ABSTRACT

Title of dissertation: QUANTUM SWITCHING NETWORKS:
UNICAST AND MULTICAST

Manish Kumar Shukla
Doctor of Philosophy, 2010

Dissertation directed by: Professor A. Y. Oruç
Department of Electrical Engineering

Quantum switching networks are analogs of classical switching networks in which classical switches are replaced by quantum switches. These networks are used to switch quantum data among a set of quantum sources and receivers. They can also be used to efficiently switch classical data, and help overcome some limitations of classical switching networks by utilizing the unique properties of quantum information systems, such as superposition and parallelism. In this thesis, we design several such networks which can be broadly put in the following three categories:

1. *Quantum unicast networks:* We give the design of quantum Baseline network (QBN) which is a self-routing and unicast quantum packet switch that uses the Baseline topology. The classical version of the network blocks packets internally even when there are no output contentions and each input packet is addressed to a different output. The QBN uses the principles of quantum superposition and parallelism to overcome such blocking. Also, for assign-

ments that have multiple input packets addressed to an output, this network creates a quantum superposition of all these packets on that output, ensuring that all packets have non-zero probabilities of being observed on that output.

2. *Quantum concentrators:* We introduce a new network called quantum concentrator, which is a key component of our quantum multicasting network design. This concentrator is also an $n \times n$ quantum switching network, to be denoted by n-QC, and which, for any m, $1 \leq m \leq n$, routes arbitrary quantum states on any m of its inputs to its top m outputs. This network uses $O(n \log n)$ quantum gates, and has a gate level depth of $O(\log^2 n)$. We also give several variations of this network, the main ones being order-preserving and priority quantum concentrators.

3. *Quantum multicast networks:* We first design a quantum multicasting network, called a *generalized quantum connector* (GQC) which can be used to multicast quantum information from n input sources to n outputs. Since general quantum states cannot be copied due to the no-cloning theorem [1], this network actually multicasts superposed classical information packets, contained in a finite number of qubits at each input. Copying needed for such multicasting is obtained by Wootters and Zurek's quantum copying machines [1] [2] or controlled-not gates. This n-input and n-output ($n \times n$) network, to be denoted by n-GQC, is recursively constructed using $n/2$-GQCs and uses $O(n \log^2 n)$ quantum gates. The time complexity of this

network in terms of gate level depth is $O(\log^3 n)$. We also give two variations of this network which improve its behavior when routing multicast assignments that have multiple input packets contending for same outputs.

QUANTUM SWITCHING NETWORKS: UNICAST AND MULTICAST

by

Manish Kumar Shukla

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2010

Advisory Committee:
Professor A. Y. Oruç, Chair/Advisor
Professor Robert Newcomb
Professor Mario Dagenais
Professor Ankur Srivastava
Professor Kenneth Berg

To my wife, Archana.

Thank you for your love and support!

Acknowledgments

First and foremost, I would like to thank my adviser, Professor Yavuz Oruç for sharing his vision on moving the field of switching theory forward using quantum networks, and his generous guidance during the course of my graduate studies. He is a mentor not only on my academic endeavors but also on other facets of life. Thank you sir for everything that you have done, and for the fun-filled fishing trips we have had together.

I would like to thank Professors Robert Newcomb, Mario Dagenais, Ankur Srivastava and Kenneth Berg for taking part in my PhD dissertation defense committee, and providing me with valuable comments and suggestions to make this thesis better.

My stay at the University of Maryland has been an experience to cherish forever because of the great friends I have had. I specially thank my fellow researcher and close friend Rahul Ratan for always being open to discussions on any topic related to research, and always being available to hit some tennis balls. I would also like to thank my tennis buddies and friends, Ravi Tandon, Anuj Rawat, Alok Singhal, Rakesh Panda, Robert Kargus and Himaanshu Gupta for keeping me moving. I thank my other close friends at UMD: Tarun Pruthi, Sharmistha, Vishal Khandelwal, Srikanth Vishnubhotla, Gaurav Agarwal, Abhinav Gupta, Swati Jarial, Pavan Turaga, Amit Agrawal, Sudarahsan Kaushik, Gunjan Sharma and Himanshu Tyagi for making my graduate studies a fun and memorable experience.

This work would not have been possible without the constant support and love from my family. My deepest gratitude goes to my dad for always believing in me and unconditionally supporting me on the decisions that I have made in my life. He is the reason of my interests in mathematics and science in general. Thank you dad for everything that you have done. I do not have words to describe the love and care we have received from my Mom. She is the bedrock of our family who has powered and guided us all along. She cooks the most delicious food that I have ever had, and ever will. I would like to thank my younger brother, Awanish, for his love despite me being harsh with him at times. He has shielded me from several responsibilities during my graduate studies, and has always supported me. Nidhi, for her best wishes, and quickly becoming a great support for all of us. My youngest brother, Vishwa Deepak (Deepu), for discussions on this thesis; cricket, tennis, and sports in general. My sister, Deepika, for her love, and keeping me up to date with her fights with Deepu. I cannot thank my lovely wife, Archana, enough for her love, care and support. She gave me the strength to pursue graduate studies, and has been with me every step of the way. I love you so much.

Last, but not the least, I thank my late grandparents, whose memories, love and blessings will always remain with us. I can never forget how happy and excited they got whenever we were home in Lahasani during our summer vacations. Pranam Daadi and Baba!

Table of Contents

List of Figures

Chapter 1: Introduction

Quantum information science is an emerging area of research that seeks to use special properties of quantum systems such as parallelism and entanglement to develop efficient solutions for classically intractable problems. Research in this area has been spurred by some key algorithms that have shown that quantum systems could be used to solve some important exponentially complex problems with speedups that are impossible in classical computing. Examples include Shor′s polynomial time algorithm [19] for finding the prime factors of a composite number and Grover′s search algorithm [20] that can find an element in an unstructured database containing n elements in $O(\sqrt{n})$ time.

Building quantum systems would require means to transport quantum information from one place to another. Several architectures are being considered to build *quantum wires* over which quantum data can be transmitted. Primary examples are quantum swapping and teleportation-based architectures for building quantum wires as described in [21]. For n quantum sources that wish to communicate with one another by sharing quantum information, $O(n^2)$ quantum wires are needed. This complexity can be greatly reduced using advanced switching architectures. The relationship between quantum circuits and permutation maps was identified in [7] and it was shown that any permutation map can be realized by a quantum circuit consisting of 6 layers of controlled-not gates. However, this result requires a different quantum circuit for each permutation map to be realized. It was also shown in [7] that the classical components of a qubit can be replicated using controlled-not gates. This copying was used in [8] to implement both unicasting and multicasting of qubits using directed graph representations of the desired maps. However, this approach also requires that each multicast assignment be realized by a separate quantum circuit, and cannot be viewed as a quantum switching network that can realize multiple or all maps.

In [5] and [6], we presented a self-routing $O(n \log n)$ quantum baseline network that can be used to permute quantum packets from n inputs to n outputs. We also showed that this network can also be used to resolve internal blocking when transmitting classical packets by creating a superposition of packets whenever they contend for the same wire in the network. However, this network can realize only $n^{n/2}$ permutations between n inputs and n outputs out of $n!$ total possible permutations. Recently, Cheng and Wang [9] proposed a quantum merge sorting-based switching network that can realize all $n!$ permutations using $O(n \log^2 n)$ quantum gates. More recently, [10] described a non-blocking quantum switch with $O(n^2)$ quantum gates. This design relies on quantum circuit representations of unicast and multicast maps given in [7] and [8]. The networks in [5, 6, 9, 10] are limited to satisfying one-to-one or unicast assignments between inputs and outputs. We presented a $O(n \log^2 n)$ quantum multicasting network called n-generalized quantum connector in [3] which can be used to multicast both superposed classical packets and arbitrary quantum states, provided the assignments are non-contending, i.e., no two packets in an assignments are addressed to the same output.

Our work on quantum unicasting, the quantum baseline network, is described in Chapter 3. In Chapters 5 and 6, we give our work on quantum multicasting networks, the generalized quantum connector, and its variants that improve its behavior when packets have output contentions. We also design quantum concentration networks which can be used to multiplex sparse quantum data from a large number of inputs to a small number of outputs. These networks are given in Chapter 4. Next, we give brief introductions to these three types of quantum switching networks.

1.1 Quantum Unicast Networks

A quantum unicast switching network routes quantum data from a set of input sources to a set of outputs, where each input source connects to at most one output. In Chapter 3, we give the design of an $n \times n$ quantum unicast network called the quantum Baseline network (QBN) which realizes $n^{n/2}$ permutations between the n inputs and outputs when routing quantum data. When routing classical packets encoded in quantum states, this network uses quantum superposition to resolve internal blocking such that when two packets contend for the same output at an internal switch, it creates a superposition of these packets on that output. If we make a measurement on the outputs of QBN, every input packet has a non-zero probability of being observed at its desired output. Additionally, we characterize the permutations which can be self-routed through the QBN without internal blocking by using the concepts of balanced matrices and frames introduced in [22] and [23] respectively. We then describe a method which determines the distribution of sub-permutations generated from a permutation assignment which suffers internal blocking while being self-routed through the QBN.

An unicast switching network is said to be non-blocking if it can route all possible one-to-one input-output mappings. For an $n \times n$ network, these mappings are the $n!$ permutations of the output addresses. A network which cannot route at least one of these $n!$ input-output mappings is said to be blocking. Many non-blocking interconnection networks exist in the literature [24], but either their crosspoint complexity is high [14, 25–27], or they do not have simple routing schemes [28]. Therefore, networks which are blocking but have the desirable properties of simple decentralized routing along with lower crosspoint complexity have been investigated. Examples of such networks are the Baseline, Banyan, Omega, Butterfly, Delta networks [29–35]. These networks are most suited for realizing quantum unicast networks because of their self-routing property.

An $n \times n$ (usually $n = 2^p$) Baseline network is composed of $\log_2 n$ stages, each having $n/2$ 2×2 switches. This network is self-routing, i.e., the routing decision for a packet at any stage in the network is made solely on the basis of the output address in that packet's header [24, 29, 30]. Thus, the routing is local, decentralized and easy to implement. Since there are $(n/2)\log_2 n$ 2×2 switches in the network, it can have $2^{(n/2)\log_2 n} = n^{n/2}$ states and because this network provides a unique path between an input-output pair, this is also equal to the number of unique input-output permutation maps it can route. As a result, a baseline network *cannot* route $n! - n^{n/2}$ permutation maps, and contentions for outputs can occur between the input packets at an internal switch. These contending packets either buffered or dropped in the classical implementations of the baseline network.

Using quantum superposition, two contending packets can be sent together to their desired output (without any increase in bandwidth or additional lines) when the packets are encoded using qubits. In a classical network, sending both packets on the same output wire would require either multiple parallel outputs or higher bandwidth outputs, both of which have a higher cost. Consider packets P_1 and P_2 input to a 2×2 switch S with outputs O_1 and O_2. Both packets are addressed to a single output, say O_1. In a classical network, either we buffer or drop one randomly chosen packet and route the other. But using quantum superposition a state of the form $1/\sqrt{2}(|P_1, P_2\rangle + |P_2, P_1\rangle)$ can be created at the outputs of the switch. The two entries in each *ket* ($|\rangle$) correspond to packets at O_1 and O_2 respectively. We can see that both P_1 and P_2 are present at O_1 and a simple measurement in computational basis ($|0\rangle, |1\rangle$) on each qubit will give either P_1 or P_2 at O_1, each with probability $1/2$. Thus, blocking does not occur until the measurement is made. One implication of this is that packets can be routed to their destinations without incurring any time overhead to resolve the contentions between them. Other clever measurements can be devised that lead to different results by yielding information about both the packets. Also, the superposition created in this way can be used to process the packets in parallel.

QBN uses the above concept to resolve internal blocking. For an input assignment that undergoes internal blocking, the output of QBN is a superposition of correctly realized subsets of the input assignment, which can be routed through the network without blocking. In Chapter 3, we also give a method to characterize all such subsets using concepts of balanced matrices and frames.

1.2 Quantum Multicasting

Multicasting or generalized connection networks have been extensively studied in the classical information domain [36–40]. A survey of these networks is given in [41]. These networks can be widely classified into three classes. The first class contains multicast networks based on the three stage Clos network. Networks in this class require complex routing algorithms and are not self-routing in general. Consequently, these networks cannot easily be implemented using quantum circuits. The second class consists of networks in which multicasting is decomposed into two stages. In the first stage, all the required copies of the input packets are created. The second stage routes these copies to their desired outputs. The third class of multicasting networks, introduced by Nassimi and Sahni [42], are based on recursive decomposition of generalized connectors into smaller ones. Nassimi and Sahni's generalized connector, however, requires a parallel computer model connected in cube or perfect shuffle topology for computing routes. Lee and Oruç used a similar approach in their $O(n \log^2 n)$ generalized connector in which routes are determined locally at internal nodes by using local packet headers [14] [41]. Thus, their network is self-routing.

As in the case of quantum Baseline or merge-sorting networks, we begin with a meaningful interpretation of multicasting in a quantum network. In a classical multicasting network, packets at inputs are routed to outputs under the assumption that a packet at any given input may be routed to one or more outputs. One restriction that is often applied is that such multicast assignments may not direct more than one packet to a given output. Furthermore, assignments in a classical multicast network can only be issued one assignment at a time even though they can be overlapped by pipelining. In contrast, in a quantum multicast network, packets themselves are made out of quantum bits and as such they represent a superposition of possible assignment patterns of packets that may be routed through such a network all at once through the principle of quantum parallelism. It is this quantum parallelism aspect of multicasting that is explored in our work. We extend Lee and Oruç's approach to the domain of quantum information processing to design an $n \times n$ generalized quantum connector, also called n-GQC. Since it is impossible to copy arbitrary quantum states due to the no-cloning theorem, multicasting of quantum states is also impossible. However, a certain set of orthogonal states can be copied by Wootters and Zurek's copying machine [1, 2]. For example, a controlled-not gate transforms the two-qubit state $(\alpha\,|0\rangle + \beta\,|1\rangle) \otimes |0\rangle$ to $\alpha\,|00\rangle + \beta\,|11\rangle$, which can be interpreted as follows: If a source has bit $|0\rangle$ that is to be copied with probability $|\alpha|^2$ and bit $|1\rangle$ to be copied with probability $|\beta|^2$ then the above operation does the required copying. We extend this to a controlled-not gate-based copy node that copies quantum information from a set of qubits to another set of qubits each of which is initialized to the blank state $|0\rangle$. This copying operation however, amounts to the copying of classical packets contained as a probabilistic superposition in a multi-qubit quantum state. Therefore, the n-GQC essentially multicasts *superposed* classical packets that will be referred to as *quantum packets* in this paper.

1.3 Quantum Concentrators

In the course of designing a quantum generalized connector (n-GQC), we introduce another quantum network, called an $n \times n$ quantum concentrator (n-QC). This network maps quantum states on any m of its inputs to its top m outputs, for any $m, 1 \leq m \leq n$. The n-QC is a self-routing multistage network that uses $O(n \log n)$ quantum gates, and has a gate-level depth of $O(\log^2 n)$. It plays a key role in the decomposition of a generalized quantum connector into smaller ones. As there is no copying involved in a concentrator, it can concentrate both arbitrary quantum states and superposed classical packets. The only other quantum concentrator in literature is the (n, m)-sparse crossbar quantum concentrator [4]. This network is based on classical sparse-crossbar concentrators given in [11, 12]. Its complexities in terms of number of gates and gate-level depth are $O(n(n-m+1))$ and $O(n+m)$ respectively.

It is known theoretically that a classical $n \times n$ concentrator can be constructed using $O(n)$ crosspoints and $O(\log n)$ delay [43–45]. However, explicit constructions either use sparse crossbars [11, 12, 46, 47] or $O(n \log n)$ multistage networks which are either based on binary sorting [48] or recursive decomposition of an n-concentrator into smaller ones [14,49]. Sparse crossbar based designs are not well suited for quantum information domain as they require external routing algorithms. The well known AKS sorting network [50] sorts in $O(\log n)$ time on an $O(n \log n)$ network. However, the constants involved in the complexity expressions are very large for small values of n. Chien and Oruç [48] give an adaptive binary sorting network which has $O(n \log n)$ cost and $O(\log^2 n)$ routing time. Lee and Oruç [14] have given an n-concentrator design which is self-routing and has the same cost and routing delay as the Chien and Oruç's concentrator. This network uses a splitting stage which distributes the input packets equally onto two $n/2$-concentrators whose outputs are then merged using a shuffle stage to achieve the concentration operation. Due to its self-routing property this network is most suitable for being used in the quantum information domain.

The n-QC is constructed by modifying Lee and Oruç's classical concentrator. The main bottleneck in transforming their concentrator to the quantum domain is the balancer network used in their design to distribute m packets from n input ports onto two $n/2$-input networks such that each $n/2$ network gets $m/2$ packets, for any $m, 1 \leq m \leq n$. Their balancer network is based on a binary tree on which bits are propagated in both forward and backward directions. We propose a reversible balancer called, an n-quantum balancer, on which data is propagated only in the forward direction. This quantum balancer then facilitates the design of the n-QC.

We also give several variations of n-QC. The order-preserving n-QC concentrates quantum packets while keeping them in the same order in which they appear on the inputs. The priority n-QC concentrates two classes of packets such that packets in one class are always above the ones in the other at the outputs. We also design a superposing quantum concentrator that uses quantum superposition to ensure that all input packets have non-zero probabilities of being concentrated among a fixed number of its top outputs. These quantum concentrators are used to introduce new features in n-GQC, and improve its routing behavior when realizing multicast assignments that have output contentions between packets. In the next section, we give a formal summary of the contributions made in this thesis, and discuss their significance.

1.4 List of Contributions and Significance

We explore the application of quantum information processing and computing concepts to the solutions of congestion and routing problems in interconnection networks, and design quantum switching networks that route both classical and quantum data. Some of our key contributions are listed below:

1. *Quantum packet switching model:* We introduce a framework to describe quantum packet switching network operations [3]. This model was used in [4] to illustrate the functioning of the sparse quantum crossbar concentrator. We introduce the concepts of quantum packets, assignment patterns, and assignments; and describe how they are routed by quantum switching networks.

2. *Quantum Baseline network (QBN):* This is the first quantum switching network that appears in the literature, and it was introduced by us in [5]. We presented an improved version of this network that requires significantly less auxiliary qubits in [6]. Prior to our work, there have been some results that map permutation and multicast assignments to quantum circuits [7,8]. Although related to switching qubits, these results cannot be viewed as quantum switching networks since they map each choice of permutation or multicast assignment to a different quantum circuit. The QBN is obtained by replacing the 2×2 switches in the classical Baseline network with quantum 2×2 switches that can work in a superposition of *through* and *cross*

states. This network routes all feasible packet permutations of a classical Baseline network in parallel. This is done without adding any extra routing or hardware complexity on top of the QBN. When routing classical packets, output contention at any internal switch is resolved by creating a superposition of both straight and through settings for that switch. Thus, the output quantum state of the network is a superposition of all the permutations that can be possibly generated by the classical Baseline network.

The QBN uses $O(n \log n)$ quantum gates, and its gate level depth is $O(\log n)$. When routing general quantum states, the QBN can realize only $n^{n/2}$ permutation assignments. After our work, other quantum unicast networks have been designed that have higher cost in terms of both number of gates and gate level depth, but can realize all permutation assignments when routing quantum states [9,10].

3. *Quantum concentrators:* We introduced the concept of a quantum concentrator in [3]. A quantum concentrator is the quantum analogue of a classical concentrator, and routes quantum data on some of its input to its top outputs. We designed a self-routing concentrator called n-QC that uses $O(n \log n)$ quantum gates, and has a gate level depth of $O(\log^2 n)$. The only other quantum concentrator design was given in [4], which is based on a n input-m output sparse-crossbar of [11,12]. This network uses $O(n(n-m+1))$ quantum gates, and has a gate level depth of $O(n+m)$. The n-QC is used to do a recursive decomposition of the GQC into smaller ones. We also give

several variations of the n-QC. The order-preserving QC (OPQC) concentrates input quantum packets to its top outputs consecutively while retaining their input order. The priority-QC (PQC) concentrates two classes of packets such that packets in one class are above the ones in the other at the outputs. The concepts introduced in these QC variations also apply to classical versions of these concentrators. The PQC is used to construct the priority generalized quantum connector (PGQC). We also introduce and design an $(n, n/2)$-superposing QC that uses the principle of quantum superposition to ensure that every input packet has a non-zero probability of being concentrated among its top $n/2$ outputs.

4. *Generalized quantum connector (GQC):* This is the first quantum multicasting network in the literature, and it was introduced by us in [3]. This network uses $O(n \log^2 n)$ quantum gates, and has a gate-level depth of $O(\log^3 n)$. The quantum merge-sorting network given in [9] has the same complexity in terms of number of gates, and has $O(\log^2 n)$ gate-level depth, but it realizes only unicast assignments. The only other result related to quantum multicasting is given by Tsai and Kuo [8]. However, their result cannot be viewed as a switching network, as it requires a different quantum circuit for every multicast assignment. Another work on quantum multicasting, [13], explores the benefits of network coding while multicasting quantum states, and does not design a switching network.

The GQC is designed based on the classical generalized connector given in [14]. The main challenge in transforming this network to the quantum domain was the construction of a quantum concentrator. The concentrator used in [14] has both forward and backward propagation of data, which cannot be done in a quantum network. The GQC multicasts superposed classical packets contained in qubits, and used controlled-not gate based copiers to generate copies of the packets. We also analyze the behavior of GQC when routing multicast assignments that have output contentions, and show that the GQC is not work-conserving when routing such assignments, i.e., some outputs may not receive any packets even when there are input packets addressed to them. We then give two variations of GQC that improve its behavior when routing contending assignments. The priority GQC (PGQC) introduces prioritized routing in GQC, and ensures that higher priority packets reach their outputs in case of output contentions. The superposing-GQC (SGQC) creates superpositions of contending packets on their desired output, such that every packet has a non-zero probability of reaching its desired output. The GQC can also be used to do approximate multicasting of general quantum states by using advanced quantum copiers [2,15–18] which can generate imperfect copies of any general quantum state.

Chapter 2: Quantum Information Processing and Packet Switching

Much of the findings of this thesis rests on the quantum information processing field that has been developed during the last two decades. In this chapter, we review and describe the quantum information processing concepts that will be used in the rest of the thesis. We also review some of the recent results in quantum copying which form the basis of quantum multicasting. Finally, we introduce our quantum switching network model, and give the terminology that will be used to describe the operation of our networks.

2.1 Background on Quantum Information Processing

We start out with an introduction to a quantum bit or a qubit and some basic quantum gates.

2.1.1 Qubits and Superposition

The indivisible unit of classical information is the *bit* which can take either one of two values: 0 or 1. The corresponding unit of quantum information is the quantum bit or *qubit* which can simultaneously be both 0 and 1. In general, a qubit's state is an unit vector in two-dimensional complex Hilbert space and can be expressed as $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^2 + |\beta|^2 = 1$. Vectors $|0\rangle$ and $|1\rangle$ are called computational bases. State $|\psi\rangle$ is also represented as a column vector $[\alpha\ \beta]^T$. On measurement, in which the qubit's state is projected onto the computational basis, the qubit is observed to be found either in state $|0\rangle$ or in state $|1\rangle$ with probability $|\alpha|^2$ and $|\beta|^2$ respectively. This act of measurement collapses the qubits state to either $|0\rangle$ or $|1\rangle$ depending on the outcome of the measurement.

The state of an n-qubit system is a vector in a 2^n-dimensional complex Hilbert space which is a tensor product of the two-dimensional spaces associated with individual qubits. In general, we can express an n-qubit state in a 2^n-dimensional complex Hilbert space as

$$\left|\bar{\psi}\right\rangle = \alpha_0\,|00\cdots 0\rangle + \alpha_1\,|00\cdots 1\rangle + \cdots + \alpha_{2^n-1}\,|11\cdots 1\rangle\,, \qquad \text{where } \alpha_i \in \mathbb{C} \text{ and } \sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1. \quad (2.1)$$

where the n-bit vectors $|00\cdots 0\rangle\,, \cdots, |11\cdots 1\rangle$ constitute the bases of the space and are called computational bases. State $\left|\bar{\psi}\right\rangle$ can also be represented as a column vector: $[\alpha_0\ \alpha_1\ \cdots\ \alpha_{2^n-1}]^T$.

Similar to the case of a single qubit system, if all the qubits in the n-qubit system are measured by projecting onto the computational bases, n-bit string $|\bar{i}\rangle$ is observed with probability $|\alpha_i|^2$, where $i = 0, \cdots, 2^n - 1$. We use the *bar* notation

$\left|\bar{\psi}\right\rangle$ to denote a multi-qubit vector or bit string in this thesis. A single qubit state is denoted without the *bar* as $|\psi\rangle$.

2.1.2 Quantum Gates

The state of qubits can be transformed via quantum gates and circuits made using such gates. These gates are linear and unitary transformations in the state space of the qubits. A general n-qubit quantum gate is represented by an unitary transformation matrix U (where $U^H U = UU^H = I$, U^H denotes the conjugate transpose of U) in the 2^n-dimensional complex Hilbert space associated with n qubits. This gate transforms an n-qubit quantum state $\left|\bar{\psi}\right\rangle$ as:

$$\left|\bar{\psi}'\right\rangle = U\left|\bar{\psi}\right\rangle \tag{2.2}$$

Unitarity implies that quantum gates are *reversible*, i.e., it is possible to uniquely identify the input state from a given output state. An n-qubit quantum gate that maps one computational basis state to another, should be one-to-one due to reversibility. This means that a quantum gate, which transforms by mapping among computational bases, should transform every n-bit input string to an unique n-bit output. Linearity implies that an n-qubit state $\left|\bar{\psi}\right\rangle = \sum \alpha_i \left|\bar{\phi}_i\right\rangle$, is transformed to $\sum_{i=0}^{2^n-1} \alpha_i U \left|\bar{\phi}_i\right\rangle$. This property of quantum gates is a source of massive parallelism, since all of the 2^n components of a general n-qubit state are transformed simultaneously by an n-qubit quantum gate. Most of the well known quantum algorithms exploit this parallelism to obtain speedups that are not possible in classical domain.

This same property of quantum gates allows us to do a probabilistic multiplexing of packets in quantum switching networks. We can form a superposition of several packet assignments at the inputs of a quantum switching network, and all of these assignments can be realized simultaneously due to quantum parallelism [6]. The quantum switching network functions in different configuration for different packet assignments in the superposition.

The simplest example of an one-qubit quantum gate is the NOT gate which transforms state $|0\rangle$ to $|1\rangle$ and state $|1\rangle$ to $|0\rangle$. Due to linearity, a general quantum state $\alpha|0\rangle + \beta|1\rangle$ is transformed by the NOT gate to state $\alpha|1\rangle + \beta|0\rangle$. Next, we describe some quantum gates which will be used in this thesis.

Another one-qubit gate that is often used to create superposition of quantum states is a Hadamard gate. This gate transforms the basis vectors $|0\rangle$ and $|1\rangle$ as:

$$|0\rangle \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \;\; |1\rangle \rightarrow \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right). \tag{2.3}$$

Therefore, its transformation matrix can be expressed as:

$$\mathbf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.4}$$

Due to linearity, the Hadamard gate transforms a general state $\alpha|0\rangle + \beta|1\rangle$ to

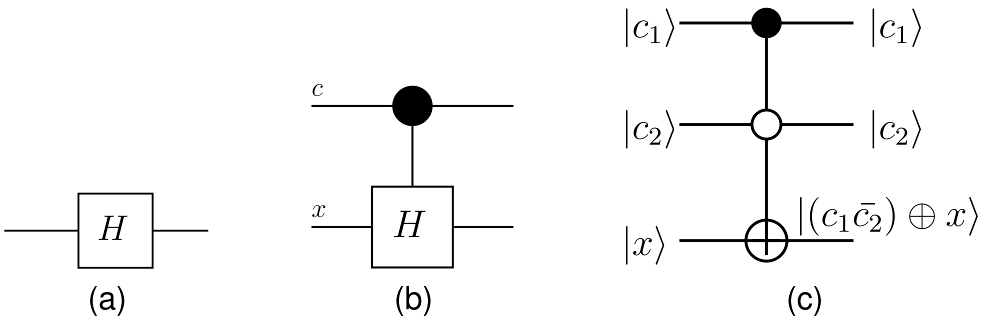


Figure 2.1: Quantum gates: (a) Hadamard gate; (b) controlled-Hadamard gate; (c) A controlled-controlled not gate.

$\frac{\alpha+\beta}{\sqrt{2}} |0\rangle + \frac{\alpha-\beta}{\sqrt{2}} |1\rangle$. This gate is shown in Figure 2.1(a).

Any quantum gate can be extended by adding one or more control qubits so that the gate operates on its input or target qubits when the control qubits are in a certain state. Otherwise, it passes the target qubits unchanged. The control qubits always remain in their original state. Such gates are called controlled quantum gates.

As an example, a controlled-not gate having two control qubits (c_1 and c_2) is shown in Figure 2.1(c). This gate does the following operation

$$|c_1, c_2, x\rangle \xrightarrow{\textbf{CC-NOT}} |c_1, c_2, (c_1.\bar{c_2}) \oplus x\rangle \tag{2.5}$$

i.e., it inverts x when $c_1 = 1$ (indicated by solid circle) and $c_2 = 0$ (indicated by open circle). Therefore, it maps basis vector $|100\rangle$ to $|101\rangle$ and $|101\rangle$ to $|100\rangle$. Rest of the basis vectors are passed unchanged. We follow this notation of solid and open circles to indicate the functioning of control qubits in the rest of the thesis.

Similarly, a controlled-Hadamard gate shown in Figure 2.1(b) applies the

Hadamard transformation to the target qubit x when the control qubit is in state $|1\rangle$, otherwise it leaves the target qubit unchanged. Thus, input states $|0\rangle_c |0\rangle_x$ and $|0\rangle_c |1\rangle_x$ are not affected by this gate. However, states $|1\rangle_c |0\rangle_x$ and $|1\rangle_c |1\rangle_x$ are transformed as

$$\begin{aligned} |1\rangle_c |0\rangle_x &\rightarrow 1/\sqrt{2}\,(|1\rangle_c |0\rangle_x + |1\rangle_c |1\rangle_x) \\ |1\rangle_c |1\rangle_x &\rightarrow 1/\sqrt{2}\,(|1\rangle_c |0\rangle_x - |1\rangle_c |1\rangle_x) \end{aligned} \tag{2.6}$$

As an example, if qubit c is in state $\alpha|0\rangle + \beta|1\rangle$, and qubit x is in state $|1\rangle$, then the controlled-Hadamard gate transforms input state $(\alpha|0\rangle + \beta|1\rangle)_c \otimes |1\rangle_x$ as

$$\alpha|01\rangle + \beta|11\rangle \rightarrow \alpha|01\rangle + \frac{\beta}{\sqrt{2}}|10\rangle - \frac{\beta}{\sqrt{2}}|11\rangle \tag{2.7}$$

Unlike controlled-not gate, it is not possible to express the transformation of a Hadamard or a controlled-Hadamard gate using simple logical operations. This is why we have left out output symbols in Figures 2.1(a) and (b).

The basic building block of quantum switching networks is a multi-qubit gate called a controlled-swap gate or *switch gate*, which is shown in Figure 2.2(a) [6] [51] [52]. It swaps two quantum packets or sets of qubits when a control qubit c is $|1\rangle$, otherwise it passes them unchanged. Therefore, it can be used as a 2×2 switch for routing quantum information. The transformation performed by this

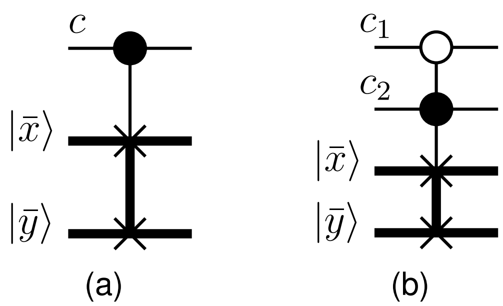


Figure 2.2: Switch gate: (a) Switch gate representations; (b) Switch gate with multiple control qubits.

gate can be expressed as

$$\begin{aligned} |0\rangle_c |\bar{x}\rangle |\bar{y}\rangle &\xrightarrow{\textbf{SWG}} |0\rangle_c |\bar{x}\rangle |\bar{y}\rangle \\ |1\rangle_c |\bar{x}\rangle |\bar{y}\rangle &\xrightarrow{\textbf{SWG}} |1\rangle_c |\bar{y}\rangle |\bar{x}\rangle \end{aligned} \tag{2.8}$$

where $|\bar{x}\rangle$ and $|\bar{y}\rangle$ are equal-length binary strings. A switch gate can have multiple control inputs. For example, the switch gate shown in Figure 2.2(b) swaps its input packets when control qubits c_1 and c_2 are $|0\rangle$ and $|1\rangle$ respectively, otherwise it passes them.

The switch gate can also be used to superpose the packets that contend for one output of a 2×2-switch and to route the superposition on that output [5] [6]. For example, when $n = 1$, if the control qubit of the gate is set in an equal superposition of states $|0\rangle$ and $|1\rangle$ then the action of the gate is: $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|x\rangle|y\rangle \longrightarrow \frac{1}{\sqrt{2}}(|0\rangle|x\rangle|y\rangle + |1\rangle|y\rangle|x\rangle)$. Thus an equal superposition (probability of observation = $1/2$) of packets x and y is created at both the outputs. Also, if we observe packet x at one of the outputs then packet y will be observed with certainty at the other and vice-versa.

While quantum switch gates are essential to perform permutation maps, we need quantum copiers to replicate either quantum states or the classical components of multi-qubit quantum states. Before describing the quantum copiers used in this thesis, we introduce the density matrix representation of quantum states which makes it easier to characterize quantum copying operations.

2.1.3 Density Matrix Representation of Quantum States

There is another useful formulation for representing quantum states, called the density matrix or density operator representation, which will be used in our work to describe quantum copying operations in a quantum multicasting network. This representation is also very useful to describe the behavior of a subset of qubits in a large quantum system. Suppose that a system of qubits is in pure quantum state $|\psi\rangle$ then its density matrix is given by the outer product $|\psi\rangle \langle\psi|$, where $\langle\psi|$ is the complex conjugate transpose of $|\psi\rangle$. If we represent an n qubit state $|\psi\rangle$ in computational basis as $|\psi\rangle = \sum_{i=0}^{2^n-1} a_i |i\rangle$, then the corresponding density matrix can be expressed as:

$$\rho = |\psi\rangle \langle\psi| = \sum_{i=0}^{2^n-1} \sum_{j=0}^{2^n-1} a_i a_j^* |i\rangle \langle j| \tag{2.9}$$

On the other hand, if the state of a quantum system is not completely known and it is only known that it exists in state $|\psi_i\rangle$ with probability p_i, then we express the density matrix of this system as:

$$\rho = \sum_i p_i \left|\psi_i\right\rangle \left\langle\psi_i\right| \tag{2.10}$$

The density operator is always positive, i.e., for an arbitrary vector ψ, $\langle\psi|\,\rho\,|\psi\rangle \geq 0$. Also, trace of the density operator is always equal to 1. If the combined density matrices of two quantum systems A and B is ρ_{AB}, then the density matrix of system A can be obtained by taking the partial trace of ρ_{AB} over system B, i.e.,

$$\rho_A = Tr_B\left(\rho_{AB}\right) \tag{2.11}$$

The evolution of a quantum system by unitary operator U transforms its density matrix ρ as:

$$\rho' = U\rho U^\dagger \tag{2.12}$$

where $U^\dagger$ is the conjugate transpose or hermitian of U. For a detailed description of quantum information systems we refer the reader to [52] and [51].

2.1.4 No-Cloning Theorem and Quantum Copying

A key requirement for a quantum multicasting networks is a copy node, which can produce an identical copy of a quantum packet or an arbitrary quantum state $|\psi\rangle$. Such a node does not exist due to Wootters and Zurek's no-cloning theorem [1] which states that an arbitrary quantum state cannot be cloned, i.e., it is impossible to have an unitary operation that can do following transformation:

$$|\psi\rangle\,|s\rangle \longrightarrow |\psi\rangle\,|\psi\rangle \tag{2.13}$$

for a pure state $|\psi\rangle$ and a 'blank paper' state $|s\rangle$. This theorem has been extended to arbitrary mixed states as well [53]. However, it has been recently shown by Bužek and Hillery that approximate cloning of quantum states is possible [2]. A copying network made using quantum gates which produces two imperfect copies of an arbitrary qubit has also been proposed by Bužek et. al. [15]. In this section, we give a brief description of these quantum copying approaches. Before doing that, we discuss some performance measures which are used to quantify the quality of copies produced.

2.1.4.1 Performance Measures for Quantum Copying

Bužek and Hillery used the Hilbert-Schmidt distance between the density matrices of the output states produced by a quantum copier and the states that would be produced by an ideal quantum copier [2]. They showed that it is a good measure in the sense that if two one-qubit quantum states are closer in terms the Hilbert-Schmidt distance then their measurement probability distributions for an arbitrary observable are also close. Hilbert-Schmidt norm of an operator or density matrix ρ is defined as

$$\|\rho\|_2 = \left(Tr(\rho^\dagger \rho)\right)^{1/2} \tag{2.14}$$

Using this norm, the distance between two density matrices ρ_1 and ρ_2 is defined as

$$D(\rho_1, \rho_2) = (\|\rho_1 - \rho_2\|_2)^2 \tag{2.15}$$

The Hilbert-Schmidt distance, however, is not a very good measure to compare density matrices in higher dimensions, i.e., for multi-qubit density matrices. Another measure that is widely used is the *fidelity* of quantum states [54,55]. For two density matrices ρ_1 and ρ_2, fidelity is defined as:

$$F(\rho_1, \rho_2) = Tr\left(\left(\rho_1^{1/2}\rho_2\rho_1^{1/2}\right)^{1/2}\right) \tag{2.16}$$

It ranges between 0 and 1 and is equal to 1 when $\rho_1 = \rho_2$. These performance measures are used to study the performance of quantum multicasting networks in Chapter 5.

2.1.4.2 Wootters and Zurek's Quantum Copying Machine

Wootters and Zurek's quantum copying machine (WZQCM) [1,2] copies a certain set of orthogonal basis states perfectly, but produces imperfect copies of any other states. This machine transforms basis states $|0\rangle_a$ and $|1\rangle_a$ of a qubit a as:

$$\begin{aligned} &|0\rangle_a |Q\rangle_x \longrightarrow |0\rangle_a |0\rangle_b |Q_0\rangle_x \\ &|1\rangle_a |Q\rangle_x \longrightarrow |1\rangle_a |1\rangle_a |Q_1\rangle_x \end{aligned} \tag{2.17}$$

where $|Q\rangle_x$ is the initial state of the copier and ${}_x\langle Q|Q\rangle_x = {}_x\langle Q_0|Q_0\rangle_x = {}_x\langle Q_1|Q_1\rangle_x = 1$. Qubit b is the copy qubit. A general state $|\psi\rangle_a = \alpha\,|0\rangle_a + \beta\,|1\rangle_a$ is copied as:

$$|\psi\rangle_a |Q\rangle_x \to \alpha\,|0\rangle_a |0\rangle_b |Q_0\rangle_x + \beta\,|1\rangle_a |1\rangle_b |Q_0\rangle_x \tag{2.18}$$

However, if qubit b was an ideal copy of a then the final state of the two qubits should have been $\alpha^2 |0\rangle_a |0\rangle_b + \alpha\beta\,|0\rangle_a |1\rangle_b + \alpha\beta\,|1\rangle_a |0\rangle_b + \beta^2 |1\rangle_a |1\rangle_b$, which is clearly not the case. By computing the density matrix of the resulting state on the right hand side of (2.18), and then taking partial trace over qubit a and copier x, the density matrix of qubit b is (assuming α and β are real):

$$\rho_b = \alpha^2 |0\rangle_b\, {}_b\langle 0| + \beta^2 |1\rangle_b\, {}_b\langle 1| \tag{2.19}$$

which is also equal to the density matrix of qubit a after copying operation. Thus, we see that the off-diagonal elements in the density matrix of both qubits a and b are destroyed by this copier. The Hilbert-Schmidt distance between the density matrices of the original and copy qubits is [2]:

$$D_a = 2\alpha^2\beta^2 \tag{2.20}$$

which shows that the WZQCM copies states $|0\rangle$ and $|1\rangle$ perfectly and generates error for any other source state. The error is also variable and dependent on source state. Also, WZQCM creates a strong entanglement between source and copy qubits, and a measurement on one of them collapses the state of the other as well, which should not happen for an ideal quantum copier.

In our quantum multicasting networks, we mainly route classical packets encoded using qubits. For such networks, we do not need to copy general quantum states, and only the classical components of these states need to be replicated. This is done by using a collection of controlled-not gates, as shown in Figure 2.3(a) by which a set of source qubits are copied to a set of target qubits initialized to state $|0\rangle$. We represent these gates collectively as one controlled-not gate using bold lines, as shown in Figure 2.3(b). The copying operation performed by this gate is: $|\bar{x}\rangle_s |\bar{0}\rangle_t \xrightarrow{\textbf{COP}} |\bar{x}\rangle_s |\bar{x}\rangle_t$. As an example, two source qubits in state $1/\sqrt{2}\,|00\rangle_s + 1/2\,|10\rangle_s + 1/2\,|11\rangle_s$ are transformed as:

$$\left(\frac{1}{\sqrt{2}}\,|00\rangle_s + \frac{1}{2}\,|10\rangle_s + \frac{1}{2}\,|11\rangle_s\right)|00\rangle_t \xrightarrow{\textbf{COP}} \frac{1}{\sqrt{2}}\,|00\rangle_s\,|00\rangle_t + \frac{1}{2}\,|10\rangle_s\,|10\rangle_t + \frac{1}{2}\,|11\rangle_s\,|11\rangle_t \tag{2.21}$$

that shows that a copy of each component of a quantum state is created. This copier, which is essentially a WZQCM, is exactly what we need when multicasting superposed classical data using a quantum switching network.

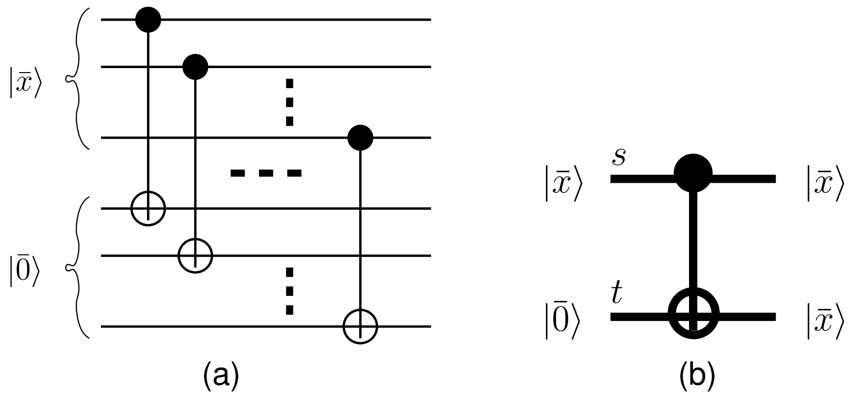


Figure 2.3: Quantum copier: (a) Controlled-not gates based quantum copier; (b) Compact representation of the quantum copier.

2.1.4.3 Universal Quantum Copying Machine (UQCM)

Bužek and Hillery proposed an universal quantum copying machine (UQCM) which copies all one qubit states equally well. The transformation implementing UQCM is:

$$\begin{aligned}
|0\rangle_a |Q\rangle_x &\rightarrow \sqrt{\frac{2}{3}}\, |00\rangle_{ab} |\uparrow\rangle_x + \sqrt{\frac{1}{3}}\, |+\rangle_{ab} |\downarrow\rangle_x \\
|1\rangle_a |Q\rangle_x &\rightarrow \sqrt{\frac{2}{3}}\, |11\rangle_{ab} |\downarrow\rangle_x + \sqrt{\frac{1}{3}}\, |+\rangle_{ab} |\uparrow\rangle_x
\end{aligned} \tag{2.22}$$

where $|+\rangle_{ab} = \frac{1}{\sqrt{2}}\left(|10\rangle_{ab} + |01\rangle_{ab}\right)$, $|-\rangle_{ab} = \frac{1}{\sqrt{2}}\left(|10\rangle_{ab} - |01\rangle_{ab}\right)$, $|\uparrow\rangle$ and $|\downarrow\rangle$ are two orthonormal basis states in two dimensional state space and $|Q\rangle_x$ is the initial copier state.

The transformation parameters of UQCM have been chosen in such a way that the Hilbert-Schmidt distance between the density matrices of copy qubit b and the source qubit a is constant, irrespective of the source state. These parameters ensure that the Hilbert-Schmidt distance between density matrices ρ_{ab}^{out} and

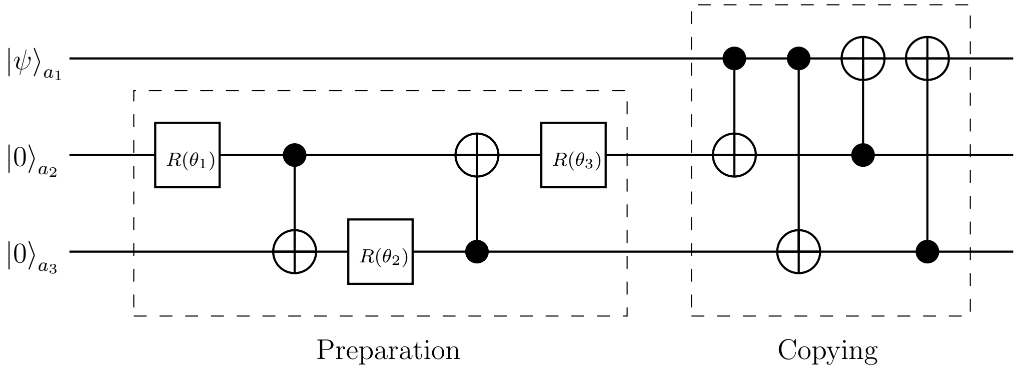


Figure 2.4: Quantum circuit for UQCM.

ρ_{ab}^{id} is also independent of the source state, where ρ_{ab}^{out} is the density matrix of the output state of $UQCM$ and ρ_{ab}^{id} is the output density matrix of an ideal quantum copier. If the source qubit state is $|\psi\rangle_a = \alpha\,|0\rangle_a + \beta\,|1\rangle_a$ then the reduced density matrix of both the source and copy qubits at the output of UQCM is:

$$\rho_a = \frac{5}{6}\,|\psi\rangle_a\ {}_a\langle\psi| + \frac{1}{6}\,|\psi_\perp\rangle_a\ {}_a\langle\psi_\perp| \tag{2.23}$$

where $|\psi_\perp\rangle_a = \beta^*\,|0\rangle_a - \alpha^*\,|1\rangle_a$. This shows that UQCM retains $5/6$ of the original state in both source and copy qubits. However, the resulting source and copy qubits are highly entangled, and a measurement on one of the qubits affects the state of the other [2].

A quantum network that implements UQCM has also been proposed [15]. Figure 2.4 shows this network which is formed using controlled-NOT gates and single qubit rotation gates. A single qubit rotation gate $R(\theta)$ with rotation angle

θ performs the following transformation:

$$\begin{aligned} |0\rangle &\to \cos(\theta)\,|0\rangle + \sin(\theta)\,|1\rangle \\ |1\rangle &\to -\sin(\theta)\,|0\rangle + \cos(\theta)\,|1\rangle \end{aligned} \tag{2.24}$$

The UQCM network consists of two parts. The first part prepares two qubits a_2 and a_3 in a specific state by choosing parameters θ_1, θ_2 and θ_3 appropriately. The second part is the copying stage that copies the source state $|\psi\rangle_{a_1}$ from qubit a_1. It was shown in [15] that by setting $\theta_1 = \theta_3 = \pi/8$ and $\theta_2 = -\sin^{-1}\left(1/2 - \sqrt{2}/3\right)^{1/2}$ this network behaves as UQCM and the two copies at the output are qubits a_2 and a_3 respectively.

2.2 Quantum Packet Switching

In this section, we introduce the terminology that will be used to describe how both classical and quantum data is routed through quantum packet switching networks. We develop a framework to represent quantum packets and assignments and discuss how they are switched via such networks. Each input of a quantum packet switching network may contain multiple classical packets in a quantum superposition where each packet is present with a certain probability. The switching elements in a quantum switching network function in a probabilistic superposition of multiple settings, and route these packets on different routes. Before giving a description of such networks, we introduce *stochastic networks*

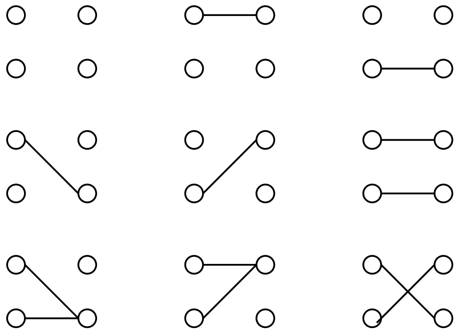


Figure 2.5: States of a deterministic 2x2 unicast switch.

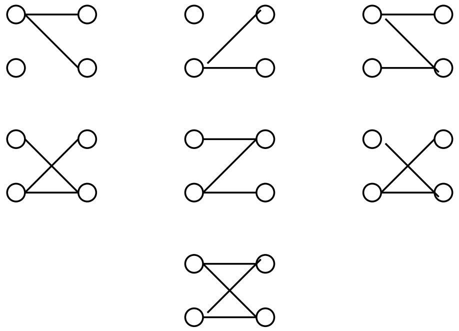


Figure 2.6: States of a stochastic 2x2 unicast switch.

which are classical analogues of quantum switching networks.

2.2.1 Stochastic Switching

Switching operations can generally be viewed as mappings from a set of callers to a set of receivers. When this mapping is such that each caller is mapped to at most one receiver then it is referred as a *deterministic switching map*. Otherwise, it called a *non-deterministic* or *stochastic switching map*. There exist $(1+m)^n$ deterministic switching maps between n callers and m receivers, and $2^{mn}-(1+m)^n$ stochastic switching maps. For $n=m=2$, there are 9 deterministic maps and 7 stochastic maps, which are shown in Figures 2.5 and 2.6 respectively.

There are two interpretations of one caller connecting to two receivers in a stochastic switch. One is that the the caller may call only one of the two receivers randomly with certain probabilities, but not both. The second interpretation is that the caller multicasts to both outputs. In either case, the set of callers that can reach a given receiver can be expressed by using the *connection matrix* of a stochastic switch. For a $n \times m$ stochastic switch, the connection matrix is an $n \times m$ matrix in which j^{th} element in the i^{th} row is 1 if receiver i can be reached by caller j. For example, the connection matrix for the second stochastic switch in the middle row of Figure 2.6 is

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

Defining matrix multiplication as union of intersections, the set of inputs that can reach the top and bottom outputs can be computed as

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \text{a} \\ \text{b} \end{bmatrix} = \begin{bmatrix} \text{a,b} \\ \text{a} \end{bmatrix} \tag{2.25}$$

A *stochastic multistage switching networks* consists of several stages of stochastic switches connected using deterministic connection patterns. An example of such a network is shown in Figure 2.7. The connection matrix of a stage of a multistage stochastic network can be expressed by placing the connection matrices of the stochastic switches in the stage along the diagonals and filling other entries with

0. For example, the connection matrix of the first stage of the shown network is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \tag{2.26}$$

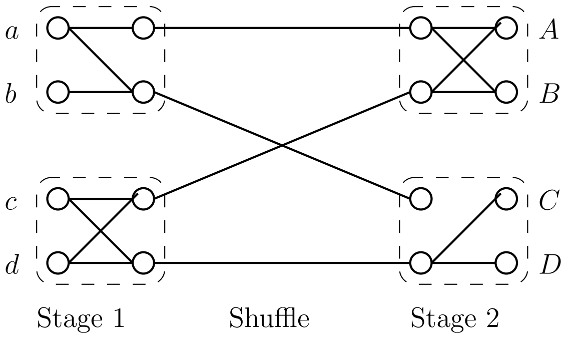


Figure 2.7: A stochastic multistage network.

The connection matrix for the overall stochastic network can be computed my multiplying the connection matrices of its stages and the deterministic connections between the stages, where matrix multiplication is defined as union of intersections.

$$\underbrace{\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{stage2}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{shuffle}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}}_{\text{stage1}} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \tag{2.27}$$

The possible mappings realized by this network are computed as

$$\begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \text{a} \\ \text{b} \\ \text{c} \\ \text{d} \end{bmatrix} = \begin{bmatrix} \text{a, c, d} \\ \text{a, c, d} \\ \text{c, d} \\ \text{c, d} \end{bmatrix} \tag{2.28}$$

The connection matrix representation of a stochastic network can be used to specify which inputs can reach a given output. In quantum switching networks, internal switching elements can work in a superposition of different settings of stochastic switches. Consequently, connection matrices can be employed to specify all packets that exists in a superposition at a given output. Connection matrices do not identify all of the mappings that can realized by the network. For example, the mapping (c, d, c, d) can never be realized by the shown stochastic network.

The most general 2×2 stochastic 2×2 switch is the one shown in the third row of Figure 2.6 in which both callers can reach both receivers. If each caller in this switch independently chooses its receiver then there is a probability of blocking or congestion. We can avoid such collisions by setting the stochastic switch only in non-blocking states. For example, the stochastic switch may be set in *through* or *cross* states only, as shown in Figure 2.8(a). Similarly, the switch may be set in one of the multicast states only with probabilities ϵ and $1 - \epsilon$ as shown in Figure 2.8(b). In the most general case, the switch may be set in one of

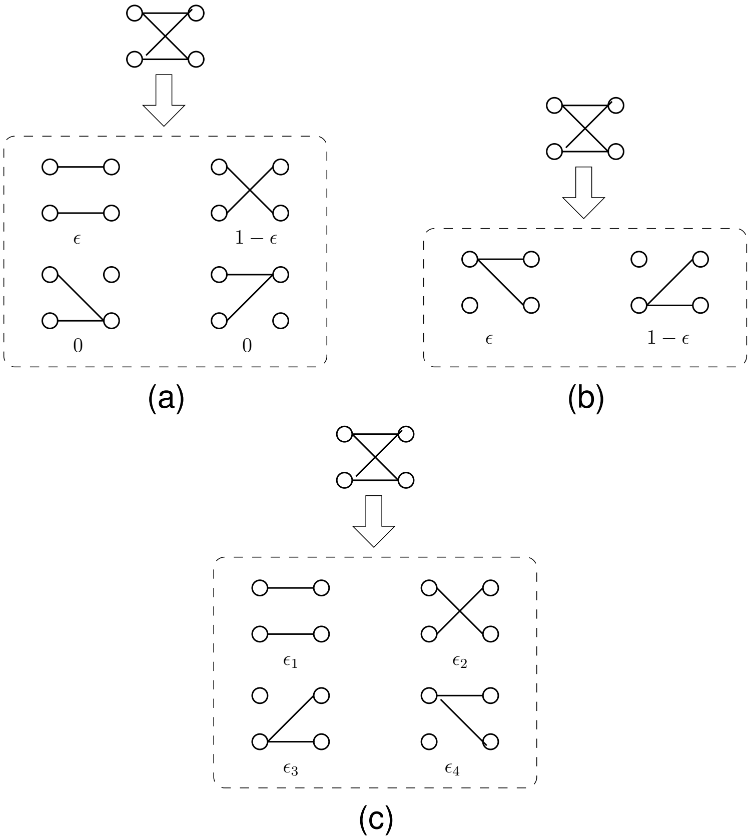


Figure 2.8: Non-blocking 2×2 stochastic switches: (a) Unicast: sets in through or cross state (b) Multicast: sets in one of the two multicast states (c) General 2×2 non-blocking stochastic switch.

the two unicast states, or two multicast states with probabilities ϵ_1, ϵ_2, ϵ_3 and ϵ_4 respectively, as shown in Figure 2.8(c), where $\epsilon_1 + \epsilon_2 + \epsilon_3 + \epsilon_4 = 1$.

Quantum circuits can not only realize such stochastic switches, but can also operate in a superposition of the shown non-blocking states simultaneously. For example, following quantum transformation realizes the stochastic switch shown in Figure 2.8(a).

$$|ab\rangle \rightarrow \alpha\,|ab\rangle + \beta\,|ba\rangle \tag{2.29}$$

where α and β are chosen such that $|\alpha|^2 = \epsilon$ and $|\beta|^2 = 1 - \epsilon$. When both inputs of the switch consist of one bit only, following 2-qubit quantum gate realizes this switch for $\epsilon = 1/2$:

$$\begin{aligned} |00\rangle &\rightarrow |00\rangle\,,\ |11\rangle \rightarrow |11\rangle \\ |01\rangle &\rightarrow \frac{1}{\sqrt{2}}\left(|01\rangle + |10\rangle\right) \\ |10\rangle &\rightarrow \frac{1}{\sqrt{2}}\left(|01\rangle - |10\rangle\right) \end{aligned} \tag{2.30}$$

This gate can be constructed using hadamard and control-not gates, as shown in Chapter 3. Quantum circuit realization of the 2×2 multicast stochastic switches shown in Figures 2.8(b) and 2.8(c) is not trivial, and auxiliary qubits are needed to accomplish copying in such switches.

2.2.2 Quantum Packet Switching Definitions

In this section, we define quantum packets, assignments and assignment patterns which would be used to describe the operation of quantum packet switches. We also provide a high level description of our quantum packet switching model using these terms.

2.2.2.1 Quantum Packets and Assignments

A quantum packet consists of two sets of qubits, called the *address field* and *data* qubits, respectively, and an extra qubit, called a *routing* qubit that indicates the presence (when set to $|1\rangle$) or absence (when set to $|0\rangle$) of a quantum packet on the corresponding address field and data qubits. The address field contains routing information that is used to route the data qubits.

Definition 2.1 (Quantum packet). A *quantum packet*, consisting of k classical packets in which packet i has routing bit r_i, address field $\bar{a}_i$ and data $\bar{d}_i$, for $i = 0, 1, \cdots, k-1$, is represented as:

$$\sum_{i=0}^{k-1} \alpha_i \left| r_i, \bar{a}_i, \bar{d}_i \right\rangle \tag{2.31}$$

where each $\left| r_i, \bar{a}_i, \bar{d}_i \right\rangle$ denotes a classical packet with probability $|\alpha_i|^2$ and it is to be multicast to the set of outputs specified by its address field $\bar{a}_i$ when r_i is 1. The set of outputs specified by the address field $\bar{a}_i$ is called the *fanout set* of packet i and is represented as F_i. The size of the fanout set is called the *fanout* of the packet i. When r_i is 0, packet $\left| r_i, \bar{a}_i, \bar{d}_i \right\rangle$ is considered to be empty, i.e., even though the packet exists, a quantum switching network routing this packet will ignore its address and data bits.

We give an example to make the definition of a quantum packet more clear. Consider a 4×4 network, in which an input has two packets A and B, which are to be routed with probabilities $3/4$ and $1/4$, respectively. Suppose that the fanout sets of A and B are $\{1, 3\}$ and $\{1\}$ respectively. We use a 4-bit address field, $o_0o_1o_2o_3$, where o_i is set to 1 when the fanout set of a packet contains output i. The quantum packet on this input is a superposition of two classical packets, expressed as:

$$\frac{\sqrt{3}}{2}\,|1,0101,A\rangle + \frac{1}{2}\,|1,0010,B\rangle \tag{2.32}$$

In general, at least n bits are needed to specify the destinations of an input in an $n \times n$ multicast network, as noted in [14]. This is because an input can have up to 2^n destination patterns.

There is more than one possible interpretation of this quantum packet representation. We mention two such interpretations. One is that if A and B denote the same packet then this packet will likely be routed to outputs 1 and 3 with probability $3/4$ and to output 2 with probability $1/4$. The second interpretation is that the input source is likely to generate one of two different packets. One of these two packets is generated and routed to outputs 1 and 3 with probability $3/4$ and the other is generated and routed to output 2 with probability $1/4$. Nonetheless, in both interpretations, either a packet appears at both outputs 1 and 3 with probability $3/4$ or at output 2 only with probability $1/4$. Therefore, for consistency of our statements, we shall assume the second interpretation.

Definition 2.2 (Assignment pattern and quantum assignment). An *assignment pattern* over an $n \times n$ quantum switching network is a sequence of classical packets, each of which belongs to a quantum packet on a distinct input of the network from top to bottom. We say that an assignment pattern is *non-contending* when no two classical packets with routing bits of 1 in the pattern are addressed to the same output. A *quantum assignment* on an $n \times n$ quantum switching network is a superposition of a set of assignment patterns. A quantum assignment is called

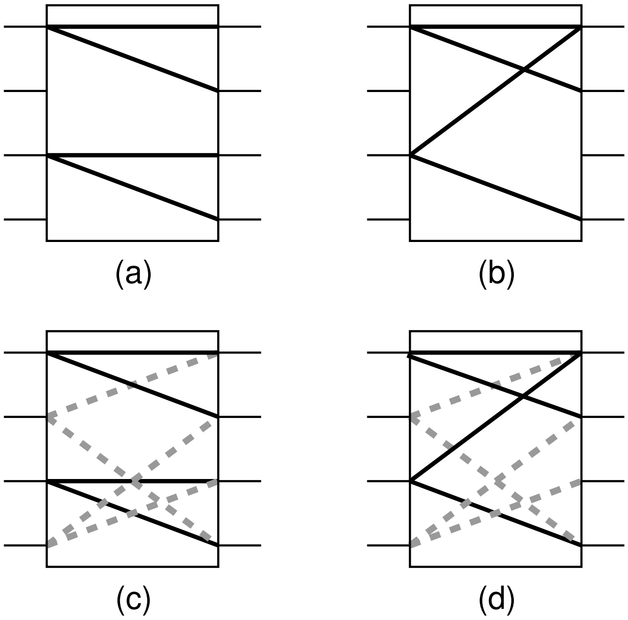


Figure 2.9: *Assignment patterns:* (a) non-contending assignment pattern; (b) contending assignment pattern; *Quantum assignments:* (solid and dashed lines show two assignment patterns) (c) non-contending quantum assignment; (d) contending quantum assignment.

non-contending if and only if all of its assignment patterns are non-contending; and called *contending* otherwise. These definitions are illustrated in Figure 2.9. An assignment pattern is said to be *unicast* when each classical packet is addressed to at most one output, otherwise, it is said to be *multicast*. A non-contending unicast assignment pattern in which every input is paired with some output is said to be a *permutation* assignment pattern.

Definition 2.3 (**Quantum M-assignment).** A quantum assignment consisting of a superposition of a set of M assignment patterns on an $n \times n$ quantum switching network is called a *quantum M-assignment* and can be expressed as:

$$\sum_{k=0}^{M-1} \beta_k \left|(r_{k,0}, \bar{a}_{k,0}, \bar{d}_{k,0}) \cdots (r_{k,n-1}, \bar{a}_{k,n-1}, \bar{d}_{k,n-1})\right\rangle \tag{2.33}$$

where assignment pattern $\left|\bar{P}_k\right\rangle$ = $|(r_{k,0}, \bar{a}_{k,0}, \bar{d}_{k,0})$, $\cdots$, $(r_{k,n-1}, \bar{a}_{k,n-1}, \bar{d}_{k,n-1})\rangle$ is a sequence of classical packets in which $(r_{k,i}, \bar{a}_{k,i}, \bar{d}_{k,i})$ is the classical packet on input i. The probability with which the k^{th} assignment pattern is realized is $|\beta_k|^2$, where $\sum_{k=0}^{M-1}|\beta_k|^2 = 1$.

When all the inputs have quantum packets of the form given in (2.31), the expression for the corresponding quantum assignment can be obtained by taking the tensor product of all the input quantum packets.

We illustrate this by considering the example of a 4×4 network in which input 0 issues the quantum packet $\frac{\sqrt{3}}{2}\,|1, 0101, A\rangle + \frac{1}{2}\,|1, 0100, B\rangle$, input 1 has no packet, input 2 issues the quantum packet $|1, 0010, D\rangle$ and input 3 issues the quantum packet $\frac{1}{\sqrt{2}}\,|1, 1001, E\rangle + \frac{1}{\sqrt{2}}\,|1, 1000, F\rangle$. Then, the quantum assignment is

$$\left(\frac{\sqrt{3}}{2}\,|1, 0101, A\rangle + \frac{1}{2}\,|1, 0100, B\rangle\right) \otimes |0, 0000, C\rangle \otimes$$
$$|1, 0010, D\rangle \otimes \left(\frac{1}{\sqrt{2}}\,|1, 1001, E\rangle + \frac{1}{\sqrt{2}}\,|1, 1000, F\rangle\right) \quad (2.34)$$

which can be written as a superposition of four multicast assignment patterns

$$\frac{\sqrt{3}}{2\sqrt{2}}\,|(1, 0101, A), (0, 0000, C), (1, 0010, D), (1, 1001, E)\rangle +$$
$$\frac{\sqrt{3}}{2\sqrt{2}}\,|(1, 0101, A), (0, 0000, C), (1, 0010, D), (1, 1000, F)\rangle +$$
$$\frac{1}{2\sqrt{2}}\,|(1, 0100, B), (0, 0000, C), (1, 0010, D), (1, 1001, E)\rangle +$$
$$\frac{1}{2\sqrt{2}}\,|(1, 0100, B), (0, 0000, C), (1, 0010, D), (1, 1000, F)\rangle \quad (2.35)$$

It is seen that the first assignment pattern in the above quantum assignment is *contending* because packets A and E in this pattern are addressed to output 3. The rest of the three assignment patterns are *non-contending*. Next we define a sub-pattern of an assignment pattern:

Definition 2.4 (sub-pattern). An assignment pattern $\left|\bar{P}'\right\rangle = |(r'_0, \bar{a}'_0, \bar{d}'_0), \cdots, (r'_{n-1}, \bar{a}'_{n-1}, \bar{d}'_{n-1})\rangle$ is called a *sub-pattern* of another assignment pattern $\left|\bar{P}\right\rangle = |(r_0, \bar{a}_0, \bar{d}_0), \cdots, (r_{n-1}, \bar{a}_{n-1}, \bar{d}_{n-1})\rangle$ if, for every i, $0 \leq i \leq n-1$, $r'_i = 1$ implies that $r_i = 1$, $\bar{d}'_i = \bar{d}_i$ and $F'_i \subseteq F_i$. A non-contending sub-pattern $\left|\bar{P}'\right\rangle$ of $\left|\bar{P}\right\rangle$ is said to be *maximal* if the fanout sets $\cup_{i=0}^{n-1}\{F'_i : r'_i = 1\}$ and $\cup_{i=0}^{n-1}\{F_i : r_i = 1\}$ are equal.

2.2.2.2 Quantum Packet Switching Model

Classical packet switches are often constructed using non-blocking switching networks. Even in an internally non-blocking switch, output contentions may occur when multiple input packets are addressed to the same output. To resolve these contentions, packets are either buffered at some place in the network or dropped. A widely studied buffering scheme uses n infinite length FIFO queues on each input of the $n \times n$ network, as shown in Figure 2.10(a). In this scheme, however, if there are some queues with their head packets addressed to the same output then only one of these queues can be served. This problem, known as head-of-line (HOL) blocking, limits the maximum throughput which can be obtained in such a network. Karol et. al. showed that the maximum achievable throughput on this network is $2 - \sqrt{2} = 0.586$, which is obtained under some ideal arrival conditions and large n [56]. Several techniques have been suggested for reducing HOL blocking using non-FIFO queues. One such technique is to examine the first k packets in a FIFO queue, where $k > 1$ [57]. It is possible to avoid HOL blocking completely by using virtual output queuing (VOQ) scheme [58] which uses n^2 input queues called virtual output queues. In this scheme, each input has n FIFO queues, where each queue holds packets which are addressed to only one output as shown in Figure 2.10(b). It was shown by Mckeown et. al. [58] that it is possible to achieve 100% throughput using VOQ scheme and maximum weight matching based scheduling. However, finding maximum weight matching is complex ($O(n^3)$) and several scheduling algorithms which approximate maximum weight matching have also been proposed [59–64]. All these algorithms rely on the fact that the queue lengths vary slowly from one time slot to the next time slot so that choosing the previous matching and a randomly generated matching can be used to achieve high throughput.

The above discussion was about unicast switches in which each packet is addressed to at most one output. In case of multicast switches, it is not possible to use the VOQ scheme to remove HOL blocking, because each input packet can be addressed to one of 2^n possible output patterns. Therefore, we would need 2^n queues at each input, which is not feasible. Several algorithms that use finite number of queues on each input and aim to maximize the throughput using some heuristic based scheduling, have been proposed [65] [66] [67].

Quantum systems can provide a novel approach to serve packets on an

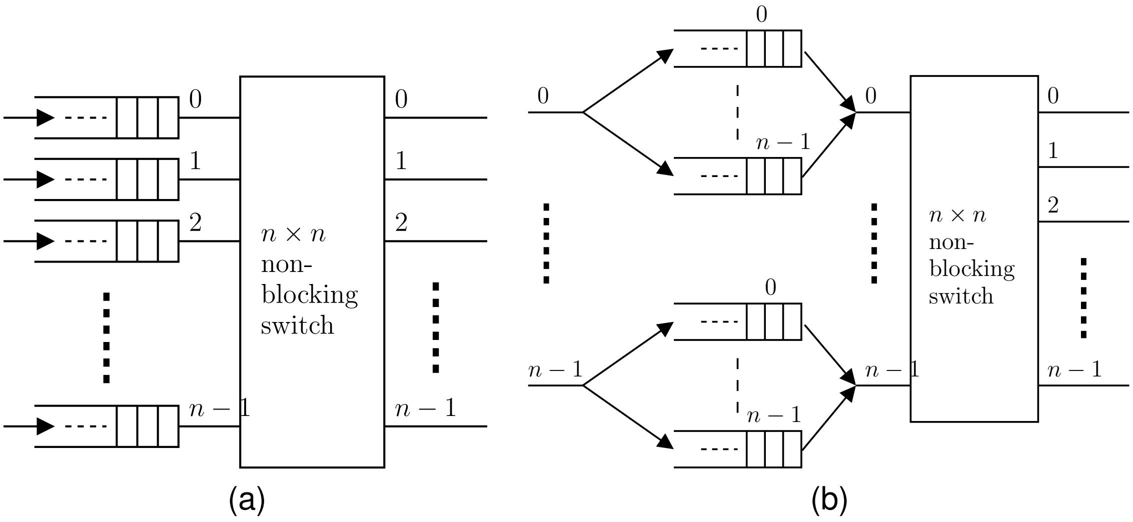

0
1
2
$n-1$
$n \times n$ non-blocking switch
0
1
2
$n-1$
(a)
0
$n-1$
0
$n-1$
0
$n-1$
0
$n-1$
$n \times n$ non-blocking switch
0
1
2
$n-1$
(b)

Figure 2.10: Input queued classical switches: (a) Each input having one queue (b) Virtual output queuing.

input-queued switch using quantum superposition and parallelism. Instead of using a centralized scheduling algorithm to determine which packets to route from the input queues, we can create a superposition of certain number of packets in a queue to form a quantum packet, which can then be routed by a quantum packet switching network. This is the model of quantum packet switching that is explored in this thesis, and is shown in Figure 2.11. The classical-to-quantum (C/Q) converters select some packets from the input queues and creates their superposition by assigning appropriate probabilities. These probabilities can be chosen suitably to shape the incoming traffic in order to maximize overall throughput. The quantum packets created by C/Q converters form a quantum assignment which is realized by a quantum packet switch. This switch routes all assignment patterns in the assignment simultaneously. The output state of the quantum packet switch is also a superposition of assignment patterns, where

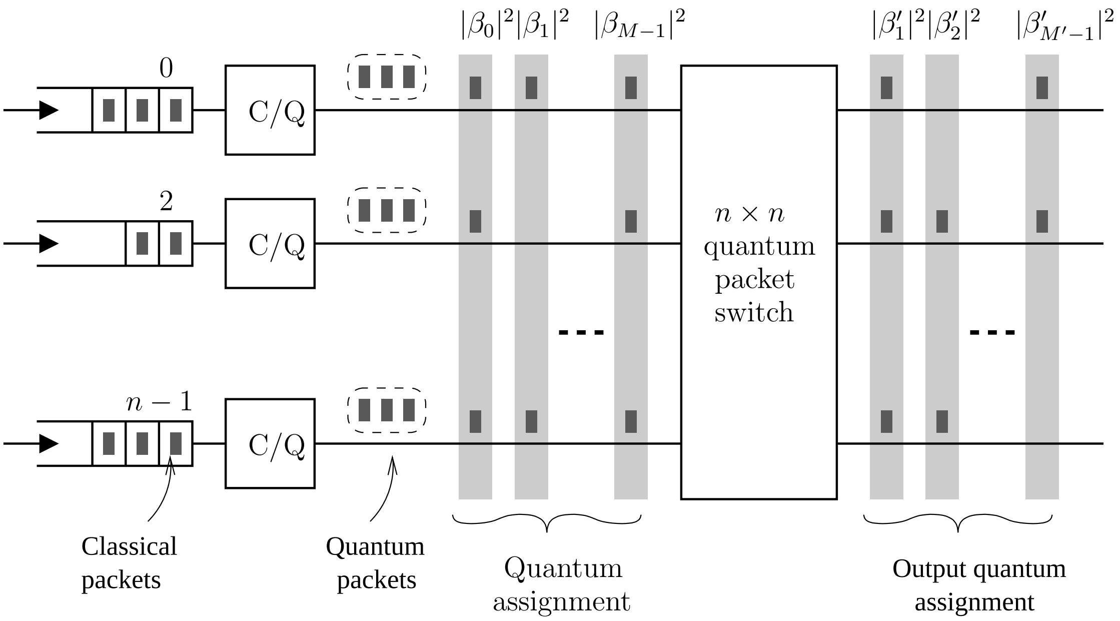


Figure 2.11: Quantum packet switching model.

classical packets in every pattern are at their desired outputs. Consequently, each output contains superposition of packets that were addressed to that output. In this thesis, we give the design of such quantum packet switches. The problem of scheduling classical packets from the input queues, and assigning them probabilities such that a certain traffic pattern in served or overall throughput is maximized, is not studied in this thesis.

Quantum packet switches are categorized based on the possible assignment patterns they can realize. An $n \times n$ quantum packet switch is said to be *non-blocking* if it transforms any non-contending assignment pattern to another pattern in which to another pattern in which all the classical packets are at their desired outputs. A set of auxiliary qubits is usually needed to construct such a switch. Non-blocking quantum packet switches can be categorized further as *unicast* or *multicast* depending on whether they can realize only unicast non-contending assignments or all of them. A non-blocking multicast quantum packet switch is also called a generalized quantum connector. Our design of generalized quantum connector is given in Chapter 5.

A contending assignment pattern, by definition, has multiple packets addressed to some outputs of the switch. For such a pattern, a classical non-blocking packet switch can route at most one packet to each output, and others are blocked. In contrast, a quantum packet switch can route all contending packets to their desired output by creating their superposition on that output. This can be done by breaking the contending assignment patterns into non-contending ones using quantum superposition, as conflicts arise while these patterns are routed through the switch. The contending packets are distributed among the resulting sub-patterns to ensure that these sub-patterns are non-contending. The probability coefficient of the contending pattern is distributed among the sub-patterns. We modify the n-GQC design to accomplish this behavior in Chapter 6

Quantum packet switches that cannot realize all non-contending assignment pattern are said to be *blocking*. In such switches, even when there is no contention for final outputs, internal conflicts may arise because of topology limitations. The above approach of creating superposition of conflicting packets by breaking the assignment patterns can be applied to such switches as well, and this is explored in the next chapter for the quantum Baseline network, which is a blocking quantum packet switch.

Chapter 3: Quantum Baseline Network

In this chapter, we present the design a self-routing quantum packet switch called quantum Baseline network [6]. This quantum packet switch is an improved version of a similar switching network presented by us in [5]. For an input assignment pattern that undergoes internal blocking or has output contentions, it creates a superposition of a set of output patterns containing all maximum size sub-patterns of the input assignment pattern which can be routed through the Baseline network without internal blocking. This cannot be achieved by any classical self-routing switching network that is internally blocking. We also give a method to characterize the output state of the switch using the concepts of frames and balanced matrices.

The chapter is organized as follows. In section 3.1, we give a brief introduction to the classical Baseline network. In Section 3.2, we present the design of a 2×2 quantum switch that is capable of creating a superposition of its input packets in the case of an output contention. In Section 3.3, we present the design of the self-routing quantum Baseline network (QBN). We discuss the output state of the QBN for permutation assignment patterns in Section 3.4.

3.1 Classical Baseline Network

An n input and n output Baseline network has $p = \log_2 n$ stages, each having $n/2$ two input-two output switches that can be set either in *through* state or in *cross* state. The switches in each stage are numbered $0, \cdots, n/2-1$ from top to bottom using $p-1$ bit binary numbers. The input and output ports of each stage are labeled $0, \cdots, n-1$ from top to bottom using p bit binary numbers. Output port $o_1 \cdots o_n$ of the m^{th} stage is connected to the input port $i_1 \cdots i_n$ of the $m+1^{th}$ stage $(1 \leq m \leq p-1)$ where binary number $i_1 \cdots i_n$ is obtained by doing a right circular shift on lower $p-m+1$ bits of binary number $o_1 \cdots o_p$. Switch $b_1 b_2 \cdots b_{p-1}$ in the m^{th} stage is connected to the switches $b_1 \cdots b'_m \cdots b_{p-1}$ and $b_1 \cdots b''_m \cdots b_{p-1}$ in the $m+1^{th}$ stage, where $b'_m = 0$, $b''_m = 1$ and $1 \leq m \leq p-1$. The 16×16 baseline network is shown in Figure 3.1.

Packets can be *self routed* in the baseline network in following manner. Suppose the output addresses of the input packets of a 2×2 switch in m^{th} stage $(1 \leq m \leq p)$ are binary numbers $a_1 a_2 \cdots a_p$ and $b_1 b_2 \cdots b_p$. This switch is set in *through* state if $a_m = 0$ and $b_m = 1$ and in *cross* state if $a_m = 1$ and $b_m = 0$. When both a_m and b_m are same there is a contention and one of the packets has to be either dropped or buffered. Next, we discuss some connection properties of the baseline network which are used later.

Consider the m^{th} stage of the n-input baseline network, where $2 \leq m \leq p$. Divide the inputs of the 1^{st} stage into 2^{p-m} consecutive disjoint sets of size 2^m each and number these sets $1, \cdots, 2^{p-m}$ from top to bottom. A 2×2 switch $b_1 b_2 \cdots b_{p-1}$ in the m^{th} stage of the baseline network can be reached only by the inputs in the input set number k, where k is binary number $b_m \cdots b_{p-1}$. Switches in the last stage of the network can be reached by every input. Also, any self routed packet reaching switch $b_1 b_2 \cdots b_{p-1}$ in the m^{th} stage should have its top $m-1$ address bits as $b_1 \cdots b_{m-1}$.

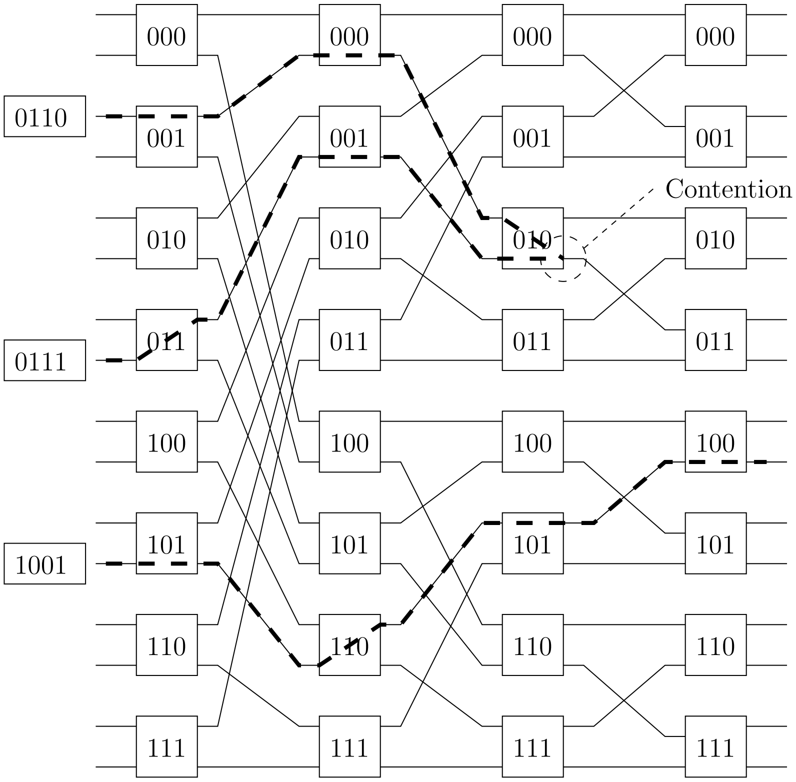


Figure 3.1: 16×16 Baseline Network.

3.2 Quantum Switch

We gave a simple design for a 2×2 quantum switch in [5]. The quantum circuit of this switch is shown in Figure 3.2. Two sets of qubits of size n_d each con-

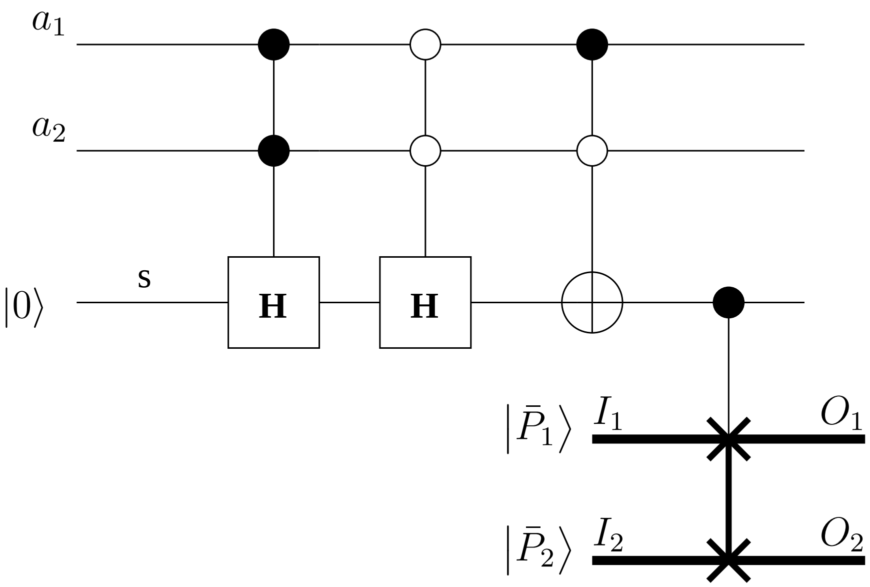


Figure 3.2: A 2×2 Quantum Switch.

tain the data parts of input packets. The address bits that determine the switch setting are a_1 and a_2 respectively. These address bits are not switched along with remaining packet content, and are discarded after use. $\bar{P}_1$ and $\bar{P}_2$ denote the remaining bits in the input packets after excluding a_1 and a_2. An auxiliary qubit s initialized to state $|0\rangle$ is employed to act as a control input to the switch gate that performs the actual switching.

When $a_1 = a_2$, i.e., when the input packets are contending for one of the outputs, one of the Hadamard gates changes the state of s to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and the switch gate creates an equal superposition of $\bar{P}_1$ and $\bar{P}_2$. If a_1 is $|1\rangle$ and a_2 is $|0\rangle$ then the NOT gate sets s to state $|1\rangle$, and the switch gate is set in *cross* state. When a_1 is $|0\rangle$ and a_2 is $|1\rangle$, s remains in state $|0\rangle$ and the switch gate is set in *through* state. Thus, representing the state of the system by $|a_1, a_2, \bar{P}_1, \bar{P}_2\rangle$, the function of the switch is described as

$$\begin{aligned}
\left|0,0,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s &\longrightarrow \frac{1}{\sqrt{2}}\left(\left|0,0,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s+\left|0,0,\bar{P}_2,\bar{P}_1\right\rangle\left|1\right\rangle_s\right)\\
\left|0,1,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s &\longrightarrow \left|0,1,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s\\
\left|1,0,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s &\longrightarrow \left|1,0,\bar{P}_2,\bar{P}_1\right\rangle\left|1\right\rangle_s \qquad (3.1)\\
\left|1,1,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s &\longrightarrow \frac{1}{\sqrt{2}}\left(\left|1,1,\bar{P}_1,\bar{P}_2\right\rangle\left|0\right\rangle_s+\left|1,1,\bar{P}_2,\bar{P}_1\right\rangle\left|1\right\rangle_s\right)
\end{aligned}$$

Even though this switch creates a superposition of the contending packets at the desired output, a complementary superposition is created on the other output as well, which is undesirable. In a baseline network made using this switch, the outputs of the network might receive packets that are not addressed to them. Also, it will not be possible to verify whether the received packet on an output was intended for that output or not because the address bits are removed by the network. This problem was solved in [5] by swapping undesired superposition with dummy packets, which are distinguishable from other data packets. This approach is costly and requires $O(\log n)$ extra qubits for each 2×2 switch. We can simply include a routing qubit in quantum packets as defined in Section 2.2.2.1, and set the routing bit on the output to which no packet should go to 0, which can be used to ignore these packets while determining switch setting further in the network.

A 2×2 quantum switch that requires that each packet should contain a routing bit, and sets this bit to 0 for the packets in the undesirable superposition,

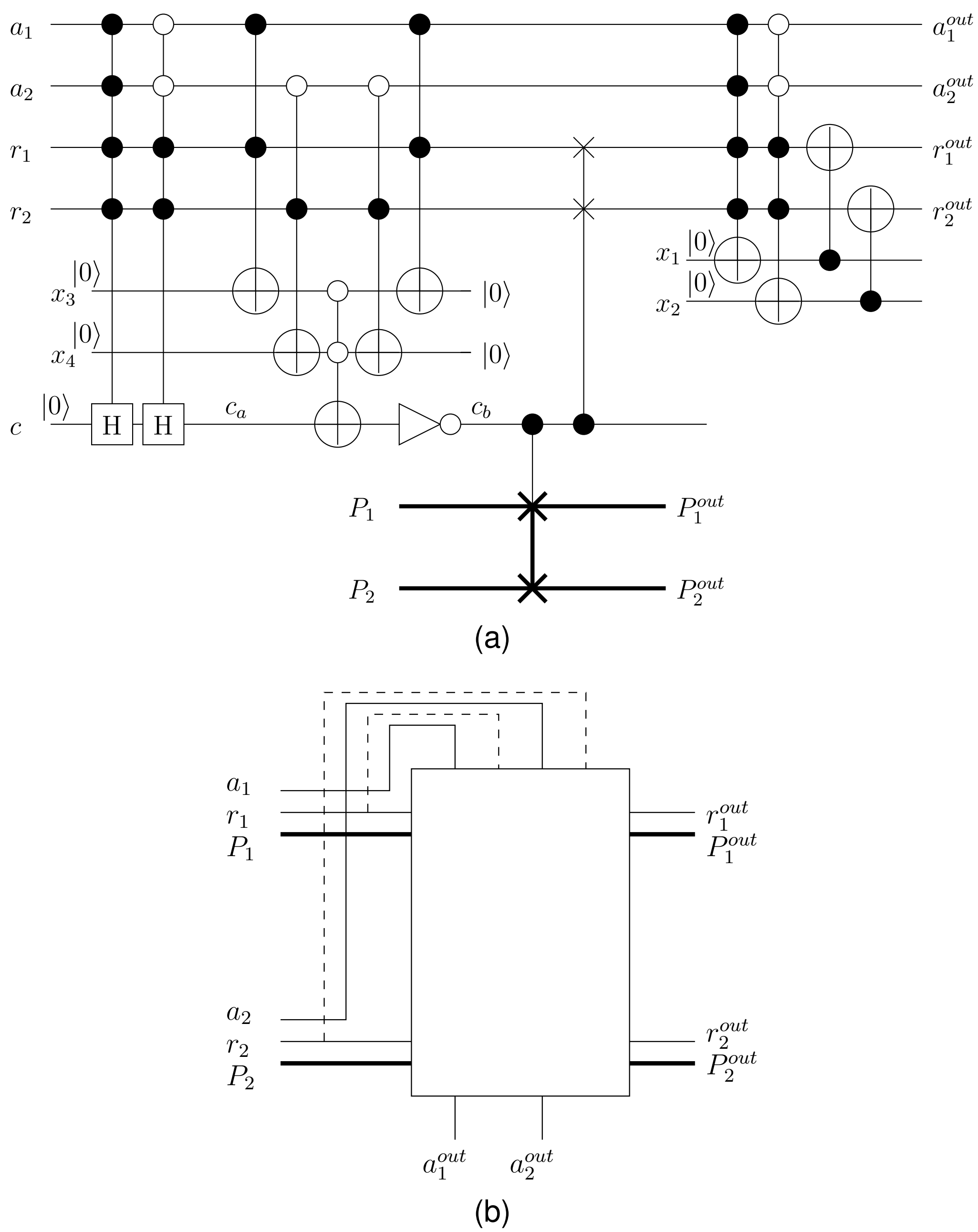


Figure 3.3: Improved 2×2 Quantum Switch: (a) quantum circuit for the 2×2 switch; (b) its block representation.

is shown in Figure 3.3(a). The switch ignores the address bits of any input packet that has a routing bit of 0.

Qubit c is used to set the switch gate in a *through* or *cross* or an equal superposition of *through* and *cross* states depending on the values of the address and routing bits of packets on the upper and lower inputs. These qubits are labeled a_1, r_1 and a_2, r_2 respectively for the two packets. The routing bits are switched along with the packets whereas the address bits are discarded. Whenever an output contention occurs, one of the first two controlled Hadamard gates changes the state of qubit c to $1/\sqrt{2}(|0\rangle + |1\rangle)$, otherwise they do not affect c and it remains in state $|0\rangle$. Note that a NOT operation on state $1/\sqrt{2}(|0\rangle + |1\rangle)$ does not affect it. When there is no contention, qubit c is set in state $|0\rangle$ or $|1\rangle$ depending on the values of a_1, r_1, a_2 and r_2 according to the following table. This gives $c = a_1r_1 + \bar{a}_2r_2$,

v_1v_2 \ a_1a_2	00	01	11	10
00	x	x	x	x
01	1	0	0	1
11	x	0	x	1
10	0	0	1	1

0: Through state
1: Cross state

which is done in the quantum circuit using reversible logic. This logic does not affect the state of c when it is in state $1/\sqrt{2}(|0\rangle + |1\rangle)$. Finally, in case of contention the routing bits of the packets on unused output are set to 0. Qubit c and other two auxiliary qubits used in the reversible logic $a_1r_1 + \bar{a}_2r_2$ remain in their original state $|0\rangle$ and can be ignored because a measurement on them does not have any effect on the functioning of the circuit. Only the auxiliary qubits x_1 and x_2 may end up in an altered state. Denoting the packets as $(r_i, a_i, \bar{P}_i)$, $i = 1, 2$, the functioning of the switch when both the input packets are valid is:

$$\begin{aligned}
&|(1,0,\bar{P}_1),(1,0,\bar{P}_2)\rangle \longrightarrow \\
&\qquad\qquad \frac{1}{\sqrt{2}}\left(|(1,0,\bar{P}_1),(0,0,\bar{P}_2)\rangle + |(1,0,\bar{P}_2),(0,0,\bar{P}_1)\rangle\right) \\
&|(1,0,\bar{P}_1),(1,1,\bar{P}_2)\rangle \longrightarrow |(1,0,\bar{P}_1),(1,1,\bar{P}_2)\rangle \\
&|(1,1,\bar{P}_1),(1,0,\bar{P}_2)\rangle \longrightarrow |(1,0,\bar{P}_2),(1,1,\bar{P}_1)\rangle \\
&|(1,1,\bar{P}_1),(1,1,\bar{P}_2)\rangle \longrightarrow \\
&\qquad\qquad \frac{1}{\sqrt{2}}\left(|(0,1,\bar{P}_1),(1,1,\bar{P}_2)\rangle + |(0,1,\bar{P}_2),(1,1,\bar{P}_1)\rangle\right)
\end{aligned} \tag{3.2}$$

When only one of the two input packets has routing bit of 0, the switch is simply set to route the other packet. Also, the switch can be set in any state when both the input packets have routing bits of 0. A block schematic diagram for this switch is shown in Figure 3.3(b).

3.3 Quantum Baseline Network Construction

The 2×2 quantum switch given in Figure 3.3 is used to form the quantum baseline network (QBN). An n input ($n = 2^p$) QBN has $p/2$ stages of 2×2 quantum switches connected in baseline topology described in Section 3.1. In this section, we give an example of 4×4 QBN shown in Figure 3.4 to explain its functionality.

Suppose inputs 0, 1, 2 and 3 have classical packets P_3, P_2, P_0 and P_1 destined for outputs 3, 2, 0 and 1 respectively. We represent the data part of these packets as d_3, d_2, d_0 and d_1 respectively. The packets are represented as tuple (r, a_1a_2, d), where a_1a_2 is the binary number equal to the output address of the packet. We write the state of the quantum wires at locations marked in Figure 3.4 by vertical dotted lines. The order of the components in ket-notation $|W_1, W_2, \cdots, W_n\rangle$, corresponds to the order in which we encounter the wires as we traverse the circuit from top to bottom.

The input assignment pattern is $|(1, 11, d_3)$, $(1, 10, d_2)$ $(1, 00, d_0)$, $(1, 01, d_1)\rangle$. The contentions for outputs at both the switches in the first stage cause them to create a superposition of four patterns at location *B* in the figure (representing packets at this location as tuple (r, a_2, d)):

$$\frac{1}{2}\Big[\big|(0,1,d_3)(1,0,d_2)(1,0,d_0)(0,1,d_1)\big\rangle + \\ \big|(0,0,d_2)(1,1,d_3)(1,0,d_0)(0,1,d_1)\big\rangle + \\ \big|(0,1,d_3)(1,0,d_2)(1,1,d_1)(0,0,d_0)\big\rangle + \\ \big|(0,0,d_2)(1,1,d_3)(1,1,d_1)(0,0,d_0)\big\rangle\Big]$$

After the shuffle, the state at *C* is obtained by interchanging the middle two packets in each of the four patterns. All the address bits of the packets are used by the networks at the end of the second stage and a packet at the output is represented as (r, d). There is no contention in any pattern at any of the switches in the second stage because one packet at the inputs of both the 2×2 switches (for every pattern) is an invalid packet and has routing bit of 0. Thus the output state of the switch is

$$\begin{aligned}\frac{1}{2}\big[&|(1,d_0)(0,d_3)(1,d_2)(0,d_1)\rangle +\\ &|(1,d_0)(0,d_2)(0,d_1)(1,d_3)\rangle +\\ &|(0,d_3)(1,d_1)(1,d_2)(0,d_0)\rangle +\\ &|(0,d_2)(1,d_1)(0,d_0)(1,d_3)\rangle\big]\end{aligned}$$

If we do a computational basis measurement at the outputs of the switch, we will observe one of the patterns in the above expression with probability $1/4$. The i^{th} component of a tuple is the packet at the i^{th} output. Thus, the probability of observing a valid packet P_i, $i = 0, 1, 2, 3$, at output i is $1/2$. Also, we will observe valid and correctly routed packets only at any two of the outputs. The other two outputs will have *invalid* packets.

An $n \times n$ quantum baseline network for larger values of n is formed in similar way. A 2×2 quantum switch at stage m of this network uses m^{th} address qubits of its input packets for routing. In the next section, we develop methods to express the output quantum state of an $n \times n$ QBN for any given permutation assignment pattern.

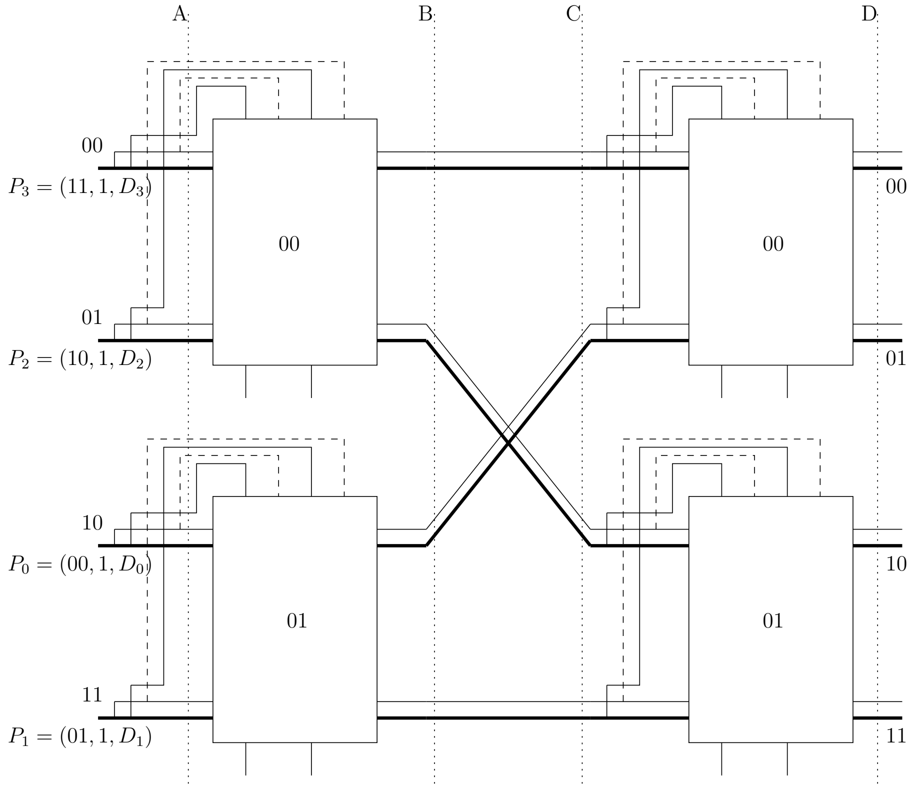
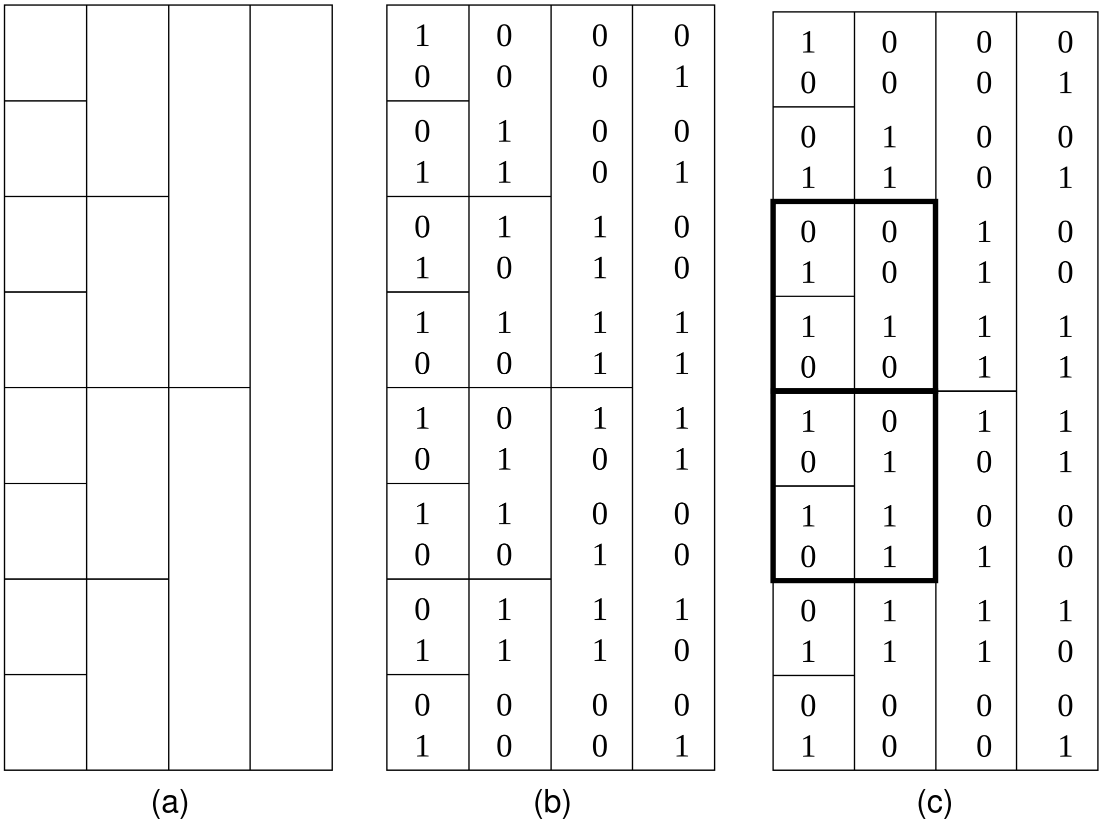


Figure 3.4: 4×4 Quantum Baseline Network.

3.4 The Output State of QBN

In this section, we discuss the structure of the output quantum state of a $n \times n$ QBN. Given any permutation assignment pattern, we give formulae for all the patterns in the output quantum state and their coefficients or probabilities. We use the concepts of balanced matrices and frames used by various authors and most recently by Çam [23]. We start with some definitions.

Definition 3.1. *Permutation Matrix:* A permutation assignment pattern π in an $n \times n$ ($n = 2^p$) switch is represented by an $n \times p$ permutation matrix Π, where $\Pi(i, j)$ is the j^{th} most significant bit in the output address of the packet on input i.



1	0	0	0
0	0	0	1
0	1	0	0
1	1	0	1
0	1	1	0
1	0	1	0
1	1	1	1
0	0	1	1
1	0	1	1
0	1	0	1
1	1	0	0
0	0	1	0
0	1	1	1
1	1	1	0
0	0	0	0
1	0	0	1

1	0	0	0
0	0	0	1
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	0
1	1	1	1
0	0	1	1
1	0	1	1
0	1	0	1
1	1	0	0
0	1	1	0
0	1	1	1
1	1	1	0
0	0	0	0
1	0	0	1

Figure 3.5: 16×4 Frame and permutation matrices: (a) A 16×4 frame; (b) A 16×4 permutation matrix which fits the frame; (c) A 16×4 permutation matrix which does not fit the frame.

Definition 3.2. *Balanced Matrix:* An $n \times p$ binary matrix B is *balanced* if and only if no p bit binary number appears more than once in its rows.

Definition 3.3. *Frame* is a set consisting of following $2^m \times m$ sub-matrices of an $n \times p$ matrix, where $1 \leq m \leq p$. Columns 1 to m (where $1 \leq m \leq p$) of the $n \times p$ matrix are divided into 2^{p-m} consecutive blocks ($2^m \times m$ sub-matrices) from top to bottom, each block having 2^m rows. The set of all these sub-matrices is called a Frame. A frame for $n = 16$ is shown in Fig 3.5a.

Definition 3.4. A $n \times p$ permutation matrix is said to *fit* an $n \times p$ frame if each sub-matrix in the Frame is balanced.

A permutation matrix which fits the 16×4 frame is shown in Fig 3.5b. A permutation matrix that does not fit this frame is shown in Fig 3.5c. For example, the highlighted sub-matrices are not balanced.

Theorem 3.1. *A permutation assignment pattern can be self-routed in an $n \times n$ baseline network without blocking if and only if its permutation matrix fits into an $n \times p$ frame.*

Proof: First, suppose that the $n \times p$ permutation matrix of the assignment pattern does not fit the $n \times p$ frame. Then for some $1 \leq m < p$, an m column sub-matrix (of size $2^m \times m$) in the frame is not balanced. Note that m can not be equal to p since a permutation matrix is always balanced. There are two packets on the 2^m inputs corresponding to this sub-matrix that have their top m output address bits same, let these bits be $a_1 \cdots a_m$. Let the input ports of these two packets be $i_1 \cdots i_{p-m} i_{p-m+1} \cdots i_p$ and $i_1 \cdots i_{p-m} j_{p-m+1} \cdots j_p$. The top $p - m$ bit in the input addresses are the same because the two inputs belong to the same block of size 2^m. Using the self routing described in section 3.1, both the packets get routed to the same switch $a_1 a_2 \cdots a_{m-1} i_1 \cdots i_{p-m}$ in the m^{th} stage. Since the address bit a_m for both input packets at this switch is same there is a contention and the input permutation cannot be routed without blocking.

For the converse, if there is a contention at one of the 2×2 switches in m^{th} stage $1 \leq m < p$, then the top m address bits of the two input packets at this switch should be the same since any packet reaching switch $b_1 \cdots b_{p-1}$ in the m^{th} stage should have its top $m - 1$ address bits as $b_1 \cdots b_{m-1}$. Also, these two inputs belong to the same input block of size 2^m. Thus, there is an unbalanced $2^m \times m$

sub-matrix in the frame. Hence the permutation matrix does not fit the $n \times p$ frame.$\|$

A similar theorem for reverse baseline network is given in [23]. Here we have proved that the same condition ensures a non-blocking routing in a baseline network as well. It can be verified that the permutation shown in Fig 3.5b passes the baseline network without any contention while the one in Fig 3.5c does not.

For any input permutation assignment pattern, the *output state* of an $n \times n$ QBN is in general as superposition of patterns

$$\sum_{i=1}^{M} \alpha_i \left|(r_0^i, d_0^i)(r_1^i, d_1^i) \cdots (r_{n-1}^i, d_{n-1}^i)\right\rangle \tag{3.3}$$

where $\sum_{i=1}^{M} |\alpha_i|^2 = 1$; and r_j^i, d_j^i $(0 \leq j \leq n-1)$ respectively are the routing bit and data part of the packet at output j of the QBN in the i^{th} output patten. On measurement the i^{th} pattern is observed with probability $|a_i|^2$. The smallest value of M can be 1, as given in the following theorem which follows directly from theorem 1.

Theorem 3.2. *Let $\pi = (o_0, \cdots, o_{n-1})$ be a permutation of numbers $0, \cdots, n-1$ where $n = 2^p$, and in a given permutation assignment pattern, input i of a QBN have a packet P_{o_i} destined to output o_i, where $0 \leq i \leq n-1$. If the permutation matrix of π fits the $n \times p$ frame then the output quantum state of the QBN is $|(1, d_0), \cdots, (1, d_{N-1})\rangle$, where d_i is the data part of packet P_i.*

If a permutation assignment pattern does not fit in the $n \times p$ frame, it is broken into sub-patterns each having some packets with routing bits of 0, as shown in the example given in Section 3.3. Thus the value of M will be more than one for such an assignment. The exact number of packets with routing bits of 0 in each resulting output pattern, and the number of patterns present in output quantum state can be given by using the concepts of frame and fitting discussed earlier. For this we first make a $n \times p$ permutation matrix fit into $n \times p$ frame by marking some of the rows as *invalid* using the following procedure called *balancing*.

Definition 3.5. *Balancing a permutation matrix:* First for $m = 1$, the row corresponding to one of each repeated 1 bit number in each 2×1 sub-matrix of the frame is marked *invalid*. Next, for $2 \leq m \leq p$, check only the valid rows and make the row corresponding to one of each repeated m bit binary number (if any) in every $2^m \times m$ sub-matrix of the frame as *invalid*. A permutation matrix which is obtained by this process is called a *balanced permutation matrix*. $\|$

Note that there are many possible balanced matrices that can be generated by the balancing procedure as there are many choices of rows that can be marked invalid in each step of the balancing procedure. The output state of the QBN is a superposition of every balancing outcome, as described next.

Theorem 3.3. *Let $\pi = (o_0, \cdots, o_{n-1})$ be a permutation of numbers $0, \cdots, n-1$ ($n = 2^p$) which does not fit an $n \times p$ frame. Consider a permutation assignment pattern in which input i has packet P_{o_i} destined to output o_i, where $0 \leq i \leq n-1$. Suppose that M balanced matrices can be generated by balancing the permutation matrix of π. Let $I_j \subset \{0, \cdots, n-1\}$ be the set of inputs corresponding to the invalid rows of the j^{th}*

balanced matrix, where $1 \leq j \leq M$*. Then the output state of the QBN is:*

$$\sum_{j=1}^{M} \alpha_j \left|(r_0^j, d_0^j)(r_1^j, d_1^j) \cdots (r_{n-1}^j, d_{n-1}^j)\right\rangle \tag{3.4}$$

where r_i^j *is* 1 *if* $\pi^{-1}(i) \notin I_j$ *and* 0 *if* $\pi^{-1}(i) \in I_j$*,* $0 \leq i \leq n-1$*. Also,* d_i^j *is the data part of packet* P_i *if* $\pi^{-1}(i) \notin I_j$*, and invalid if* $\pi^{-1}(i) \in I_j$*,* $0 \leq i \leq n-1$*. Furthermore,*

$$\alpha_j = \left(\frac{1}{\sqrt{2}}\right)^{|I_j|} \tag{3.5}$$

where $|I_j|$ *is the cardinality of set* I_j*.*

Proof: The 2×1 frames in the permutation matrix contain the address bits that determine the settings of the 2×2 switches in the first stage. Consequently, an unbalanced 2×1 frame indicates an output contention on the corresponding first stage switch. Similarly, as shown earlier in the proof of Theorem 3.2, input packets corresponding to identical rows in an unbalanced $2^m \times m$ frame contend at a 2×2 switch in the m^{th} stage of the network. Also, it can be observed that there can be at most two rows having same values in a $2^m \times m$ frame if the two $2^{m-1} \times (m-1)$ frames have been balanced. Therefore, a 2×2 switch in the m^{th} stage receives at most two contending packet if the earlier stages of the network have balanced the permutation matrix until the $m - 1^{th}$ step of the balancing procedure.

The above observations imply that the switching operation of the m^{th} stage of the network is equivalent to the m^{th} step of the balancing procedure, where

$0 \leq m \leq p-1$. One of the rows in every pair of identical rows in a $2^m \times m$ frame is marked invalid by setting the routing bit of one of the contending packets at the corresponding 2×2 switch to 0. For an input pattern to the m^{th} stage, if there are k switches in the stage that have contending input packets then the input pattern is broken into 2^k output patterns, with k of the contending packets replaced by the ones with routing bits of 0 in each output pattern. The stage introduces a coefficient multiplier of $(1/\sqrt{2})^k$ into each output pattern. These output patterns correspond to the possible outcomes of the m^{th} step of balancing procedure applied to the matrix representing incoming pattern. Consequently, the network generates all possible outcomes of the balancing procedure, and also routes the input packets corresponding to the valid rows in each resulting balanced matrix to their outputs. Other outputs receive packets with routing bits of 0. The output pattern that corresponds to the j^{th} balanced matrix outcome has $|I_j|$ packets with routing bits of 0, and thus has undergone $|I_j|$ contentions. Hence its coefficient is $(1/\sqrt{2})^{|I_j|}$. $\|$

On a computational basis measurement on all outputs, one of the patterns in the output quantum state will be observed, and the probability of finding the pattern having I_j packets with routing bits of 0 and $n - I_j$ packets with routing bits of 1 is $(1/2)^{|I_j|}$. This is desirable as the output patterns having larger number of packets with routing bits of 1 have higher probability of being measured. Probability of finding any packet at its desired output can be obtained by summing the probabilities of the output patterns in which that packet has routing bit of 0.

Chapter 4: Quantum Concentrators

A quantum concentrator is a quantum switching network that transforms an input assignment pattern in which only the routing bits matter and the address fields are ignored. The packets with routing bits of 1 are routed consecutively to the top outputs of the network in some non-specifiable order. The rest of the packets (with routing bits of 0) are routed to the remaining outputs. When presented with general quantum states on its inputs, where the presence of a quantum state on an input is indicated by setting the routing qubit on that input to $|1\rangle$, a quantum concentrator would route these states to its top outputs.

4.1 Quantum Concentrator Definitions

In this section, we introduce different types of quantum concentrators and define them in terms of quantum transformations. These concentrators are constructed using quantum circuits later in the chapter. Since output addresses are not needed in a concentrator, we combine the address and data bits and represent a classical packet as $(r, \bar{p})$ in this chapter, where $\bar{p}$ denotes the data part.

Definition 4.1 (Quantum concentrator of first type). An *n-quantum concentrator of first type* (denoted as n-QC1) is an $n \times n$ quantum packet network which transforms an assignment pattern such that data parts of all classical packets having routing bits of 1 are concentrated to the top outputs in some non-specifiable order. The routing bits are not concentrated and are passed to the outputs, i.e., routing bit on input i is passed to output i, for $i = 0, 1, \cdots, n-1$. Also, two or more assignment patterns having the same number of coinciding routing bits of 1 are concentrated by applying the same permutation to their data parts. $\|$

We first show that n-QC1 represents a reversible quantum transformation. Since the routing bits $r_0, \cdots, r_{n-1}$ are copied to the outputs by an identity mapping, two input assignment patterns having different sequences of routing bits cannot be mapped to the same output pattern. Therefore, we need to show that there is a one-to-one mapping for the assignment patterns having the same sequence of routing bits. In this case, since the routing bits are the same, we can only check if two different assignment patterns having different packets are mapped to the same output pattern. Without loss of generality, consider two different packets with routing bits of 1 on a given input. Since both packets will have to be routed to the same output as the permutations are fixed for a given sequence of routing bits, the two input assignments which differ on only this input cannot be mapped to the same output pattern. This completes the proof that the n-QC1 is reversible. We list the set of all input patterns and the corresponding output patterns for a 2-QC1 in Table 4.1(a) for $d = 1$.

The n-QC1 would work well as a stand-alone quantum concentrator. However, in some switching networks in which quantum concentrators are used as

Input				Output			
r_0	p_0	r_1	p_1	r_0	p_0	r_1	p_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	1	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	1	1
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	1
1	1	0	0	1	1	0	0
1	1	0	1	1	1	0	1
1	0	1	0	1	0	1	0
1	0	1	1	1	0	1	1
1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	1

(a)

	Input				Output			
row index	r_0	p_0	r_1	p_1	r_0	p_0	r_1	p_1
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	1	0	0	0	1	0	0
3	0	1	0	1	0	1	0	1
4	0	0	1	0	1	0	0	0
5	0	0	1	1	1	1	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	1	0	1
8	1	0	0	0	1	0	0	0
9	1	0	0	1	1	0	0	1
10	1	1	0	0	1	1	0	0
11	1	1	0	1	1	1	0	1
12	1	0	1	0	1	0	1	0
13	1	0	1	1	1	0	1	1
14	1	1	1	0	1	1	1	0
15	1	1	1	1	1	1	1	1

(b)

Table 4.1: 2-input reversible and non-reversible transformation tables: (a) 2-QC1 transformation table; (b) A non-reversible 2-input transformation table.

building blocks, it may be required that the routing qubits are also concentrated along with the packets. This will invalidate the reversibility of the quantum concentrator unless some auxiliary qubits are used. As an example, for $n = 2$ and $d = 1$, the following two assignment patterns get mapped to the same output pattern:

$$\begin{aligned} &|(0,0)(1,0)\rangle \rightarrow |(1,0)(0,0)\rangle \\ &|(1,0)(0,0)\rangle \rightarrow |(1,0)(0,0)\rangle \end{aligned} \tag{4.1}$$

A complete list of input and output patterns is shown in Table 4.1(b). It is seen that a number of pairs of input patterns are mapped to the same output pattern in the table. These input patterns are shaded and correspond to row pairs $(4,8)$, $(5,10)$, $(6,9)$ and $(7,11)$ in the table. Obviously, this makes the table non-reversible. Still, the table exhibits a concentrator behavior in all the rows. A desirable goal is to maintain the concentrator property of the table for as many rows as possible while making it reversible. There are two ways to ensure reversibility. The first approach is to use a certain number of auxiliary qubits which will be discussed later in this section. The second approach is to change the data parts of the packets whose routing bits are set to 0, since these packets belong to the unused inputs and may be considered as empty packets. While this leads to a reduction of the number of allowable input patterns, it ensures that all concentration assignments are still realized. To accomplish this, we fix the data parts of the input packets with routing bits of 0 and only allow the input packets with routing bits of 1 to have arbitrary data parts. Assuming that the data part consists of d qubits, the number of allowable packets on each input is $1+2^d$, one packet for the routing bit of 0 and 2^d packets for the routing bit of 1. Therefore, the total number of distinct concentration assignment patterns, in which every packet with a routing bit of 0 has a fixed data part, is $(1 + 2^d)^n$. The concentration operation is defined only for these $(1 + 2^d)^n$ input assignment patterns. In order to ensure reversibility, we can modify the data parts of the packets with routing bit of 0 to break ties whenever two such assignment patterns are mapped to the same output pattern. As will be shown below, this requirement imposes a constraint on the minimum number of data bits which must be used in each input packet. We denote this minimum number of data bits by d_{min} and define a second type of quantum concentrator. We first have following definitions:

Definition 4.2. Two assignment patterns are said to be *similar* if they have the same sequences of routing bits and have the same data parts at inputs having routing bits of 1. Otherwise, they are said to be *distinct*. ||

Definition 4.3. A set of assignment patterns ζ is said to be a *concentration set* if no two assignment patterns in this set are similar and for any assignment pattern $\left|\bar{P}'\right\rangle$ which is not in ζ, there is an assignment pattern $\left|\bar{P}\right\rangle \in \zeta$, such that $\left|\bar{P}\right\rangle$ and $\left|\bar{P}'\right\rangle$ are similar. ||

For a given sequence of n routing bits in which m bits are 1, where $0 \leq m \leq n$, there are 2^{md} distinct assignment patterns, because each input with a routing bit of 1 can have 2^d possible data parts. There are $\binom{n}{m}$ sequences of routing bits having m bits set to 1, therefore, the number of distinct assignment patterns having m packets is $\binom{n}{m}2^{md}$. Consequently, the size of a concentration set is $\sum_{m=0}^{n} \binom{n}{m}2^{md} = (1 + 2^d)^n$. Also, any of the $\binom{n}{m}2^{md}$ distinct patterns described above can be chosen in $2^{(n-m)d}$ ways because packets on an input with routing bit of 0 can have one of the 2^d possible data parts. This leads us to conclude that there are $\prod_{m=0}^{n} 2^{(n-m)d\binom{n}{m}2^{md}} = 2^{nd(1+2^d)^{n-1}}$ different concentration sets. Also, any two concentration sets ζ and ζ' are similar in the sense that for every pattern $\left|\bar{P}\right\rangle \in \zeta$, there is an unique $\left|\bar{P}'\right\rangle \in \zeta'$ such that $\left|\bar{P}\right\rangle$ and $\left|\bar{P}'\right\rangle$ are similar.

Next, we define a quantum concentrator which, unlike n-QC1, also concentrates routing bits along with the packets. This network modifies the data parts of the unused inputs in order to obtain reversibility.

Definition 4.4 (Quantum concentrator of second type). An $n \times n$ quantum switching network is called an *n-quantum concentrator of second type* for a concentration set ζ (denoted as n-QC2), if for any $m = 0, 1, \cdots, n-1$, an assignment pattern $\left|\bar{P}\right\rangle \in \zeta$ in which m packets have routing bits of 1, is transformed such that these packets are routed to the top m outputs along with the routing bits, where the order of these packets on the top outputs is non specifiable. Two or more assignment patterns in ζ, which have coinciding routing bits of 1, are concentrated by applying the same permutation to the packets with routing bits of 1. The bottom $n-m$ outputs have packets with routing bits of 0. The data parts of these packets are set such that overall transformation is reversible. $\|$

The n-QC2 concentrates only the assignment patterns contained in set ζ. The remaining $2^{n(d+1)} - (1+2^d)^n$ input assignment patterns can be mapped to the same number of remaining output patterns one an one-to-one basis in any order. Also, if we can construct an n-QC2 for concentration set ζ, then we can construct an n-QC2 for any other concentration set ζ' by mapping an assignment pattern $\left|\bar{P}'\right\rangle \in \zeta'$ to output pattern $\left|QC2(\bar{P})\right\rangle$, where $\left|\bar{P}\right\rangle$ is the assignment pattern in ζ which is similar to $\left|\bar{P}'\right\rangle$. To this end, in Section 4.2, we construct an n-QC2 for a specific concentration set called zero-concentration set, which is defined next.

Definition 4.5. An assignment pattern $\left|\bar{P}\right\rangle = |(r_0, \bar{p}_0), \cdots, (r_{n-1}, \bar{p}_{n-1})\rangle$ is called a *zero-assignment pattern* if for every i, $0 \leq i \leq n-1$, $r_i = 0$ implies $\bar{p}_i = \bar{0}$. Also, the set of all zero-assignment patterns is called the *zero-concentration set* and is denoted as ζ_0 now onward.

As noted earlier, the data parts of the outputs with routing bits of 0 are used to obtain reversibility in an n-QC2. This can be done only if the data part of each packet has sufficient number of bits. We denote the minimum number of these bits as d_{min}. Before computing d_{min} for a general n, we point out that for a 2-QC2, $d_{min} = 1$. It can be seen that the reversible transformation shown in Table 4.2 is a 2-QC2 transformation for the zero-concentration set ζ_0. There are nine zero-assignment patterns (shown by shaded rows in the table) in this set, and they are concentrated as desired. These nine patterns cover all the concentration assignments for $n = 2$. It can be observed in the table that the data bits of some of the packets with routing bits of 0 are modified to ensure reversibility. For example, two zero-assignment patterns $|(0,0),(1,0)\rangle$ and $|(1,0),(0,0)\rangle$ in rows 4 and 8, which were mapped to the same output pattern in Table 4.1(b), are now mapped to output patterns $|(1,0),(0,0)\rangle$ and $|(1,0),(0,1)\rangle$ respectively. In order to ensure that these two assignment patterns are not mapped to the same output pattern,

	Input				Output			
Row index	r_0	p_0	r_1	p_1	r_0	p_0	r_1	p_1
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	1	0	0	0	1	0	0
3	0	1	0	1	0	1	0	1
4	0	0	1	0	1	0	0	0
5	0	0	1	1	1	1	0	0
6	0	1	1	0	0	0	1	0
7	0	1	1	1	0	0	1	1
8	1	0	0	0	1	0	0	1
9	1	0	0	1	0	1	1	0
10	1	1	0	0	1	1	0	1
11	1	1	0	1	0	1	1	1
12	1	0	1	0	1	0	1	0
13	1	0	1	1	1	0	1	1
14	1	1	1	0	1	1	1	0
15	1	1	1	1	1	1	1	1

Table 4.2: 2-QC2 transformation table in which shaded rows show the zero-assignment patterns.

packet $(0, 0)$ is modified to $(0, 1)$ for the second input pattern.

Theorem 4.1. *At least $\lceil \log_2 n \rceil$ bits are needed in the data parts of input packets in order to realize an n-QC2, that is, $d_{min} = \lceil \log_2 n \rceil$.* ‖

Proof: without loss of generality, we will work with the zero-concentration set ζ_0. We have already shown using the example in Table 4.2 that for $n = 2$, we have $d_{min} = 1$. For $n > 2$, we first observe that, for a given $m > 0$, there are $\binom{n}{m}$ zero-assignment patterns which are mapped to an output pattern of the form $|(1, \bar{p}_0)(1, \bar{p}_1), \cdots, (1, \bar{p}_{m-1}), (0, \bar{p}_{m+1}), \cdots, (0, \bar{p}_{n-1})\rangle$ in which $\bar{p}_0, \cdots, \bar{p}_{m-1}$ are fixed. This is because there are $\binom{n}{m}$ sequences of routing bits in which m routing bits are set to 1, and for each such sequence, there is only one zero-assignment pattern which is mapped to the above output pattern. The reason for this is that for the zero-assignment patterns having a fixed sequence of routing bits, the order in which m input packets appear on the top m outputs is the same, irrespective of packet contents. Also, for $m = n$, there is only one input pattern which can be mapped to a given output pattern $|(1, \bar{p}_0) \cdots, (1, \bar{p}_{n-1})\rangle$, therefore, we only need to ensure reversibility for $m < n$. For $0 \leq m \leq n-1$, the outputs $\bar{p}_{m+1}, \bar{p}_{m+2}, \cdots, \bar{p}_{n-1}$ should be different for each of these $\binom{n}{m}$ zero-assignment patterns in order to ensure reversibility. Since these outputs have $(n-m)d$ bits, we require that

$$2^{(n-m)d} \geq \binom{n}{m} \text{ or } d \geq \left\lceil \frac{1}{n-m} \left\lceil \log_2 \binom{n}{m} \right\rceil \right\rceil \quad \text{for every } 0 \leq m \leq n-1 \quad (4.2)$$

Denoting the following expression as d_m

$$d_m = \frac{1}{n-m} \left\lceil \log_2 \binom{n}{m} \right\rceil \quad \text{where } 0 \leq m \leq n-1 \quad (4.3)$$

we show that d_m is maximum for $m = n-1$. This is because $d_{m+1} > d_m$, for every

$0 \leq m \leq n-2$. To show this, we express d_{m+1} as:

$$\begin{aligned} d_{m+1} = \frac{1}{n-m-1}\left\lceil \log_2 \binom{n}{m+1} \right\rceil &= \frac{1}{n-m-1}\left\lceil \log_2 \binom{n}{m} + \log_2\left(\frac{n-m}{m+1}\right)\right\rceil \\ &\geq \frac{1}{n-m}\log_2\binom{n}{m} + \frac{1}{n-m}\left(\frac{n-m}{n-m-1}-1\right)\log_2\binom{n}{m} + \\ &\qquad \frac{1}{n-m-1}\log_2\left(\frac{n-m}{m+1}\right) \\ &= \frac{1}{n-m}\log_2\binom{n}{m} + \frac{1}{(n-m)(n-m-1)}\left[log_2\binom{n}{m} + (n-m)\log_2\left(\frac{n-m}{m+1}\right)\right] \end{aligned} \tag{4.4}$$

From (4.3), we have $d_m \leq \frac{1}{n-m}\log_2\binom{n}{m} + \frac{1}{n-m}$ or $\frac{1}{n-m}\log_2\binom{n}{m} \geq d_m - \frac{1}{n-m}$. Using this in (4.4), we have

$$d_{m+1} \geq d_m - \frac{1}{n-m} + \frac{f_m}{n-m} \quad \text{where } f_m = \frac{1}{n-m-1}\log_2\left[\binom{n}{m}\left(\frac{n-m}{m+1}\right)^{n-m}\right] \tag{4.5}$$

Therefore, to show that $d_{m+1} > d_m$, we need to establish that $f_m > 1$ for $0 \leq m \leq n-2$. Expressing f_m as: $f_m = \frac{1}{n-m-1}\log_2(c_m)$, where $c_m = \binom{n}{m}\left(\frac{n-m}{m+1}\right)^{n-m}$, we need to show that $c_m > 2^{n-m-1}$ for $0 \leq m \leq n-2$. We observe that for $m = n-2$, $c_{n-2} = \frac{2n}{n-1} > 2 = 2^{n-m-1}$. Next,

$$\begin{aligned} \frac{c_m}{c_{m+1}} = \left[\frac{(m+2)(n-m)}{(m+1)(n-m-1)}\right]^{(n-m-1)} &> \left(\frac{n-m}{n-m-1}\right)^{n-m-1} \\ &= \left(1 + \frac{1}{n-m-1}\right)^{n-m-1} \geq 1 + \frac{n-m-1}{n-m-1} = 2 \end{aligned} \tag{4.6}$$

Since $c_m > 2c_{m+1}$ and $c_{n-2} > 2$, we have $c_m > 2^{n-m-1}$ for $0 \leq m \leq n-2$. Therefore, $f_m > 1$ for every $0 \leq m \leq n-2$. This completes the proof of $d_{m+1} > d_m$ for $0 \leq m \leq n-2$. Therefore, from (4.2), we conclude that it is necessary to have $d \geq \lceil d_{n-1} \rceil = \lceil \log_2 n \rceil$.

We now show that $d \geq \lceil \log_2 n \rceil$ is also sufficient. Two zero-assignment patterns with different values of m are always mapped to different output patterns, because the routing bits of the top m packets in the output pattern are 1. For a given m, the $\binom{n}{m}2^{dm}$ zero-assignment patterns are mapped to uniquely identifiable output patterns by choosing different $\bar{p}_{m+1}, \cdots, \bar{p}_{n-1}$ sequences for each of them. This can be done since $d \geq \lceil \log_2 n \rceil$. Having uniquely mapped all $(1+2^d)^n$ zero-assignment patterns, rest of the $2^{n(d+1)} - (1+2^d)^n$ assignment patterns can be mapped in a one-to-one fashion to the remaining output patterns. Therefore, we have $d_{min} = \lceil \log_2 n \rceil$. $\|$

The n-QC2 has an undesirable constraint that each input packet must have a minimum number of qubits which is dependent on n. Also, for the construction of the n-GQC given in this paper, we require a quantum concentrator in which there is no restriction on the data bits contained in the input packets, since we need to use the output of one quantum concentrator as an input to another. This cannot be done using an n-QC2 since the output assignment pattern of an n-QC2 is not necessarily an allowable input assignment for another n-QC2. Therefore, we require a quantum concentrator which concentrates arbitrary assignment patterns and also sends the routing bits along with input packets. A certain number of auxiliary qubits initialized to state $|0\rangle$ are needed to ensure reversibility in such a concentrator. As an example, when $n = 2$ and $d = 1$, we can design such a concentrator with one auxiliary qubit, using the following quantum transformation:

$$
\begin{aligned}
|(0,p_0)(0,p_1)\rangle \, |x\rangle_{aux} &\rightarrow |(0,p_0)(0,p_1)\rangle \, |x\rangle_{aux} \\
|(1,p_0)(1,p_1)\rangle \, |x\rangle_{aux} &\rightarrow |(1,p_0)(1,p_1)\rangle \, |x\rangle_{aux} \\
|(0,p_0)(1,p_1)\rangle \, |0\rangle_{aux} &\rightarrow |(1,p_1)(0,p_0)\rangle \, |p_0 \oplus p_1\rangle_{aux} \\
|(0,p_0)(1,p_1)\rangle \, |1\rangle_{aux} &\rightarrow |(0,p_0)(1,p_1)\rangle \, |\overline{p_0 \oplus p_1}\rangle_{aux} \\
|(1,p_0)(0,p_1)\rangle \, |0\rangle_{aux} &\rightarrow |(1,p_0)(0,p_1)\rangle \, |\overline{p_0 \oplus p_1}\rangle_{aux} \\
|(1,p_0)(0,p_1)\rangle \, |1\rangle_{aux} &\rightarrow |(0,p_1)(1,p_0)\rangle \, |p_0 \oplus p_1\rangle_{aux}
\end{aligned}
\tag{4.7}
$$

This transformation works as desired when the auxiliary qubit is initialized to state $|0\rangle$. It is seen that we trivially have one-to-one mapping when $r_0 = r_1$. For $r_0 \neq r_1$, we list all the input and output patterns in Table 4.3. For a given output pattern $|(1,q_0)(0,q_1)\rangle$, there are two possible input patterns: $|(0,q_1),(1,q_0)\rangle$ and $|(1,q_0),(0,q_1)\rangle$. These two input patterns are differentiated at the output using the auxiliary qubit. Therefore, we have reversibility. Next, we define such a quantum concentrator for the general case.

Definition 4.6. An $n \times n$ quantum network is called an *n-quantum concentrator* (n-QC) if, for each $0 \leq m \leq n$, and any subsequence $i_0, i_1, \cdots, i_{m-1}$ of $0, 1, \cdots, n-1$, it transforms an assignment pattern $\left|\bar{P}\right\rangle = |\,(r_0, \bar{p}_0)\,(r_1, \bar{p}_1) \cdots (r_{n-1}, \bar{p}_{n-1})\rangle$ in which $r_{i_0}, r_{i_1}, \cdots, r_{i_{m-1}}$ are 1 and rest of the routing bits are 0, and a set of d_a auxiliary

Input					Output				
r_0	p_0	r_1	p_1	x	r_0	p_0	r_1	p_1	x
0	0	1	0	0	1	0	0	0	0
0	0	1	1	0	1	1	0	0	1
0	1	1	0	0	1	0	0	1	1
0	1	1	1	0	1	1	0	1	0
1	0	0	0	0	1	0	0	0	1
1	0	0	1	0	1	0	0	1	0
1	1	0	0	0	1	1	0	0	0
1	1	0	1	0	1	1	0	1	1

(a)

Input					Output				
r_0	p_0	r_1	p_1	x	r_0	p_0	r_1	p_1	x
0	0	1	0	1	0	0	1	0	1
0	0	1	1	1	0	0	1	1	0
0	1	1	0	1	0	1	1	0	0
0	1	1	1	1	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0
1	0	0	1	1	0	1	1	0	1
1	1	0	0	1	0	0	1	1	1
1	1	0	1	1	0	1	1	1	0

(b)

Table 4.3: Table for transformation in (4.7) when $r_0 \neq r_1$, concentration occurs for input auxiliary qubit $|x\rangle = |0\rangle$: (a) Input $|x\rangle = |0\rangle$; (b) Input $|x\rangle = |1\rangle$.

qubits initialized to state $|\bar{0}\rangle$, as:

$$\begin{aligned} \left|\bar{P}\right\rangle |\bar{0}\rangle_{aux} &\xrightarrow{n\text{-QC}} \left|QC(\bar{P})\right\rangle \left|\Phi(\bar{P})\right\rangle_{aux} \\ &= \Big| \underbrace{(1, \bar{p}_{j_0})}_{\text{output } 0} \cdots \underbrace{(1, \bar{p}_{j_{m-1}})}_{\text{output } m-1} \underbrace{(0, \bar{p}_{k_0})}_{\text{output } m} \cdots \underbrace{(0, \bar{p}_{k_{n-m-1}})}_{\text{output } n-1} \Big\rangle \left|\Phi(\bar{P})\right\rangle_{aux} \end{aligned} \tag{4.8}$$

where $j_0, j_1, \cdots, j_{m-1}$ is a permutation of the indices $i_0, i_1, \cdots, i_{m-1}$ and k_0, $\cdots$, k_{n-m-1} is a permutation of the rest of the indices in $0, 1, \cdots, n-1$. These permutations are fixed for all the assignment patterns having the same sequence of routing bits. The output state $\left|\Phi(\bar{P})\right\rangle_{aux}$ of the auxiliary qubits is a d_a-bit binary string such that for any two input assignment patterns $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ for which $\left|QC(\bar{P}_1)\right\rangle = \left|QC(\bar{P}_2)\right\rangle$, we have $\left|\Phi(\bar{P}_1)\right\rangle_{aux} \neq \left|\Phi(\bar{P}_2)\right\rangle_{aux}$.

Also, due to linearity, a quantum assignment $\sum_{k=0}^{M-1} \beta_k \left|\bar{P}_k\right\rangle$ is transformed as

$$\left(\sum_{k=0}^{M-1}\beta_k\left|\bar{P}_k\right\rangle\right)|\bar{0}\rangle_{aux}\xrightarrow{n\text{-QC}}\sum_{k=0}^{M-1}\beta_k\left|QC(\bar{P}_k)\right\rangle\left|\Phi(\bar{P}_k)\right\rangle_{aux} \tag{4.9}$$

i.e., all the assignment patterns in the quantum assignment are individually concentrated in parallel. ‖

Theorem 4.2. *At least d_a^{min} auxiliary qubits are needed in order to realize an n-QC, that is $d_a \geq d_a^{min}$, where*

$$d_a^{min}=\left\lceil \log_2\binom{n}{\lceil\frac{n}{2}\rceil}\right\rceil \tag{4.10}$$

Proof: Using similar arguments as given in the proof of Theorem 4.1, we observe that there are $\binom{n}{m}$ assignment patterns which are mapped to a fixed output pattern $|(1,\bar{p}_0)\cdots\ (1,\bar{p}_{m-1})\ (0,\bar{p}_m)\cdots(0,\bar{p}_{n-1})\rangle$, where $0 \leq m \leq n$. The output state of the auxiliary qubits should be different for each of these patterns, therefore, we require that $d_a \geq \left\lceil\log_2\binom{n}{m}\right\rceil$ for every $0 \leq m \leq n-1$. Since $\binom{n}{m}$ is maximum for $m=\lceil\frac{n}{2}\rceil$, we have $d_a \geq \left\lceil\log_2\binom{n}{\lceil\frac{n}{2}\rceil}\right\rceil$.

To show that $d_a \geq \left\lceil\log_2\binom{n}{\lceil\frac{n}{2}\rceil}\right\rceil$ is also sufficient, we observe that two input assignment patterns with different values of m are always mapped to different output patterns, which are uniquely identifiable by the output routing bits. For any fixed m, we have already shown that the output patterns can be uniquely identified by the auxiliary qubits if $d_a \geq \left\lceil\log_2\binom{n}{\lceil\frac{n}{2}\rceil}\right\rceil$. Having uniquely mapped the $2^{n(d+1)}$ input patterns which have all of the auxiliary qubits set to $|0\rangle$, we can map the remaining $(2^{d_a}-1)2^{n(d+1)}$ possible input patterns in which initial state of auxiliary qubits is different from $|0\rangle$, to the remaining $(2^{d_a}-1)2^{n(d+1)}$ output patterns in a one-to-one fashion in any order. Therefore, $d_a^{\min} = \left\lceil \log_2 \binom{n}{\lceil \frac{n}{2} \rceil} \right\rceil$. ∥

Observe that $\left\lceil \log_2 \binom{n}{\lceil \frac{n}{2} \rceil} \right\rceil$ is $O(n)$, therefore an n-QC requires at least $O(n)$ auxiliary qubits. We can construct an n-QC from an n-QC1 using exactly n auxiliary qubits as shown in Figure 4.1. The auxiliary qubits are labeled as $c_0, \cdots, c_{n-1}$ and are initialized to $|0\rangle$. Using controlled-not gates, copies of the routing qubits are created on the auxiliary qubits. These auxiliary qubits are then used as routing qubits in the n-QC1. The input packets to the n-QC along with their routing qubits form the input packets for the n-QC1. At the output side, the routing qubit outputs of the n-QC1 become the auxiliary outputs for the n-QC. It can be verified that the overall circuit is an n-QC, and for an input assignment pattern $\left|\bar{P}\right\rangle = |(r_0, \bar{p}_0) \cdots (r_{n-1}, \bar{p}_{n-1})\rangle$ in which $r_{i_0}, r_{i_1}, \cdots, r_{i_{m-1}}$ are 1, its concentration operation can be expressed as:

$$
\begin{aligned}
&\left|\bar{P}\right\rangle |\bar{0}\rangle_{aux} \xrightarrow{\text{c-not}} |\underbrace{(r_0, (r_0, \bar{p}_0))}_{\text{input } 0}, \cdots, \underbrace{(r_{n-1}, (r_{n-1}, \bar{p}_{n-1}))}_{\text{input } n-1}\rangle \xrightarrow{n\text{-QC1}} \\
&|\underbrace{(r_0, (r_{j_0}, \bar{p}_{j_0}))}_{\text{output } 0} \cdots \underbrace{(r_{m-1}, (r_{j_{m-1}}, \bar{p}_{j_{m-1}}))}_{\text{output } m-1} \underbrace{(r_m, (r_{k_0}, \bar{p}_{k_0}))}_{\text{output } m} \cdots \underbrace{(r_{n-1}, (r_{k_{n-m-1}}, \bar{p}_{k_{n-m-1}}))}_{\text{output } n-1}\rangle \\
&\longrightarrow |\underbrace{(r_{j_0}, \bar{p}_{j_0})}_{\text{output } 0} \cdots \underbrace{(r_{j_{m-1}}, \bar{p}_{j_{m-1}})}_{\text{output } m-1} \underbrace{(r_{k_0}, \bar{p}_{k_0})}_{\text{output } m} \cdots \underbrace{(r_{k_{n-m-1}}, \bar{p}_{k_{n-m-1}})}_{\text{output } n-1}\rangle |r_0, \cdots, r_{n-1}\rangle_{aux}
\end{aligned}
\tag{4.11}
$$

where indices $j_0, \cdots, j_{m-1}$ and $k_0, \cdots, k_{n-m-1}$ are as defined for n-QC1. This is clearly an n-QC transformation. We give a recursive construction of n-QC using $O(n \log n)$ auxiliary qubits in Section 4.2. Also, since the n-QC operation is de-

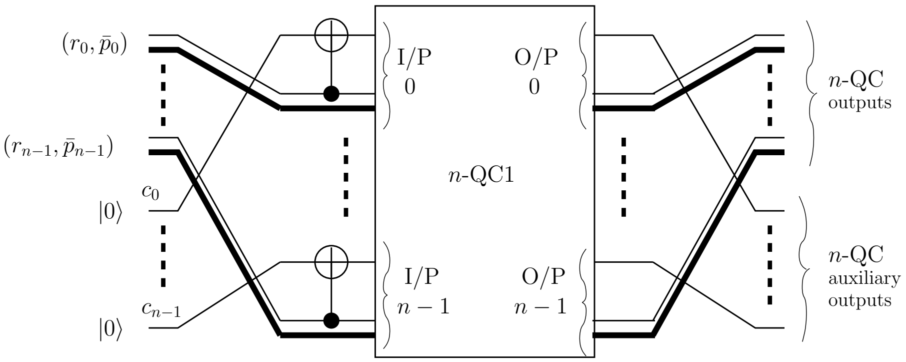


Figure 4.1: Construction of n-QC using an n-QC1.

fined as a permutation of the packets in an assignment pattern, it is easy to verify that the n-QC routes arbitrary quantum states on any m of the inputs to its top m outputs [68]. This can be accomplished by setting the routing qubits on the inputs which have quantum states to be concentrated to $|1\rangle$.

4.2 Quantum Concentrator Construction

We present an n-QC (n being a power of 2) using the classical n-concentrator given by Lee and Oruç [14] [41] as a starting point. A key component of the classical concentrator is a balancer network, which, when combined with an odd-even splitter network, is used to divide the packets in an assignment pattern into two equal sets. This balancer network, as presented in [14] [41], however, requires both forward and backward propagation of routing bits, which is not suitable for implementation using quantum gates. In this section, we first present a recursive quantum balancer network which is designed using controlled-not gates and does not require backward data propagation. This quantum balancer is used to obtain a quantum odd-even splitter which divides the packets with routing bits of 1 in an assignment pattern equally between its odd and even numbered outputs. The packets on odd and even sets of outputs are then recursively concentrated by two $n/2$-QCs and are finally merged using a shuffle stage to obtain an n-QC. In what follows, we give a detailed description of this concentrator and its key building blocks.

4.2.1 Quantum Balancer

An *n-quantum balancer*, denoted as n-QB, is an n-qubit quantum gate which transforms a computational basis vector or bit-string $|b_0 b_1 \cdots b_{n-1}\rangle$, in which m bits are 1, to an output bit-string with $\lfloor m/2 \rfloor$ of these bits converted to 0 and $\lceil m/2 \rceil$ left unchanged, where $1 \leq m \leq n$. The bits b_i which are 0, can be converted to either 0 or 1 so that the overall mapping is reversible, i.e., there is a one-to-one mapping between the 2^n input and output bit-strings. We show by induction that it is possible to construct an n-QB.

For $n = 1$, an identity gate works as a 1-QB. For $n = 2$, the following transform, which is a controlled-not gate, works as a 2-QB: $|00\rangle \rightarrow |00\rangle$, $|01\rangle \rightarrow |01\rangle$, $|10\rangle \rightarrow |11\rangle$, $|11\rangle \rightarrow |10\rangle$. For $n > 2$, suppose that we have a reversible k-QB for every $k < n$. Choose non-zero n_1 and n_2 such that $n_1 + n_2 = n$. The first n_1 qubits are balanced using an n_1-QB and the rest of the qubits are balanced using an n_2-QB. Suppose that there are m_1 ones in string $b_0 \cdots b_{n_1-1}$ and m_2 ones in $b_{n_1} \cdots b_{n-1}$. We complement every output of the n_2-QB if m_1 is odd, otherwise we leave them unchanged. To see that we now have a n-QB: if m_1 is even, $\lfloor m_1/2 \rfloor + \lfloor m_2/2 \rfloor = \lfloor (m_1 + m_2)/2 \rfloor$ of the ones are converted to zeros. If m_1 is odd, $\lfloor m_1/2 \rfloor + \lceil m_2/2 \rceil = \lfloor (m_1 + m_2)/2 \rfloor$ of the ones are converted to zeros. Next, we verify that this network is reversible. Since n_1-QB is reversible, we can uniquely determine $b_0 \cdots b_{n_1-1}$ from the top n_1 outputs, from which we can obtain m_1. We complement the bottom n_2 output bits if m_1 is odd, otherwise we leave them unchanged. Hence, we can uniquely determine $b_{n_1} \cdots b_{n-1}$ using the resulting bits, since n_2-QB is reversible. Therefore, we can construct an n-QB for any $n \geq 1$ using the procedure that has been described.

For n being a power of 2, i.e., $n = 2^p$, we can recursively construct an $O(\log n)$ depth n-QB by choosing $n_1 = n_2 = n/2$. We present a quantum circuit realization of the n-QB using this procedure in Figure 4.2. We place an additional requirement on the n-QB that its last or bottom-most output bit is always equal to the parity of the input string $|b_1 b_2 \cdots b_n\rangle$, while satisfying the functionality of an n-QB given in the above definition. This network consists of two similar $n/2$-QB gates followed by an $(n/2 + 1)$-input controlled-not gate which complements the output bits of the bottom $n/2$-QB when the parity output of the top $n/2$-QB is 1, otherwise it does not affect any of the outputs. Next, we show that this network is an n-QB.

Theorem 4.3. *The recursive network shown in Figure 4.2(a) is an n-QB (for n being a power of 2, i.e., $n = 2^p$) in which the last output is the parity of the input bits.*

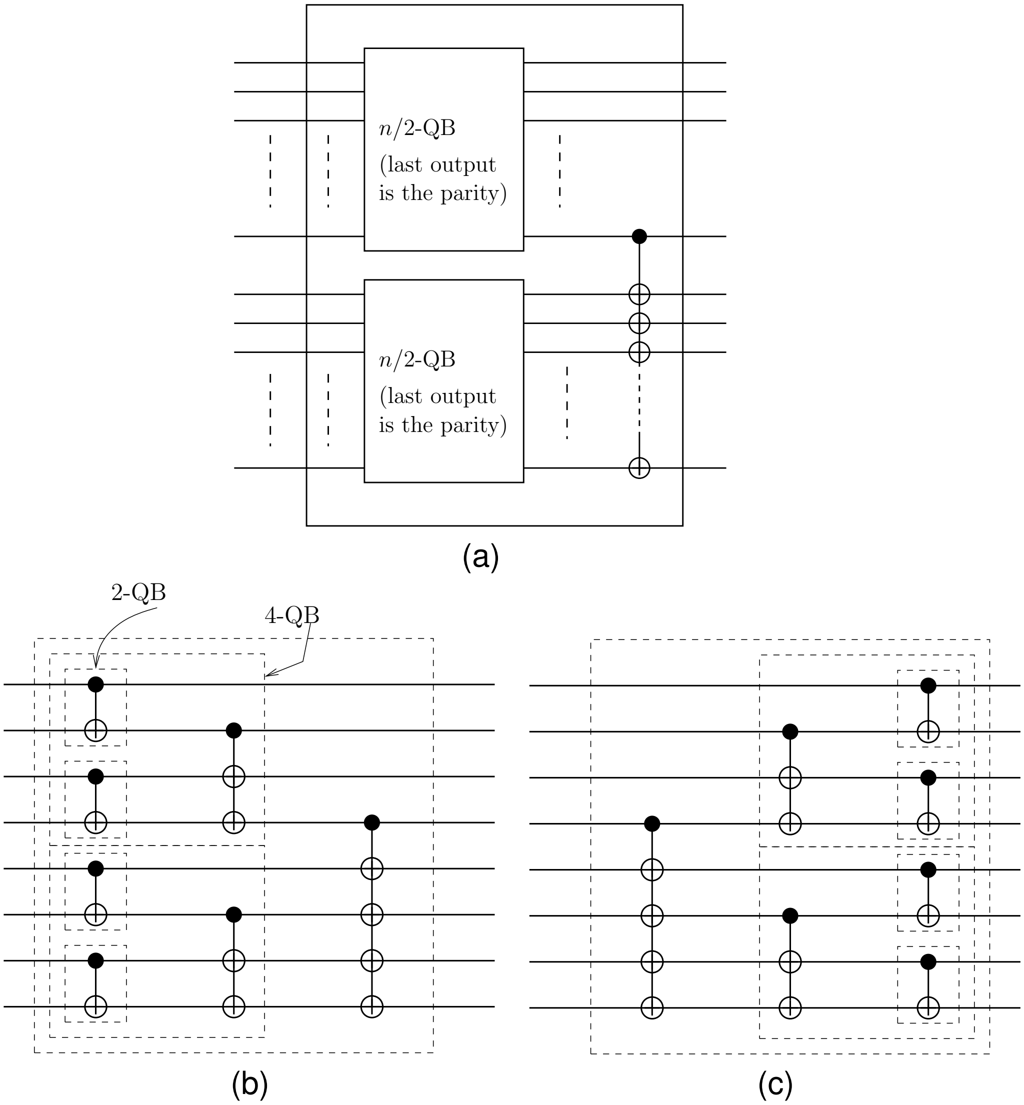


Figure 4.2: Quantum balancer: (a) An n-quantum balancer in which the last output bit is 1 if the inputs have odd parity and is 0 if the input bits have even parity; (b) 8-quantum balancer; (c) Inverse 8-QB.

Proof: For $p = 1$, input sequences 00, 01, 10 and 11 are mapped to 00, 01, 11 and 10 respectively by a controlled-not gate, therefore it is a 2-QB and it also satisfies the parity requirement. For $p > 1$, since the $(n/2 + 1)$-input controlled-not gate complements all the outputs of the lower $n/2$-QB when the last output of upper $n/2$-QB is 1, it is sufficient to show that the bottom-most or last output bit of n-QB represents the parity of its input bits. Suppose that 2^p-QB satisfies this condition. Assuming that m_1 and m_2 are the number of input bits that are 1 in the upper and lower halves of a 2^{p+1}-QB respectively, we have the following four cases for the 2^{p+1}-network.

1. *Both m_1 and m_2 are even*: In this case, the $(2^p + 1)$-input controlled-not gate is not active since the parity output of the top 2^p-QB is 0. The output parity bit is 0 since the parity output of bottom 2^p-QB is 0.

2. *m_1 even and m_2 odd*: Again, the controlled-not gate is not active and the output parity bit is 1 since the parity output of lower 2^p-QB is 1.

3. *m_1 odd and m_2 even*: In this case, the controlled-not gate is active since the parity output of top 2^p-QB is 1. It complements all the output bits of the bottom 2^p-QB. Thus, the output parity bit becomes 1.

4. *Both m_1 and m_2 odd*: In this case also, the controlled-not gate complements all the output bits of the bottom 2^p-QB. Thus, the output parity bit becomes 0.

Therefore, the parity requirement is satisfied for the 2^{p+1}-QB, and this completes the proof. $\|$

It can be easily verified that the mirror image of the n-QB, as shown in Figure 4.2(c), restores the output qubits of the quantum balancer to their initial state. This network is needed for restoring some of the auxiliary qubits used in the quantum odd-even splitter to their initial state as explained in the next section.

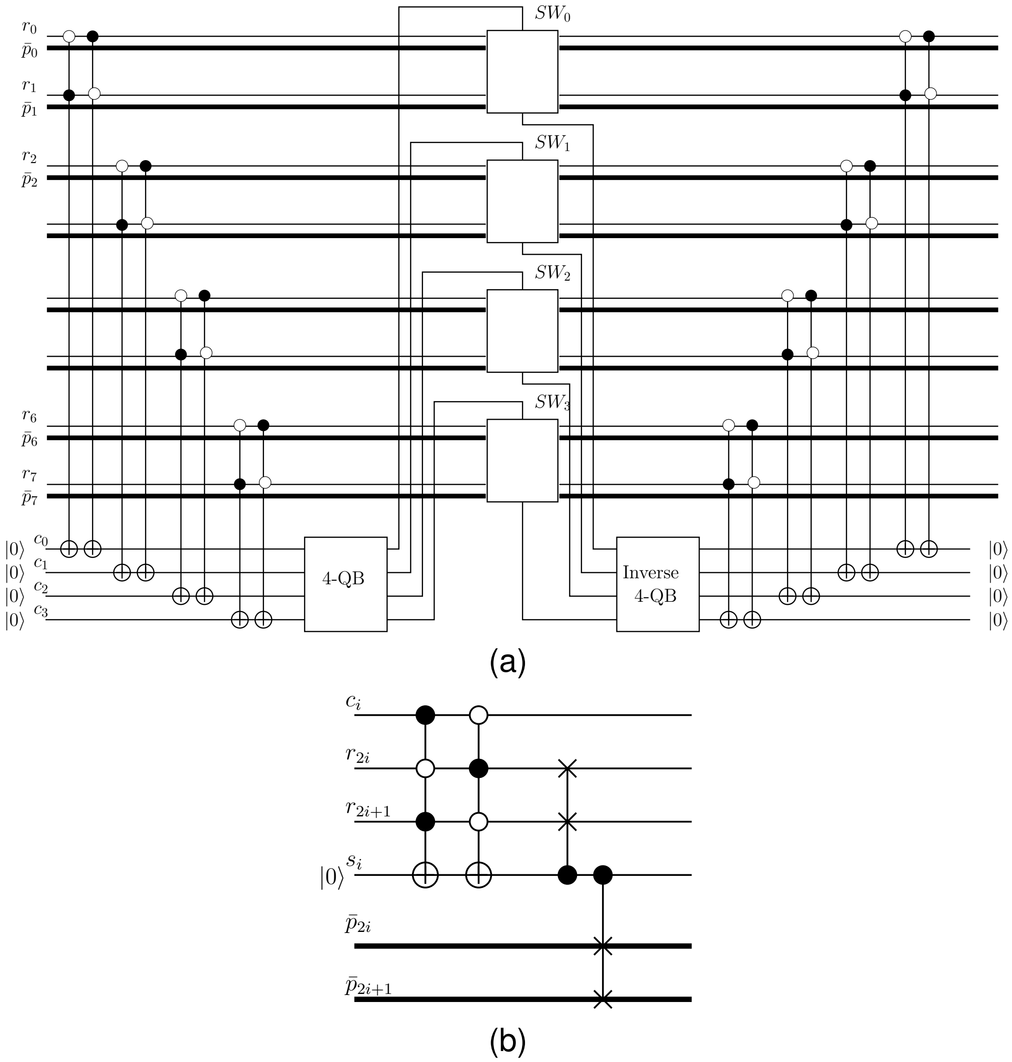


Figure 4.3: Quantum odd-even splitter: (a) 8-input quantum odd-even splitter; (b) Splitter switch (SW_i).

4.2.2 Quantum Odd-Even Splitter

An *n-quantum odd-even splitter* is an $n \times n$ switching network, which permutes an input assignment pattern in which m packets have routing bits of 1, such that $\lceil m/2 \rceil$ of these packets appear on the even outputs and remaining appear on the odd outputs, where $1 \leq m \leq n-1$. A set of auxiliary qubits initialized to $|\bar{0}\rangle$ are used to ensure reversibility.

Our construction of an 8-input odd-even splitter is shown in Figure 4.3. The n-quantum odd-even splitter network consists of $n/2$ splitter switches, SW_0, $SW_1, \cdots, SW_{n/2-1}$ driven by control qubits $c_0, c_1, \cdots, c_{n/2-1}$, respectively. For a given assignment pattern, switch SW_i is said to be *balanced* if it has packets on both of its inputs or no packets at all, i.e., $r_{2i} = r_{2i+1}$. The address and data fields of the packets in the assignment pattern are collectively represented as $\bar{p}_i$ in the figure. Using the $n/2$-QB on qubits $c_0, c_1, \cdots, c_{n/2-1}$, half of the packets at the unbalanced switches are routed to the even numbered outputs and the other half are routed to the odd numbered outputs of quantum odd-even splitter. The packets on the balanced switches are always equally distributed between the odd and even numbered outputs, irrespective of the switch settings. Therefore, the quantum odd-even splitter equally distributes the input packets between the odd and even outputs. A brief description of the quantum circuit follows.

Using two controlled-not gates, qubit c_i is set in state $|1\rangle$ if switch SW_i is unbalanced otherwise it is set in state $|0\rangle$. The $n/2$-QB balances the control qubits c_i, $i = 0, \cdots, n/2 - 1$, which control the splitter switches. The quantum circuit for a splitter switch is shown in Figure 4.3(b). This switch uses one extra qubit s_i, called *switching qubit*, which was not shown in Figure 4.3(a). By setting this qubit appropriately, and using switch gates as shown in the figure, the input packet on an unbalanced switch is routed to the upper (or even) output when the control qubit c_i is $|1\rangle$, otherwise it is routed to the lower (or odd) output. A balanced splitter switch may be set either way without affecting the splitting property of

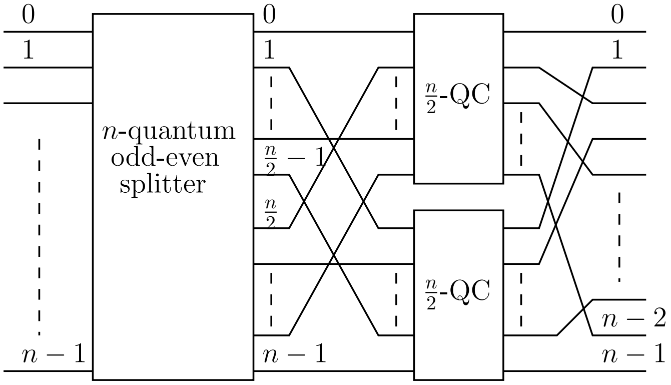


Figure 4.4: Recursive construction of n-QC.

the odd-even splitter.

The quantum gates and the inverse $n/2$-QB shown on the right side of switches SW_i in Figure 4.3(a) are used for restoring qubits c_i to $|0\rangle$. Even though the routing bits may have been switched by the splitter switches along with the input packets, they would still maintain their balanced or unbalanced status. Therefore, these qubits can be used to restore the control qubits c_i to their original state $|0\rangle$ as shown in the figure, so that decoherence on qubits c_i does not have any effect on the performance of the network. Only the switching qubits s_i have not been restored to their initial state.

4.2.3 Construction of n-QC

We can recursively realize an n-QC by using an n-quantum odd-even splitter and two $n/2$-QCs as shown in Figure 4.4. Using a Banyan connection pattern, the even outputs of the odd-even splitter are connected to the upper $n/2$-QC and the odd outputs are connected to the lower $n/2$-QC. The outputs of $n/2$-QCs are connected alternately to the final n outputs using a shuffle connection pattern.

Theorem 4.4. *The network shown in Figure 4.4 is an n-QC.*

Proof: For $n = 2$, a splitter switch in which the control input is set to $|1\rangle$, works as a 2-QC. For $n = 2^p$, where $p > 1$, consider an input assignment pattern in which m packets have routing bits of 1, where $0 \leq m \leq n$. The odd-even splitter transforms it to an assignment pattern in which $\lceil m/2 \rceil$ of these packets are at the inputs of the upper $n/2$-QC and $\lfloor m/2 \rfloor$ are at the inputs of the lower $n/2$-QC. The $n/2$-QCs concentrate their input assignment patterns and these packets are concentrated at their top $\lceil m/2 \rceil$ and $\lfloor m/2 \rfloor$ outputs respectively. These packets are routed to the top m outputs of the n-QC using the shuffle connection. Therefore by induction, the packets having routing bits of 1 are concentrated to the top m outputs and the packets with routing bits of 0 are sent to the bottom $n - m$ outputs.

We now show that when an n-QC maps the packets in two or more input assignment patterns with the same number of coinciding routing bits of 1, it applies same permutation to all of them. Since the sequence of routing bits is fixed for all assignment patterns, the settings of the splitter switches in the odd-even splitter are fixed, therefore, the sequences of routing bits at the inputs of the upper and lower $n/2$-QCs are also fixed. This inductively implies that the setting of every splitter switch in the network is fixed. Consequently, same permutation is applied to all assignment patterns having the same sequence of routing bits.

Next, we show that if two different assignment patterns $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ are mapped to the same output pattern $\left|QC(\bar{P}_1)\right\rangle = \left|QC(\bar{P}_2)\right\rangle$, then $\left|\phi(\bar{P}_1)\right\rangle_{aux} \neq \left|\phi(\bar{P}_2)\right\rangle_{aux}$. Since the control qubits are restored to their initial state, only the switching qubits constitute as auxiliary qubits. We need to show that the output state of at least one of the switching qubits is different for these two input assignment patterns. Clearly, $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ must have the same number of packets with routing bits of 1. Also, since any two assignment patterns with coinciding routing bits of 1 are concentrated using the same permutation, there must be at least one input at which the routing bits in $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ are different. Therefore, this input would be connected to one of the top m outputs for one pattern and one of the bottom $n - m$ outputs for the other pattern. Hence, the overall $n \times n$ permutations applied to concentrate $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ must be different. The topology of the n-QC is easily seen to provide a unique path between each input and each output. Therefore, for the two permutations to be different, at least one of the 2×2 splitter switches must be set in through state for one of the assignment patterns and in cross state for the other. Consequently, the output states of the two assignments are different when the switching qubits s_i are taken into account and we have $\left|\phi(\bar{P}_1)\right\rangle_{aux} \neq \left|\phi(\bar{P}_2)\right\rangle_{aux}$. ||

To illustrate the concentration operation done by n-QC, we give an example for $n = 4$. An expanded quantum circuit for the 4-QC is shown in Figure 4.5. The input quantum assignment pattern has three classical packets A, B and C on inputs 1, 2 and 3 respectively. Input 0 has no packet, i.e., $r_0 = 0$. The address

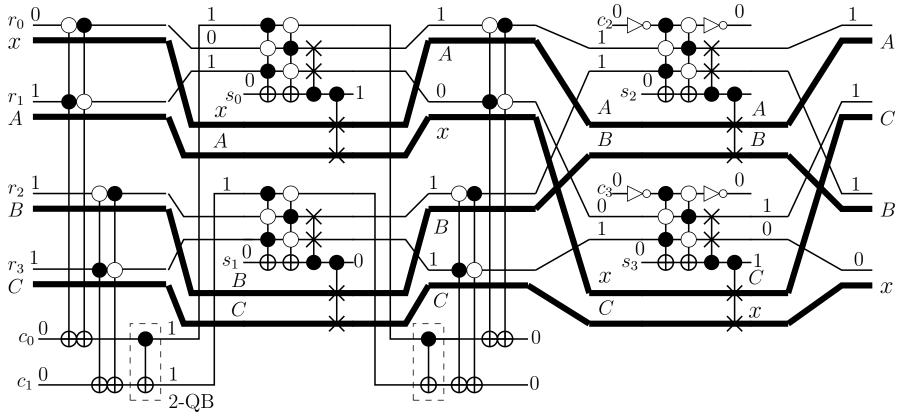


Figure 4.5: 4-QC quantum circuit.

and data fields of this input are denoted as x in the figure. Eight auxiliary qubits labeled $c_0, \cdots, c_3$ and $s_0, \cdots, s_3$ are used. All of them are initialized to state $|0\rangle$. For clarity, we have not used the *ket* notation in the figure. The transformation done by the 4-QC in this example is expressed as:

$$|(0,x)(1,A),(1,B),(1,C)\rangle\, |\bar{0}\rangle_{aux} \xrightarrow{\text{4-QC}} |(1,A)(1,C),(1,B),(0,x)\rangle\, |0_{c_0}0_{c_1}0_{c_2}0_{c_3}1_{s_0}0_{s_1}0_{s_2}1_{s_3}\rangle_{aux}$$

It is seen that the input packets are concentrated on the top three outputs of the 4-QC. Control qubits c_i are restored to $|0\rangle$, whereas some of the switching qubits s_i are not restored.

In the next section, we modify the n-QC given in this section to design an order-preserving quantum concentrator. We show later in Chapter 5 that if we use the order-preserving quantum concentrator to design an n-GQC then it is easier to characterize the routing of contending assignment patterns through the

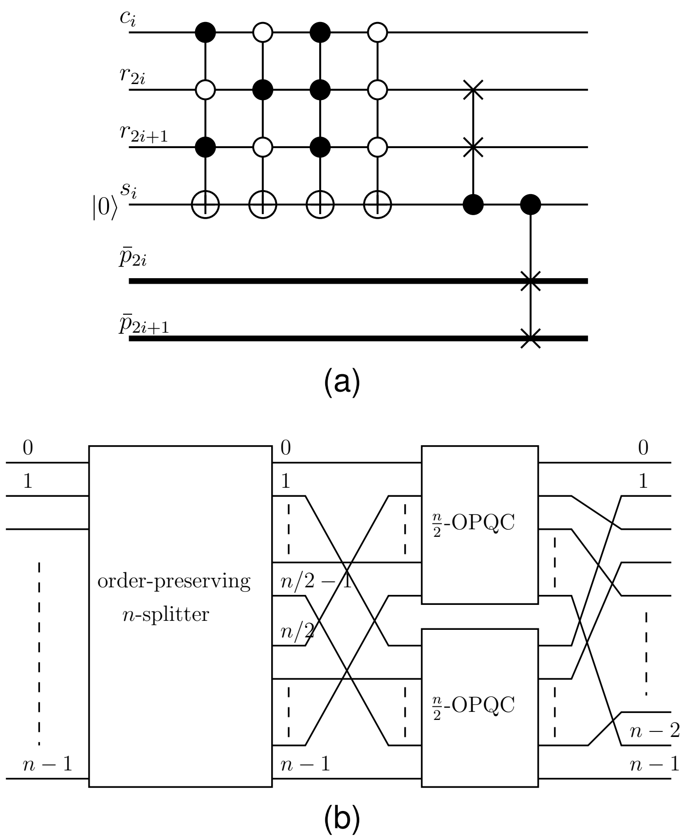


Figure 4.6: Order-preserving quantum concentrator: (a) Splitter switch used in the order preserving n-splitter; (b) Order preserving n-quantum concentrator.

n-GQC. This concentrator is also used to construct the priority quantum concentrator given in Chapter 6.

4.3 Order-preserving Quantum Concentrator

An n-QC is called an *order-preserving n-quantum concentrator* (n-OPQC) if it concentrates the input packets in an assignment pattern to the top outputs while maintaining the order in which they appear on the inputs. In other words, when the sequence of indices $j_0, j_1, \cdots, j_{m-1}$ is the same as the sequence $i_0, i_1, \cdots, i_{m-1}$ in the n-QC definition given in Definition 4.6, the n-QC is called an n-OPQC. For an n-OPQC, the output pattern given in Definition 4.6 becomes

$$\big|\underbrace{(1,\bar{p}_{i_0})}_{\text{output } 0}\cdots\underbrace{(1,\bar{p}_{i_{m-1}})}_{\text{output } m-1}\underbrace{(0,\bar{p}_{k_0})}_{\text{output } m}\cdots\underbrace{(0,\bar{p}_{k_{n-m-1}})}_{\text{output } n-1}\big\rangle \tag{4.12}$$

We can construct an n-OPQC using the n-QC design given in Section 4.2.3 by modifying the quantum odd-even splitter stage such that, for $0 \leq k \leq m$, packet $(1, p_{i_k})$ is sent to the top output of its splitter switch if k is even, otherwise it is send to the bottom output. We call this odd-even splitter an *order-preserving n-splitter*. It can be constructed by introducing a small modification in the splitter switches of the n-quantum odd even splitter. The new splitter switch routes its input packets in the following way:

- If the switch is unbalanced, then it works in exactly the same way as the n-QC splitter switch.
- If the switch is balanced and both the input packets have their routing bits set to 1, then it works in through state if its control input is 0, otherwise it works in cross state. A balanced splitter switch in which both input packets have routing bits of 0 can be set in either state.

The quantum circuit for this splitter switch is shown in Figure 4.6(a). To show that the resulting odd-even splitter is an order-preserving n-splitter, we first observe the following for our n-QB construction:

Theorem 4.5. *The n-QB (where $n = 2^p$) given in Figure 4.2 transforms an input string $|b_0, \cdots, b_{n-1}\rangle$ such that*

1. *Bit b_0 is not modified.*
2. *For $1 \leq k \leq n-1$, bit b_k is complemented if string $b_0, \cdots, b_{k-1}$ contain odd number of ones otherwise it is left unchanged.* $\|$

Proof: We can prove this inductively using Figure 4.2(a). For $n = 2$, controlled-not gate transformation $|00\rangle \rightarrow |00\rangle$, $|01\rangle \rightarrow |01\rangle$, $|10\rangle \rightarrow |11\rangle$, $|00\rangle \rightarrow |00\rangle$ satisfies above. For $n = 2^{p+1}$, we assume that the two 2^p-QB transformations satisfy the above property. Since the top 2^p outputs of the 2^{p+1}-QB are the same as those of the upper 2^p-QB, they satisfy the above property. Bits $b_{2^p}, \cdots, b_{2^{p+1}-1}$ are balanced by the bottom 2^p-QB, with bit b_{2^p} serving as b_0. Since these outputs are complemented by the controlled-not gate when string $b_0, \cdots, b_{2^p-1}$ has odd number of ones and left unchanged otherwise, the property given in the theorem is satisfied for the bottom 2^p outputs of the 2^{p+1}-QB as well. $\|$

We now have the following:

Theorem 4.6. *The odd-even splitter of Figure 4.3(a) in which the modified splitter switches of Figure 4.6(a) are used, works as an order preserving n-splitter.*

Proof: Splitter switch $SW_{\lfloor i_0/2 \rfloor}$, which has input packet $(1, p_{i_0})$ is either the topmost splitter switch (when $i_0 = 0$ or 1), or is preceded by $\lfloor i_0/2 \rfloor$ balanced switches having packets with routing qubit set to 0. Therefore, using the above theorem, the control input for this switch is not modified by the n-QB. If $SW_{\lfloor i_0/2 \rfloor}$ is balanced then its control input is 0 and it works in through state. In this case, i_0 is the top input, and therefore, packet $(1, p_{i_0})$ is sent to the top output of $SW_{\lfloor i_0/2 \rfloor}$. If $SW_{\lfloor i_0/2 \rfloor}$ is unbalanced then its control input is 1 and $(1, p_{i_0})$ is sent to the top output.

For an even k, where $k > 0$, if packet $(1, p_{i_k})$ is on the top input of its splitter switch $SW_{\lfloor i_k/2 \rfloor}$, then this switch is preceded by an even number of unbalanced switches. Consequently, using Theorem 4.5, its control input is not modified by the n-QB. Using similar arguments as given above for $SW_{\lfloor i_0/2 \rfloor}$, we observe that $(1, p_{i_k})$ is routed to the top output of switch $SW_{\lfloor i_k/2 \rfloor}$. If $(1, p_{i_k})$ is on the bottom input of $SW_{\lfloor i_k/2 \rfloor}$, then this switch is preceded by an even number of unbalanced switches if it is unbalanced otherwise it is preceded by an odd number of such switches. In either case, the control input of $SW_{\lfloor i_k/2 \rfloor}$ is 1, therefore, $(1, p_{i_k})$ is sent to the top output.

For an odd k, if $(1, p_{i_k})$ is on the top input of its splitter switch $SW_{\lfloor i_k/2 \rfloor}$, then this switch is preceded by an odd number of unbalanced switches. In this case, the control input is complemented by n-QB and it is 1 if $SW_{\lfloor i_k/2 \rfloor}$ is balanced and 0 otherwise. Therefore, $(1, p_{i_k})$ is sent to the bottom output. If packet $(1, p_{i_k})$ is on bottom input, $SW_{\lfloor i_k/2 \rfloor}$ is preceded by an even number of unbalanced switches if it is balanced, otherwise it is preceded by an odd number of unbalanced switches. Therefore, the control input to switch $SW_{\lfloor i_k/2 \rfloor}$ is 0 and $(1, p_{i_k})$ is routed to the bottom output. This completes the proof of the fact that the new odd-even splitter is an order-preserving n-splitter. $\|$

Theorem 4.7. *The recursive network shown in Figure 4.6(b) is an n-OPQC.*

Proof: The proof is by induction. For $n = 2$, an order-preserving 2-splitter works as a 2-OPQC. For $n = 2^p$, where $p > 1$, the order-preserving n-splitter routes an assignment pattern $\left|\bar{P}\right\rangle = |\,(r_0, \bar{p}_0)\,(r_1, \bar{p}_1) \cdots (r_{n-1}, \bar{p}_{n-1})\rangle$ in which $r_{i_0}, r_{i_1}, \cdots, r_{i_{m-1}}$ are $|1\rangle$, such that packets $(1, p_{i_k})$ with even k are routed to the upper $n/2$-OPQC and appear at its inputs in increasing order of k from top to bottom. Packets $(1, p_{i_k})$ with odd k are sent to the bottom $n/2$-OPQC and appear at its inputs in increasing order of k. These packets are concentrated to the top outputs of the $n/2$-OPQCs in the same order in which they appear on their inputs. Therefore, the packets on the top $\lceil m/2 \rceil$ outputs of the upper $n/2$-OPQC are $(1, p_{i_0}), (1, p_{i_2}), \cdots, (1, p_{i_{2(\lceil m/2 \rceil - 1)}})$ and the output packets on the top $\lfloor m/2 \rfloor$ packets of the lower $n/2$-OPQC are $(1, p_{i_1}), (1, p_{i_3}), \cdots, (1, p_{i_{2\lfloor m/2 \rfloor}+1})$. The shuffle stage routes these packets alternately to the top m outputs, therefore, the packets on the top m outputs are $(1, p_{i_0}), (1, p_{i_1}), \cdots, (1, p_{i_{m-1}})$ in that order. This proves that the overall network is an n-OPQC. $\|$

In the next section, we give a construction for n-QC2. This network is obtained by modifying the n-QC to restore all auxiliary qubits to their initial states using the data parts of the packet with routing bits of 0.

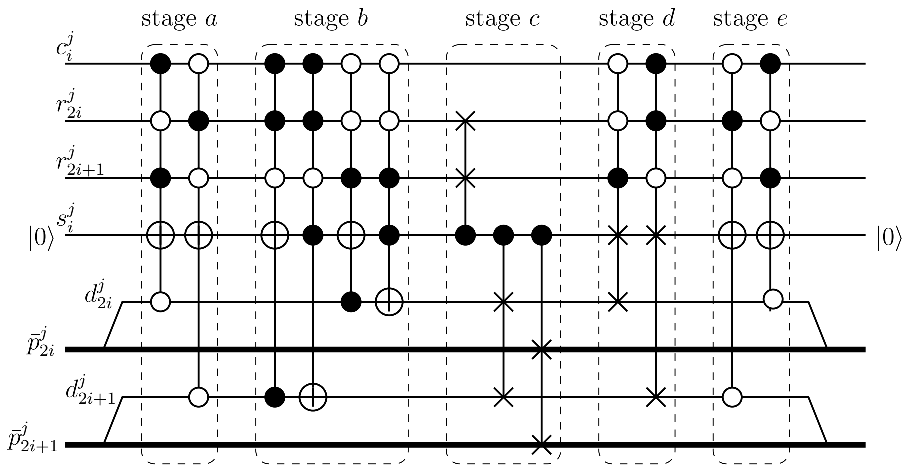


Figure 4.7: Quantum circuit for n-QC2 splitter switch SW_i^j.

4.4 Quantum Concentrator of Second Type (n-QC2)

In Theorem 4.1, given in Section 4.1, we showed that we can construct an n-QC2 for zero-concentration set ζ_0 if the data part of each input packet has at least $d_{min} = \lceil \log_2 n \rceil$ bits. For $n = 2^p$, we have $d_{min} = p$, which is equal to the number of stages in the n-QC construction given in Section 4.2.3. Assuming that each packet has at least p data bits, we show that an n-QC2 for the zero-concentration set ζ_0 can be realized by modifying the splitter switches used in the n-QC.

The quantum circuit for the modified splitter switch SW_i^j, which is the i^{th} splitter switch in the j^{th} stage of the network, is shown in Figure 4.7, where $0 \leq i \leq \frac{n}{2} - 1$ and $0 \leq j \leq p - 1$. Similar to the splitter switches used in n-QC, SW_i^j works in through state when it is balanced and the switching qubit s_i^j remains in state $|0\rangle$. When SW_i^j is unbalanced, we restore s_i^j to $|0\rangle$ by swapping it with the j^{th} data bit of the packet with routing bit of 0. Denoting the j^{th} data bits of the two input packets as d_{2i}^j and d_{2i+1}^j respectively, the functionality of SW_i^j, when it is unbalanced, is described as follows:

Case 1: *When* $(r^j_{2i}, d^j_{2i}) = (0, 0)$ *or* $(r^j_{2i+1}, d^j_{2i+1}) = (0, 0)$: This always holds for zero-assignment patterns because the j^{th} data bit of the packet with routing bit of 0 is not modified in any of the previous stages. In this case, the splitter switch works similar to the n-QC splitter switch. Only the quantum gates in stages a, c and d of the circuit operate. If s^j_i is 1 after stage b, i.e., the switch is set in cross state, then s^j_i is swapped with the j^{th} data bit of the packet with routing bit of 0, in stage d. Therefore, s^j_i is restored to $|0\rangle$. The quantum gates in stages a and d work only in Case 1 and not in the next case in which the input packet with routing bit of 0 does not have its j^{th} data bit set to 0.

Case 2: *When* $(r^j_{2i}, d^j_{2i}) = (0, 1)$ *or* $(r^j_{2i+1}, d^j_{2i+1}) = (0, 1)$: This may happen only those assignment patterns which are not in ζ_0. In this case when c^j_i is $0/1$, we send the packet with routing bit of 1 to the bottom/top output and the packet with routing bit of 0 to the top/bottom output. The j^{th} data bit of the packet having routing bit of 0 is reset to 0. All of this is accomplished by stages b and c. Finally, stage e resets s^j_i to $|0\rangle$ if it was set to 1 in stage b. The quantum gates in stages b and e work only in Case 2 and not in Case 1.

We now have the following:

Theorem 4.8. *An n-QC which uses splitter switches SW^j_i shown in Figure 4.7, for $0 \le i \le \frac{n}{2} - 1$ and $0 \le j \le p$, is an n-QC2 for the zero-concentration set ζ_0.* $\|$

Proof: Since switching qubits are always restored to state $|0\rangle$, all the auxiliary qubits are restored to their initial state of $|0\rangle$ for every assignment pattern. Also, for a zero-assignment pattern, the data part of any packet having a routing bit of 1 is not altered and all such packets are concentrated to the top outputs in the same way as in n-QC.

For a given zero-assignment pattern $|\bar{P}\rangle$ having m packets, we can express the output pattern $|QC2(\bar{P})\rangle$ as $|QC2(\bar{P})_m, (0, \bar{p}'_m), \cdots, (0, \bar{p}'_{n-1})\rangle$, where $QC2(\bar{P})_m$ denotes the packets on top m outputs, each having a routing bit of 1. An input packet having a routing bit of 0 is sent to one of the bottom $n-m$ outputs, and its j^{th} data bit at the output of the network is 1 if the splitter switch in its path at the j^{th} stage of the network was set in a cross state, otherwise, it is 0, where $0 \leq j \leq p-1$. Therefore, the first p bits of the data part of an output packet $(0, \bar{p}'_k)$, $m \leq k \leq n-1$, uniquely identify its input address. String $(0, \bar{p}'_m), \cdots, (0, \bar{p}'_{n-1})$ uniquely identifies all of the inputs which have packets with routing bits of 0, consequently, this string is unique for all assignment patterns having a fixed sequence of routing bits. As we have shown earlier, two zero-assignment patterns $|\bar{P}_1\rangle$ and $|\bar{P}_2\rangle$ having m packets for which $QC2(\bar{P}_1)_m = QC2(\bar{P}_2)_m$ must have different sequences of routing bits, which implies that string $(0, \bar{p}'_m), \cdots, (0, \bar{p}'_{n-1})$ is different for these patterns. Therefore, for any $|\bar{P}_1\rangle, |\bar{P}_2\rangle \in \zeta_0$, we have $|QC2(\bar{P}_1)\rangle \neq |QC2(\bar{P}_2)\rangle$, which completes the proof. $\|$

4.5 Complexity Analysis

In this section, we compute the complexities of our n-QC design in terms of the total number of quantum gates and the gate level depth. Representing these complexities as $C_{qc}(n)$ and $D_{qc}(n)$ respectively, we have

$$C_{qc}(n) = 2C_{qc}(n/2) + C_{split}(n) \tag{4.13}$$

$$D_{qc}(n) = D_{qc}(n/2) + D_{split}(n) \tag{4.14}$$

where $C_{split}(n)$ and $D_{split}(n)$ are the corresponding complexities for an n-quantum odd-even splitter. Since an n-QB has $n-1$ controlled-not gates, and each splitter switch has a constant number of controlled-not gates, $C_{split}(n)$ is $O(n)$. Also, the depth of the n-quantum odd-even splitter is mainly determined by the depth of n-QB, which is equal to $\log_2 n$. Consequently, $D_{split}(n)$ is $O(\log n)$. Thus, we have $C_{qc}(n) = O(n \log n)$ and $D_{qc}(n) = O(\log^2 n)$.

Chapter 5: Quantum Multicasting

In this chapter, we give our work on the design of a quantum multicasting network called generalized quantum connector (GQC) [3]. Since general quantum states cannot be replicated, this network multicasts classical packets contained in quantum states. Copying needed for such multicasting is obtained using controlled-not gate based copier described in Section 2.1.4.2. The GQC is constructed recursively using the n-QC given in Chapter 4. We also study the behavior of GQC when routing contending assignment patterns. We start out by defining this network in terms of how it transforms multicast assignment patterns.

5.1 Generalized Quantum Connector

A classical switching network is called a *generalized connector* if it realizes any non-contending multicast assignment pattern between its inputs and outputs. In this section, we extend this definition to the quantum domain for multicasting quantum packets. We go back to the notation introduced in Section 2.2.2.1 to represent quantum assignment patterns, in which an input classical packet is represented by the 3-tuple $(r, \bar{a}, \bar{p})$ where $\bar{a}$ and $\bar{p}$ are m_a and m_d bit binary strings respectively.

An *n-generalized quantum connector or n-GQC* is an $n \times n$ quantum switching network which transforms any non-contending assignment pattern $\left|\bar{P}\right\rangle = \left|(r_0, \bar{a}_0, \bar{d}_0), \cdots, (r_{n-1}, \bar{a}_{n-1}, \bar{d}_{n-1})\right\rangle$ such that, for $i = 0, \cdots, n-1$, if $r_i = 1$ then the data $\bar{d}_i$ of the packet on input i is copied onto the data fields of all the outputs in its fanout set. The routing bits on these outputs are set to 1. The routing bit of an output to which no input packet is addressed is set to 0. Each output consists of routing and data qubits only and does not contain address qubits. We represent this transformation as:

$$\left|\bar{P}\right\rangle \left|\bar{0}\right\rangle_{\text{aux}} \xrightarrow{n\text{-GQC}} \left|GQC(\bar{P})\right\rangle \left|\Psi(\bar{P})\right\rangle = \left|(r'_0, \bar{d}'_0), \cdots, (r'_{n-1}, \bar{d}'_{n-1})\right\rangle \left|\Psi(\bar{P})\right\rangle \qquad (5.1)$$

The auxiliary qubits on the left hand side are needed for two reasons: to ensure reversibility and to create copies of the input packets. The auxiliary and address qubits are transformed to a state $\left|\Psi(\bar{P})\right\rangle$ such that, for any two different input assignment patterns $\left|\bar{P}_1\right\rangle$ and $\left|\bar{P}_2\right\rangle$ for which $\left|GQC(\bar{P}_1)\right\rangle = \left|GQC(\bar{P}_2)\right\rangle$, we have $\left|\Psi(\bar{P}_1)\right\rangle \neq \left|\Psi(\bar{P}_2)\right\rangle$. Again, due to the linearity of quantum networks, an n-GQC simultaneously realizes all the assignment patterns in a non-contending quantum assignment, i.e., a quantum assignment $\sum_{k=0}^{M-1} \beta_k \left|\bar{P}_k\right\rangle$, in which every assignment pattern is non-contending, is transformed as:

$$\left(\sum_{k=0}^{M-1} \beta_k \left|\bar{P}_k\right\rangle\right) \left|\bar{0}\right\rangle_{aux} \xrightarrow{n\text{-GQC}} \sum_{k=0}^{M-1} \beta_k \left|GQC(\bar{P}_k)\right\rangle \left|\Psi(\bar{P}_k)\right\rangle \qquad (5.2)$$

The address bits in our quantum packet representation denote the outputs of the n-GQC and they are not needed once the packets have reached their desired n-GQC outputs. When the n-GQC is used as a part of a larger network and the packets are to be routed further, extra address bits are needed. In our representation, these address bits can be included in the data parts of the packets and they are not discarded by the n-GQC.

Before giving our n-GQC construction, we describe the addressing schemes that we use to represent address fields in quantum packets.

5.1.1 Addressing Schemes

As mentioned in Section 2.2.2.1, since there are 2^n possible fanout sets for each input in an $n \times n$ network, at least n bits are needed per input to address these patterns. The most straightforward way to code these fanout sets is to allocate n bits $o_0, o_1, \cdots, o_{n-1}$ for each input, in which o_j is set to 1 when that input is paired with output j. In this paper, we use a $(2n-2)$-bit addressing scheme that is more suitable for quantum circuit realization of a multistage n-GQC. Both of these addressing schemes were introduced in [14].

In the $(2n-2)$-bit addressing scheme, where n is a power of 2, each input uses a binary address of the form $b_{00}b_{01}, b_{10}b_{11}b_{12}b_{13}, \cdots$, $b_{p-1,0}b_{p-1,1} \cdots b_{p-1,n-1}$ to specify the outputs it is paired with, where $p = \log_2 n$. The first two bits specify whether the packet at an input is routed to the upper half or lower half or both upper and lower halves of outputs or not routed at all. The next group of four

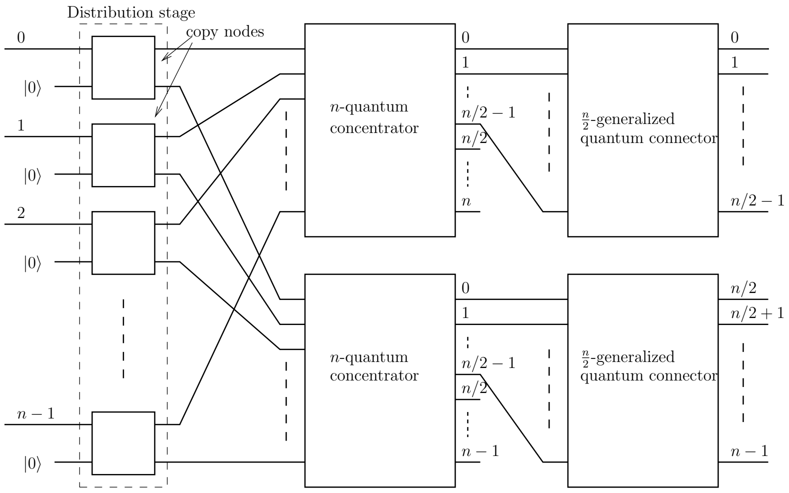


Figure 5.1: Generalized quantum connector.

bits is then used to resolve the location of the same packet within each half of the upper and lower halves of outputs, and this is inductively extended. More specifically, for $k = 0, \cdots, p-1$, the outputs are divided into 2^{k+1} sets of size 2^{p-k-1} of the form $i2^{p-k-1} \leq j \leq (i+1)2^{p-k-1} - 1$, where $i = 0, \cdots, 2^{k+1} - 1$. For a given input, an address bit b_{ki} is set to 1 when that input is paired with at least one output in set $i2^{p-k-1} \leq j \leq (i+1)2^{p-k-1} - 1$.

We use the $(2n-2)$-bit addressing scheme in this paper since it leads to a simpler quantum circuit implementation. These bits collectively form the address field $\bar{a}_i$ in the representation of a quantum packet given in (2.31).

5.1.2 Construction of n-GQC

In this section, we present a multistage quantum network realization of n-GQC, where $n = 2^p$. It is a recursive network consisting of a distribution stage

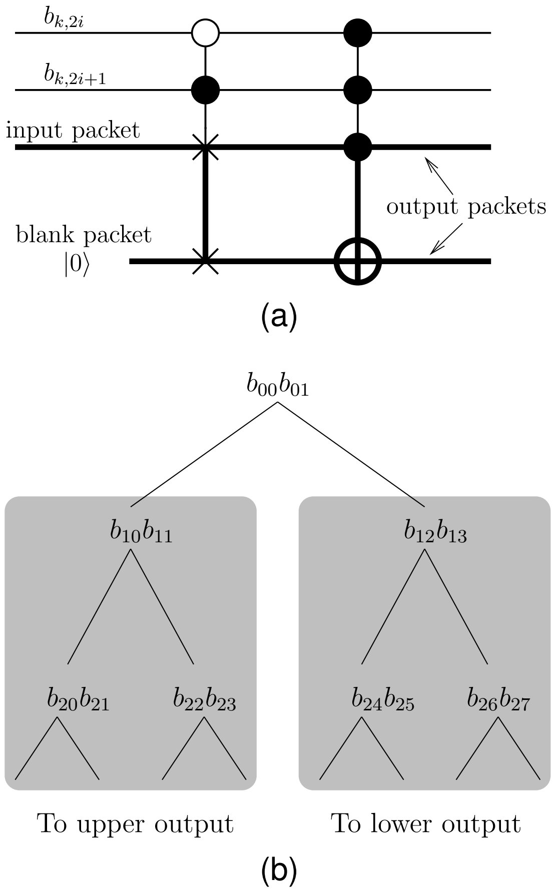


Figure 5.2: Copy node: (a) Quantum circuit for a copy node; (b) Division of the address field at a copy node for the $(2n-2)$-bit addressing scheme.

followed by two n-QCs and two $n/2$-GQCs, as shown in Figure 5.1. The distribution stage consists of n copy nodes, labeled 0 to $n-1$ from top to bottom. Given an input assignment pattern for the distribution stage, a copy node does one of the following:

- It creates copies of its input packet on both of its outputs if the fanout set of the packet contains at least one output from both top and bottom $n/2$ outputs of the n-GQC. For the $(2n-2)$-bit addressing scheme, this happens when $b_{00}b_{01} = 11$.

- It routes the input packet on its upper output if that input packet has at least one output from the top $n/2$ outputs but no output from the bottom $n/2$ outputs of the n-GQC in its fanout set. No packet is sent to the bottom half of outputs in this case. This happens when $b_{00}b_{01} = 10$.

- It routes the input packet on its lower output if that input packet has at least one output from the bottom $n/2$ outputs but no output from the top $n/2$ outputs of the n-GQC in its fanout set. No packet is sent to the top half of outputs in this case. This happens when $b_{00}b_{01} = 01$.

- It divides the remaining address bits between the two outputs using the scheme shown in Figure 5.2(b). Consequently, the size of the address fields in the packets received by the $n/2$-GQCs in Figure 5.1 is $n - 2$. The same addressing scheme is recursively followed in the subsequent stages and a copy node always uses the first two address bits of its input packet to determine its settings, irrespective of its stage.

The address bits $b_{00}b_{01}$ are never set to 00 for an input packet with routing bit of 1. Consequently, the routing and data bits of an input packet having $b_{00}b_{01} = 00$ can be passed to either of the outputs without affecting the n-GQC functionality.

The quantum circuit implementation of a copy node is shown in Figure 5.2(a). When b_{00} and b_{01} are 00, routing and address bits are passed to the upper output by convention. When b_{00} and b_{01} are 10, the routing and data bits are passed to the upper output. When they are 01, the routing and data bits of the input packet are sent to the lower output by swapping them with a blank quantum packet initialized in state $|\bar{0}\rangle$, using the switch gate as shown. When b_{00} and b_{01} are 11, copies of the routing and data bits are created on both outputs of the copy node by using the multi-qubit quantum copier gate. The remaining address bits are divided between the two outputs as described earlier.

The routing and data qubits on an output to which no packet is sent are set to $|0\rangle$. The top $n/2$ outputs of the two n-QCs are connected to two $n/2$-GQCs in the next stage, as shown in Figure 5.1. The bottom $n/2$ outputs of the n-QCs are dropped. We now have the following theorem:

Theorem 5.1. *The network shown in Figure 5.1 is an n-GQC.*

Proof: For a non-contending assignment pattern of size n, a maximum of $n/2$ packets are assigned to be routed to the top as well as to the bottom $n/2$ outputs of the n-GQC. Therefore, the output of the distribution stage is an assignment pattern of size $2n$, which is a concatenation of two n-assignment patterns at the inputs of the two n-QCs, each pattern having a maximum of $n/2$ packets with routing bits of 1. These packets are concentrated on the top $n/2$ outputs of the n-QCs. The bottom $n/2$ outputs of the n-QCs receive no packets. These outputs are in state $|\bar{0}\rangle$ and are dropped. Therefore, all the auxiliary qubits used in the distribution stage are restored to their initial state. The two $n/2$-GQCs receive non-contending assignment patterns of size $n/2$ each, which are inductively realized. For $n = 2$, a copy node works as a 2-GQC. Consequently, the shown quantum network is an n-GQC by induction, where n is a power of 2.

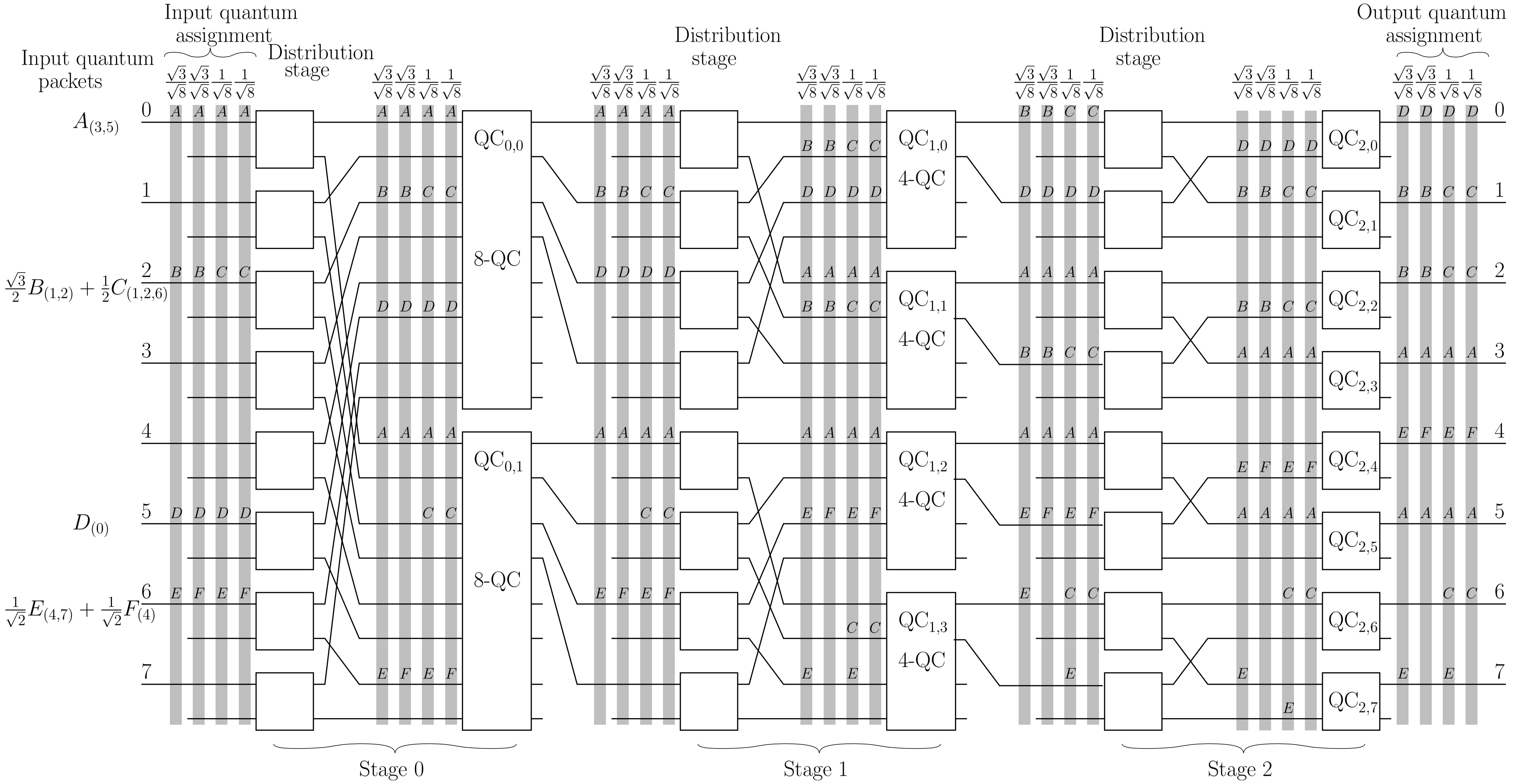


Figure 5.3: 8-GQC realizing a non-contending quantum assignment.

An expanded version of n-GQC is shown in Figure 5.3 for $n = 8$. For simplicity, we have not expanded the quantum concentrators in the figure. We denote the 2^{p-k}-QCs in the k^{th} concentrator stage of the n-GQC as $\mathrm{QC}_{k,0}$, $\mathrm{QC}_{k,1}$, $\cdots$, $\mathrm{QC}_{k,2^{k+1}-1}$ from top to bottom, where $p = \log_2 n$ and $0 \leq k \leq p - 1$. The set of outputs of n-GQC which can be reached from $\mathrm{QC}_{k,j}$ is represented as $\mathrm{O}_{k,j} = \{j2^{p-k-1}, \cdots, (j+1)2^{p-k-1} - 1\}$, where $0 \leq j \leq 2^{k+1} - 1$.

Figure 5.3 also illustrates how a quantum multicast assignment is realized by the 8-GQC. The input quantum packets to the network are shown in the figure. We use subscripts to show the output addresses in a quantum packet. For example, input 2 has quantum packet $\frac{\sqrt{3}}{2}B_{(1,2)} + \frac{1}{2}C_{(1,2,6)}$, where packet B having fanout set $\{1, 2\}$ is to be routed with probability $3/4$ or packet C having fanout set $\{1, 2, 6\}$ is to be routed with probability $1/4$. Inputs 1, 3, 4 and 7 do not have any packets. The corresponding quantum assignment is a superposition of four assignment patterns with coefficients $\sqrt{3}/\sqrt{8}$, $\sqrt{3}/\sqrt{8}$, $1/\sqrt{8}$ and $1/\sqrt{8}$ respectively, which are shown in the figure by grey vertical columns on the input side with coefficients on top. This quantum assignment is non-contending since all the four assignment patterns are non-contending. The figure illustrates how the four assignment patterns are realized by the 8-GQC, by showing the output quantum state at each stage of the network. On measurement, one of the four patterns shown at the output will be observed with probabilities $3/8$, $3/8$, $1/8$ and $1/8$ respectively. Therefore, packets D, A and A reach outputs 0, 3 and 5 with probability 1. Packet B is observed on outputs 1 and 2 with probability $3/4$. With probability $1/4$, packet C is observed on these two outputs. The probability of observing a packet on output 6 is $1/4$ and on output 7 is $1/2$.

We now count the number of auxiliary qubits used in the n-GQC. Every copy node requires $m_d + 1$ auxiliary qubits as a blank packet, where m_d is the number of bits in the data part of a packet. Consequently, the total number of auxiliary qubits used in the distribution stages of the network is $(m_d + 1)n \log_2 n$. Also, since an n-QC uses $\frac{n}{2} \log_2 n$ auxiliary qubits, the total number of auxiliary qubits used in the concentration stages of the n-GQC is

$$\sum_{k=0}^{p-1} 2^{k+1} \frac{n/2^k}{2} \log_2(n/2^k) = \frac{n}{2}(\log_2 n + 1) \log_2 n \quad (5.3)$$

Therefore, the n-GQC uses $O(n \log^2 n)$ auxiliary qubits.

It is always possible to restore all of the auxiliary qubits used in a switching network to their initial state once the desired switching operation has been completed, i.e., the data fields of the input packets have reached their desired destinations. This can be accomplished by the inverse of the switching network, which uses routing, address and auxiliary qubits. However, it is desirable that the auxiliary qubits are restored as early as possible so that they can be reused, and also their decoherence does not affect the switching operation [51] [52]. We adhere to this policy by restoring all of the auxiliary qubits used in the n-QC except the switching qubits. For non-contending assignment patterns, the switching qubits used in the quantum concentrators are the only auxiliary qubits which are not restored in an n-GQC. These can always be restored afterwards using the inverse of n-GQC as described earlier.

So far, we have seen that an n-GQC realizes a non-contending assignment pattern without any blocking. In the next section, we describe how blocking occurs when a contending assignment pattern is routed through the n-GQC design given in this section.

5.2 Behavior of n-GQC for Contending Assignments

A substantial amount of recent research on multicast switching has been focused on developing scheduling algorithms which aim to maximize the throughput of an input-queued multicast switch [69] [65] [70] [71]. A number of such algorithms require that the packets in a contending assignment are routed using a *fanout-splitting* policy, where a multicast packet can be sent to a subset of the outputs in its fanout set. The rest of the fanout set is realized in subsequent attempts. It is usually desired that the multicast switch has the capability to do such a fanout-splitting internally. It is also desired that the multicast switch be *work-conserving*, which means that if an output is in the fanout set of at least one of the packets in a contending assignment, then it should not happen that this output does not receive any packet. In this section, we show that, due to internal blocking in case of contending assignment patterns, the n-GQC works in fanout-splitting fashion. However, it is not work-conserving.

Consider a contending assignment pattern to an n-GQC that has m classical packets, out of which m_u are addressed to only upper $n/2$ outputs, m_l are addressed to only lower $n/2$ outputs and m_b are addressed both upper and lower $n/2$ outputs. Then, in the recursive construction of n-GQC, the upper n-QC receives m_u+m_b packets and lower n-QC receives m_l+m_b packets. Since a contending assignment can have more than $n/2$ packets addressed to the top $n/2$ outputs, $m_u + m_b$ can be more than $n/2$. In this case, $m_u + m_b - n/2$ packets are blocked or dropped since only $n/2$ outputs of the n-QC are connected to the next stage. Similarly, if $m_l + m_b > n/2$ then $m_l + m_b - n/2$ packets are dropped at the lower n-QC. It is easily seen that, for such blocking to occur at either of the n-QCs, it is necessary but not sufficient that $m > n/2$. This is because when $m_l = m_u = 0$, m_b has to be greater than $n/2$ for blocking. Also, for a contending assignment, such a blocking will certainly occur at one of the quantum concentrators in the n-GQC. Thus, we see that the n-GQC realizes a subset of the set of output addresses for every packet in a contending multicast assignment pattern. This subset can also be empty, which means that some of the packets may be blocked entirely. Therefore, the n-GQC works in a fanout-splitting fashion.

This is illustrated by an example of a 4-GQC, as shown in Figure 5.4. The contending assignment pattern has four packets A, B, C and D with the fanout sets shown in the figure. Packets C and D are blocked at the 4-QCs. The fanout set of A is fully realized, but the fanout set of B is partially realized, since a copy of B is blocked at the bottom 2-QC in the final stage.

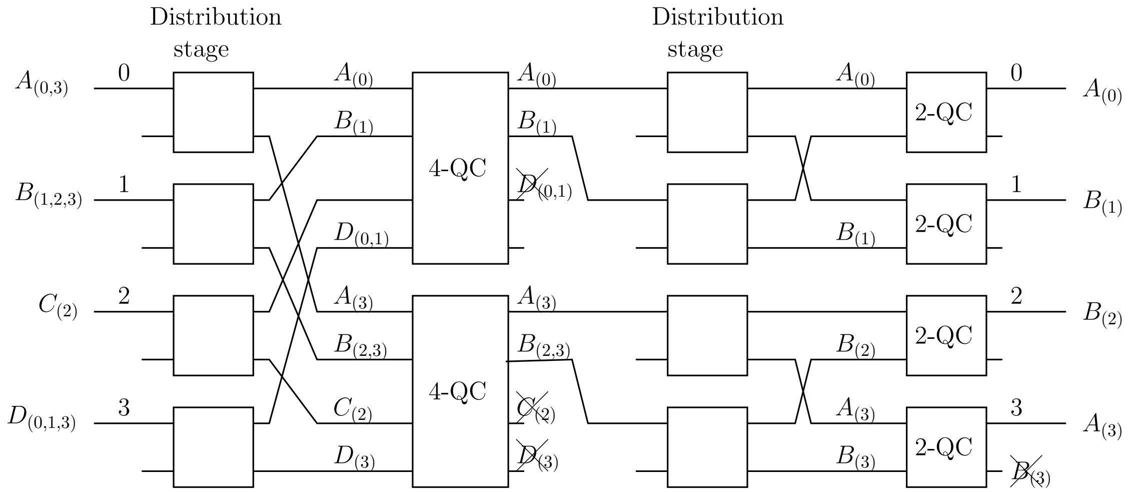


Figure 5.4: Contending assignment through a 4-GQC.

Because of blocked packets on the unused outputs of the intermediate quantum concentrators, the n-GQC is not robust against decoherence on these outputs when realizing a contending quantum assignment. Such a decoherence would collapse the quantum assignment to only those assignment patterns that contain the observed packets on these outputs. This problem does not arise when realizing a non-contending quantum assignment. This is because the unused outputs of the intermediate concentrators are always empty for every assignment pattern in the assignment, as explained earlier in Section 5.1.2. Next, we define the work-conserving property for an n-GQC.

An assignment pattern $\left|\bar{P}'\right\rangle = |(r'_0, \bar{a}'_0, \bar{d}'_0), \cdots, (r'_{n-1}, \bar{a}'_{n-1}, \bar{d}'_{n-1})\rangle$ is called a *sub-pattern* of another assignment pattern $\left|\bar{P}\right\rangle = |(r_0, \bar{a}_0, \bar{d}_0), \cdots, (r_{n-1}, \bar{a}_{n-1}, \bar{d}_{n-1})\rangle$ if, for every i, $0 \leq i \leq n-1$, $r'_i = 1$ implies that $r_i = 1$, $\bar{d}'_i = \bar{d}_i$ and $F'_i \subseteq F_i$. A non-contending sub-pattern $\left|\bar{P}'\right\rangle$ of $\left|\bar{P}\right\rangle$ is said to be *maximal* if the fanout sets $\cup_{i=0}^{n-1}\{F'_i : r'_i = 1\}$ and $\cup_{i=0}^{n-1}\{F_i : r_i = 1\}$ are equal.

An n-GQC is called *work-conserving* if it realizes a maximal non-contending sub-pattern of every contending multicast assignment pattern. It is seen that a contending multicast assignment pattern can have several maximal non-contending sub-patterns and a work-conserving n-GQC realizes one of these sub-patterns. We have the following for the n-GQC design given in this paper:

Theorem 5.2. *The n-GQC shown in Figure 5.1 is not work-conserving.*

Proof: This is seen by considering a contending multicast assignment pattern having n classical packets, in which the fanout set of every packet on the top $n-1$ inputs is $\{1, \cdots, n-1\}$ and the fanout set of the packet on the n^{th} input is $\{0\}$. Copies of all the packets on top $n-1$ inputs and the packet on the n^{th} input are routed to the upper n-QC. The packet with fanout $\{0\}$ is concentrated to the n^{th} output of the n-QC, and is therefore dropped. Thus, no packet is routed to output 0. Which means that the n-GQC is not work-conserving.

As the example in Figure 5.4 shows, there are some contending assignment patterns which do get routed in a work-conserving fashion by the n-GQC. For a given contending assignment, this happens when no fanout loss occurs at any of the quantum concentrators in the n-GQC. In other words, at any quantum concentrator, the fanout-set union of the output packets on top half of outputs is equal to fanout-set union of input packets, where the unions are restricted to the outputs reachable by the quantum concentrator.

Also, if we were able to construct an n-QC which concentrates in such a way that it always maximizes the union of the fanout sets of its top $n/2$ output packets when receiving more than $n/2$ input packets, then the n-GQC constructed using such n-QCs will be work-conserving. Designing such an n-QC is a potential approach towards the realization a work-conserving n-GQC.

Finally, we observe that the results given in this section for contending assignments also hold for the classical version of generalized connector given in [14], i.e., this network works in a fanout-splitting manner and is not work-conserving. This is due to the fact that the classical generalized connector is functionally similar to the n-GQC and realizes a classical non-contending assignment in the same fashion as n-GQC without any blocking. In the next section, we modify the n-GQC such that it creates a superposition of contending packets on their desired outputs when realizing contending assignment pattern.

5.3 Complexity Analysis

In this section, we compute the complexities of the n-GQC in terms of the total number of quantum gates and the gate level depth. Representing these complexities for the n-QC as $C_{gqc}(n)$ and $D_{gqc}(n)$ respectively, we have

$$C_{gqc}(n) = 2C_{gqc}(n/2) + 2C_{qc}(n) + C_{dist}(n) \tag{5.4}$$

$$D_{gqc}(n) = D_{gqc}(n/2) + D_{qc}(n) + D_{dist}(n) \tag{5.5}$$

where $C_{dist}(n)$ and $D_{dist}(n)$ are the corresponding costs for the distribution stage, and $C_{qc}(n)$, $D_{qc}(n)$ are the corresponding complexity for the n-QC . From Chapter 4, we have $C_{qc}(n) = O(n \log n)$ and $D_{qc}(n) = O(\log^2 n)$. It is easy to verify that $C_{dist}(n) = O(n)$ because each copy node uses a constant number of gates, and $D_{dist}(n) = O(1)$. Thus, we have $C_{gqc}(n) = O(n \log^2 n)$ and $D_{gqc}(n) = O(\log^3 n)$.

Chapter 6: Priority and Superposing Generalized Quantum Connectors

In this chapter, we give variations of the n-GQC to improve its behavior when routing contending assignments. First, we introduce prioritized routing in this network. The input assignment patterns consist of packets belonging to two classes. For contending assignment patterns, the network gives priority to packets in one class over the ones in the other class such that these packets are more likely to reach their desired outputs. In the second variation, we use quantum superposition to ensure that all packets in a contending assignment pattern have non-zero probabilities of reaching all of the outputs in their fanout sets.

6.1 Priority Generalized Quantum Connector

We include another qubit, called *priority qubit* in the quantum packet, and represent the classical component of a quantum packet as $|r, t, \bar{a}, \bar{p}\rangle$, where t denotes the *priority bit*. A classical packet $|r, t, \bar{a}, \bar{p}\rangle$ is called a *high-priority packet* if $t = 1$, and *low-priority* otherwise. We first describe a variation of n-QC which is used to introduce prioritized routing in an n-GQC. Again, we merge address and data bits and represent packets as $|r, t, \bar{p}\rangle$ when describing concentration operations.

6.1.1 Priority Quantum Concentrator

An n-QC is is called a *priority* n*-QC* (abbreviated as n-PQC), if it concentrates an assignment pattern such that all high-priority packets appear above the low-priority ones at the outputs. More specifically, for any $0 \leq m_1, m_2 \leq n$, where $m_1+m_2 \leq n$ and an assignment pattern $\left|\bar{P}\right\rangle = |(r_0, t_0, \bar{p}_0), \cdots, (r_1, t_1, \bar{p}_1)\rangle$ in which packets on inputs $i_0, \cdots, i_{m_1-1}$ have both their routing and priority bits set to 1, packets on inputs $j_0, \cdots, j_{m_2-1}$ have their routing bits set to 1 and priority bit set to 0, and packets on inputs $k_0, \cdots, k_{n-m_1-m_2-1}$ have their routing bits set to 0, is transformed as:

$$\begin{aligned} \left|\bar{P}\right\rangle |\bar{0}\rangle_{aux} \xrightarrow{n\text{-PQC}} & \left|(1,1,\bar{p}_{i'_0}) \cdots (1,1,\bar{p}_{i'_{m_1-1}}),\right. \\ & \left.(1,0,\bar{p}_{j'_0}) \cdots (1,0,\bar{p}_{j'_{m_2-1}}), (0, t_{k'_0}, \bar{p}_{k'_0}) \cdots (0, t_{k'_{m_3-1}}, \bar{p}_{k'_{m_3-1}})\right\rangle \left|\Phi(\bar{P})\right\rangle_{aux} \end{aligned} \quad (6.1)$$

where $m_3 = n - m_1 - m_2$; sequence of indices $i'_0, \cdots, i'_{m_1-1}$ is a permutation of $i_0, \cdots, i_{m_1-1}$, sequence $j'_0, \cdots, j'_{m_2-1}$ is a permutation of $j_0, \cdots, j_{m_2-1}$, and sequence $k'_0, \cdots, k'_{m_3-1}$ is a permutation of $k_0, \cdots, k_{m_3-1}$. These permutations are kept the same for all assignment patterns having the same sequences of routing and priority bits.

An n-PQC can be constructed by cascading an n-QC and an n-OPQC (order-preserving quantum concentrator) as shown in Figure 6.1. The priority qubits are used as routing qubits in the n-QC, which concentrates packets with priority bits of 1 to its top outputs. Therefore, all input packets with priority bits of 1 are above

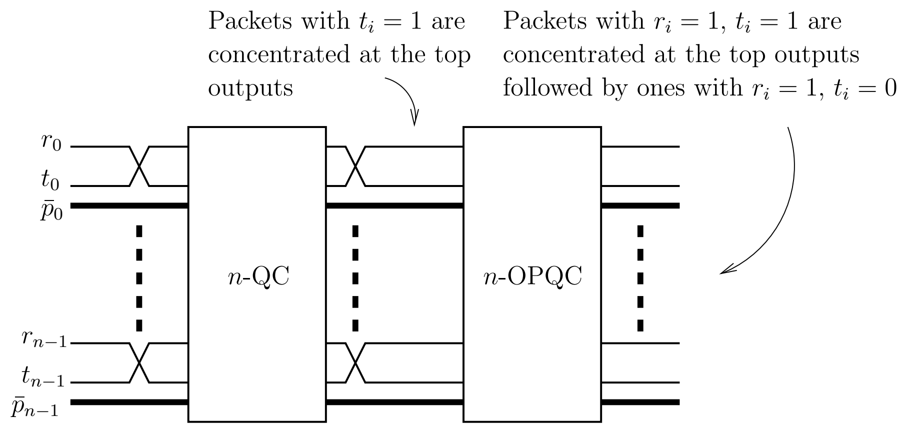

Packets with $t_i = 1$ are concentrated at the top outputs
Packets with $r_i = 1$, $t_i = 1$ are concentrated at the top outputs followed by ones with $r_i = 1$, $t_i = 0$
r_0
t_0
$\bar{p}_0$
r_{n-1}
t_{n-1}
$\bar{p}_{n-1}$
n-QC
n-OPQC

Figure 6.1: Constructing an n-PQC using an n-OPQC and an n-QC.

the ones having priority bits of 0 at output of n-QC, irrespective of their actual routing bits. The n-OPQC concentrates the resulting pattern using the routing qubits. Since it concentrates the packets with routing bits of 1 to the top outputs while preserving their input order, all the high-priority packets are concentrated at the top outputs followed by the low-priority ones, which are then followed by the packets with routing bits of 0.

The gate level depth of the above n-PQC is almost twice as that of the n-QC. We can construct an n-PQC having almost the same gate-level depth as that of the n-QC by modifying the odd-even splitter used in the n-QC such that it splits the input packets as follows:

- If m_1 is even: it routes half of the high-priority packets to the even outputs and the other half to the odd outputs. It routes $\lceil m_2/2 \rceil$ of the low priority packets to the even outputs and $\lfloor m_2/2 \rfloor$ to the odd outputs.
- If m_1 is odd: it routes $\lceil m_1/2 \rceil$ high-priority packets to the even outputs and

the remaining $\lfloor m_1/2 \rfloor$ to the odd outputs. In this case, $\lfloor m_2/2 \rfloor$ of the low priority packets are routed to the even outputs and the remaining $\lceil m_2/2 \rceil$ are routed to the odd outputs.

In order to design the above splitter, we divide the splitter switches of an odd-even splitter into four classes depending on the priorities of their input packets. A splitter switch is said to be in:

- Class-1, if it has only one input packet and it is a high-priority one.
- Class-2, if it has only one input packet and it is a low-priority one.
- Class-3, if it has one low priority and one high-priority packet at its inputs.
- Class-4, if it has either no input packets, or two input packets having same priorities.

These classes account for all combinations of input packet types. The settings of Class-4 switches do not affect output priority order, and only Class-1, 2 and 3 switches are considered. Splitter switch SW_i uses three control qubits c_{i1}, c_{i2} and c_{i3} initialized to state $|0\rangle$, where qubit c_{ij} is used to determine switch setting if SW_i is a Class-j switch. Using controlled-not gates, we transform qubit c_{ij} to 1 if SW_i is a Class-j splitter switch. The other two control qubits are left unchanged. Bit-strings $c_{01} \cdots c_{(n/2-1,1)}$, $c_{02} \cdots c_{(n/2-1,2)}$ and $c_{03} \cdots c_{(n/2-1,3)}$ are balanced separately using three $n/2$-QBs. The outputs of Class-2 and Class-3 balancers are complemented using a controlled-not gate if there are odd number of ones in $c_{01} \cdots c_{(n/2-1,1)}$, otherwise they are left unchanged. We now set the splitter switch SW_i as follows depending on its class:

- *Class*-1: If c_{i1} is 1, the input packet is sent to the top output. Otherwise, it is sent to the bottom output.
- *Class*-2: If c_{i2} is 1, the input packet is sent to the top output. Otherwise, it is sent to the bottom output.
- *Class*-3: If c_{i3} is 1, the high-priority input packet is sent to the top output and the low-priority one is sent to the bottom output. Otherwise, high-priority input packet is sent to the bottom output and the low-priority one is sent to the top output.

We now show that the resulting stage is an n-priority odd-even splitter. Assume that $m_1 = m_{11} + m_{13} + m_{14}$ and $m_2 = m_{22} + m_{23} + m_{24}$, where m_{11}, m_{13} and m_{14} are the numbers of high priority packets on Class-1, 3 and 4 switches respectively, m_{22}, m_{23} and m_{24} are the numbers of low priority packets on Class-2, 3 and 4 switches respectively. Obviously, we have $m_{13} = m_{23}$. Also, m_{14} and m_{24} are always even. We now have the following cases:

- m_{11} even: In this case, the controlled-not gate does not complement the outputs of Class-2 and 3 n-QBs. Hence, if:
  - m_1 even: m_{13} and m_{23} are even and $m_{11}/2+m_{13}/2+m_{14}/2 = m_1/2$ high

priority packets are sent to the even outputs. Also, $\lceil m_{22}/2\rceil + m_{23}/2 + m_{24}/2 = \lceil m_2/2\rceil$ low priority packets are sent to the even outputs.

  - m_1 odd: m_{13} and m_{23} are odd and $m_{11}/2 + \lceil m_{13}/2\rceil + m_{14}/2 = \lceil m_1/2\rceil$ high priority packets are sent to the even outputs. $\lceil m_{22}/2\rceil + \lfloor m_{23}/2\rfloor + m_{24}/2 = \lfloor m_2/2\rfloor$ low priority packets are sent to the even outputs.

- m_{11} odd: In this case, the controlled-not gate complements the outputs of Class-2 and 3 n-QBs. Hence, if:

  - m_1 even: m_{13} and m_{23} are odd and $\lceil m_{11}/2\rceil + \lfloor m_{13}/2\rfloor + m_{14}/2 = m_1/2$ high priority packets are sent to the even outputs. $\lfloor m_{22}/2\rfloor + \lceil m_{23}/2\rceil + m_{24}/2 = \lceil m_2/2\rceil$ low priority packets are sent to the even outputs.

  - m_1 odd: m_{13} and m_{23} are even and $\lceil m_{11}/2\rceil + m_{13}/2 + m_{14}/2 = \lceil m_1/2\rceil$ high priority packets are sent to the even outputs. $\lfloor m_{22}/2\rfloor + m_{23}/2 + m_{24}/2 = \lceil m_2/2\rceil$ low priority packets are sent to the even outputs.

This completes the proof that the odd-even splitter using the above n-QBs and the new splitter switches is an n-priority odd-even splitter. The quantum circuit for splitter switch is shown in Figure 6.2(c). We can restore all of the control qubits to their initial states by using the inverses of quantum balancers, as described earlier in Section 4.2.2. Only the switching qubits s_i, where $i = 0, \cdots, n/2 - 1$, are not restored. The n-PQC is constructed in the same way as n-QC, by using the n-priority odd-even splitter and the recursive construction shown in Figure 6.2(d).

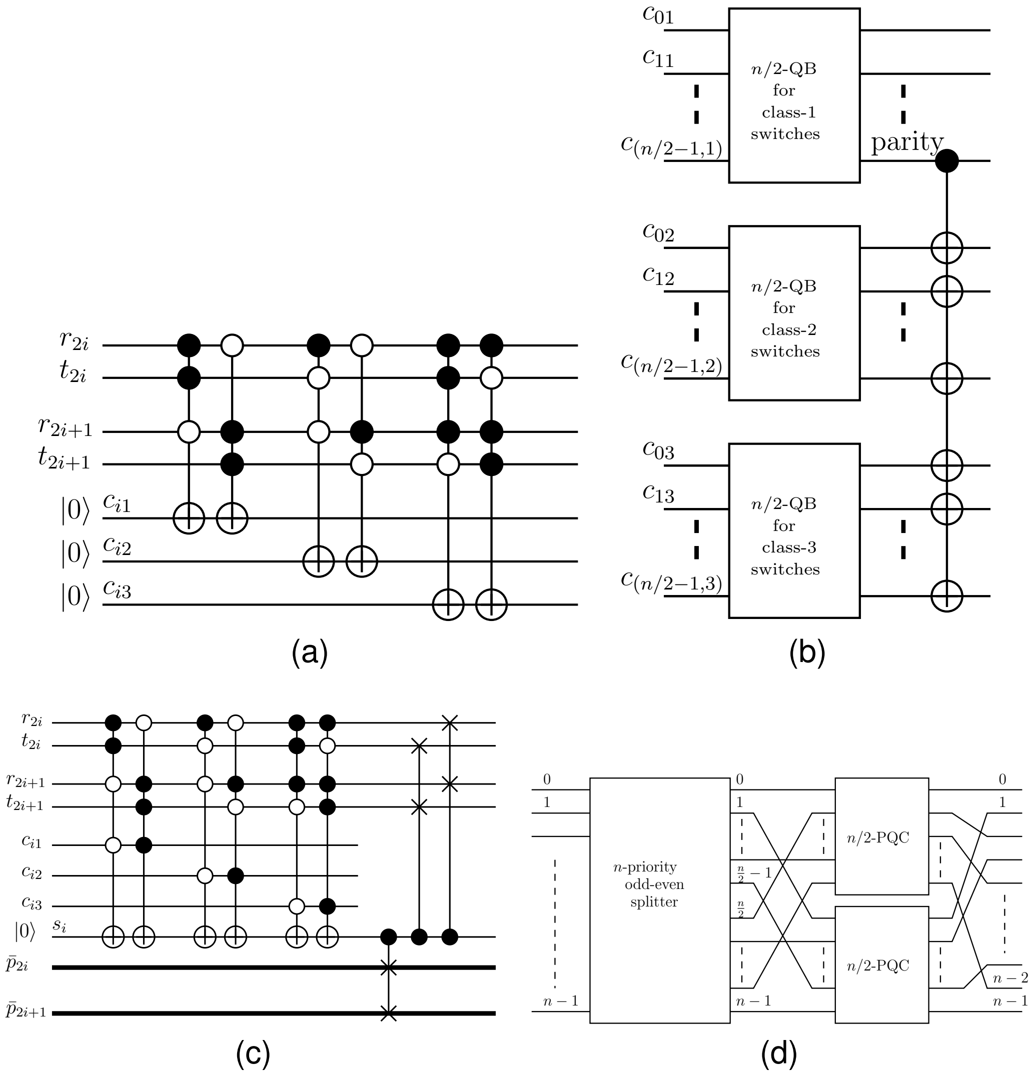


Figure 6.2: Various components of n-PQC and its recursive realization: (a) Generating control qubits; (b) Balancing the control qubits; (c) Splitter switch for n-priority odd-even splitter; (d) Recursive construction of n-PQC.

6.1.2 Priority GQC Construction

A priority n-GQC is obtained by replacing n-QCs in the n-GQC by n-PQCs. We denote this network as n-PGQC. Since this network remains an n-GQC, it realizes non-contending assignment patterns without any blocking. For contending assignment patterns, high-priority packets are generally given precedence over low-priority ones when blocking occurs at the intermediate concentrators. This is described in the results that follow.

Theorem 6.1. *For a contending assignment pattern having both low and high priority packets, the n-PGQC routes all of the high-priority ones to their desired outputs without any blocking if and only if the sub-pattern consisting of high priority packets is non-contending.*

Proof: Since high-priority packets are always concentrated to the top outputs of n-PQCs, and are above the low-priority ones, we can treat them as the only ones with routing bits of 1, and the n-PGQC works as n-GQC for the sub-pattern of high-priority packets. This sub-pattern is realized without any blocking if it is non-contending. The converse is trivial, because if all high priority packets are routed to all of their destinations without any blocking then the sub-pattern of these packets must be non-contending. $\|$

When the sub-pattern of high priority packets is also contending, it may happen that some high priority packets are blocked, and low priority ones go through. However, this is possible for only those low-priority packets that are not contending for an output with a high-priority packet:

Theorem 6.2. *For a contending input assignment pattern to an n-PGQC in which the sub-pattern of high-priority packets is also contending, an output to which both high and low priority are addressed, receives either a high-priority packet or none.*

Proof: It is observed that in an n-GQC, any two inputs will have to be connected through to same sets of concentrators to reach a given output. Consequently, if both a low and a high-priority packets are addressed to a given output and the low-priority one reaches that output, then the high-priority packet must have been below the low-priority one on the outputs of one of the concentrators. This cannot happen because all the concentrators are n-PQCs. Hence either the high-priority packet is routed to that output, or both packets are blocked. $\|$

These properties of the n-PGQC are illustrated by examples given in Figure 6.3. Figure 6.3(a) shows a contending assignment pattern in which the sub-pattern of high-priority packets is non-contending. All high priority packets are routed to their outputs, and only the low-priority ones (D_0, B_3 and D_2) are blocked. A low-priority packet, B_2, is also routed to its output. Figure 6.3(b) illustrates a contending assignment pattern in which the sub-pattern of high-priority packets is also contending. In this case, some high priority packets (D_1^* and C_0^*) are also blocked, and a low-priority packet B_2 reaches its output. This happens since no high-priority packet is addressed for output 2. This example also shows that the n-PGQC is not work-conserving in general, since no packet reaches output 1 even when there is a packet (D_1^*) addressed to it.

6.2 Superposing Generalized Quantum Connector

An *n-superposing generalized quantum connector*, or n-SGQC, is a quantum multicast network which creates superpositions of contending packets on their

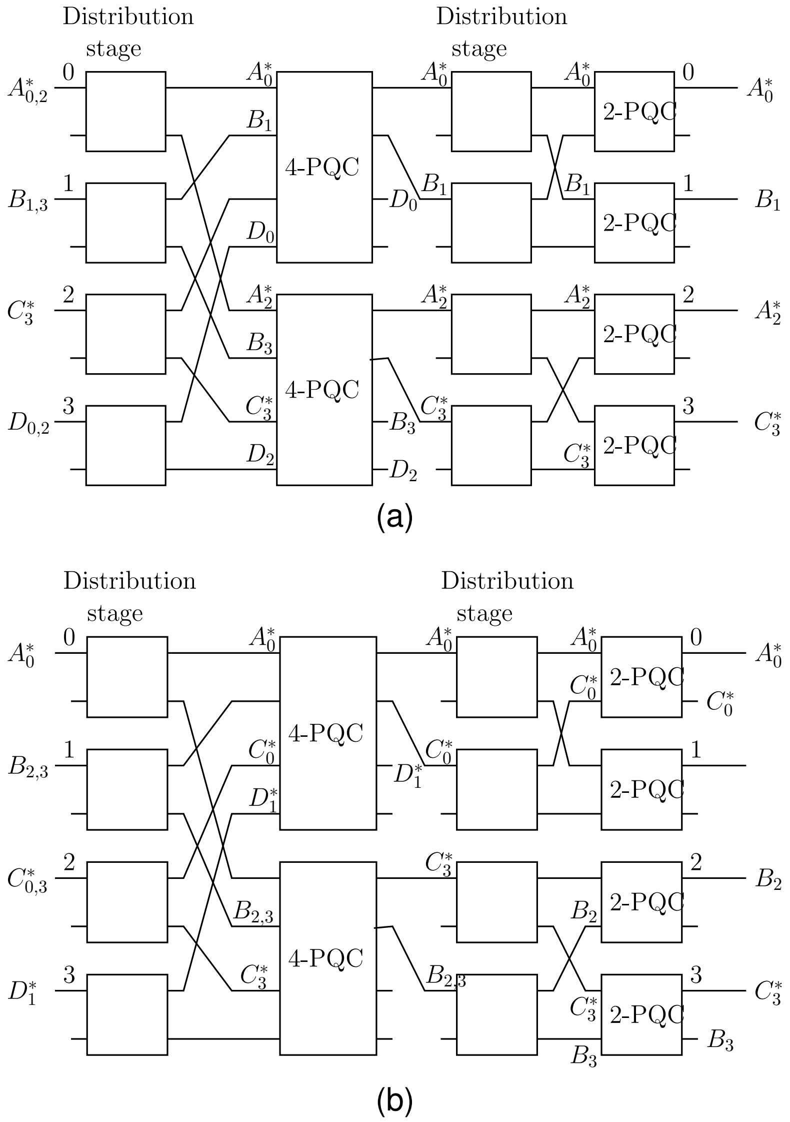


Figure 6.3: 4-PGQC examples: (a) All high-priority packets (denoted with *) routed without blocking; (b) 4-PGQC example when sub-pattern of high-priority packets is also contending.

desired outputs when routing contending assignment patterns. It realizes non-contending patterns like an n-GQC. However, unlike n-GQC, it transforms a contending assignment pattern to a superposition of output patterns such that every input packet has non-zero probabilities of being observed at all outputs in its fanout set. In order to construct our n-SGQC, we modify the n-QC as described next.

6.2.1 Superposing Quantum Concentrators

Definition 6.1 (**Superposing quantum concentrator (SQC)).** An $n \times n$-quantum switch is called a (n, m)-*superposing quantum concentrator* (where $m \leq n$), or (n, m)-SQC, if it transforms an input assignment pattern having k packets with routing bits of 1, where $1 \leq k \leq n$, to a quantum superposition of a set of output patterns such that each output patterns in this set has all k packets concentrated on top k outputs, and if $k > m$ then for each input packet with routing bit of 1, there is at least one output pattern in which this packet appears on one of the top m outputs.

The sum of probability coefficients of all output patterns that have a given input packet among the top m outputs, is called the *concentration probability* of that input packet on an (n, m)-SQC. By definition, every input packet to an (n, m)-SQC has a non-zero concentration probability. For an assignment pattern with $k \leq m$, every input packet has a concentration probability of 1. This is because all these packets are included in the top m outputs in every output pattern. For such a assignment pattern, the (n, m)-SQC does not need to generate a superposition of output patterns, and only one is sufficient. Still, having a superposition is okay because the concentration probabilities of packets are not affected. Since each output pattern can have at most m concentrated input packets, an (n, m)-SQC must have at least $\lceil k/m \rceil$ output patterns for an input assignment pattern with $k > m$.

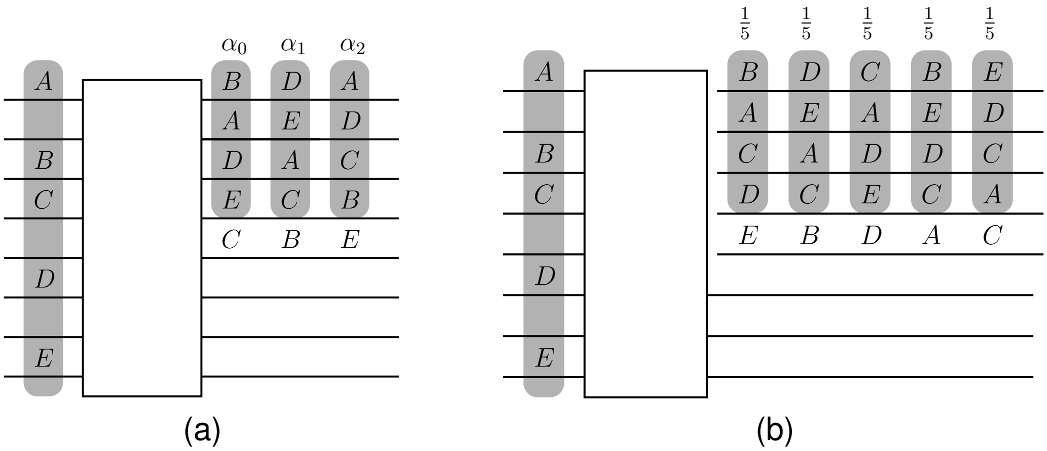


Figure 6.4: Superposing quantum concentrators: (a) $(8,4)$-SQC; (b) $(8,4)$-USQC.

We did not place any restriction on the probability coefficients that are assigned to the output patterns, due to which, different input packets may have different concentration probabilities, when $k > m$. We next define a stronger version of (n,m)-SQC in which every input packet has the same concentration probability.

Definition 6.2 (Uniform superposing quantum concentrator (USQC)). An (n,m)-SQC is said to be *uniform* if, for an input assignment pattern having k packets with routing bits of 1, every packet has concentration probability of m/k when $k > m$, and 1 otherwise. This concentrator is referred as (n,m)-USQC.

On a computational basis measurement at the output of the (n,m)-USQC, any packet in the input assignment pattern is observed among the m outputs with probability m/k and among the bottom $n-m$ outputs with probability $1-m/k$. One way to realize an (n,m)-USQC is to generate an equal superposition of $\binom{k}{m}$ output patterns that have all the size m subsets of the set of input packets at the top m outputs, when $k > m$. Both (n,m)-SQC and USQC concentrators are il-

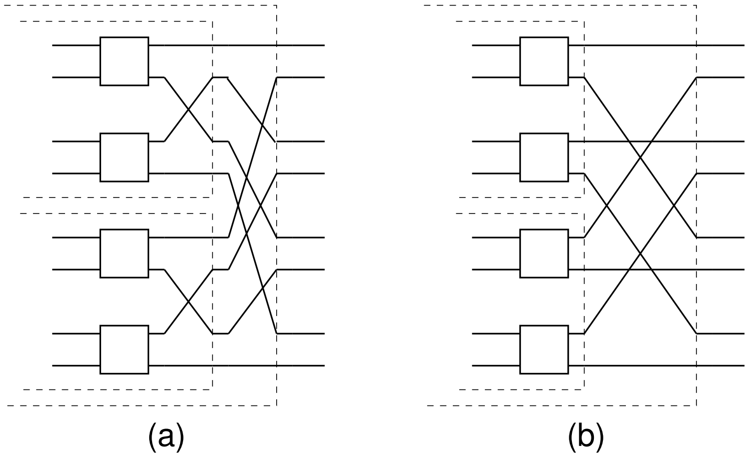


Figure 6.5: Final stage of the 8-QC: (a) Output shuffles for $n = 2, 4$ and 8; (b) Output shuffles combined.

lustrated by the examples given in Figure 6.4 for $n = 8$, $m = 4$, and an input assignment pattern having $k = 5$ packets with routing bits of 1. The probability coefficients of the output patterns are shown over them. Figure 6.4(a) shows a network that is a $(8, 4)$-SQC, but not a $(8, 4)$-USQC. The concentration probabilities of packets A and B are $\alpha_1^2 + \alpha_2^2 + \alpha_3^2 = 1$ and $\alpha_0^2 + \alpha_2^2$ respectively, which cannot be equal. Figure 6.4(b) shows a $(8, 4)$-USQC which concentrates every packet in this input pattern among its top 4 outputs with probability $4/5$. The output has 5 patterns with coefficients of $1/\sqrt{5}$, and each packet appears among the top 4 outputs in four of the output patterns.

6.2.2 Construction of $(n, n/2)$-SQC

In this section, we modify the n-QC given in Section 4.2.3 to construct an $(n, n/2)$-SQC. First, we observe that the upper outputs of the splitter switches in the last stage of the n-QC are connected the top $n/2$ outputs consecutively, i.e., the upper output of i^{th} splitter switch SW_i^{p-1} is connected to the i^{th} output, where

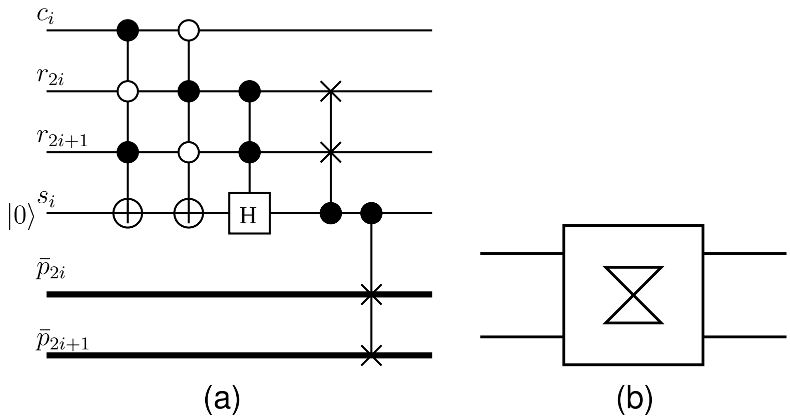


Figure 6.6: Modifying n-QC to obtain $(n, n/2)$-SQC: (a) splitter switch circuit; (b) its block representation.

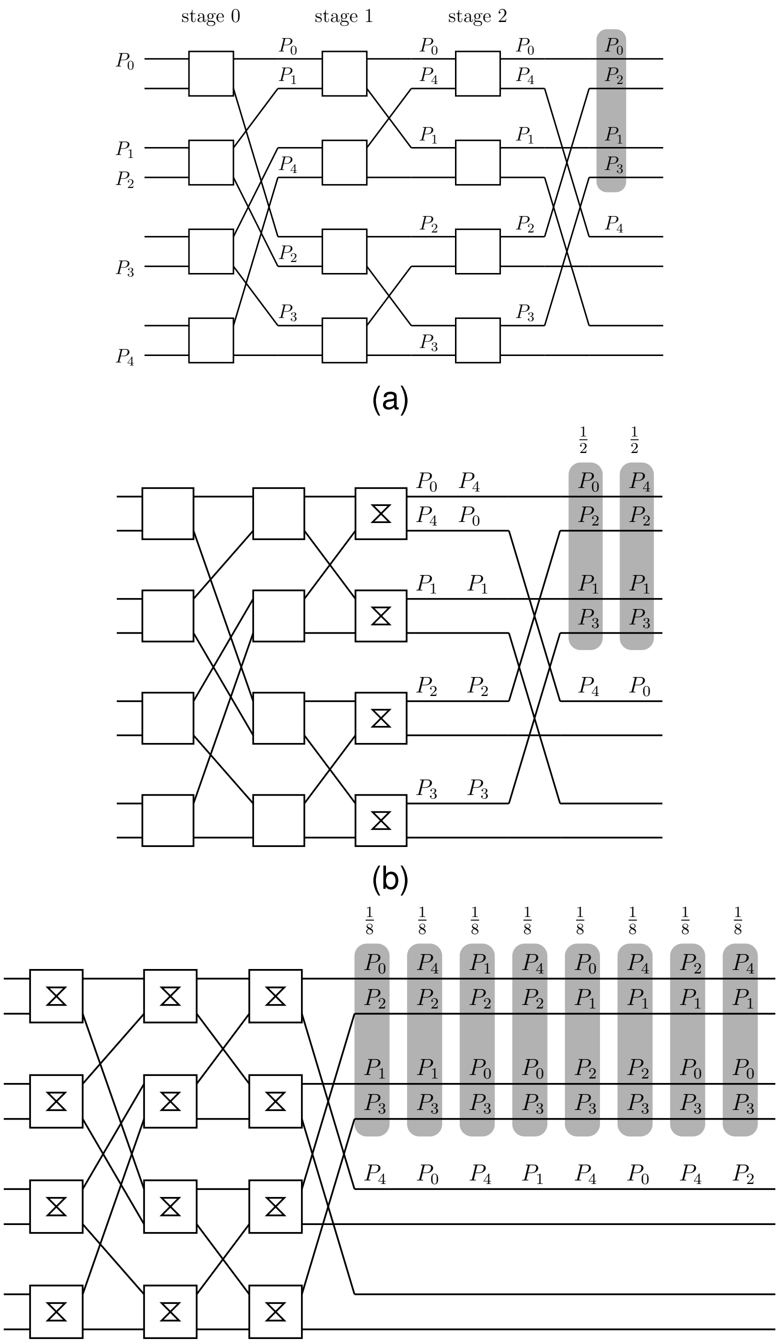
$0 \leq i \leq \frac{n}{2} - 1$. This is shown in Figure 6.5 for $n = 8$.

Consider an input assignment pattern to the n-QC having k classical packets with routing bits of 1, where $1 \leq k \leq n-1$. Since the n-QC concentrates all packets to its top k outputs, if $k \leq n/2$, all these packets appear on the upper outputs of the top k splitter switches in the final stage. If $k > n/2$, the upper outputs of all $n/2$ splitter switches in the final stage have packets, and remaining $k - n/2$ packets appear on the lower outputs of the top $k - n/2$ splitter switches in this stage. These $k - n/2$ packets are the ones that do not reach the upper $n/2$ outputs, due to which the n-QC is not an $(n, n/2)$-SQC.

Since the setting of a balanced splitter switch does not affect the concentration operation of the n-QC, we can set these top $k - n/2$ switches in a superposition of through and cross states, and convert the n-QC to an $(n, n/2)$-SQC, as explained next. The quantum circuit for the modified splitter switch is shown in Figure 6.6(a). We add a controlled-Hadamard gate to the n-QC splitter switch circuit shown earlier in Figure 4.3(b). This gate operates only when routing bits of both input packets are 1, and it sets the control qubit driving the swap gates in state $1/\sqrt{2}(|0\rangle + |1\rangle)$, which creates a superposition of input packets. It does not affect the splitter switch operation in any other case. We next show that by using the modified splitter switches in the final stage of n-QC, we get an $(n, n/2)$-SQC.

Consider the 8-QC shown in Figure 6.7(a). In order to simplify the figure, we have not shown the quantum balancers, and only the splitter switches and the interconnections are shown. The input assignment pattern has 5 packets with routing bits of 1. The figure shows the intermediate pattern generated at every stage as this input assignment pattern is routed through the network. In the output pattern, packet P_4 appears on the 5^{th} output and has zero probability of appearing among the top 4 outputs, due to which this network is not a $(8, 4)$-SQC. If we replace all of the splitter switches in the last stage of the network with the modified splitter switch, the topmost switch in this example works in a superposition. Consequently, we get two output patterns, each with a probability coefficient of $1/2$. Now, P_4 is also concentrated among the top 4 outputs, with probability $1/2$. Packets P_1, P_2 and P_3 have concentration probabilities of 1. Packets P_0 and P_4 end up sharing the remaining output with concentration probabilities of $1/2$.

In general, for an assignment pattern having $k > n/2$ packets, this network generates $2^{k-n/2}$ output patterns with probability coefficients of $\frac{1}{2^{k-n/2}}$ such that $n - k$ of the input packets have concentration probabilities of 1, and remaining



(c)

Figure 6.7: Superposing quantum concentrator: (a) An 8-QC concentrating an assignment pattern having 5 packets; (b) Using modified splitter switches in the last stage, we get a $(8, 4)$-SQC; (c) Using modified splitter switches in all stages we get more uniformity.

$2k - n$ have concentration probabilities of $1/2$. Since all the input have non-zero concentration probability, this network is an $(n, n/2)$-SQC.

We can make the $(n, n/2)$-SQC more uniform by replacing all the splitter switches in the n-QC with the modified splitter switch. This is illustrated in Figure 6.8(c). The second switch in the first stage works in a superposition, resulting in two patterns at the output of the first stage. For both of these patterns, the top switch in the second stage works in a superposition, resulting in 4 patterns at the output of the second stage. For each of these patterns, the top switch in the third stage works in a superposition. Consequently, we have 8 output patterns, each having a probability coefficient of $1/8$ and a different pattern of 5 packets at the top 5 outputs. We observe that only P_3 has a concentration probability of 1. The concentration probabilities of P_0, P_1, P_2 and P_4 are $3/4$, 7/8, $7/8$ and $1/2$ respectively. This is because these packets respectively appear among the top 4 outputs of the network in 6, 7, 7 and 4 of the output patterns.

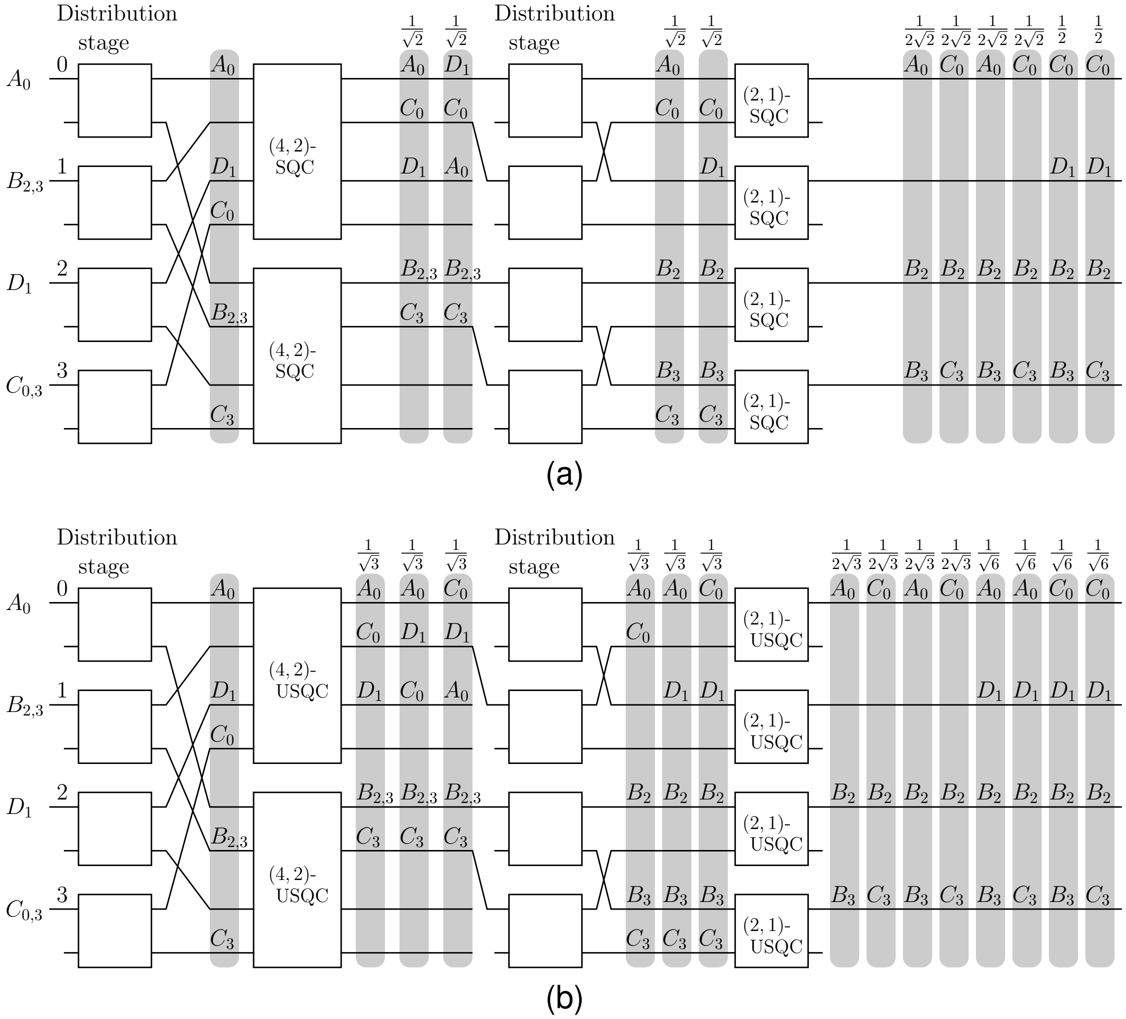
In general, for an input assignment pattern having $k > n/2$ packets with routing bits of 1, at least $k - n/2$ splitter switches in each stage of the network are balanced, and therefore, there are at least $2^{(k-n/2)\log_2 n}$ output concentration patterns, as compared to $2^{k-n/2}$ for the network in which the modified splitter switches are employed only in the last stage. This leads to more uniformity in general. A detailed analysis of the concentration properties of this network, and construction of an uniform superposing quantum concentrator is left as future work.

6.2.3 Superposing GQC Construction

As described earlier, the n-GQC blocks packets when routing a contending assignment pattern, and these packets are dropped at the bottom halves of outputs at the internal quantum concentrators. Also, due to this blocking, some outputs may not receive any packet even when there are packets addressed to them. We propose n-GQC variations that use the principle of quantum superposition to create superpositions of all contending packets on their desired output, such that every input packet has non-zero probabilities of reaching all of the outputs in its fanout set.

Definition 6.3. An n-GQC is said to be a *superposing*, denoted as n-SGQC, if it transforms any contending assignment pattern to a superposition of output patterns that are n-GQC realizations of distinct non-contending sub-patterns of the input pattern, such that for every output which has one or more packets (with routing bit of 1) addressed to, there is at least one output pattern realizing each of these packets.

If output i is in the fanout sets of k_i input packets, where $i = 0, \cdots, n-1$ and $0 \leq k_i \leq n$, then an n-SGQC has to generate at least $max(k_i)$ output patterns. The probability of observing a packet on an output in its fanout set is given by the sum of the probability coefficients all output patterns that contain the packet on that output. This probability is non-zero for every input packet and output pair. The probability of observing a packet on an output is sum of probability



(a)

(b)

Figure 6.8: Contending assignment pattern on a 4-SGQC (dropped packets at the output stage are not shown): (a) using $(n, n/2)$-SQC of Figure 6.7(b); (b) using $(n, n/2)$-USQCs.

coefficients of all output patterns that have a packet with routing bit of 1 on that output. The input assignment pattern is said to be realized in a work-conserving fashion if this probability is 1 for every output which is in the fanout set of at least one packet. An n-SGQC is said to be work-conserving, if it is work-conserving for all contending assignment patterns.

We construct an n-SGQC by replacing the n-QCs in the n-GQC by $(n, n/2)$-

SQCs. Since every input packet has a non-zero probability of making into the top halves of outputs of the SQCs in its path, all input packets in a contending assignment pattern have non-zero probabilities of reaching all of the outputs in their fanout sets. We illustrate this in Figure 6.8 which shows how a contending assignment pattern of size $n = 4$ is realized in the resulting network. The figure does not show how this pattern is routed by the 4-GQC. It can be verified that the 4-GQC blocks packets D_1 (at its upper 4-QC), C_0 (at top 2-QC) and C_3 (at bottom 2-QC). Packets A_0, B_2 and B_3 reach their desired outputs, and no packet is routed to output 1.

Figure 6.8(a) shows an 4-SGQC which is constructed using the $(n, n/2)$-SQC of Figure 6.4(b) (with modified splitter switches only in the last stage). We get two patterns with coefficients of $1/\sqrt{2}$ at the output of the first concentrator stage because the top splitter switch (not shown in the figure) in the final stage of the upper $(4, 2)$-SQC is balanced and works in a superposition. The first pattern is broken into four at the second concentration stage, because the topmost and bottommost $(2, 1)$-SQC work in a superposition – sharing output 0 between A_0 and C_0, and output 3 between B_2 and B_3. The second pattern is broken into two as only the bottommost $(2, 1)$-SQC works in a superposition – routing C_0, B_1 and B_2 to outputs 0, 1, 2, and sharing output 3 between B_3 and C_3. Probabilities of observing A_0 and C_0 on output 0 are $1/4$ and $3/4$. Packet B_2 reaches output 2 with probability 1. Packets B_3 and C_3 have probabilities $1/2$ each of being observed at output 3. Output 1, which does not receive any packet on the 4-GQC, receives

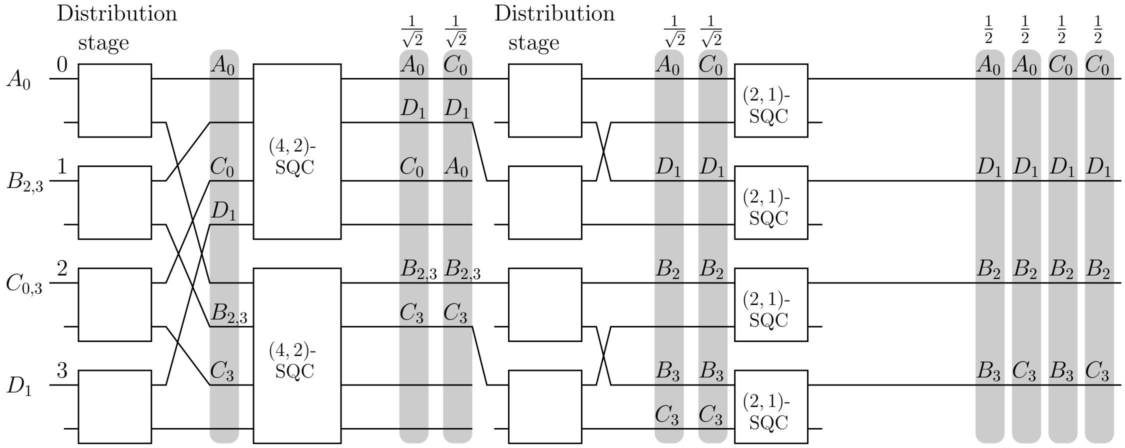


Figure 6.9: 4-SGQC example showing a work-conserving realization.

packet D_1 with probability $1/2$. Every input packet has non-zero probability of reaching its desired output, and overall throughput is increased because outputs 0, 2, and 3 still receive packets with probabilities 1, and output 1 also receives a packet with probability $1/2$, as opposed to none in the 4-GQC.

Figure 6.8(b) shows the same assignment pattern being routed on an n-SGQC which uses $(n, n/2)$-USQC of Figure 6.7(b) which creates an equal superposition of all $n/2$ size subsets of input packets on its top $n/2$ outputs when receiving more than $n/2$ packets. On this n-SGQC, we get more uniform sharing of contending packets on their desired outputs, as packets A_0 and C_0 now share output 0 equally with probabilities of $1/2$. Also, the probability of observing packet D_1 on output 1 increases from $1/2$ to $2/3$. Consequently, the throughput is increased even further for this assignment pattern.

The above examples cannot be generalized about either uniformity or work-conserving property for all contending-assignment patterns. The n-SGQC may have less throughput than the n-GQC for some contending assignment patterns which are realized by n-GQC in a work-conserving or close to work-conserving manner. Also, for some contending assignment patterns, an $(n, n/2)$-USQC based n-SGQC may have less uniform sharing of outputs and less throughput than the $(n, n/2)$-SQC based one. Still, it is expected that for most of the assignment patterns, we get more uniform sharing of outputs in an $(n, n/2)$-USQC based n-SGQC.

The example given in Figure 6.9 illustrates these ideas. The input assignment pattern is similar to the earlier examples, the only difference is that the input locations of packets D_1 and $C_{0,3}$ are interchanged. It can be verified that an 4-GQC realizes this assignment pattern in a work-conserving fashion, and blocks packets C_0 and C_3. By using the $(n, n/2)$-SQC (with modified splitter switches only in the last stage), we retain work-conserving property, and also get uniform sharing of outputs between contending packets on all outputs. In this case, it happens that there is no fanout loss in any of the output patterns at the internal quantum concentrators, and packet D_1, which is the only packet addressed for output 1, has a concentration probability of 1 on all quantum concentrators on its route. It can be verified that the USQC-based 4-SGQC gives same output as in Figure 6.8(b) for this assignment pattern as well, and output 1 has only a probability of $2/3$ of getting packet D_1. This happens because of the fanout loss in the first output pattern at the $(4, 2)$-USQC stage. On the upper USQC, union of input packet fanouts is $\{0, 1\}$, but the union of fanout sets of packets on the upper half of outputs in the first output pattern is $\{0\}$. Due to this fanout loss, the USQC based 4-SGQC does not realize the input assignment pattern in a work-conserving fashion.

A stronger version of n-SGQC that is work-conserving and also creates an uniform superposition of contending packets on their desired outputs, is defined next. Construction of this network is left as future work.

Definition 6.4. An n-SGQC is said to be *uniform* if, for any contending assignment pattern in which k_i input packets with routing bits of 1 have output i in their fanout sets, $i = 0, \cdots, n-1$, all these packets have the same probability $1/k_i$ of being observed at output i when $k_i \neq 0$, and a packet is observed at this output with with probability 1.

In order to construct an uniform n-SGQC using the n-GQC network topology given in this thesis, we would need to construct superposing quantum concentrators that keep track of fanout sets of the multicast packets, and ensure no fanout loss in any of the output patterns. Also, the concentration probabilities of packets have to be proportional to the sizes of their fanout sets so that a packet which is addressed to just one output does not have an unfair advantage over one which is addressed to multiple outputs.

Chapter 7: Conclusion and Future Work

In this chapter, we briefly describe how the quantum interconnection networks we have designed can be used to switch and multicast general quantum states. We then give our concluding remarks, and list some future research directions.

7.1 Switching General Quantum States

In this thesis, we have primarily focused on unicasting and multicasting of classical data through quantum switching networks. In this section, we describe how the networks we have designed can be used to unicast and multicast general quantum states as well.

7.1.1 Unicasting and Concentration of Quantum States

In both n-QC and quantum Baseline networks, the switch settings are determined using the routing and address qubits only, and are not affected by the data parts of the packet. Consequently, these networks route two or more assignment patterns having the same routing and address bits by applying the same permutation. Also, the final states of the auxiliary qubits are also dependent on the routing and address bits only. Consider a 4×4 unicast quantum switching network which permutes an unicast assignment pattern $|\psi\rangle = |(r_0, a_0, p_0)\ (r_1, a_1, p_1)\ (r_2, a_2, p_2)\ (r_3, a_3, p_3)\rangle$ with fixed routing and address bits as:

$$|\psi\rangle \longrightarrow \Pi_{r_i,a_i}(|\psi\rangle)\,|\phi_{aux}(r_i, a_i)\rangle = |(r_2, p_2)(r_1, p_1)(r_3, p_3)(r_0, p_0)\rangle\,|\phi_{aux}(r_i, a_i)\rangle \quad (7.1)$$

where Π_{r_i,a_i} denotes the permutation applied, and $|\phi_{aux}(r_i, a_i)\rangle$ is the output state of the auxiliary qubits when routing and address bits on input i are r_i and a_i, $i = 0, \cdots, 3$. Without loss of generality, we assume that the address qubits become a part of the auxiliary qubit outputs.

Suppose inputs 0, 1, 2 and 3 have quantum states $\alpha\,|A\rangle + \beta\,|B\rangle$, $|C\rangle$, $|D\rangle$ and $\gamma\,|E\rangle + \delta\,|F\rangle$ respectively, and they are to be routed using the same routing and address bits as above. We set the routing qubit on input i in state $|r_i\rangle$ and address qubit in state $|a_i\rangle$. The input quantum assignment is a superposition of four assignment patterns, and is expressed as:

$$\begin{aligned}&\alpha\gamma\,|(r_0, a_0, A)(r_1, a_1, C)(r_2, a_2, D)(r_3, a_3, E)\rangle\\ &\qquad + \alpha\delta\,|(r_0, a_0, A)(r_1, a_1, C)(r_2, a_2, D)(r_3, a_3, F)\rangle\\ &\qquad + \beta\gamma\,|(r_0, a_0, B)(r_1, a_1, C)(r_2, a_2, D)(r_3, a_3, E)\rangle\\ &\qquad\qquad + \beta\delta\,|(r_0, a_0, B)(r_1, a_1, C)(r_2, a_2, D)(r_3, a_3, F)\rangle \quad (7.2)\end{aligned}$$

The network transforms this quantum assignment to:

$$
\begin{aligned}
&|(r_2, D)(r_1, C)(r_3, E)(r_0, A)\rangle \, |\phi_{aux}(r_i, a_i)\rangle \\
&\qquad\qquad + |(r_2, D)(r_1, C)(r_3, F)(r_0, A)\rangle \, |\phi_{aux}(r_i, a_i)\rangle \\
&\qquad\qquad + |(r_2, D)(r_1, C)(r_3, E)(r_0, B)\rangle \, |\phi_{aux}(r_i, a_i)\rangle \\
&\qquad\qquad + |(r_2, D)(r_1, C)(r_3, F)(r_0, B)\rangle \, |\phi_{aux}(r_i, a_i)\rangle \\
= &\left(\underbrace{|r_2\rangle \, |D\rangle}_{\text{output 0}} \otimes \underbrace{|r_1\rangle \, |C\rangle}_{\text{output 1}} \otimes \underbrace{|r_3\rangle \, (\gamma \, |E\rangle + \delta \, |F\rangle)}_{\text{output 2}} \otimes \underbrace{|r_0\rangle \, (\alpha \, |A\rangle + \beta \, |B\rangle)}_{\text{output 3}} \right) |\phi_{aux}(r_i, a_i)\rangle
\end{aligned}
\tag{7.3}
$$

Consequently, the quantum states are also routed (along with their routing qubits) using permutation Π_{r_i,a_i}. If the auxiliary qubits were dependent on the data parts of the inputs and became entangled with the data parts, we would not be able to separate the auxiliary output states in the above expression.

The above example can be generalized to any $n \times n$ unicast quantum switching network which routes input assignment patterns by permuting them based on routing and address qubits only. All unicast quantum switching networks given in this thesis – the QBN, n-QCs, and n-GQC when used as a unicast network, satisfy this condition. Hence, the QBN can realize n^n permutation assignments without blocking when routing quantum states. The n-GQC can realize any non-contending unicast assignment (including all permutation assignments) when routing quantum states. The n-QC can concentrate general quantum states on any m of its inputs to the top m outputs, where the inputs that issue quantum states have routing qubits of $|1\rangle$, and $m = 1, \cdots, n$. Also, the order-preserving and priority quantum concentrators retain their behavior when concentrating general quantum states.

7.1.2 Multicasting Quantum States

Although copying of general quantum states is not possible due to the no-cloning theorem, recent research demonstrates that approximate or imperfect copying is possible [2, 15–18]. We have given a survey of such quantum copiers in Chapter 1. One of the main copiers is the universal quantum copying machine (UQCM) which copies all one qubit quantum states with same fidelity of $5/6$. We can use the UQCM in the copy nodes of the n-GQC, instead of the controlled-not gate based copiers. We expect that this will enable us to do an approximate multicasting of quantum states on the n-GQC, when routing non-contending assignments. For example, if the fanout of each input in the multicast assignment is limited to 2, and inputs issue single-qubit quantum states, then each input quantum state will be cloned just once as it is routed through the n-GQC, and will appear on its intended outputs with a fidelity of $5/6$. A detailed study of such multicasting is left as future work.

7.2 Concluding Remarks

We have given a model to describe routing in quantum packet switching networks, and have introduced the concepts of quantum packets, assignments and assignment patterns. We have designed a quantum unicasting network called the quantum Baseline network (QBN) which creates a superposition of permutations of non-contending or balanced sub-patterns of an input permutation assignment pattern. This is done by using the routing bits in packets contending for the same output. As a result, all packets are routed through the network without blocking. We have also given a characterization of the output permutations generated from a given input permutation and the associated probability distribution. A simple measurement destroys the output superposition state and gives only one output permutation, which is equivalent to classical routing through a baseline network with random packet drops in case of contentions. More sophisticated measurements can be done to get more information about the packets from the output state. Another advantage of the QBN over a conventional baseline network is that there is no need to take local decisions to resolve contentions.

We have given the design of a self-routing $n \times n$ multistage quantum switching network called quantum generalized connector (n-GQC) that can realize quantum multicast assignments. The quantum packets at each input consist of a number of classical multicast packets in a probabilistic quantum superposition. We have shown that quantum assignments can be expressed as superpositions of multicast assignment patterns, where each assignment pattern is a sequence of classical packets across the inputs of the network. All the assignment patterns are simultaneously realized by the n-GQC due to quantum parallelism. All the packets in a non-contending assignment pattern are routed to their desired outputs. However, the n-GQC is not work-conserving when the input assignment pattern is contending and it realizes a sub-pattern of such an assignment pattern due to internal blocking.

The main motivation behind the design of n-GQC and QBN is that these networks have inherent quantum parallelism that provides high throughput by definition, while the packets are en route to their destinations. To give an example, consider unicast assignments, which are a subset of multicast traffic. Unicast switches with nearly 100% throughput have been reported in the literature [58, 63], but such switches require complex scheduling algorithms. To avoid using such algorithms, one can potentially employ a multi-layer switch to route a fixed number of packets from the head of each input queue and an n-GQC would be a natural replacement for such a switch in the quantum domain.

The n-GQC can be used to unicast arbitrary quantum states between its n inputs and outputs, provided that there is no output contention among the input quantum states. This is due the fact that no copying would be needed at any of the distribution stages. It can also be used to do an approximate multicasting of quantum states, as described earlier in this chapter. We have also established that the n-GQC is not work-conserving when routing contending assignment patterns, and have given two variations of this network that improve its behavior when routing such assignment patterns. The priority n-GQC (n-PGQC) introduces prioritized routing in the n-GQC and it gives priority to packets in one class over ones in another, when they contend for the same outputs. The superposing n-GQC (n-SGQC) creates a superposition of all contending packets on their desired outputs when routing contending assignment patterns.

We have also introduced and designed a new class of quantum switching networks called quantum concentrators, which route an n-assignment pattern having $k < n$ packets by concentrating these k packets onto their top k outputs in an unspecified order. The n-QC is one such network that is constructed based on a classical concentrator given in [14]. This network plays a key role in the construction of n-GQC. We have also given some variations of the n-QC. The n-OPQC concentrates assignment patterns while preserving the input order of packets. The n-PQC concentrates assignment patterns consisting of two classes of packets, and concentrates the packets in one class on the topmost outputs followed by the ones in the other class. The (n, m)-SQC creates a superposition of concentrated output patterns when routing an assignment pattern having $k > m$ packets, such that every input packet has a non-zero probability of being concentrated among the top m outputs. The n-PQC and (n, m)-SQC are used to construct n-PGQC and n-SGQC networks respectively.

7.3 Future Work

The results presented in this thesis provide a good foundation for the design of future quantum switching networks but several open problems remain. Some of these are listed as follows:

- Randomization has been used in classical switching networks to minimize congestion and improve throughput [72]. Quantum switching networks offer inherent randomization, for example, the quantum Baseline network given in this thesis can fully randomize its input assignment if all 2×2 switches are set to work in a superposition of through and cross states, and creates a superposition of $n^{n/2}$ permutations of the input assignment. We can cascade this network with another quantum switching network which routes a subset of these permutations without blocking, and control the randomization in the QBN so that the probability coefficients of the permutations in this subset are maximized. This approach can help us realize nonblocking switching networks which are self-routing and optimal in terms of number of gates and gate-level depth.

- The quantum switching networks given in this thesis create a superposition of all contending packets on their desired output. A simple measurement that projects onto orthonormal bases collapses the output state so that only one of these packets is observed. More sophisticated measurement schemes (such as Bell measurement [51]) can be developed that get more out of the output states of these networks by extracting some shared information about all packets in the superposition. Also, these packet superpositions can be fed into other quantum systems which will process these packets in parallel.

- Realization of an uniform superposing generalized quantum connector (USGQC) defined in Section 6.2.3 will strengthen the quantum packet switching model proposed in this thesis. This network will obviate the need to have a scheduler which selects packets from input queues such that these packets form a non-contending multicast assignment while ensuring that the inputs are served fairly. We expect that the design of a work-conserving generalized quantum connector will be an initial step towards the realization of an USGQC. The n-GQC given in this thesis can potentially be made work-conserving by building a quantum concentrator which ensures that the union of the fanout sets of packets on its top half of outputs is always equal to the union of the fanouts of its input packets. Construction of the (n, m)-uniform superposing concentrator defined in Section 6.2.1 is also an open problem.

- As mentioned earlier in this chapter, an evaluation of the n-GQC performance when multicasting general quantum states using advanced quantum copying machines that do approximate cloning of of such states, is needed. This performance can be measured in terms of the Hilbert-Schmidt distance between the output state of the n-GQC and an ideal (but impossible) multicasting network which produces perfect copies of the input quantum states. Also, as in the case of quantum copying machines, there is going to be a trade-off between the amount of entanglement in the output copies and the above mentioned Hilbert-Schmidt distance. It would be interesting to study how this trade-off depends on the parameters of the quantum copying nodes.

- Quantum entanglement is a powerful property exhibited by multiqubit systems, due to which two qubits can be coupled such that an operation or a measurment on one instantly reflects on the other, and information can thus be encoded in the correlation between two qubits. We have not exploited this property of quantum systems in our work, and expect that it could help build a self-routing $O(n \log n)$ quantum switching network that realizes all $n!$ permutation maps. Switches can be entangled with each other such that the state of one switch affects the states of others. This way switching elements in the network can set themselves in a self-routing fashion.

Bibliography

[1] W. Wootters and W. Zurek, “A single quantum cannot be cloned,” *Nature*, vol. 299, pp. 802–803, 1982.

[2] V. Bužek and M. Hillery, “Quantum copying: Beyond the no-cloning theorem,” *Physical Review A*, vol. 54, no. 3, pp. 1844–1852, September 1996.

[3] M. K. Shukla and A. Y. Oruç, “Multicasting in quantum switching networks,” *IEEE Transactions on Computers*, to appear 2010.

[4] R. Ratan, “Random routing and concentration in quantum switching networks,” Ph.D. dissertation, University of Maryland, College Park, 2008.

[5] M. K. Shukla, R. Ratan, and A. Y. Oruç, “A quantum self-routing packet switch,” in *Proceedings of the 38th Annual Conference on Information Sciences and Systems CISS’04*, Princeton, NJ, USA, March 2004, pp. 484–489.

[6] ——, “The quantum baseline network,” in *Proceedings of the 39th Annual Conference on Information Sciences and Systems CISS’05*, Johns Hopkins University, Baltimore, MD, March 2005.

[7] C. Moore and M. Nilsson, “Parallel quantum computation and quantum codes,” *SIAM J. Comput.*, vol. 31, no. 3, pp. 799–815, 2001.

[8] I. M. Tsai and S. Y. Kuo, “Digital switching in the quantum domain,” *IEEE Transactions on Nanotechnology*, vol. 1, no. 3, pp. 154–164, 2002.

[9] S. T. Cheng and C. Y. Wang, “Quantum switching and quantum merge sorting,” *IEEE Transactions on Circuits and Systems*, vol. 53, no. 2, pp. 316–325, February 2006.

[10] C. C. Sue, “An enhanced universal n x n fully nonblocking quantum switch,” *IEEE Transactions on Computers*, vol. 58, no. 2, pp. 238–250, 2009.

[11] A. Y. Oruç and H. M. Huang, “Crosspoint complexity of sparse crossbar concentrators,” *IEEE Transactions on Information Theory*, pp. 1466–1471, September 1996.

[12] W. Guo and A. Y. Oruç, “Regular sparse crossbar concentrators,” *IEEE Transactions on Computers*, pp. 363–368, 1998.

[13] Y. Shi and E. Soljanin, “On multicast in quantum networks,” in *40th Annual Conference on Information Sciences and Systems*, march 2006, pp. 871–876.

[14] C. Y. Lee and A. Y. Oruç, “Design of efficient and easily routable generalized connectors,” *IEEE Transactions on Communications*, pp. 646–650, Feb 1995.

[15] V. Bužek, S. L. Braunstein, M. Hillery, and D. Bruss, “Quantum copying: A network,” *Physical Review A*, vol. 56, no. 5, pp. 3446–3452, November 1997.

[16] N. Gisin and S. Massar, “Optimal quantum cloning machines,” *Physical review letters*, vol. 79, no. 11, pp. 2153–2156, September 1997.

[17] D. Bruss, A. Ekert, and C. Macchiavello, “Optimal universal quantum cloning and state estimation,” *Physical review letters*, vol. 81, no. 12, pp. 2598–2601, September 1998.

[18] A. Lamas-Linares, C. Simon, J. C. Howell, and D. Bouwmeester, “Experimental quantum cloning of single photons,” *Science*, vol. 296, 2002.

[19] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM Review*, vol. 41, no. 2, pp. 303–332, Jun 1999.

[20] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, May 1996, pp. 212–219.

[21] M. Oskin, F. T. Chong, I. L. Chuang, , and J. Kubiatowicz, “Building quantum wires: The long and the short of it,” in *Proc. 30th Annu. Int. Symp. Comput. Architecture*, 2003, pp. 374–385.

[22] N. Linial and M. Tarsi, “Interpolation between bases and the shuffle-exchange network,” *European Journal of Combinatorics*, vol. 10, pp. 29–39, 1989.

[23] H. Çam, “Rearrangeability of $(2n - 1)$-stage shuffle-exchange networks,” *SIAM Journal on Computing*, vol. 32, no. 3, pp. 557–585, March 2003.

[24] F. K. Hwang, *The Mathematical Theory of Nonblocking Switching Networks*, ser. Series on applied mathematics, F. K. Hwang, Ed. Singapore: World Scientific Publishing Co. Pte. Ltd, 1998, vol. 11.

[25] C. Clos, “A study of non-blocking switching networks,” *Bell System Technical Journal*, vol. 32, no. 2, pp. 406–424, March 1953.

[26] D. G. Cantor, “On nonblocking switching networks,” *Networks*, vol. 1, pp. 367–377, 1971.

[27] D. M. Koppelman and A. Yavuz Oruç, “A self-routing permutation network,” *Journal of Parallel and Distributed Computing*, vol. 10, no. 2, pp. 140–151, October 1990.

[28] V. E. Beneš, *Mathematical Theory of Connecting Networks and Telephone Traffic*. Academic Press, 1965.

[29] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Transactions on Computers*, vol. 25, pp. 1145–1155, 1976.

[30] J. H. Patel, "Performance of processor memory interconnections for multiprocessors," *IEEE Transactions on Computers*, vol. 30, no. 10, pp. 771–780, October 1981.

[31] L. R. Goke and G. J. Lipovski, "Banyan networks for partitioning multiprocessor systems," in *ISCA '73: Proceedings of the 1st annual symposium on Computer architecture*. New York, NY, USA: ACM, 1973, pp. 21–28.

[32] H. J. Siegel and S. D. Smith, "Study of multistage simd interconnection networks," in *ISCA '78: Proceedings of the 5th annual symposium on Computer architecture*. New York, NY, USA: ACM, 1978, pp. 223–229.

[33] I. Pease, M.C., "The indirect binary n-cube microprocessor array," *Computers, IEEE Transactions on*, vol. C-26, no. 5, pp. 458 –473, may 1977.

[34] C.-L. Wu and T.-Y. Feng, "On a class of multistage interconnection networks," *Computers, IEEE Transactions on*, vol. C-29, no. 8, pp. 694 –702, aug. 1980.

[35] J. Y. Hui, *Switching and Traffic Theory for Integrated Broadband Networks*, R. Gallager, Ed. Kluwer Academic Publishers, 1990.

[36] Y. Yang and Masson, "Nonblocking broadcast switching networks," *IEEE Transactions on Computers*, pp. 1005–1015, Dec 1991.

[37] J. P. Ofman, "A universal automaton," *Transactions of the Moscow Mathematical Society*, pp. 200–215, 1965.

[38] G. M. Masson and B. W.Jordan, "Generalized multistage connection networks," *Networks*, pp. 191–209, 1972.

[39] C. D. Thompson, "Generalized connection networks for parallel processor intercommunication," *IEEE Transactions on Computers*, pp. 1119–1125, Dec 1978.

[40] J. Turner, "Design of a broadcast packet switching network," *Communications, IEEE Transactions on*, vol. 36, no. 6, pp. 734 –743, jun 1988.

[41] C. Y. Lee and A. Y. Oruç, "Design of efficient and easily routable generalized connectors," Institute for advanced computer studies, University of Maryland, College Park, MD, Tech. Rep. UMIACS-TR-92-22,CS-TR-2846, 1992.

[42] D. Nassimi and S. Sahni, "Parallel permutation and sorting algorithms and a new generalized connection network," *Journal of the ACM*, pp. 642–667, July 1982.

[43] N. Pippenger, "Superconcentrators," *SIAM Jour. on Comput.*, vol. 6, no. 2, pp. 298–304, 1977.

[44] L. A. Bassalygo, "Asymptotically optimal switching circuits," *Prob. of Inform. Trans.*, vol. 17, no. 3, pp. 206–211, 1981.

[45] N. Alon, "Eigenvalues and expanders," *Combinatorics*, vol. 6, pp. 83–96, 1986.

[46] S. Nakamura and G. Masson, "Lower bounds on crosspoint in concentrators," *IEEE Trans. on Comp.*, pp. 1173–1178, December 1982.

[47] E. Gunduzhan and A. Y. Oruç, "Structure and density of sparse crossbar concentrators," *DIMACS Series in Discrete Mathematics and Computer Science. Advances in Switching Networks*, vol. 42, pp. 169–180, 1998.

[48] M. V. Chien and A. Y. Oruç, "Adaptive binary sorting schemes and associated interconnection networks," *IEEE Trans. on Par. and Distr. Systems*, vol. 5, pp. 561–571, June 1994.

[49] C. Y. Jan and A. Y. Oruç, "Fast self-routing permutation switching on an asymptotically minimum cost network," *Computers, IEEE Transactions on*, vol. 42, no. 12, pp. 1469–1479, 1993.

[50] M. Ajtai, J. Komlos, and E. Szemeredi, "An $o(n \log n)$ sorting network," in *Proc. 15th ACM Symp. Theory Comput.*, 1983, pp. 1–9.

[51] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, September 2000.

[52] J. Preskill. (1998) Physics 229: Advanced mathematical methods of physics – quantum computation and information. [Online]. Available: http://www.theory.caltech.edu/people/preskill/ph229

[53] H. Barnum, C. Caves, C. Fuchs, R. Jozsa, and B. Schumacher, "Noncommuting mixed states cannot be broadcast," *Physical Review Letters*, vol. 76, pp. 2818–2821, April 1996.

[54] R. Jozsa, "Fidelity for mixed quantum states," *Journal of modern optics*, vol. 41, no. 12, pp. 2315–2323, 1994.

[55] B. Schumacher, "Quantum coding," *Physical Review A*, vol. 51, no. 4, pp. 2738–2747, April 1995.

[56] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input versus output queueing on a space-division packet switch," *IEEE Transactions on Communications*, vol. 35, no. 12, pp. 1347–1356, Dec 1987.

[57] M. Karol and M. Hluchyj, "Queueing in high-performance packet-switching," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1587–1597, Dec 1988.

[58] N. Mckeown, A. Mekkittikul, V. Anantharam, and J. Walrand, “Achieving 100% throughput in an input-queued switch,” *IEEE Transactions on Communications*, vol. 47, no. 8, Aug. 1999.

[59] T. Troudet and S. Walters, “Hopfield neural network architecture for crossbar switch control,” *IEEE Transactions on Circuits and Systems*, vol. CAS-38, pp. 42–57, Jan. 1991.

[60] N. Mckeown, J. Walrand, and P. Varaiya, “Scheduling cells in an input-queued switch,” *IEE Electronics Letters*, pp. 2174–2175, Dec. 1993.

[61] T. Anderson, S. Owicki, J. Saxe, and C. Thacker, “High speed switch scheduling for local area networks,” *ACM Transactions on Computer Systems*, pp. 319–352, Nov. 1993.

[62] L. Tassiulas, “Linear complexity algorithms for maximum throughput in radio networks and input queued switches,” in *IEEE INFOCOM'98*, vol. 2, New York, 1998, pp. 533–539.

[63] D. Shah, P. Giaccone, and B. Prabhakar, “Efficient randomized algorithms for input-queued switch scheduling,” *IEEE Micro*, vol. 22, no. 1, pp. 10–18, Jan.-Feb. 2002.

[64] P. Giaccone, D. Shah, and B. Prabhakar, “An implementable parallel scheduler for input-queued switches,” *IEEE Micro*, vol. 22, no. 1, pp. 19–25, Jan.-Feb. 2002.

[65] B. Prabhakar, N. McKeown, and R. Ahuja, “Multicast scheduling for input-queued switches,” *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 5, pp. 855–866, June 1997.

[66] L. Veltri, “Maximum throughput in multicast queued packet switches,” in *ICC*, vol. 7, 2001, pp. 2033–2037.

[67] S. Gupta and A. Aziz, “Multicast scheduling for switches with multiple input-queues,” in *Hot Interconnects X*, Stanford, CA, August 2002.

[68] R. Ratan, M. K. Shukla, and A. Y. Oruç, “Quantum switching networks with classical routing,” in *Proceedings of the 41st Annual Conference on Information Sciences and Systems CISS'07*, March 2007, pp. 798–793.

[69] N. McKeown and B. Prabhakar, “Scheduling multicast cells in an input queued switch,” in *Proceedings of IEEE INFOCOM'96*, vol. 1, Mar 1996, pp. 261–278.

[70] M. A. Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, “Multicast traffic in input-queued switches: Optimal scheduling and maximum throughput,” *IEEE/ACM Transactions on Networking*, pp. 465–477, June 2003.

[71] D. Pan and Y. Yang, “FIFO-based multicast scheduling algorithm for virtual output queued packet switches,” *IEEE Transactions on Computers*, vol. 54, no. 10, pp. 1283–1297, Oct 2005.

[72] L. G. Valiant and G. J. Brebner, “Universal schemes for parallel communication,” in *Proceedings of the thirteenth annual ACM symposium on Theory of computing*. Milwaukee, Wisconsin, United States: ACM Press, 1981, pp. 263–277.