum nodes, and continue to provide the service of end-to-end Bell pair generation. Once a network is public and widely deployed, it is very difficult to change the protocols and architecture that make up the network.

Therefore, it is valuable to implement QRNA on a simulator to explore its feasi-

bility. Compared to verification on actual hardware, it is less expensive and quicker, and various cases can be verified. On the other hand, since the implementation is on a simulator, it is necessary to be careful not to make the implementation impossible to realize.

3.2 System Requirements

In this section, we describe the system requirements for the protocols and architectures that comprise the quantum Internet in terms of scalability and management.

There are various requirements for the protocol and architecture of quantum Internet. For example, the service provider must be stable and uninterrupted even as the number of participating nodes and requests increase; the participating nodes must be manageable, different types of nodes can participate, and networks with different mechanisms can interconnect.

Many of these requirements are complex. Network throughput and latency are highly dependent on multiplexing and resource allocation strategies, routing, and swapping strategies. On the other hand, QRNA itself connects networks to each other, and the impact of QRNA itself on throughput and latency is not dominant. QRNA allows networks to be connected to each other or, from a different perspective, to be partitioned into multiple smaller networks from a single large network. This has a significant impact in that it makes it easier to manage the nodes participating in the network and to connect different networks.

Here we break down the various requirements for the quantum Internet into two aspects of scalability and manageability for implementing and evaluating QRNA.

3.2.1 Scalability

Here, scalability refers to the number of nodes that can participate in the quantum Internet and the number of requests that the quantum Internet can process. Since QRNA has the greatest impact on the number of nodes that can participate and the number of requests that can be processed, we will focus our discussion on those two points.

The protocols and architectures that make up the quantum Internet should be able to achieve high scalability. As one indicator of high scalability, we can refer to the classical Internet. Ref. [44] shows the number of users and devices connected to the classical Internet. As of 2018, the number of Internet users was 3.9 billion and is projected to increase to 5.3 billion by 2023. The number of devices was 18.4 billion in 2018 and is projected to grow to 29.3 billion by 2023. Anticipating the widespread adoption of the quantum Internet, a mechanism to accommodate such a large number of nodes will be necessary.

The number of nodes or hops on the communication path is an important indicator when generating end-to-end Bell pairs using the quantum Internet. To generate long-

range end-to-end Bell pairs, entanglement swapping is essential. However, Shchukin *et al.*'s algorithm for determining the optimal order of entanglement swapping has complexity that is exponential in the number of hops [45]. As the quantum Internet grows, the classical processing time to process a request will increase rapidly.

3.2.2 Manageability

The management aspect of the quantum Internet deals primarily with autonomy and security. The quantum Internet consists of many different multi-layered networks, each of which must operate autonomously and be capable of connecting diverse types of networks.

Hiding the inner network by internetworking also improves security. For example, it becomes more difficult for an attacker to learn about nodes that hold sensitive information inside the network from outside the network.

Chapter 4

Proposal

4.1 Design Considerations

Many design considerations must be taken into account when implementing internetworking. In order to actually implement it, all of these points must be decided, even if only temporarily. This section discusses some of the most important points.

4.1.1 Link Recursion and Node Recursion

In this study, we introduce the concepts of link recursion and node recursion. In QRNA, link recursion abstracts a network as one link and the nodes on both ends. Conversely, in node recursion, a network is abstracted as a node (see Fig. 4.1). The original RNA is for classical networks. In the store-and-forward approach of classical networks, it is not crucial whether node recursion or link recursion is used. However, the difference between link and node recursion significantly impacts quantum networks.

The difference between the two is particularly noticeable in the extent to which the upper layer must be aware of the topology of the lower layer. The important thing is to have appropriate modeling that can represent the important features of the lower layer.

For example, there is a difference between node and link recursion when the upper layer decides in what order Entanglement Swapping and Entanglement Purification should be performed. Shchukin *et al.*'s algorithm for finding the optimal Entanglement Swapping strategy uses the generation probability of the Bell pair and the success probability of the entanglement swapping [45]. To work well with this algorithm, we believe link recursion is more suitable because it models link-by-link behavior rather than node-by-node behavior.

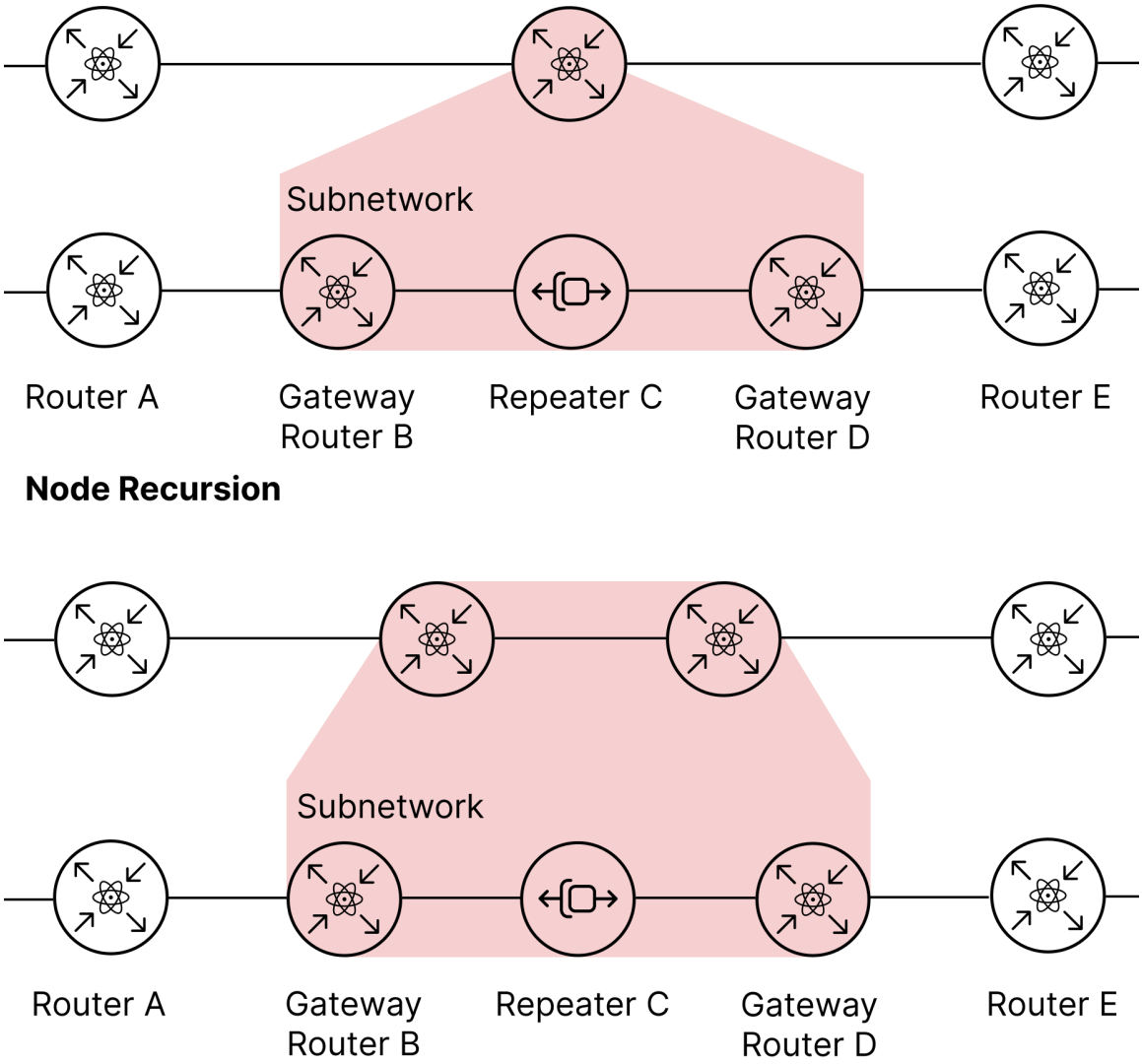

Figure 4.1: Node Recursion (top) and Link Recursion (bottom). Pink shadings represent subnetworks that consist of three nodes. The nodes (white) outside the subnetwork cannot know the detail of the subnetworks. In QRNA, we can apply this structure recursively to build a huge quantum network.

4.1.2 Inter-layer Resource Promotion

Naturally, information must be exchanged between the upper and lower layers. Resources like Bell pairs or multipartite quantum entanglement generated at one layer are provided to the upper layers and finally passed to the application at the highest layer. This promotion of resources between layers is performed by the gateway router. Therefore, the gateway router needs to be implemented according to the combination of network protocols used in the upper and lower layers.

Classical communication with another node or a node in an external network should be avoided when promoting resources between layers. Ideally, resource promotion should be accomplished only by operations local to the node.

However, since a resource is a quantum entanglement across two or more nodes, different nodes in a network are involved. Therefore, depending on the design of the quantum network protocol, there may be situations where classical communication with different nodes is unavoidable during resource promotion. Also, it is necessary to take care that the promoted resources are not used for different communications.

4.2 QRNA with RuleSet-based Protocol

Fig. 4.2 shows an overview of a two-pass connection setup with internetworking. EndNode A is an Initiator and wishes to share a Bell pair with EndNode E. Nodes B, C, and D are an embedded sub-network on this path, and these three nodes are regarded as a single link or node by the higher layer nodes.

To realize internetworking using RuleSet, generating a new request for a different layer is necessary when a request is passed to a different layer in order to hide the details of a subnetwork and increase autonomy. Here, gateway router B receives a request A-E (green) at the higher layer, creates a new request B-D (blue), and puts the request A-E inside the new one. Another gateway router, D, receives the request B-D and retrieves the request A-E. Then, based on the path information in the subnetwork, information about the link between B and D is included in the request A-E, and the request A-E is sent to E. Gateway router D needs to keep the information about the request B-D. In this way, request A-E is finally received by Responder E in the higher layer, and RuleSets are generated. The generated RuleSet is distributed to each node on the path. At this time, node D recognizes the subnetwork as a single link or node.

The network gateway node D functions as a special Responder, generating and distributing new RuleSets for B, C, and D based on the received request B-D and the RuleSet generated by node E.

In this way, the higher layer can treat the lower layer network as a link or node without knowing the detailed behavior at the lower layer. This connection setup sequence can be realized by extending the two-pass connection setup.

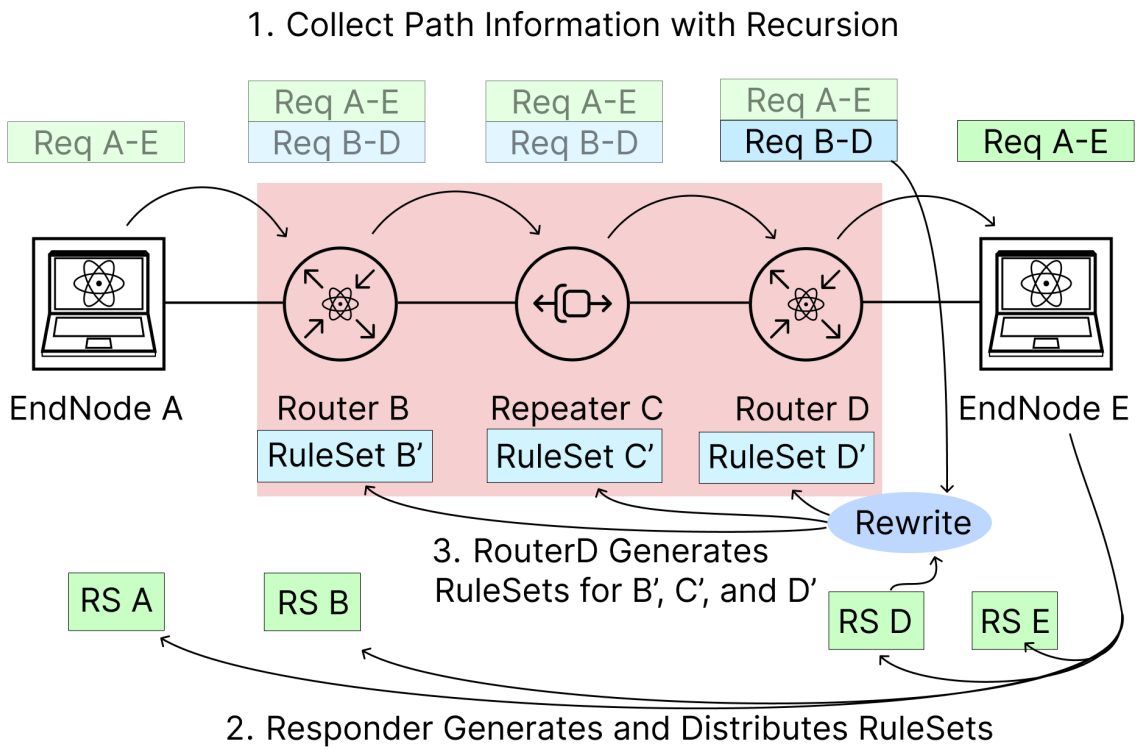

Figure 4.2: Two-pass connection setup flow with internetworking on a simplified linear network. Green boxes represent messages in the upper layer, blue boxes represent messages in the lower layer, and pink shading represents the subnetwork.

4.2.1 RuleSet Rewriting for Internetworking

The gateway Responder generates RuleSets for nodes in the network based on the request in the network and the RuleSet generated by the global Responder. The Rewriting algorithm can be divided into four types, depending on whether the RuleSet is divided into higher and lower layers or whether link or node recursion is employed. The subnetwork RuleSet generated by Rewriting should behave as a link or node as expected by the higher layer RuleSet.

Same RuleSet IDs vs. Different RuleSet IDs

RuleSet ID is an ID to identify RuleSets and is different for each connection. The RuleSets distributed to satisfy a given request have a common RuleSet ID determined by the Responder. If there are multiple different requests, the RuleSet for each must have a different ID. So, when recursive requests are generated in QRNA, should the RuleSet for each request have a shared ID or a different ID?

It is possible to assign different IDs to the RuleSet for the higher and lower layers, i.e., to treat the connection as separate or, conversely, to use the same RuleSet ID. This depends on the implementation of the gateway Responder.

If identical IDs are used, the higher layer RuleSets of the gateway Initiator and gateway Responder are rewritten to generate new RuleSets for the nodes in the subnetwork. The advantage of this is that the procedure that processes the RuleSet itself does not need to be extended. On the other hand, there is a possibility that unknown RuleSets cannot be handled in order to rewrite the higher layer RuleSet. This can be a factor that hinders the extensibility of the RuleSet.

The advantage of using different IDs is that this can be easily achieved by simply generating and distributing a lower layer RuleSet without changing the higher layer's RuleSet. On the other hand, passing a Bell pair generated in the lower layer to the RuleSet in the higher layer is necessary. For this purpose, it is necessary to extend the PROMOTE of the Action clause proposed in Ref. [8] to be able to pass qubit to another RuleSet. This extended PROMOTE is still a node-local operation.

Link Recursion vs. Node Recursion

In link recursion, the Responder of the higher layer recognizes the Initiator and Responder of the subnetwork and generates a RuleSet for each of them. On the other hand, in node recursion, the sub-network is treated as a single node, so the Responder of the higher layer generates a single RuleSet for the sub-network.

Node recursion requires swapping link Bell pairs with nodes outside the subnetwork, and swapping can be done at any node on the subnetwork path. However, if a node other than the gateway node swaps Bell pairs with a node outside the subnetwork, the gateway node needs to rewrite the destination of the Pauli frame

appropriately in order to send the Pauli frame to the outside of the subnetwork. Furthermore, since the two-pass connection setup is designed to collect link information rather than node information, it is necessary to find a way to include the impact of intermediate nodes in the subnetwork in the request.

In link recursion, only the gateway node performs classical communication with the node outside the subnetwork. The node inside the subnetwork only performs Bell pair generation between the gateway Initiator and Responder. Since the subnetwork is treated as a single link, the speed of Bell pair generation in the subnetwork can be embedded in the request as link information without major changes to the two-pass connection setup.

4.3 Four Patterns of QRNA Implementations

As mentioned in the previous section, integrating QRNA into a RuleSet-based protocol requires two major decisions: whether to use link recursion or node recursion and whether to use the same or different RuleSet IDs inside and outside the subnetwork. Four possible patterns of configuration are possible by combining each of these. Here we will present the four.

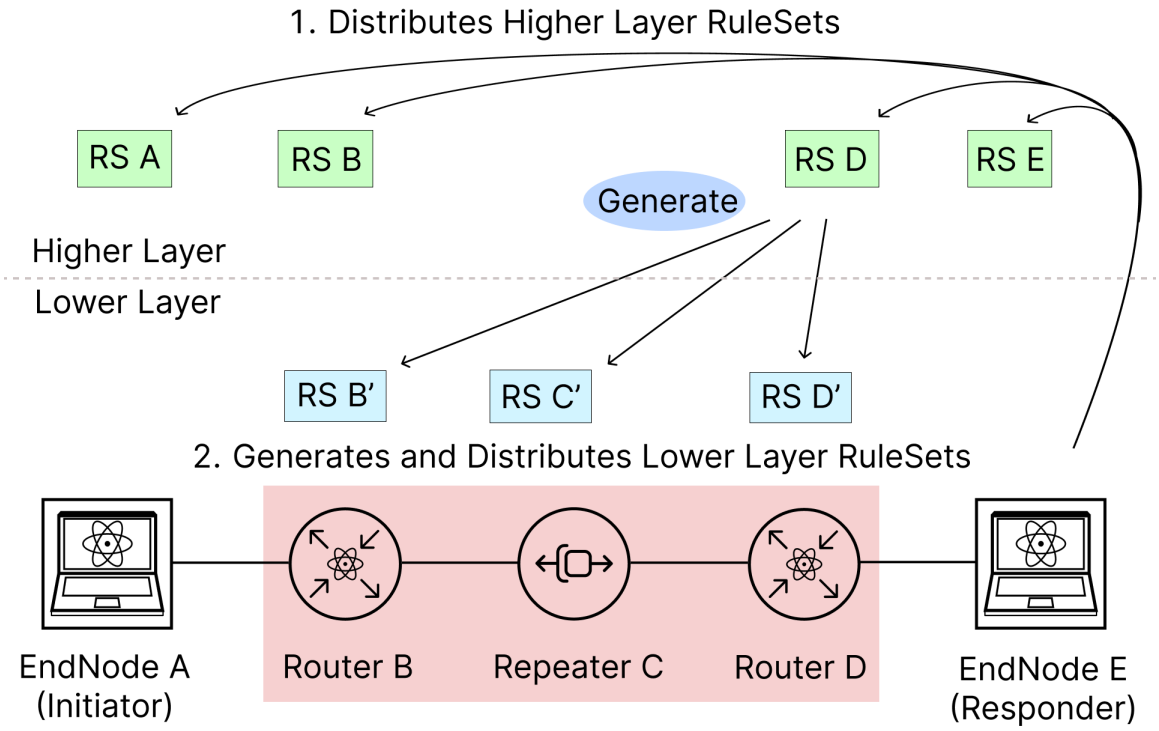

Figure 4.3: Link Recursion and Different RuleSet IDs

4.3.1 Link Recursion and Different RuleSet IDs

Fig. 4.3 shows an overview of the behavior when using different RuleSet IDs under link recursion. Green RuleSets indicate that they were generated by EndNode E, while light blue RuleSets indicate that they were generated by Router D. Router D is the Gateway Responder for the subnetwork and receives the RuleSets from EndNode E, which is the Responder for the outer network. In the case of link recursion, the RuleSets generated by EndNode E are used without modification. All RuleSets fully correspond to the request for each layer.

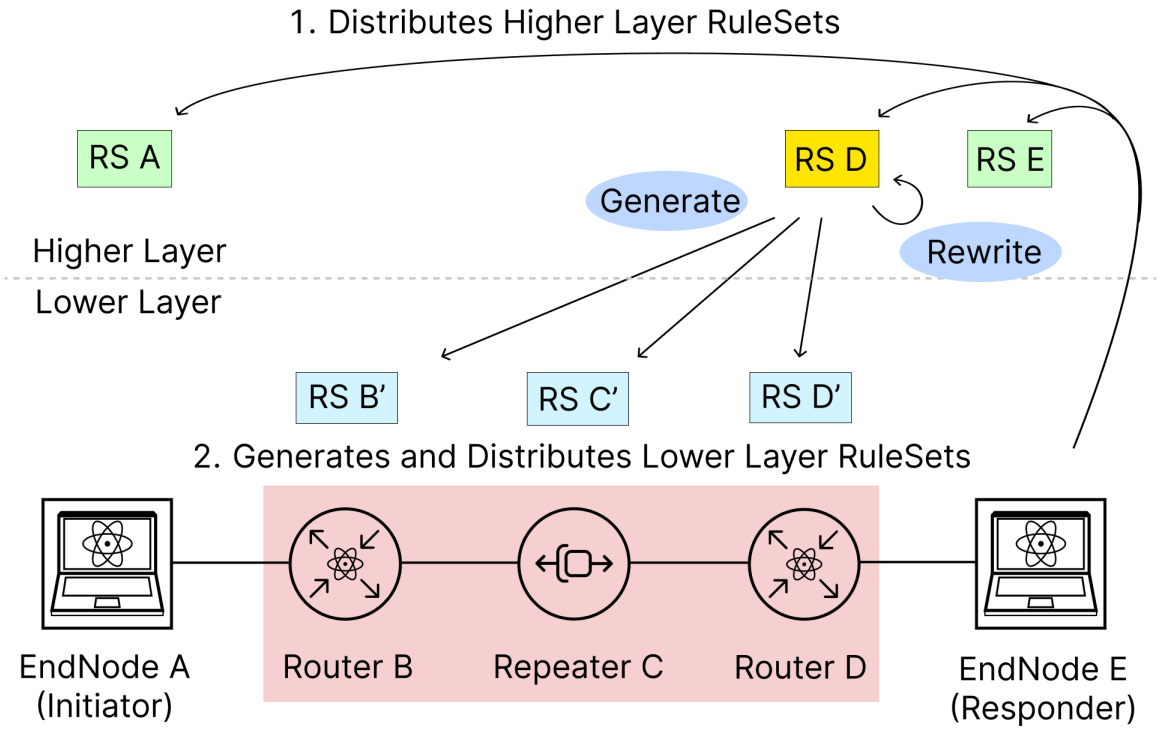

Figure 4.4: Node Recursion and Different RuleSet IDs

4.3.2 Node Recursion and Different RuleSet IDs

Fig. 4.4 shows the behavior with different RuleSet IDs under node recursion. The yellow RuleSet indicates that it was rewritten by Router D. Since EndNode E considers the subnetwork as a single node, the subnetwork must behave so that it appears as a single node in its entirety. There are multiple possible behaviors of the nodes in the subnetwork to achieve this. However, in any case, the RS D received from EndNode E must be modified.

The difficult part of this pattern is resource promotion: how does the Bell pair created between Router B and Router D perform entanglement swapping with the higher layers, EndNodes A and E? The other party that RS A communicates with is Router D, which has RS D, so a new classical communication is required.

Since Router B is the gateway router, a method of intercepting classical communication from EndNode A to Router D is also considered, but this requires that the communication path of the classical network be the same as that of the quantum network. It is also not acceptable for RS A to be changed by Router D for the convenience of its subnetwork, as this would violate layer and network independence.

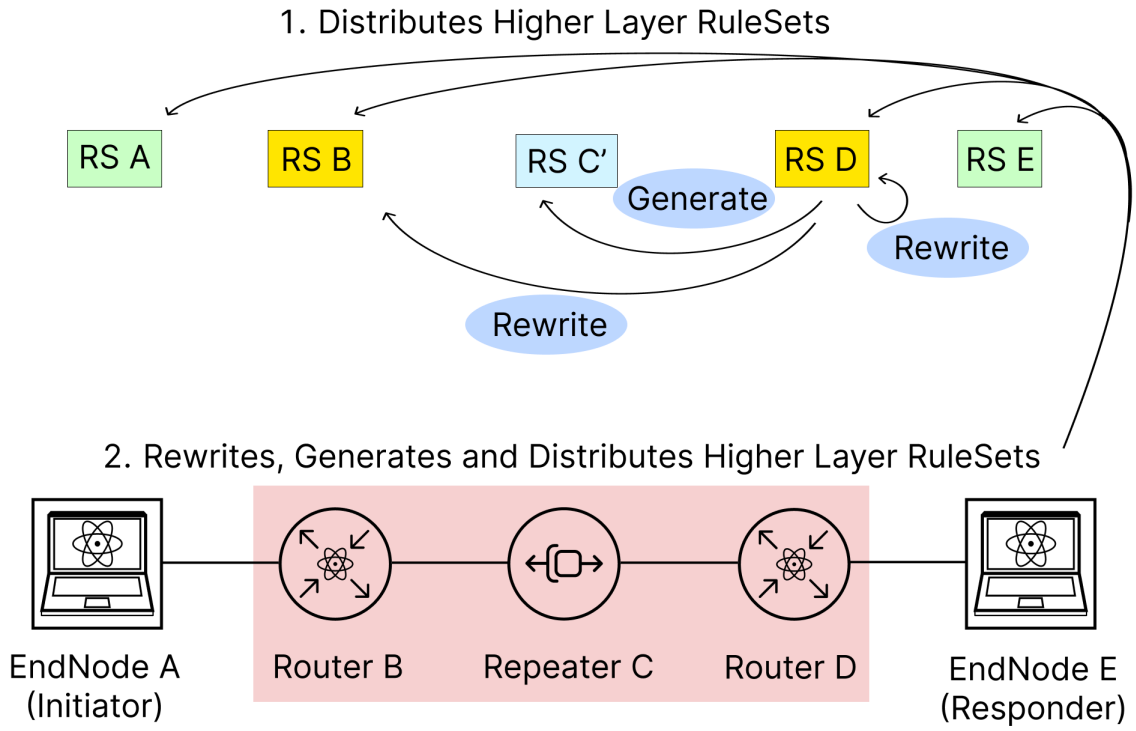

Figure 4.5: Link Recursion and Same RuleSet IDs

4.3.3 Link Recursion and Same RuleSet IDs

Fig. 4.5 shows the behavior when the same RuleSet IDs are used under link recursion. In this case, Router D rewrites RuleSets B and D to create RuleSet C. Router B needs to rewrite RuleSet B received from EndNode E based on the information received from Router D. Although it is technically feasible, other nodes should not rewrite the RuleSet received by one node because of the increased complexity.

While technically feasible, it should be avoided having another node rewrite the RuleSet received by one node because of the additional complexity. Rewriting RS B requires control over the order of arrival of classical packets, and Router B needs to wait for classical packets from two different sources and combine them. Router B needs to wait for classical packets from two different sources and combine them. Furthermore, Router D needs to make changes to RS B without knowing RS B and a mechanism for new RuleSet changes is needed between Routers B and D.

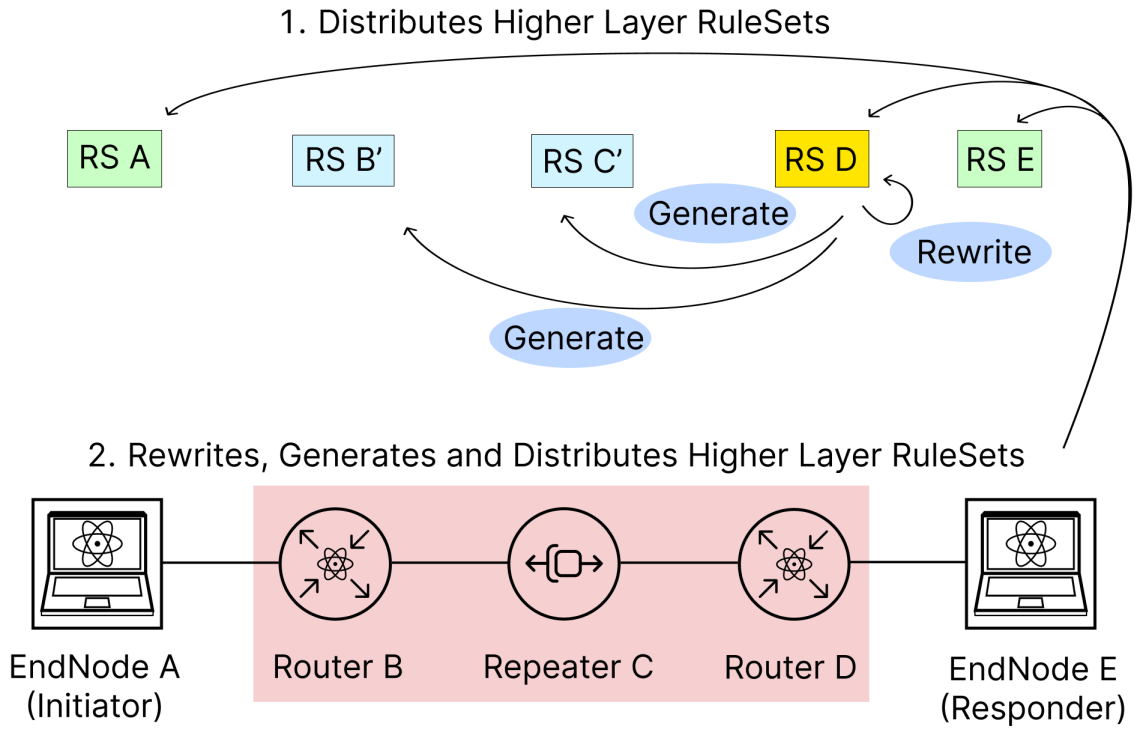

Figure 4.6: Node Recursion and Same RuleSet IDs

4.3.4 Node Recursion and Same RuleSet IDs

Fig. 4.6 shows the behavior when the same RuleSet IDs are used under node recursion. In this case, the received RuleSet D is rewritten, and Router D generates and distributes RuleSets B and C.

4.3.5 Layers and Subnetworks

A layer corresponds to a subnetwork, but it is not clear to which layer the concepts of RuleSet, connection, etc., are tied and how the recursive structure is organized. The recursive structure is deepened or unraveled when the request enters or leaves the subnetwork at the gateway router. Thus, the request is fully corresponding to the layer and the subnetwork. On the other hand, as we have seen with RuleSets, they do not necessarily correspond perfectly to the layer. When a gateway responder in a subnetwork rewrites and realizes the distributed higher-layer RuleSets, the subnetwork's RuleSets are treated the same as the higher-layer RuleSets. This means that the RuleSets do not correspond to the layer.

4.4 Rewriting Algorithm

Algorithm 1 RuleSet Rewriting Algorithm for Link-Level Recursion at Gateway Responder.

Require: $lowerRequest$, $higherRuleSet$
Ensure: RuleSets for lower layer nodes
1: $id \leftarrow higherRuleset.id$
2: $rulesets \leftarrow generateRuleSets(lowerRequest, higherRuleSet)$
3: $initiator \leftarrow lowerRequest.initiatorAddress$
4: $responder \leftarrow lowerRequest.responderAddress$
5: $initiatorRule \leftarrow Rule(RES(responder), PROMOTE(id))$
6: $responderRule \leftarrow Rule(RES(initiator), PROMOTE(id))$
7: $rulesets[initiator].addRule(initiatorRule)$
8: $rulesets[responder].addRule(responderRule)$
9: **return** $rulesets$

The simplest RuleSet rewriting algorithm for different RuleSet IDs and link recursion is shown in Alg. 1. This algorithm is executed by the gateway Responder. The input *lowerRequest* is a new request generated by the gateway Initiator, which contains link information on the path in the subnetwork, and *higherRuleSet* is a RuleSet generated by the global Responder for the gateway Responder. In this algorithm, RuleSets for nodes on the subnetwork path are first generated by the *generateRuleSets* procedure and stored in the *rulesets* variable. This *rulesets* is a dictionary that includes the node address as a key and the RuleSet to be executed by the node. The gateway Initiator and gateway Responder then add a rule containing the extended PROMOTE action clause to each RuleSet in order to pass the generated Bell pair to the higher layer's RuleSet. Finally, returning *rulesets*, the gateway Responder distributes the RuleSets to the nodes on the subnetwork.

Fig. 4.7 illustrates the state of the Bell pair generated in the subnetwork after all RuleSets on the path have been distributed. The generated Bell pair is promoted to the higher layer RuleSet. The link between B and D is thus created in the higher layer, and an end-to-end Bell pair is created between the Initiator and Responder by further execution of the RuleSet.

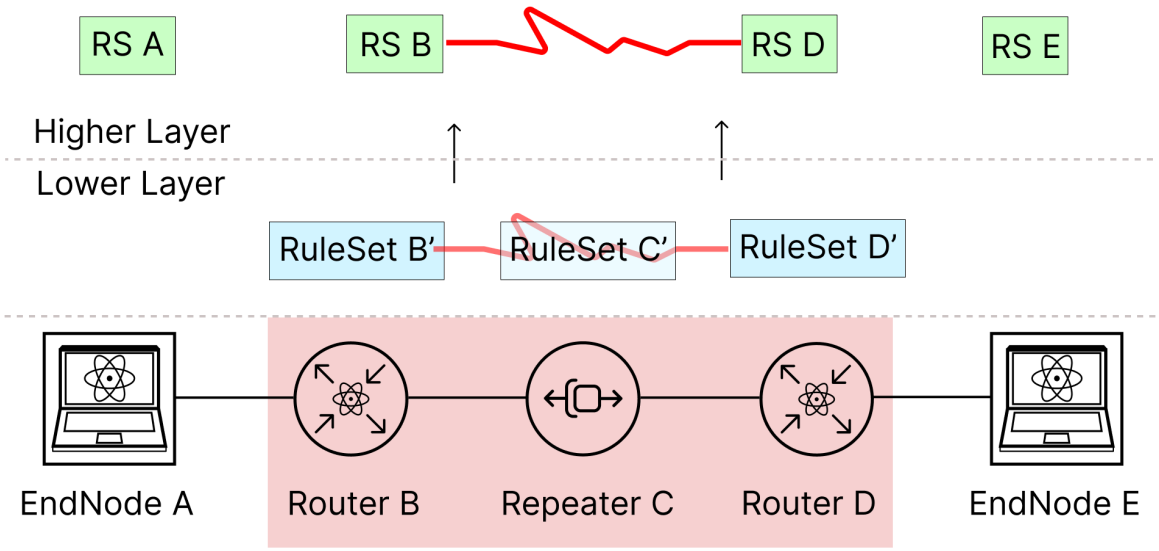

Figure 4.7: Inter-layer Resource Promotion. Lower layer RuleSets (blue) promote a generated Bell pair (red line) between B and D to higher layer RuleSets (green). This promotion action is executed in nodes B and D locally. Thus the higher layer RuleSets handle the subnetwork as a link.

Chapter 5

Implementation

This chapter describes quantum network simulation and implementation of QRNA on the RuleSet-based quantum networks. We implemented QRNA with the pattern of link recursion and different RuleSet IDs as in Sec. 43.

5.1 QuISP

QuISP (Quantum Internet Simulation Package)[37, 46] is an open-source simulator designed for large-scale simulations of the quantum Internet. QuISP is built on OMNeT++ [47], a framework for discrete event simulation, which allows users to quickly define new network topologies, check the network component's status during the simulation via GUI, and log events during the simulation.

Various methods for simulating qubits have been proposed , and diverse implementations exist. The method using density matrices can simulate various noises and quantum states but is unsuitable for large-scale simulations because it consumes a large amount of memory. For example, in the representation by a state vector, it is represented by a column vector consisting of 2^n complex numbers for n qubits. In QuISP, there are multiple QNICs per quantum node. In each QNIC, there are tens or even more qubits, so the representation of qubits by a density matrix or state vector is unsuitable because the network is constructed using tens to thousands of quantum nodes.

Therefore, when QuISP was first developed, the simulation of qubits was implemented using an error basis model. This method defines the probability of occurrence of the Pauli X, Z, and Y errors for a qubit, plus the relaxation and excitation errors. Each time a quantum operation and time lapse occurs, a random number is generated and compared to the error probability, recording only which errors have occurred. Memory consumption can be significantly reduced by keeping only the errors that occurred in the qubits instead of the actual quantum state.

5.2 Quantum Repeater Software Architecture

Quantum repeater software architecture (QRSA) is a software architecture for quantum repeaters proposed and implemented in QuISP [37, 46], consisting of five elements: Connection Manager (CM), Rule Engine (RE), Hardware Monitor (HM), Real-Time Controller (RC), and Routing Daemon (RD). This section introduces CM, RE, and RD, which will require major changes to implement QRNA on QuISP.

Routing Daemon: RD manages routing tables in quantum networks. Quantum networks do not necessarily have the same topology and path selection metrics as classical networks and require different management. In the real world, routing protocols are used to exchange information with neighboring nodes to construct paths, but in the QuISP implementation, the routing table is generated based on knowledge of the entire network using the current OMNeT++ functionality.

Connection Manager: CM is mainly responsible for the processing related to two-pass connection setup. The most important role is that of the Responder CM, which receives the request and generates RuleSets; the non-Responder CMs receive the request, add information about the node and its previous link to the request, and send the request to the next node. and sends the request to the next node.

The Responder CM is responsible for determining the strategy for how to generate end-to-end Bell pairs. The strategy of when each node in the path will perform entanglement swapping and when and what type of purification will be performed is embedded in the RuleSet and distributed by the Responder CM. This strategy greatly affects the fidelity and generation rate of the end-to-end Bell pair, and hence the time taken up by the qubit.

Rule Engine: RE is responsible for the management of the Bell pair and the execution of the RuleSet; it receives and manages the RuleSet generated by the Responder CM, and executes the RuleSet it holds in response to events such as notification of Bell pair generation and receipt of classic messages.

5.3 Implementation of QRNA on QuISP

The implementation of the QRNA functionality on QuISP can be broadly divided into the following: (i) the extension of node addresses, (ii) the addition of new actions to RuleSet, (iii) the extension of requests, (iv) the recursive processing of requests in the gateway router, (v) RuleSet rewriting in the gateway Responder, and (vi) the modification of the routing table for recursive networks. Fig. 5.1 shows how these implementation points correspond to protocol behavior.

5.3.1 Extension of Node Addresses

First of all, the address of the quantum node needs to be extended. Addresses are used to identify quantum nodes and to indicate the source and receiver of classical

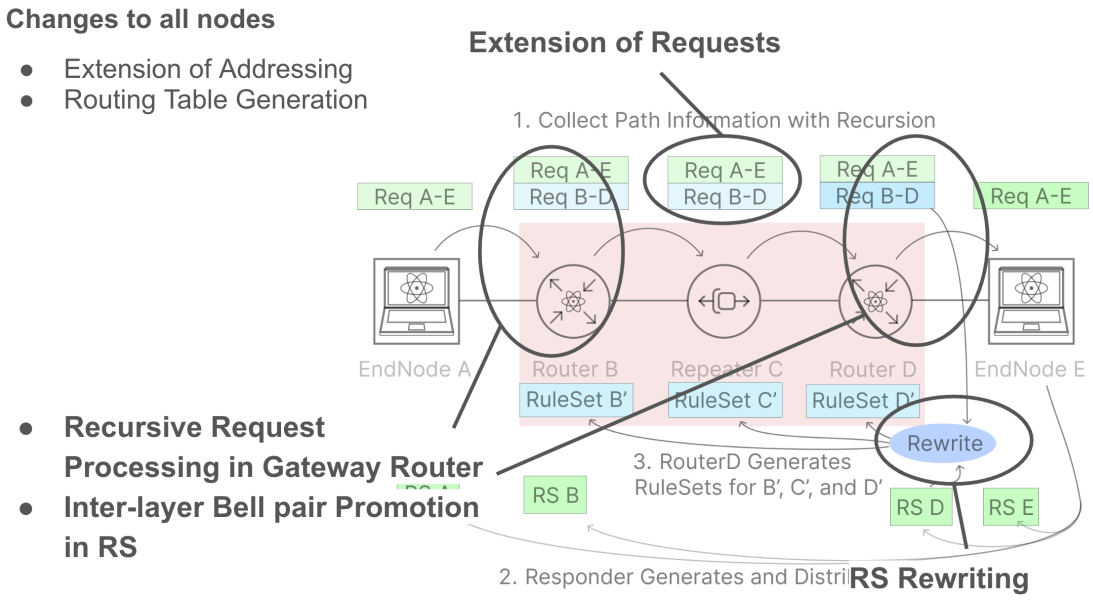

Figure 5.1: Overview of what we changed in QuISP for QRNA implementation.

packets. Each quantum node must have a mechanism to obtain a network identifier from its assigned address in order to know to which network it belongs. In QRNA, a single gateway router may belong to multiple networks. Therefore, a gateway router must be able to have multiple addresses.

In this thesis, addresses are not used for sophisticated routing algorithms. Research on the addressing for quantum networks and the quantum Internet already exists [48, 7]. However, here we focus on the realization of QRNA and give a minimal implementation.

The simulator implementation allows two integer values to be assigned to the node address for the network part and the host part. This allows the network to be set up so that each quantum node can determine whether another quantum node connected to it is in the same network or not.

5.3.2 Recursive Requests

When a request enters the subnetwork, the gateway router generates a new request for a connection inside the subnetwork and includes the original request for the external network received. For this reason, the request needs to be extended to include the request for the external network.

In QRNA, since the network is recursive, the request must also be recursive. Because a network corresponds to a recursive network layer, packets for each network layer can be closed to that network.

In other words, inside a given subnetwork, a request contains the requests of its

upper layers. This is similar to the structure of packets in a classical network. For example, in the usual classical network, the TCP layer lies on the IP layer, and an IP packet contains a TCP packet. In this quantum network, the lower layer packet contains the upper layer packet.

5.3.3 Recursive Request Processing in Gateway Router

The most important aspect of implementing QRNA is the processing at the gateway routers that takes place at the point of entry and exit of the subnetwork. At this point, new connections for the inside of the subnetwork can be handled. The information inside the subnetwork is then hidden from the outside of the subnetwork.

When a connection setup request enters a subnetwork in a two-pass connection setup, the gateway router of that subnetwork needs to generate a request for that subnetwork. The Initiator of the new request is the address of the gateway router, and the Responder is the address of the gateway router at the exit of the subnetwork.

When a connection setup request goes out from inside the subnetwork, the gateway router processes the request from the lowest layer. The Responder of the first request must be the gateway router. Therefore, the gateway router holds the lowest request, processes the higher layer requests contained in that request, and forwards them to the appropriate node outside the network. At this time, the gateway router adds the subnetwork as a link to the routing information of the upper layer request.

5.3.4 RuleSet Rewriting

A gateway Responder in a subnetwork receives a RuleSet generated by a Responder in an external network. Using this RuleSet, the gateway Responder needs to generate RuleSets for the nodes in that subnetwork. Alg. 1 (p.47) is used for this process. This process requires the corresponding request for the subnetwork internals previously stored and the RuleSet for that gateway Responder passed from the upper layers. The generated RuleSets generate the necessary number of Bell pairs between the Initiator and Responder inside the subnetwork so that the generated Bell pairs can be assigned to RuleSets at the upper layers.

The content of the algorithm for generating RuleSets inside the subnetwork depends on the implementation of the gateway Responder. When the desired Bell pair is obtained between the Initiator and Responder, a Rule that assigns it to the RuleSets of the upper layers should be able to be inserted into the RuleSets of the Initiator and Responder.

5.3.5 New Action on RuleSet

In order to assign a Bell pair obtained by a RuleSet at a lower layer to a RuleSet at a higher layer, a new action clause must be added to the RuleSet mechanism. To

promote a Bell pair, the management information of the qubits is changed at the two quantum nodes that hold the qubits at both ends of the Bell pair.

The PROMOTE action essentially assigns the qubits assigned to one Rule in one RuleSet to another [8]. This classically only changes the handling of the qubits within the Rule Engine. In this implementation on the simulator, this PROMOTE action is extended to allow the assignment of a qubit to another RuleSet within the same quantum node. In this case, no interaction with the external node occurs.

5.3.6 Routing Table for Recursive Networks

In order to implement QRNA, the routing table must also be modified. This is because when a request enters a subnetwork, the gateway router generates a new request and specifies as the Responder the address of the gateway router that is the exit point of that subnetwork. The gateway router must know what other gateway routers belonging to the same network are connected to the same node or network.

To avoid going into the details of the routing protocol, the implementation on the simulator uses the OMNeT++ functionality to access the list of all the nodes on the network and then generates the routing table.

5.4 Code Availability

The source code for QRNA implemented on QuISP is available online under the BSD-3 clause license[1].

[1] https://github.com/sfc-aqua/quisp/tree/internetworking-dev

Chapter 6

Evaluation

In this chapter, we will check whether the proposed implementation is suitable for the requirements described in 3.2. We will check the operation through simulations and evaluate the scalability based on the results. Next, the designed protocol and implementation will be evaluated in terms of manageability.

6.1 Scalability

6.1.1 Experiment Settings

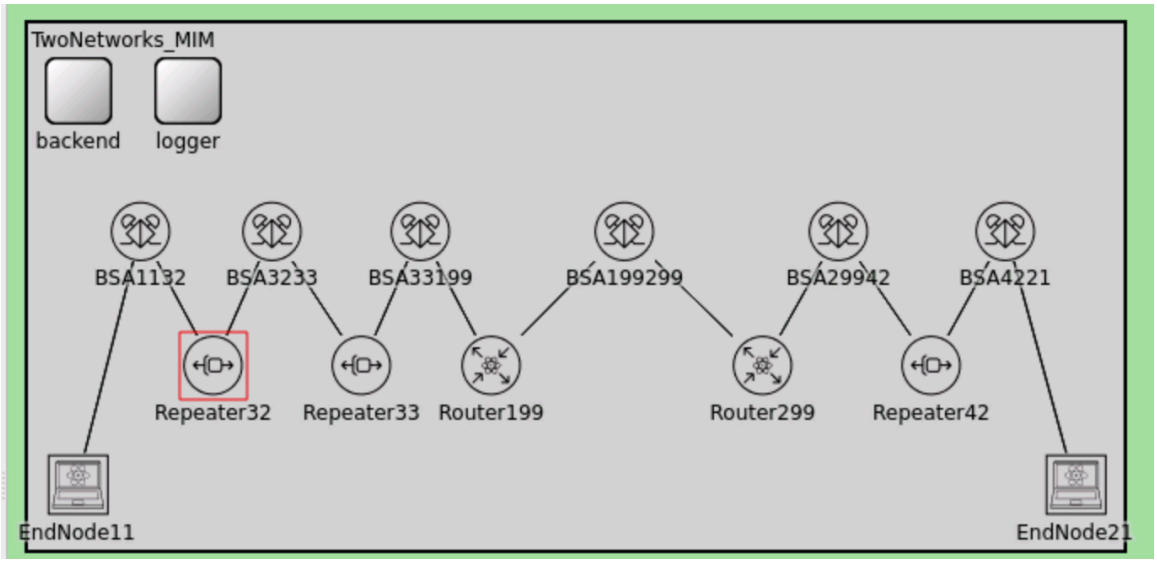

Figure 6.1: Screenshot of two networks simulation on QuISP. Each node has its own address that can specify which network the node belongs to.

Fig. 6.1 is a screenshot of the simulation. In this example, there are three networks. The central BSA node, BSA199299, separates the left and right networks. The end

nodes and gateway routers in these left and right networks also belong to the global network. In this simulator implementation, we need to manually assign an address to each node. The address has a network part that can indicate which network the node belongs to, so each node can determine which network it belongs to. Thus we can make up networks. To run this simulation in your machine, see Appendix. A.

In the experiment, a single connection request was sent within a linear network consisting of multiple sub-networks. The reason for the linear network is that the simulation focuses on a single request, so there is no need for nodes that are not involved in the communication.

Simulations were performed by changing the total number of nodes in the linear network by five from 5 nodes to 50 nodes and the number of subnetworks to be partitioned from 0 to 10. The size of the subnetworks was fixed to divide the network almost equally.

In this experiment, the ideal environment is set to zero error on the network. The fidelity of the generated end-to-end Bell pair is confirmed to be 1. This confirms that the QRNA implementation is correct.

6.1.2 Result

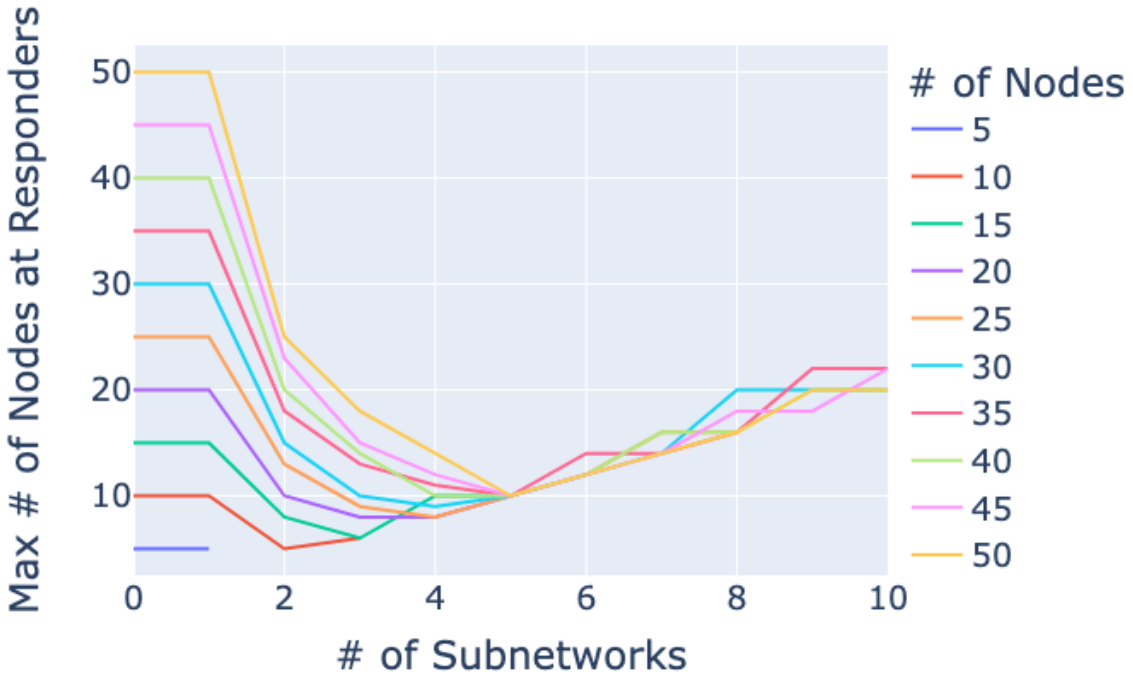

Figure 6.2: The largest number of RuleSets generated at each Responder in a network and the number of subnetworks and the total number of nodes in the network

Fig. 6.2 is a plot of the number of subnetworks and the largest number of RuleSets generated by each Responder. The number of RuleSets is the number of nodes that must be considered by that Responder. The lower this number, the lower the Responder's load. This number is the number of nodes that make up the path. As this number increases, the number of ES to generate end-to-end Bell pairs also increases. Since the amount of classical computation required to optimize the ES strategy increases exponentially in the number of nodes, minimizing this number is crucial.

As can be seen from this figure, the load on the Responder varies greatly depending on whether it is divided into subnetworks or not (between 1 and 2 for the number of subnetworks). This dip is caused by the existence of the subnetworks. Note that the networks are almost evenly divided, and the dip is an optimal case. Subsequently, as the number of subnetworks increased, the number of nodes to be considered by the global Responder steadily increased. This increase is not dependent on the total number of nodes in the network but on the number of subnetworks.

6.2 Manageability

Information about connections should not be mixed inside and outside the network, and the information inside should be well hidden when viewed from the outside of the network. This increases autonomy and contributes to improved security. This separation of information between layers leads to a reduction in the number of states to be managed for each network, which improves manageability.

This is taken into account from the design phase, and in the two-pass connection setup phase, information is separated between layers in all four patterns. However, due to the separation of information, the gateway responder needs to keep the request in its subnetwork and wait for a RuleSet from the upper layers. This increases the load on the gateway responder.

Even at the RuleSet execution stage, separation of information between layers is possible. link recursion can be realized by simply adding node-local operations to the RuleSet mechanism. However, node recursion requires additional classical communication to separate information between layers.

Chapter 7

Conclusion

7.1 Conclusion

In this thesis, we have compared recursion types and the handling of resource promotion between layers, which are major design considerations for implementing QRNA, a method of internetworking in the quantum Internet. We designed QRNA by applying it to a RuleSet-based protocol, implemented it on a simulator, and verified its behavior. We then evaluated the designed QRNAs in terms of scalability and manageability.

By recursively constructing a subnetwork with QRNA, we can show that it is scalable by being able to partition the Responders involved in a connection into multiple Responders and divide the number of nodes on the path that each Responder has to consider. We have shown through a simulator implementation that internetworking with QRNA is workable.

During the design phase of QRNA, detailed information, including requests for lower layers, is hidden from the higher layers. We implemented this on the simulator and confirmed that it is workable. We showed that this makes it easy to interconnect networks managed by different organizations. This leads to QRNA being manageable.

We proposed the concepts of link recursion and node recursion in the structure of the network as points to be considered in the design of architectures for the quantum Internet. This is an important difference from classical networks because, in quantum networks, basic operations such as link-level Bell pair creation, entanglement swapping, and purification are performed between a link and two nodes at both ends of the link.

7.2 Future Work

In this study, we have implemented and simulated the case where multiple quantum networks implemented with RuleSet-based protocols are connected by QRNA. Because QRNA hides the internal details of the subnetwork from the upper layers and absorbs the dependencies across the layers, it is also possible for subnetworks to use protocols other than the RuleSet-based protocol. The connection of different types of networks via QRNA needs to be explored further.

In QRNA, an address is needed to identify the QNode and the network to which the QNode belongs. An integer value address can work fine for a single network, but when multiple networks are connected, at least the address must have a network part and a host part. Realizing a scalable quantum Internet requires efficient routing table generation and an addressing design that allows autonomous network operation.

Bibliography

[1] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[2] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7–11, 2014.

[3] Artur K. Ekert. Quantum cryptography based on Bell's theorem. *Phys. Rev. Lett.*, 67:661–663, Aug 1991.

[4] Zheshen Zhang and Quntao Zhuang. Distributed quantum sensing. *Quantum Science and Technology*, 6(4):043001, jul 2021.

[5] Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412):eaam9288, 2018.

[6] Jessica Illiano, Marcello Caleffi, Antonio Manzalini, and Angela Sara Cacciapuoti. Quantum internet protocol stack: A comprehensive survey. *Computer Networks*, 213:109092, 2022.

[7] Jorge Miguel-Ramiro, Alexander Pirker, and Wolfgang Dür. Genuine quantum networks with superposed tasks and addressing. *npj Quantum Information*, 7(1):135, 2021.

[8] R. Van Meter, R. Satoh, N. Benchasattabuse, K. Teramoto, T. Matsuo, M. Hajdušek, T. Satoh, S. Nagayama, and S. Suzuki. A quantum internet architecture. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 341–352, Los Alamitos, CA, USA, sep 2022. IEEE Computer Society.

[9] Rodney Van Meter, Joe Touch, and Dominic Horsman. Recursive quantum repeater networks. *Progress in Informatics*, (8):65, March 2011.

[10] Takaaki Matsuo, Clément Durand, and Rodney Van Meter. Quantum link bootstrapping using a RuleSet-based communication protocol. *Physical Review A*, 100(5), 2019.

[11] James L Park. The concept of transition in quantum mechanics. *Foundations of physics*, 1:23–33, 1970.

[12] William K Wootters and Wojciech H Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.

[13] Wolfgang Dür and Hans J Briegel. Entanglement purification and quantum error correction. *Reports on Progress in Physics*, 70(8):1381, 2007.

[14] Joseph F. Fitzsimons. Private quantum computation: an introduction to blind quantum computing and related protocols. *npj Quantum Information*, 3(1), June 2017.

[15] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Universal blind quantum computation. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 517–526, 2009.

[16] Daniele Cuomo, Marcello Caleffi, and Angela Sara Cacciapuoti. Towards a distributed quantum computing ecosystem. *IET Quantum Communication*, 1(1):3–8, July 2020.

[17] Craig Gidney and Martin Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, 5:433, 2021.

[18] Christian L Degen, Friedemann Reinhard, and Paola Cappellaro. Quantum sensing. *Reviews of modern physics*, 89(3):035002, 2017.

[19] Timothy J Proctor, Paul A Knott, and Jacob A Dunningham. Multiparameter estimation in networked quantum sensors. *Physical review letters*, 120(8):080501, 2018.

[20] Daniel Gottesman, Thomas Jennewein, and Sarah Croke. Longer-baseline telescopes using quantum repeaters. *Phys. Rev. Lett.*, 109:070503, Aug 2012.

[21] Sreraman Muralidharan, Linshu Li, Jungsang Kim, Norbert Lütkenhaus, Mikhail D Lukin, and Liang Jiang. Optimal architectures for long distance quantum communication. *Scientific reports*, 6(1):20463, 2016.

[22] Koji Azuma, Kiyoshi Tamaki, and Hoi-Kwong Lo. All-photonic quantum repeaters. *Nature communications*, 6(1):6787, 2015.

[23] Naphan Benchasattabuse, Michal Hajdušek, and Rodney Van Meter. Architecture and protocols for all-photonic quantum repeaters. *arXiv preprint arXiv:2306.03748*, 2023.

[24] Cody Jones, Danny Kim, Matthew T Rakher, Paul G Kwiat, and Thaddeus D Ladd. Design and analysis of communication protocols for quantum repeater networks. *New Journal of Physics*, 18(8):083015, 2016.

[25] Axel Dahlberg, Matthew Skrzypczyk, Tim Coopmans, Leon Wubben, Filip Rozpdek, Matteo Pompili, Arian Stolk, Przemyslaw Pawelczak, Robert Knegjens, Julio De Oliveira Filho, Ronald Hanson, and Stephanie Wehner. A link layer protocol for quantum networks. *SIGCOMM 2019 - Proceedings of the 2019 Conference of the ACM Special Interest Group on Data Communication*, pages 159–173, 2019.

[26] Natasha Tomm, Alisa Javadi, Nadia Olympia Antoniadis, Daniel Najer, Matthias Christian Löbl, Alexander Rolf Korsch, Rüdiger Schott, Sascha René Valentin, Andreas Dirk Wieck, Arne Ludwig, et al. A bright and fast source of coherent single photons. *Nature Nanotechnology*, 16(4):399–403, 2021.

[27] Fabian Steinlechner, Sebastian Ecker, Matthias Fink, Bo Liu, Jessica Bavaresco, Marcus Huber, Thomas Scheidl, and Rupert Ursin. Distribution of high-dimensional entanglement via an intra-city free-space link. *Nature communications*, 8(1):15971, 2017.

[28] 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, 2013.

[29] Kaushik Chakraborty, Axel Dahlberg, Filip Rozpedek, and Stephanie Wehner. Distributed routing in a quantum internet. *Bulletin of the American Physical Society*, 64, 2019.

[30] M. Caleffi. Optimal routing for quantum networks. *IEEE Access*, 5:22299–22312, October 2017.

[31] Tim Coopmans, Robert Knegjens, Axel Dahlberg, David Maier, Loek Nijsten, Julio de Oliveira Filho, Martijn Papendrecht, Julian Rabbie, Filip Rozpedek, Matthew Skrzypczyk, et al. Netsquid, a network simulator for quantum information using discrete events. *Communications Physics*, 4(1):164, 2021.

[32] Axel Dahlberg and Stephanie Wehner. Simulaqron—a simulator for developing quantum internet software. *Quantum Science and Technology*, 4(1):015001, 2018.

[33] Axel Dahlberg, Bart van der Vecht, Carlo Delle Donne, Matthew Skrzypczyk, Ingmar te Raa, Wojciech Kozlowski, and Stephanie Wehner. Netqasm—a low-level instruction set architecture for hybrid quantum–classical programs in a quantum internet. *Quantum Science and Technology*, 7(3):035023, 2022.

[34] Ben Bartlett. A distributed simulation framework for quantum networks and channels. *arXiv preprint arXiv:1808.07047*, 2018.

[35] Lutong Chen, Kaiping Xue, Jian Li, Nenghai Yu, Ruidong Li, Qibin Sun, and Jun Lu. Simqn: a network-layer simulator for the quantum network investigation. *IEEE Network*, pages 1–8, 2023.

[36] Stephen DiAdamo, Janis Nötzel, Benjamin Zanger, and Mehmet Mert Beşe. Qunetsim: A software framework for quantum networks. *IEEE Transactions on Quantum Engineering*, 2:1–12, 2021.

[37] Ryosuke Satoh, Michal Hajdušek, Naphan Benchasattabuse, Shota Nagayama, Kentaro Teramoto, Takaaki Matsuo, Sara Ayman Metwalli, Takahiko Satoh, Shigeya Suzuki, and Rodney Van Meter. QuISP: A Quantum Internet Simulation Package. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 353–364, 2022.

[38] Xiaoliang Wu, Alexander Kolar, Joaquin Chung, Dong Jin, Tian Zhong, Rajkumar Kettimuthu, and Martin Suchara. Sequence: a customizable discrete-event simulator of quantum networks. *Quantum Science and Technology*, 6(4):045027, 2021.

[39] Rodney Van Meter and Takaaki Matsuo. Connection Setup in a Quantum Network. Internet Draft draft-van-meter-qirg-quantum-connection-setup-01, Internet Engineering Task Force, September 2019.

[40] H. J. Kimble. The quantum internet. *Nature*, 453(7198):1023–1030, June 2008.

[41] Joe Touch. A Recursive Network Architecture. *ISI-TR*, 626:00, 2006.

[42] John Day. *Patterns in Network Architecture: A Return to Fundamentals*. 2008.

[43] A. Pirker and W. Dür. A quantum network stack and protocols for reliable entanglement-based networks. *New Journal of Physics*, 21(3):033003, March 2019.

[44] Cisco annual internet report - cisco annual internet report (2018 – 2023) white paper - cisco. `https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html`, March 2020. (Accessed on 07/05/2023).

[45] Evgeny Shchukin and Peter van Loock. Optimal Entanglement Swapping in Quantum Repeaters. *Physical Review Letters*, 128(15):150502, April 2022.

[46] sfc-aqua/quisp: Open source implementation of quantum internet simulation package. `https://github.com/sfc-aqua/quisp`. (Accessed on 06/05/2023).

[47] OMNeT++ discrete event simulator. `https://omnetpp.org/`. (Accessed on 06/05/2023).

[48] Angela Sara Cacciapuoti, Jessica Illiano, Michele Viscardi, and Marcello Caleffi. Quantum internet addressing. *arXiv preprint arXiv:2306.05982*, 2023.

Appendix A

Running the Simulations

A.1 Network Definition of Recursive Quantum Networks

To run a simulation in QuISP, you need an executable file in which QuISP is built, a NED file, and an INI file. The NED file describes the network topology definition. The INI file contains the parameters necessary for the simulation. The parameters can be the error rate for each quantum gate operation, the duration of the simulation, etc. The OMNeT++ simulation manual[1] explains how to write NED and INI files in detail.

Listing. A.1 shows the network definition used in Fig. 6.1 (p.54). Recursive network definition is enabled by the network address specified for each node.

The difference here from the original QuISP is how the address of each node is specified. The address of a node consists of a string of two integer values separated by a dot, in the form of "`{network address}.{host address}`". Each QNode has one `address` field. And for the QNode that can be a gateway router, `available_addresses` field is specified in the form of a list of addresses.

Each node determines which network it belongs to and whether or not its neighbors are in the same network, based on the node address assigned to it and the addresses of the nodes to which it is connected. Then, when it receives a request, it determines whether it should act as a gateway router or an intermediate node.

Listing A.1: Network Definition for Simple Two Networks

```
1 package networks;
2
3 import ned.DatarateChannel;
4 import ned.IdealChannel;
5
6 import modules.*;
7 import modules.Backend.Backend;
8 import modules.Logger.Logger;
```

[1] https://doc.omnetpp.org/omnetpp/manual/

```
import channels.*;

network TwoNetworks_MIM
{
    parameters:
        **.speed_of_light_in_fiber = 208189.206944 km;

    submodules:
        backend: Backend;
        logger: Logger {
            @display("p=95,40");
        }
        EndNode11: QNode {
            address = "1.1";
            available_addresses = ["3.1"];
            node_type = "EndNode";
            @display("i=COMP;p=29,273,m,5,60,60");
        }
        Repeater32: QNode {
            address = "3.2";
            node_type = "Repeater";
            @display("i=REP1G;p=110,203,m,5,60,60");
        }
        Repeater33: QNode {
            address = "3.3";
            node_type = "EndNode";
            @display("i=REP1G;p=198,203,m,5,60,60");
        }
        Router199: QNode {
            address = "1.99";
            available_addresses = ["3.99"];
            node_type = "Router";
            @display("i=RTR;p=275,203,m,5,60,60");

        }
        Router299: QNode {
            address = "2.99";
            available_addresses = ["4.99"];
            node_type = "Router";
            @display("i=RTR;p=436,203,m,5,60,60");
        }
        Repeater42: QNode {
            address = "4.2";
            node_type = "Router";
            @display("i=REP1G;p=536,203,m,5,60,60");
        }
        EndNode21: QNode {
            address = "2.1";
            available_addresses = ["4.1"];
            node_type = "EndNode";
            @display("i=COMP;p=621,273,m,5,60,60");
        }
        BSA1132: BSANode {
            address = "3.10";
            @display("p=67,123,m,5,60,60");
        }
        BSA3233: BSANode {
            address = "3.11";
            @display("p=145,123,m,5,60,60");
        }
        BSA33199: BSANode {
            address = "3.12";
            @display("p=233,123,m,5,60,60");
        }
```

```
        BSA199299: BSANode {
            address = "1.10";
            @display("p=354,123,m,5,60,60");
        }
        BSA29942: BSANode {
            address = "4.10";
            @display("p=481,123,m,5,60,60");
        }
        BSA4221: BSANode {
            address = "4.11";
            @display("p=580,123,m,5,60,60");
        }

    connections:
        EndNode11.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA1132.port++;
        BSA1132.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> Repeater32.port++;
        EndNode11.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> BSA1132.quantum_port++;
        BSA1132.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> Repeater32.quantum_port++;

        Repeater32.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA3233.port++;
        BSA3233.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> Repeater33.port++;
        Repeater32.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> BSA3233.quantum_port++;
        BSA3233.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> Repeater33.quantum_port++;

        Repeater33.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA33199.port++;
        BSA33199.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> Router199.port++;
        Repeater33.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> BSA33199.quantum_port++;
        BSA33199.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> Router199.quantum_port++;

        Router199.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA199299.port++;
        BSA199299.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> Router299.port++;
        Router199.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> BSA199299.quantum_port++;
        BSA199299.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> Router299.quantum_port++;

        Router299.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA29942.port++;
        BSA29942.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> Repeater42.port++;
        Router299.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> BSA29942.quantum_port++;
        BSA29942.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <--> Repeater42.quantum_port++;

        Repeater42.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> BSA4221.port++;
        BSA4221.port++ <--> ClassicalChannel {  distance = 0.5km; } <--> EndNode21.port++;
        Repeater42.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <-->
```

```
          BSA4221.quantum_port++;
        BSA4221.quantum_port++ <--> QuantumChannel {  distance = 0.5km; } <-->
          EndNode21.quantum_port++;

}
```

A.2 quisp.py

quisp.py is a library that assists in the generation of complex network definitions and the execution of a large number of simulations. As seen in Listing. A.1, network definition files for simulating multiple networks files tend to be long and complex and are not suitable for writing directly by hand. They are also not suitable for simulating gradually increasing numbers of nodes. Therefore, we created quisp.py, which facilitates network definition and allows processes to be assigned to each simulation and run simultaneously in Python. Quisp.py is available online under the BSD-3 clause license[2].

Listing. A.2 generates the same topology as Fig. 6.1 (p.54), runs the simulation and displays the results in pandas DataFrame format.

Listing A.2: Example Code for quisp.py

```python
from quisp import (
    ChannelOption,
    QNodeAddr,
    NativeSimulator,
    NetworkBuilder,
)

# define new network named "simple_7nodes_internetworking"
n = NetworkBuilder("simple_7nodes_internetworking")

# add new QNode that has addresses 1.1 and 3.1
n.add_qnode([QNodeAddr(1,1), QNodeAddr(3,1)], is_initiator=True) # Initiator
n.add_qnode([QNodeAddr(3,2)])
n.add_qnode([QNodeAddr(3,3)])
n.add_qnode([QNodeAddr(1,99),QNodeAddr(3,99)])
n.add_qnode([QNodeAddr(2,99),QNodeAddr(4,99)])
n.add_qnode([QNodeAddr(4,2)])
n.add_qnode([QNodeAddr(2,1), QNodeAddr(4,1)], is_resipient=True) # Responder

# build a linear network
n.connect_linear(ChannelOption(quantum_channel_distance=5, classical_channel_distance=5))

# specify the executable file of QuISP
sim = NativeSimulator("../quisp")

# generate ned file and ini file and setup workspace folder
sim.load(n.network)

# start simulation
await sim.run()

```

[2]https://github.com/sfc-aqua/quisp.py/tree/internetworking

```
32 # collect results information
33 sim.read_result()
34
35 # show the collected results
36 print(sim.df)
```