A High-Throughput Energy-Efficient Passive Optical Network with Multiple Planes

by

Yang An

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Applied Science

in

Electrical and Computer Engineering

Carleton University
Ottawa, Ontario
December 2015

Abstract

Datacenter applications impose heavy demands on bandwidth and also generate a variety of communication patterns (unicast, multicast, incast, and broadcast). Supporting such traffic demands leads to networks built with exorbitant facility costs and formidable power consumption if conventional design is followed. In this thesis, we propose a novel high-throughput datacenter network that leverages passive optical technologies to efficiently support communications with mixed traffic patterns. Our network enables a dynamic traffic allocation that caters to diverse communication patterns at low power consumption. Specifically, our proposed network consists of two optical planes, each optimized for specific traffic patterns. We compare the proposed network with its optical and electronic counterparts and highlight its potential benefits in terms of facility costs and power consumption reductions. To avoid frame collisions, a high-efficient distributed protocol is designed to dynamically distribute traffic between the two optical planes. Moreover, we formulate the scheduling process as a mixed integer programming problem and design three greedy heuristic algorithms. Finally, simulation results show that our proposed scheme outperforms the previous POXN architecture in terms of throughput and mean packet delay.

Acknowledgements

I would like to express my gratitude to all the people who helped me to finish this research to fruition. First, I would like to thank Professor Changcheng Huang for providing me the opportunity of taking part in Master of Applied Science program. I am deeply grateful for his help, valuable instructions and financial support during my program of study. I do not have enough words to express my deep and sincere appreciation.

Many thanks go to the entire group of my laboratory for their numerous discussions and help.

Finally, I also express my very profound gratitude to my parents for providing me with faithful support and continuous encouragement throughout my study. The accomplishment of this research would not have been possible without them.

To my parents

Table of Contents

List of Tables

List of Figures

List of Acronyms

AWG	Arrayed-waveguide Grating
AWGR	Arrayed-waveguide Grating Router
CSMA/CD	Carrier Sense Multiple Access Protocol with Collision Detection
ECMP	Equal-cost Multi-path routing
EPSN	Electronic Packet Switched Network
FSO	Free Space Optics
HEDAP	High Efficiency Distributed Access Protocol
LQF	Longest Queue First
LR	Long-range
MCDAP	Multiple Channels Distributed Access Protocol
MEMS	Micro Electro Mechanical System
OSS	Optical Space Switch
PERCS	Productive, Easy-to-use, Reliable Computing System
POXN/MP	Passive Optical Cross-connection Network with Multiple Planes
SDN	Software Defined Network
SR	Short-range
SQF	Shortest Queue First
SQF/CF	Shortest Queue First with Cut-cover Function
ToR	Top of Rack
WDM	Wavelength Division Multiplexing
WFFOC	Wavelength Flattened Fiber Optic Coupler

Mathematical Notations

C^E – the Capex per link of the EPSN

C^{13} – the cost of LR (1310 nm) transponder

$\mathrm{C^O}$ – the Capex per link of POXN/MP

C – the cost per port of a passive coupler fabric

$\mathrm{C^W}$ – the cost of a 2x1 WFFOC and a 1x2 AWG optical splitter

C^{15} – the cost of LR (1550 nm) transponder

C^X – the cost of tunable LR transponder

E – the algorithm efficiency of a transmitting port

i – the index of a transmitting port

j – the index of a receiving port

N – the port count number of a passive coupler fabric

S – the cost per port of an electronic switch (includes the cost of line card and switch fabric)

T_i^L – the loopback time for transmitting port *i*

T^P – the corresponding one-way propagation delay

T^C – the constant time value for message processing

T^D – the transmission delay of the CONFIRMATION message

T_{ij}^Q – the transmission time of packets in the queue that will be sent from transmitting port *i* to receiving port *j*

T^T – the constant tuning time for a tunable transponder

T^I – the inter-port guard interval

t_{ij}^{S} – the starting time of transmitting port i sending to port j

t_{ij}^{F} – the finishing time of transmitting port i sending to port j

t_{ik}^{Mx} – the k-th Type x mismatch that transmitting port i experiences

$\overline{\mathrm{X}}$ – the per-frame service time

ρ – the offered load to a transmitting port

λ – the frame arrival rate of a transmitting port

Chapter 1: Introduction

The emerging trend in cloud computing is to group computing devices into large-scale datacenters that provide existing and growing cloud services and diverse Internet applications to many independent end users. To leverage the rich computing resources that are available, advanced computing technologies such as MapReduce [1-3] are being widely adopted. Application tasks are partitioned into multiple smaller pieces that are assigned to various servers. This leads to extensive data exchange among servers to complete a single job. The result is massive traffic volumes flowing within a datacenter. Moreover, with free-of-charge datacenter applications gaining traction (e.g., Google Drive), datacenter service providers are confronted with the challenge of accommodating exponentially increasing demands for network bandwidth while avoiding excessive increases in facility costs and power consumption [4]. Additionally, recent measurement studies [5-9] indicate that various types of traffic, such as unicast, incast, and multicast, coexist in a datacenter network and exhibit highly dynamic and unpredictable patterns. These patterns change constantly at a granularity of 15 ms [6, 9]. To adjust for this variability, datacenter networks are typically engineered with excessive bandwidth [9-12]. However, supporting the continued exponential growth of bandwidth requires more-expensive electronic switches with higher transmission rates and higher power consumption.

In this thesis, we propose a high-throughput passive optical cross-connection network that enables dynamical traffic allocation to satisfy communication demands with mixed traffic patterns. In particular, we use N x N optical coupler fabrics instead of active

optical devices to construct an optical cross-connection network, which dramatically reduces power consumption. Our network consists of two planes, where each optical plane is optimized for specific traffic patterns. Both planes maintain the power and capacity advantages deriving from the deployed optics. Moreover, one of the two planes is designed with a new mechanism that enables high-throughput communications for unicast traffic. We name this novel network Passive Optical Cross-connection Network with Multiple Planes (POXN/MP). We also propose a link layer protocol to coordinate traffic transmissions among connected ports. The proposed protocol enables dynamical traffic distribution between the two optical planes and among the different optical wavelengths within the same optical plane. We evaluate the protocol's performance using simulations.

In this chapter, we first state the discovered problems. We then briefly introduce the overview of our main results. Finally, we summarize our contributions, and outline the rest of this thesis.

1.1 Problem Statement

First, in this thesis we ask the following question: Is there a way to construct a high-throughput datacenter network with low power consumption, which can also provide a dynamic traffic allocation that accommodates to mixed traffic patterns? The problem outlined above leads us to define three main goals underlying our work:

- To allow the proposed datacenter network to offer high-throughput communications.
- To utilize low facility costs, low power consumption, and reliable passive optical devices to construct datacenter networks.

- To enable the proposed network to provide different transmission mechanisms for various traffic patterns and enable a dynamic traffic allocation.

Previous research studies introduced different architectures for building datacenter networks. However, the improvements of these studies are limited by different kinds of drawbacks, such as high facility costs, high power consumption, or slow configuration speeds [9-12, 25-43]. To our best knowledge, there is no such study that can improve the overall performance for datacenter networks without excessive increase in facility costs and power consumption. Thus, we propose the POXN/MP to achieve these goals. Our work is heavily influenced by the emerging new directions that use passive optical devices to construct datacenter networks [13-14].

Second, when we implemented the POXN/MP, we discovered another problem which point to the need of an access protocol to coordinate traffic transmission among connected ports in the POXN/MP. Moreover, current access protocols are not suitable for the POXN/MP, which we will discuss in Chapter 3. Thus, we propose a high-throughput Multiple Channels Distributed Access Protocol (MCDAP) to schedule traffic transmission. Furthermore, the MCDAP supports dynamic traffic distribution between the two optical planes.

Third, we identified a scheduling problem when we implemented the MCDAP. In Chapter 3, we discuss that there is no such study that can address our scheduling problem in the MCDAP. Thus, we formulate the scheduling process as a mathematical programming problem and design three heuristic algorithms for the MCDAP.

1.2 Overview of Results

In the POXN/MP, multiple transmitting ports can communicate with multiple receiving ports in parallel through the new designed plane for unicast traffic as long as traffic is delivered through different wavelengths. Moreover, a transmitting port can send traffic through the two optical planes simultaneously, which can make the throughput of POXN/MP even larger. In addition, we demonstrate the benefits of POXN/MP in terms of facility costs and power consumption by comparing the POXN/MP with its main electronic and optical counterparts. Furthermore, simulation results show that the per-port maximum efficiency of the MCDAP is higher than that of the High Efficiency Distributed Access Protocol (HEDAP) [14]. Packets experience less delay in the MCDAP compared with the HEDAP under the same per-port offered load. At last, the proposed network enables dynamic traffic allocation between the two optical planes to cater to dynamic traffic pattern.

1.3 Contributions of Thesis

Our main contributions can be summarized in the following three aspects:

- Network: We proposed the POXN/MP for datacenter networks, where different planes are designed for different traffic patterns. Because no active device is involved in the optical domain, the proposed network has low facility costs and low power consumption.
- Protocol: We proposed a distributed access protocol that enables collision-free transmission and dynamic traffic allocation between the two planes.

- Algorithm: We formulated the scheduling problem as a mathematical programming problem and then designed three heuristic algorithms that enable transmissions without collisions and optimize the bandwidth utilization.

1.4 Organization of Thesis

This section provides an extended summary of the thesis. The rest of this thesis is organized as follows:

In chapter 2, we present the background information about the architecture of typical datacenter networks. Then, we describe the major observation results about datacenter traffic characteristics from prevalent datacenter traffic studies.

In chapter 3, we first describe existing related datacenter networks and their drawbacks. Then, we introduce related access protocols and their shortages. Finally, we describe a scheduling algorithm that is used for computing the optimized optical circuit in c-Through [26] and present reasons why this algorithm is not suitable in our case.

In chapter 4, we present the physical layer of POXN/MP and show its benefits in terms of facility costs and power consumption.

In chapter 5, we present the working process of our proposed distributed protocol. It enables collision-free transmission and dynamic traffic allocation.

In chapter 6, we formulate the scheduling process and describe the proposed heuristic algorithms in detail.

In chapter 7, we present simulation results of our proposed protocol in two parts. One part focuses only on the efficiency of the proposed heuristic algorithms. The other part presents the performance of MCDAP at a system level.

In chapter 8, we conclude this thesis and then present suggestions for future research.

1.5 Submitted Manuscript

Y. An and C. Huang, “A High-Throughput Energy-Efficient Passive Optical Network”, under review in the *Journal of Lightwave Technology*. (Submission: December 2015).

Chapter 2: Background Information

This chapter describes the architecture of typical datacenter networks. Then, we review the details about datacenter traffic characteristics investigated from prevalent datacenter network studies.

2.1 Architecture of Typical Datacenter Networks

Current datacenter networks are typically constructed in a tiered architecture [15, 16]. The high level diagram of the architecture of typical deployed datacenter networks is depicted in Figure 2.1 as an example. Racks of servers are connected through Top of Rack (ToR) switches, which form the edge-switch tier. ToR switches are then interconnected with core switches to build a 2-tier datacenter network. However, in order to accommodate more servers in some warehouse-scale datacenters, a middle tier, called aggregation tier, is usually added. The ToR switches are connected to the aggregation switches first, which themselves are connected to core switches instead of the ToR switches being connected to core switches directly. When constructed in this manner datacenters can host tens to hundreds of thousands of servers.

2.2 Datacenter Traffic Characteristics

The major findings of datacenter traffic studies show that a datacenter network typically consists of mixed traffic patterns and exhibits highly dynamic and unpredictable characteristics. Moreover, datacenter traffic patterns are different from the wide area networks. Misusing wide area network traffic patterns in datacenters causes serious impairments for designing and constructing datacenter networks. Therefore, it is necessary to survey datacenter traffic characteristics.

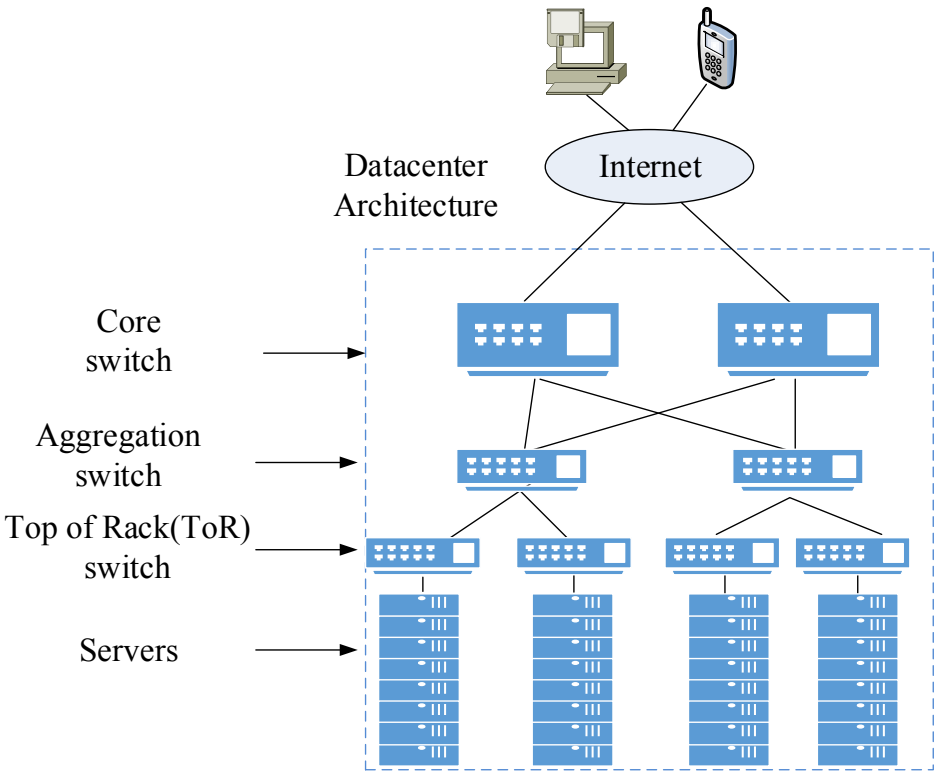

Figure 2.1 The architecture of typical datacenter networks.

Below, a summary of datacenter traffic characteristics is presented based on published papers [5-7, 9, 16-21]. All the mentioned statistics come from observations of current participating datacenters, which are typically constructed following the architecture depicted in Figure 2.1.

Datacenter traffic shows common characteristics as well as dissimilarities. The following descriptions present the major observation results of datacenter traffic characteristics:

- Annual global datacenter traffic will reach 8.6 zettabytes by the end of 2018, which means there will be nearly triple the amount of the traffic volume in 2013, with an annual growth rate of 23% from 2013 to 2018 [20].

- Although application distribution varies across different kinds of datacenters, datacenter traffic studies have consistently reported that 80% of datacenter traffic flows have an inter-arrival time within millisecond time scale [6] for ToR switches in participate datacenters.
- Datacenter traffic is frequently reported to be bursty and unstable, consisting of mixed traffic patterns [7]. These patterns change constantly at a granularity of 15 milliseconds [6, 9].
- Datacenter studies have consistently reported a bimodal packet size among some of investigated datacenters [17], with packet either approaching the maximum transmission unit or remaining quite small. Flow sizes of 80% of the datacenter traffic are considerably small (i.e., less than 10 KB). However, less than 20% of the total bytes are contributed by a few large flows [6]. Moreover, statistics show that most of the transmitted bytes in datacenters are carried by larger flows varying from 1 MB to 50 MB [7].
- Different kinds of datacenters show various traffic characteristics. For commercial datacenters, most of the generated traffic is intra-rack traffic. Some datacenter traffic studies show that commercial datacenter traffic is found to be heavily rack local that a majority of traffic originated by servers (80%) stays within the rack [6, 18-19]. The reason for this can be explained by the fact that datacenter administrators prefer to locate dependent servers and applications in the same rack to avoid more extra-rack traffic. In contrast, datacenters owned by universities or enterprises, such as Facebook, show different results; in these datacenters, more than half of the server-originated traffic is extra-rack traffic [17].

- Most of datacenter traffic studies reveal the burst and dynamic features of datacenter traffic. It appears that there is no predictability for traffic with long time-scales. However, some studies indicate that an amount of traffic is predictable for a short timescale [21]. It can be found that an approximately 35% of the total traffic exchanged between pairs of ToR switches remains stable across 1 second time-scale [21]. This is can be explained by the existence of some large flows. Datacenter network designers can utilize the historical route records for the last 1 second to optimize data transmission of predictable traffic.
- Most of statistics mentioned above come from the observations of Microsoft datacenters. A report upon the network traffic observed in some of Facebook's datacenters reveals other interesting findings [17]. In this report, traffic demands are uniform and stable with rapidly changing, internally bursty heavy hitters. Moreover, most of packets have small packet sizes and show continuous arrival patterns at the end-host level. Furthermore, there are many concurrent flows within datacenter networks.

In summary, datacenter applications impose heavy bandwidth demands to the underlying networks. Datacenter traffic tends to have a small flow size and stay within a rack; while, there are some large flows existing within datacenter networks. Moreover, datacenter traffic consists of mixed traffic patterns and exhibits highly dynamic and unpredictable patterns. These patterns change constantly at a small granularity.

Accordingly, researchers have focused a great deal of effort on designing datacenter networks to accommodate to the aforementioned datacenter traffic characteristics. In this thesis, we propose the POXN/MP that offers high-throughput communications, effectively handles mixed traffic patterns, and enables a dynamic traffic allocation.

Chapter 3: Review of the State of the Art

Based on the afore mentioned discussions, the design of an efficient high-throughput datacenter network that can accommodate to above mentioned datacenter traffic characteristics is a challenging inter-disciplinary research area, requiring expertise from several fields. First, this chapter surveys some of the existing related datacenter networks that have been proposed so far and points out their drawbacks. Second, we describe related access protocols and present reasons why they are not suitable to our case. Third, we discuss a scheduling algorithm that has been used for computing optimized optical circuits and describe its shortages in our case.

3.1 State of the Art in Datacenter Networks

In recent years, a number of research studies have focused on the design of datacenter networks that can provide high-throughput communications, low transmission latency, low facility costs, and low power consumption. A major distinction when it comes to the interconnection technologies is whether they utilize electronic packet switches or optical switches.

3.1.1 Electronic Packet Switched Datacenter Networks

Typical electronic datacenter networks are constructed with high oversubscription ratios to reduce facility costs. This may lead to long transmission latency caused by hotspots. To reduce transmission latency with low facility costs, datacenter providers utilize a large number of inexpensive commodity electronic switches to construct datacenter networks [9-12]. Moreover, the adoptions of Software Defined Network

(SDN) have inspired some studies to build centralized datacenter networks [21-24] to provide efficient communications.

However, discussions in section 2.2 show that high-throughput communications are required to handle the large volumes of exchanged traffic flows within a datacenter network. Existing proposed Electronic Packet-Switched Networks (EPSNs) leverage a large number of commodity switches to construct datacenter networks to reduce facility costs. To meet the exponential growth of bandwidth requirements within a datacenter, it is important to scale the transmission rate of network interfaces to provide excessive bandwidth. Thus, the speeds of the switching ports need to be increased accordingly, leading to formidable high facility costs and power consumption. Therefore, the improvements of these proposed EPSNs are eventually limited by various aspects of electronic packet switches (e.g., facility costs, power consumption, port speeds, etc.).

3.1.2 Optical Switched Datacenter Networks

Optical technologies, which feature high bandwidth and low power consumption, offer viable solutions to meeting the high bandwidth and low power consumption requirements. Many optical devices are transparent to signal formats and bit rate [14], which makes it easy to upgrade to a higher bit-rate transmission. Moreover, power consumption is dramatically reduced with optical devices compared with their electronic counterparts, where signals are processed at the packet level. Therefore, it is important to explore the feasibility of utilizing optical technologies in constructing datacenter networks [15, 25-38, 40-43].

Some studies advocate offloading high-volume traffic to optical circuit-switched networks for stand-alone point-to-point bulk transfers. Basic optical modules that are

utilized to implement optical interconnections are typically composed of Micro-Electro Mechanical System (MEMS) based optical space switches. Examples of this architecture include c-Through [26], Helios [25], and Proteus [34]. Currently, optical circuit switches are being proposed to replace a fraction of the core electronic switches, to construct an all-optical architecture, and/or to interconnect edge ToR switches as shortcut paths.

However, commercially available optical circuit switches are constrained by their high costs and slow configuration speeds. Both limit the performance of optical circuit switches. Therefore, optical circuit switched networks are more suitable for slowly-varying traffic with aggregate bandwidth. Moreover, utilizing point-to-point optical links for multicast/broadcast traffic transmission requires multiple optical links to be set up to send redundant traffic, which introduces multiple reconfigurations that greatly reduce the effectiveness of the optical networks.

Another important research trend is the use of optical packet switching and Arrayed Waveguide Grating Router (AWGR) techniques to construct datacenter networks [15, 35-38, 42]. For instance, C. Nitta *et al.* leveraged AWGR technologies to construct an optical packet switched datacenter network [37-38], which was mainly inspired by the Productive, Easy-to-use, Reliable Computing System (PERCS) interconnections designed for high performance computers [39].

Because of the difficulties associated with optical buffering, optical packet switched networks require extremely complex systems and electronic control mechanisms for contention resolution, which negates the benefits of optical technologies.

The next subsection presents two unconventional optical interconnections that have been recently proposed for datacenter networks.

3.1.2.1 **Rethinking the Physical Layers of Datacenter Networks of the Next Decade: Using Optics to Enable Efficient-*Cast Connectivity**

To deal with diverse datacenter traffic patterns, H. Wang *et al.* proposed an unconventional approach, termed by *Cast [13, 27-28]. *Cast follows a typical two-tiered datacenter network. In *Cast, ToR switches are connected to servers and a hybrid electronic/optical core switch tier. In this approach, different physical optical modules are designed and attached to Optical Space Switches (OSSs) to cater to different traffic patterns.

The high-level block diagram of *Cast is depicted in Figure 3.1. The hybrid core switch tier consists of two different kinds of switches: electronic packet switches that are used for transmitting latency sensitive unicast packets; and optical circuit switches that transmit large flows in other traffic patterns.

By utilizing different physical optical modules, this approach achieves the goal of providing a physical layer that is intrinsically compatible with mixed traffic patterns. However, different optical modules must be employed at the physical layer to address various traffic patterns, which construct a rather complex physical layer. Moreover, the traffic patterns generated within a datacenter are unpredictable. Hence, some deployed optical modules designed for different traffic patterns have to be overprovisioned. Finally, because of the variability of datacenter traffic, it is extremely challenging to achieve flexible reconfigurations through hardware.

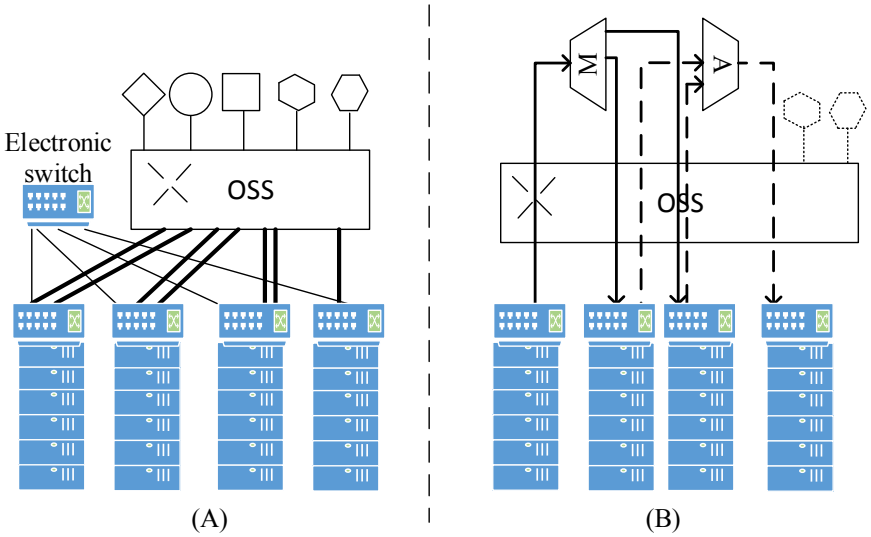

Figure 3.1 A sample of optical *-Cast connectivity.
(A) Optical *-Cast network architecture (B) Using the OSS as a connectivity substrate to deliver multicast and incast.

3.1.2.2 **A Reconfigurable Wireless Datacenter Fabric Using Free-Space Optics**

N. Hamedazimi *et al.* from Stony Brook University proposed a hybrid electrical/optical architecture for datacenter networks called FireFly [29]. FireFly utilizes Free-Space Optics (FSO) techniques to construct an optical wireless datacenter network. The key motivation of FireFly is that providing fully flexible interconnections among racks can yield near-optimal performance even without tiered switch architecture. In FireFly, instead of utilizing wired line interconnections, it leverages the established technology of infrared laser beams to transmit packets wirelessly.

The high-level block diagram of FireFly is depicted in Figure 3.2. It uses ToR switches to connect to servers; however, unlike the aforementioned architectures, there are no aggregation or core tier switches. Instead, a number of steerable FSO devices are deployed on every ToR switch. To prevent FSO devices from obstructing each other, a

ceiling mirror is installed on the space above the ToR switches. Hence, if one server transmits packets to another server in another rack, packets first arrive at the connected ToR switch, then the network manager integrated on the ToR switch configures the steerable FSO devices according to a routing table to create a wireless link between source ToR switch and destination ToR switch. Finally, packets are transmitted to the destination ToR switch through a wireless link and then to the desired server.

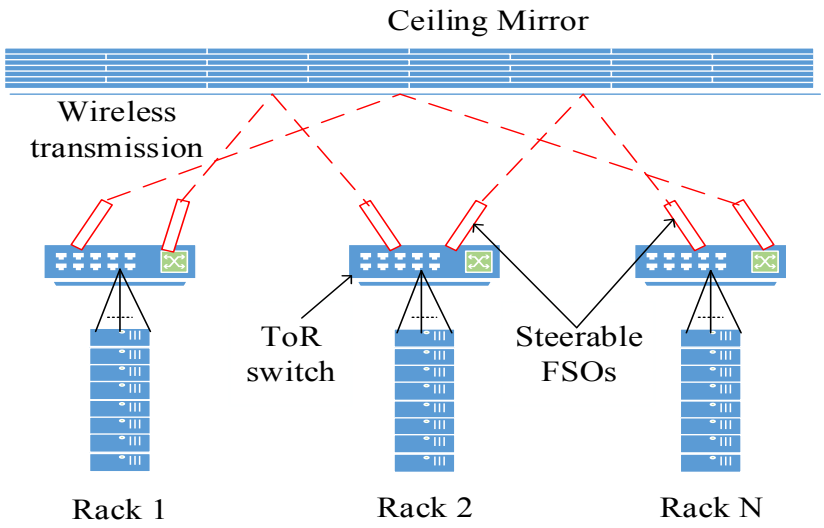

Figure 3.2 A sample of FireFly architecture.

FireFly can provide additional flexibility, reliability and scalability with its wireless links. However, its improvements are limited by the long reconfiguration speed of the deployed steerable FSOs. Thus, FireFly is more suitable to work as shortcut paths to mitigate long transmission latency caused by hotpots.

3.1.3 Summary of Shortages for Existing Related Datacenter Networks

It can be found that the aforementioned datacenter networks have different kinds of drawbacks. It is helpful to summarize these drawbacks according to the deployed

technologies. We list those shortages in Table 3.1 according to different datacenter network categories.

Table 3.1 Shortages of related datacenter networks.

Electronic packet switched datacenter network	Slow port speed, high facility costs, and high power consumption [9-12, 21].
Optical circuit switched datacenter network	Slow reconfiguration time, require multiple copies for multicast/broadcast traffic, and introduce extra reconfiguration overheads [25-34, 43].
Optical packet switched datacenter network	Extremely complex systems and control mechanisms are required to address contention problems in the optical domain [15, 35-38, 42].

3.2 Related Access Protocols

Access protocols are widely used as the building blocks in many access networks, such as EPONs [44], Wi-Fi [45], and cellular networks. Access protocols can be classified into two categories: random and scheduled. Scheduled access protocols can be further divided into two classes: polling and reservation. There are two potential candidates to be considered for the POXN/MP: random access protocols and polling protocols.

3.2.1 Random Access protocols

One potential candidate to consider is the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol. The mechanism of how the CSMA/CD protocol works can be simplified as follows. First, a port may begin to transmit traffic at any time, but before the transmission, it listens to the channel, if it senses that another port is transmitting, the port waits a random amount of time and then listens to the channel again. If the channel is sensed to be idle, then the port begins to transmit frames.

If the channel is sensed to be busy again, the port has to wait another random back-off time. Second, during the transmission, the port also listens to the channel. If the port detects that another port is transmitting frames, it stops transmitting and start the next transmission when the channel is sensed to be idle. Collisions can be detected through measuring amplitude in receivers [46].

Moreover, early works in [45] and [47] introduced and improved random access protocols for local area optical networks. An optical star coupler is used to interconnect end users in these networks. However, these random access protocols are limited by providing low efficiency.

For CSMA/CD to work properly, the transmission delay of one Ethernet frame must be larger than the worst case propagation delay so that all the frame collisions can be detected. However, in an intra-datacenter network, when the worst-cast loopback fiber distance is set to 20 km and links work at 10 Gb/s line rate, the propagation delay goes far beyond the region in which CSMA/CD can work properly.

3.2.2 Polling Protocols

Many polling protocols are typically centralized access protocols, where one of the nodes works as the master node to handle resource scheduling and collision avoidance. These protocols are implemented in networks that have a hierarchical physical structure so that the nodes that deploy at the high level of the hierarchy naturally serve as the master.

For polling protocols to work properly, a master-slave hierarchy is desired. However, such a hierarchy is not suitable for datacenter networks, where all the servers are homogeneous and peers by nature within a datacenter [48]. Moreover, to reduce facility

costs, current datacenters deploy large numbers of commodity PCs as servers. The inherent unreliability of these servers leads to the "single point failure" problem being inevitable in datacenter networks if polling protocols are used. We propose the MCDAP, which is a distributed access protocol, to address the link layer contention problems of POXN/MP.

3.3 Related Scheduling Algorithm

Given a traffic matrix, Edmond's algorithm [49] is adopted in c-Through [26] to determine how to connect the server racks by optical paths in order to maximize traffic volume offloaded to the optical network. The traffic matrix is a graph $G = (E, V)$. V is the vertex set, in which each vertex denotes a rack, and E is the edge set. The weight of an edge $e, w(e)$ is the traffic volume between the racks. A matching is a set of edges where each vertex is incident with at most one edge. Here, a matching represents a cross-connect pattern of an optical switch. The problem of carrying the maximum amount of traffic with an optical switch can be summarized as finding a matching with maximum aggregated weight. The remaining traffic demands will be carried by electronic switches. The approach is optimal in the sense that it can offload traffic from electronic switches to the highest degree.

However, the scheduling problem in this thesis requires that the POXN/MP carry the demands between all transmitting and receiving ports during a cycle period. This means that a transmitting port needs to be connected to multiple receiving ports at different times during a cycle period, which makes Edmond's algorithm inapplicable. One might think that we can divide a cycle into multiple rounds and apply Edmond's algorithm at the beginning of each round. To do so, a new round can only start after all optical ports

have finished transmitting all scheduled amounts of data for the current round so that each round is a fresh start. However, the realignment process at the end of each round will make the POXN/MP less efficient over the period of a cycle because ports finishing earlier have to wait until all ports finish.

Another problem with Edmond's algorithm is its processing time requirement. Although Edmond's algorithm can be completed in polynomial time [49], it is still too complicated to be executed within less than a round time, which is extremely small (e.g., submicron-second range), as we discussed later.

In addition, to our best knowledge, there is no such study that can address our scheduling problem for the MCDAP. Thus, we formulate the scheduling process as a mathematical programming problem and design three heuristic algorithms to solve the scheduling problems.

Chapter 4: POXN/MP

In this chapter, we describe the physical layer of POXN/MP in detail.

Ni et al. proposed POXN for datacenters [14]. The transmission system of a sample POXN is depicted in Figure 4.1. A coupler fabric can have *N* inputs and *N* outputs. Optical power from each input port is equally divided among the *N* outputs so that a message sent by one transmitting port will be received by all the receiving ports.

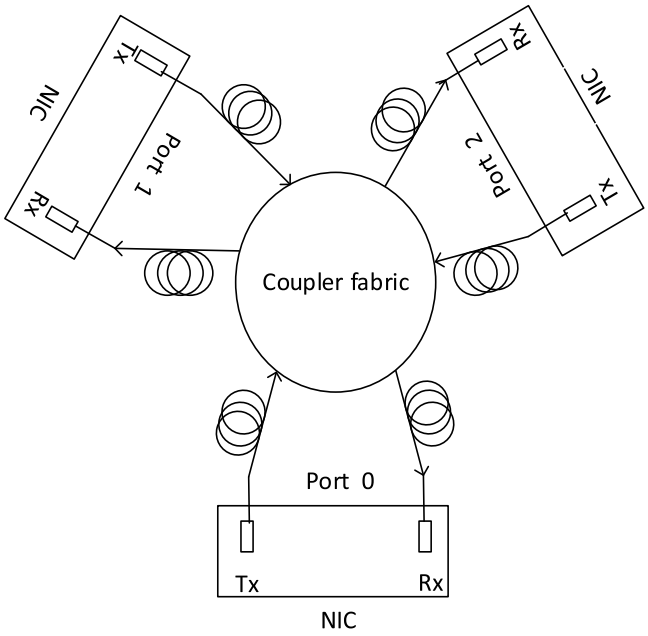

Figure 4.1 Physical layer of a sample POXN.

In POXN, ports connected to the same optical coupler fabric share a common wavelength and transmit data sequentially within their assigned time intervals. Due to the broadcast property of coupler fabrics, all the connected ports can receive the same data without requiring duplication by intermediate fabrics, which is not the case for electronic switches. Therefore, POXN is efficient in carrying multicast traffic, but it suffers from

low throughput for unicast traffic because ports sharing a common wavelength must transmit their traffic in sequence.

To overcome the performance limitations of POXN under unicast traffic, we propose the POXN/MP, which introduces an extra plane to address unicast traffic. Additionally, POXN/MP efficiently enables high-throughput communication with adaptive response to varying communication patterns in datacenters. This chapter explores the physical-layer system of POXN/MP and studies its benefits in terms of capital expenditure (Capex) and power consumption. The benefits are demonstrated through a comparison of POXN/MP, POXN, and EPSNs.

4.1 Physical Interconnections of POXN

Similar to POXN, the core of our hardware construct is a large-scale optical coupler fabric that forms a passive optical cross-connection without involving active optical devices. However, unlike POXN, each input port of the proposed scheme is connected to two transmitters in a transmitting port of an electronic switch or server through a 2x1 Wavelength Flattened Fiber Optic Coupler (WFFOC), as shown in Figure 4.2. One of the two transmitters uses a fixed wavelength that is common to all transmitting ports, while the other is tunable. In the reverse direction, one of the output ports of the scheme is connected to the corresponding receiving port of the electronic switch or server through a 1x2 Arrayed-Wavelength Grating (AWG) optical splitter [15]. Each receiving port is equipped with two fixed receivers of different wavelengths. One receiving port uses the same wavelength as the fixed wavelength transmitter, while the other works at a wavelength that varies from receiving port to receiving port. Throughout this thesis, input/output ports denote ports of the coupler fabric. Transmitting/receiving ports refer to

ports of electronic switches or servers.

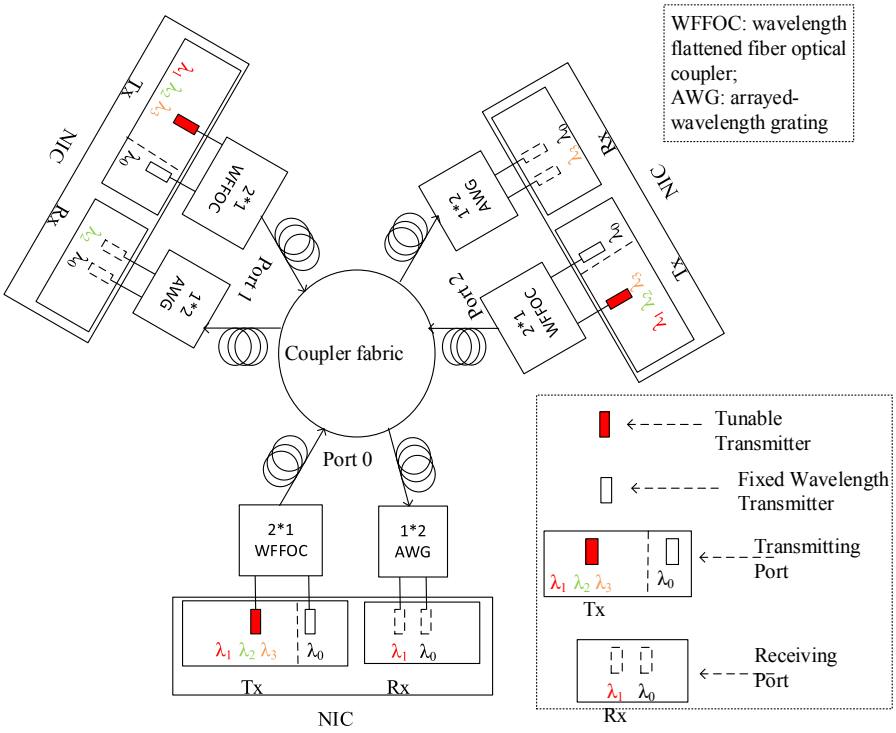

Figure 4.2 Physical layer of a sample POXN/MP.

All the transmitters and receivers with a common wavelength that are associated with different ports form an optical plane. This plane works on a shared transmission channel that is similar to the POXN. The plane, referred to as a multicast plane, can carry multicast traffic efficiently [14]. All the tunable transmitters and fixed receivers with different wavelengths that are associated with different ports form the second optical plane, which we call the unicast plane. The unicast plane is a new mechanism designed for unicast traffic pattern. For the unicast plane, the different wavelengths used by different receiving ports are called different channels. A tunable transmitter can send unicast traffic to any receiving port by tuning to its channel as long as there is no other transmitter trying to communicate with the same receiving port. With today's technology,

a tunable transmitter can tune from one wavelength to another in 5 ns [50], which is significantly faster than the switching time of optical MEMS switches (typically, approximately 10 ms [14]). However, to reduce facility costs, we use tunable transponders with tuning speeds on the 5-microsecond time scale [51].

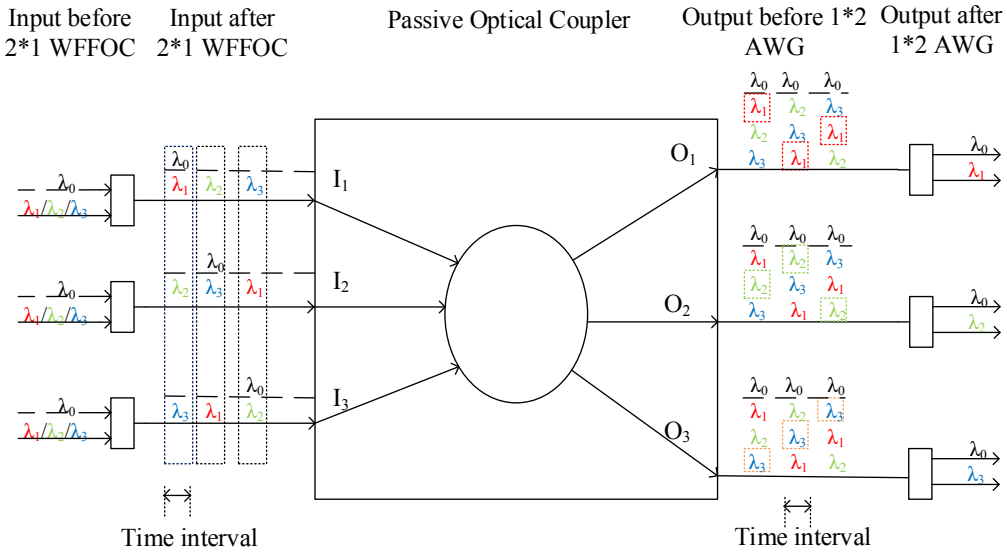

Figure 4.3 Wavelength routing map of a sample 3-port POXN/MP.

As an example, a wavelength routing map of a 3-port POXN/MP, constructed with 3 input ports, 3 output ports, and 4 wavelengths is depicted in Figure 4.3. All ports are connected to the same optical coupler fabric through two planes. The wavelength λ_0 is specifically reserved for the multicast plan. Traffic sent from any transmitting port can be delivered to all receiving ports through λ_0 in the multicast plane. All the receivers in the unicast plane use fixed wavelengths, (i.e., receiving port 1 can only receive traffic transmitted through λ_1; receiving port 2 can only be reached through λ_2, etc.). A transmitting port can use λ_1, λ_2, or λ_3 to send unicast traffic through dynamic tuning to receiving ports 1, 2, or 3 respectively. Meanwhile, multiple transmitting ports can

communicate with multiple receiving ports in parallel through the unicast plane as long as traffic is delivered through different unicast channels. Moreover, a transmitting port can simultaneously send traffic to a receiving port through both the multicast and unicast planes. Thus, in theory, an *N*-port POXN/MP achieves *N+1* times bandwidth compared with the POXN.

4.2 Advantages of the Proposed Architecture

This subsection discusses the power budget for POXN/MP and analyzes its benefits in terms of Capex and power consumption. These benefits are demonstrated by comparing the POXN/MP with its main optical and electronic counterparts. In particular, a general cost formula is developed for POXN/MP.

4.2.1 Power Budget

Similar to POXN, there are no extra active optical devices involved in the optical domain of POXN/MP. The calculation of the power budget for POXN [14] shows that the deployed coupler fabric in POXN/MP causes extra power split loss. The power split loss at each output port is $5.47 \cdot [log_3^N] - 0.2\, dB$ [14], where *N* stands for the port number of a coupler fabric. In addition, it is essential to consider the 2x1 WFFOC (2.5 dB), the 1x2 AWG optical splitter insertion (3.5 dB), and the fiber transmission loss (4 dB) [14, 52-53]. Together, these devices cause a power loss of 10 dB. Given a power budget of 35 dB, offered by a Long-Range (LR) (1550 nm) transponder with the currently available optical technology, the port count number of a coupler fabric can scale up to *N*=81.

4.2.2 POXN/MP vs POXN

A Wavelength Division Multiplexing (WDM) tunable LR transponder costs 1.5 times

that of a fixed-wavelength LR transponder [4]. A new POXN/MP deploys one more tunable transponder per port than a POXN [14]. Additionally, the facility costs for a port of an optical coupler fabric, a 2x1 WFFOC, and a 1x2 AWG optical splitter are much cheaper compared with the cost of a transponder. Hence, based on the aforementioned cost assumption, an *N*-port POXN/MP achieves *N+1* times bandwidth, resulting in about 2.5 times the cost of the POXN [14]. In the long run, when *N* scales up, it satisfies the general cost/bandwidth rule in industry development—that a new generation technology should achieve 10 times the bandwidth with only 4 times the cost increase [4]. In our case, using a 64-port network as an example, the POXN/MP can achieve 65 times the bandwidth with 2.5 times the cost compared with the POXN.

4.2.3 POXN/MP vs EPSN

To demonstrate the Capex benefits of POXN/MP, it is also essential to compare it with the Capex of EPSN competitors, which can be undertaken by developing a general formula based on a price comparison model. In this model, we assume that the use of the POXN/MP is placed closer to the end servers in the Fat Tree topology [54]. In theory, the unicast plane of the POXN/MP can sustain the same performance level in terms of network bandwidth capacity when compared to EPSNs. The high level block diagram of an 8-port EPSN is depicted in Figure 4.4 as an example.

The typical maximum range for a Short-Range (SR) transponder is up to 300-400 m, which is insufficient for current warehouse-scale datacenter network environments. We assume the fiber length in our model to be 1 km, which is longer than the maximum range for an SR transponder [14]. Therefore, the EPSNs deploy two LR (1310 nm) transponders per link, one at the server end and one at the switch end. Its price formula

can be written as

$$C^E = S + 2C^{13} \tag{4.1}$$

where C^E represents the Capex per link of the EPSN, S stands for the cost per port of an electronic switch, which includes the cost of line card and switch fabric, and C^{13} represents the cost of an LR (1310 nm) transponder.

Each port of a server in the POXN/MP equips itself with one fixed wavelength transponder and one tunable transponder. The total number of transponders deployed in a POXN/MP is the same as in an EPSN. Based on the architecture depicted in Figure 4.2, the price formula for POXN/MP can be computed as

$$C^O = C + C^W + C^{15} + C^X \tag{4.2}$$

where C^O represents the Capex per link of POXN/MP, C stands for the cost per port of the deployed coupler fabric, C^W represents the cost of a 2x1 WFFOC and a 1x2 AWG optical splitter, C^{15} represents the cost of an LR (1550 nm) transponder, and C^X represents the cost of an LR tunable transponder.

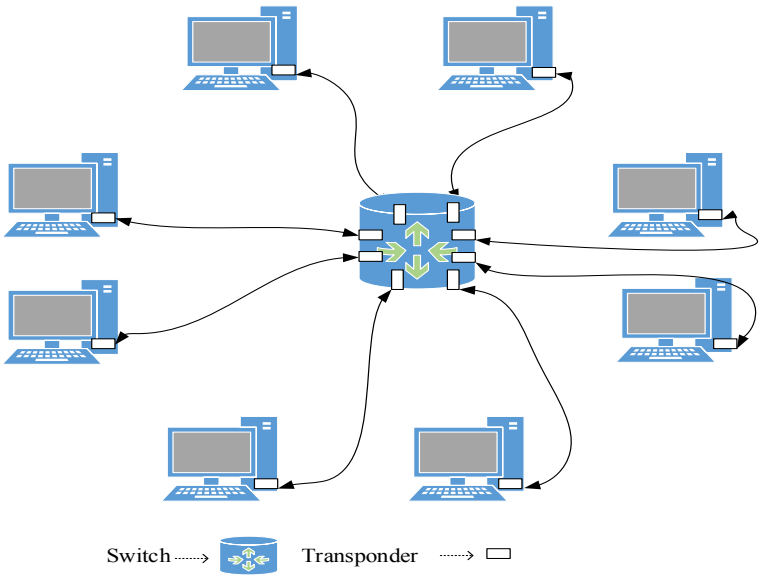

Figure 4.4 A sample of 8- port EPSN.

To illustrate the cost comparison in a real network, we use the datacenter network suggested by Cisco as an example. This datacenter network is typically constructed in the Fat Tree structure, where the popular deployed ToR switches are Cisco Nexus 3548 switches [55]. These switches have 48x10GBits SFP+ ports. We use the 48-port coupler fabric to replace the Cisco Nexus 3548 switch as an example.

The actual prices for the aforementioned elements vary among customers, depending on the quantity of equipment sold. In this model, the price of an LR (1310 nm) transponder is based on the vendor's online sale price. However, the prices of an electronic packet switch and a WDM tunable transponder are based on published Capex studies. We list the prices that are relevant to our study in Table 4.1.

Table 4.1 Price list of deployed devices.

Device	Usage	Price
Cisco Nexus 3548	Used for edge and aggregation tier switches	$21600 [34]
Passive Optical Coupler Fabric	Used for replacing edge electronic switches (48 ports)	$480
2x1 WFFOC and 1x2 AWG optical splitter	Used for combining two links into one and splitting one link into two	$40
10G LR(1310 nm) Transponder	Used for LR (1-40 km), 10G, transponder (1 wavelength)	$200 [56]
10G LR WDM Tunable Transponder	Used for LR, 10G, transponder (48 wavelengths)	$525 [4]
10G LR(1550 nm) Transponder	Used for LR (40-80 km), 10G, transponder (1 wavelength)	$350 [56]

To explore the energy-related economic benefits, we take the power consumption of the deployed devices into consideration. The coupler fabric we deployed is a passive optical device that consumes zero power, while the Cisco Nexus 3548 switch consumes 265 W [55]. Moreover, Cisco's public product data sheets [57-58] reveal that the tunable LR and fixed wavelength LR (1550 nm) transponders deployed in POXN/MP both consume 1.5 W, while the LR (1310 nm) transponder deployed in EPSNs only consumes 1 W.

Therefore, based on the prices given in Table 4.1, we can compute the Capex per link for the EPSN and the POXN/MP through equations (4.1) and (4.2), respectively. Considering the power consumption mentioned above, we summarize the facility costs and power consumption per link for these two networks in Table 4.2.

Table 4.2 Facility costs and power consumption per link for EPSN and POXN/MP.

Network	Facility Cost/link	Power Consumption/link
EPSN	$450 + 2 \times 200 = \$950$	$^{265}/_{48} + 2 = 7.52\ W$
POXN/MP	$50 + 350 + 525 = \$925$	$1.5 + 1.5 = 3\ W$

Figure 4.5 shows the resulting facility cost savings upon replacing the EPSN with the POXN/MP. As an example, we set the maximum number of ports to 81 because a coupler fabric can scale up to 81 ports without active optical amplifiers.

Table 4.2 demonstrates that the POXN/MP saves 50% power per link compared with its electronic counterpart. Figure 4.6 depicts the power consumption of the EPSN and the POXN/MP. We choose port numbers 24, 32, 48, and 64 as examples, since these port numbers are common to commercial electronic switches.

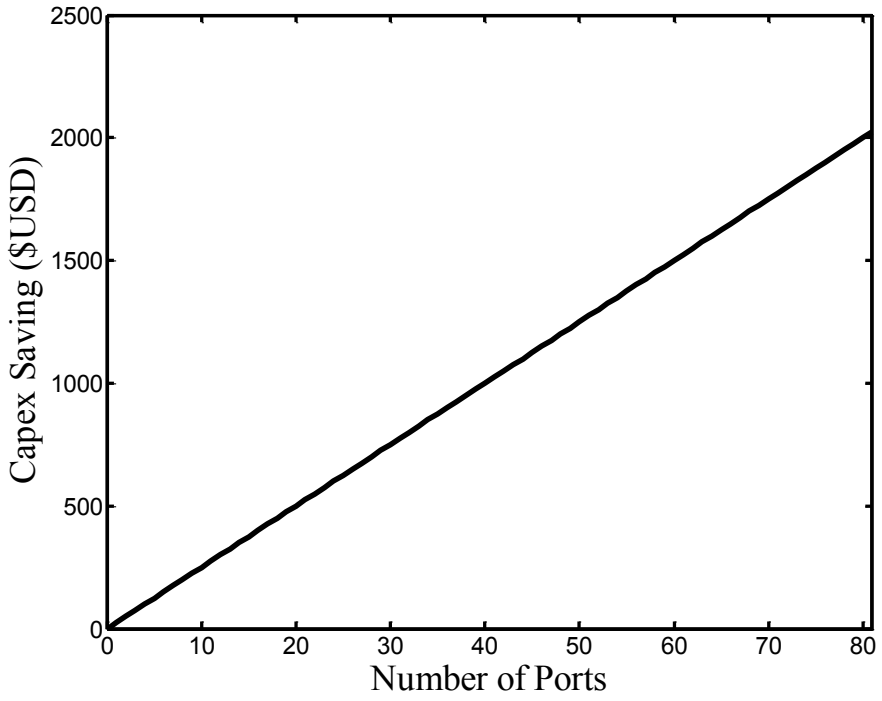

Figure 4.5 Capex savings of replacing EPSN with POXN/MP.

Notably, the transponders are primarily responsible for the cost of the POXN/MP. In addition, the cost of tunable transponders is expected to fall with commoditization and increased production volume. Many of these benefits have already been reaped for electrical technology [33]. Thus, it is reasonable to infer that the POXN/MP will be even more cost competitive if tunable transponders are produced on a massive scale.

POXN/MP does not use electrical equipment, which leads to high-energy efficiency and easier migration to 40-GigE and beyond. However, to avoid collisions among different transmitting ports at receiving ports and to optimize the unicast plane's bandwidth utilization, we propose a high-throughput MCDAP next to schedule traffic transmission among connected transmitting ports. The MCDAP also supports dynamic

traffic allocation between the two optical planes. Our system results in efficient communications for both unicast and multicast/incast traffic with a fully distributed operation.

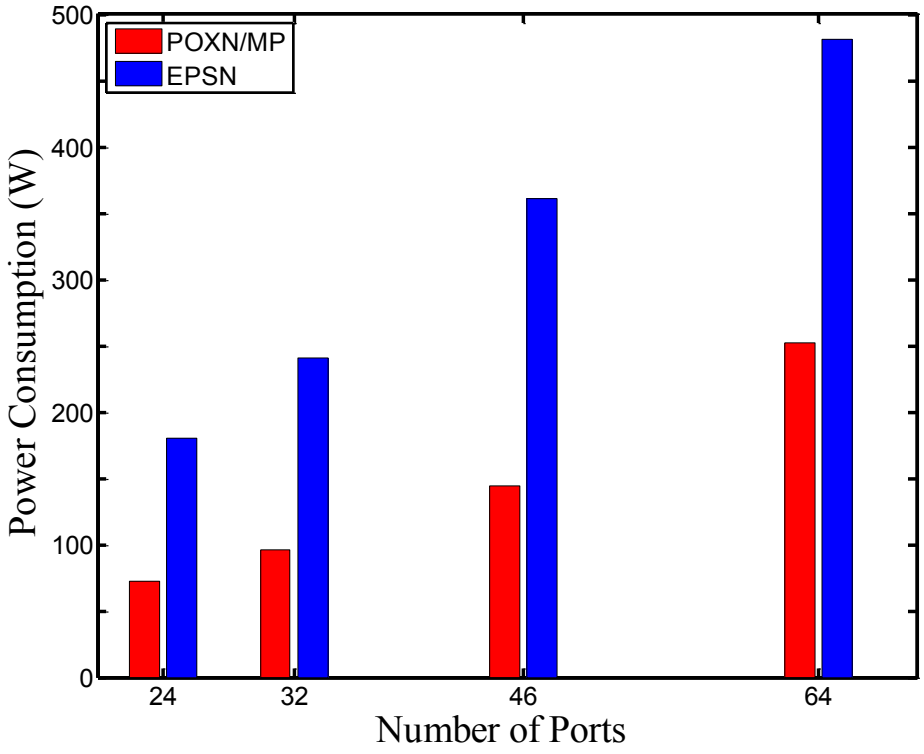

Figure 4.6 Power consumption (W) of the POXN/MP and the EPSN.

Chapter 5: Multiple Channels Distributed Access Protocol

We now describe how the MCDAP works. There are three major types of messages in the MCDAP: the control messages that are carried by the multicast plane; the unicast data messages that are mainly carried by the unicast plane but that can also be carried by the multicast plane in sequence when this plane has extra capacity; and the multicast data messages that are only carried by the multicast plane. The working process of the MCDAP can be further divided into two phases for each plane: the discovery phase and the data transfer phase for the multicast plane and the idle phase and data transfer phase for the unicast plane.

The working process of the multicast plane of the MCDAP is similar to that of the HEDAP [14]. We begin with a brief introduction about how the HEDAP works, and then we present the working process of the MCDAP in detail.

The HEDAP protocol can be divided into two phases: the discovery phase and the data transmission phase [14]. The discovery phase is designed to achieve the plug-and-play objective and to discover other ports in POXN. The data transmission phase follows the discovery phase, and it can be further divided into multiple scheduling cycles. Within each cycle, each transmitting port has a chance to send a burst of frames during its corresponding requested time interval. At the end of the burst, each transmitting port also broadcasts the amount of traffic (the requested bandwidth of the next transmission) it needs to send for the next data transmission cycle through a REQUEST message. All the receiving ports will hear the same request information for the next cycle from the other transmitting ports. Based on the same information, all the transmitting ports will follow a

common scheduling order that results from an identical algorithm. To accommodate port churns, the discovery phase is allowed to repeat after running the data transmission phase long enough. During the repeated discovery phase, new ports can be discovered, the clock can be re-referenced and resynchronized, and the round-trip time and loopback time can be re-measured.

We build the multicast plane of the MCDAP based on the HEDAP, but with several modifications, which include altering the original control information that the HEDAP broadcasts through ANNOUNCEMENT, CONFIRMATION, and REQUEST messages. We require that each transmitting port indicate the addresses for unicast traffic and the exact amount of unicast and multicast traffic to each receiving port through the multicast plane by the corresponding control messages. After the discovery phase of the multicast plane ends, each transmitting port can also generate a traffic request list of the first scheduling cycle for all the unicast channels. This list contains the amount of unicast traffic to be sent among all transmitting and receiving ports in addition to the corresponding receiving port addresses. Based on this global identical request list, each transmitting port locally runs a common scheduling algorithm. Following the results of the algorithm, each transmitting port sends its traffic without any collisions. Notably, while unicast traffic can be carried by the unicast channels in parallel, it can also be carried by the multicast plane in sequence. Thus, the MCDAP can achieve load balance between the multicast and unicast planes by dynamically allocating unicast traffic to the two planes.

Figure 5.1 illustrates the corresponding message sequence chart for the MCDAP system. The left side of the figure depicts the sequence chart for the multicast plane, while the right side presents the unicast plane's sequence chart.

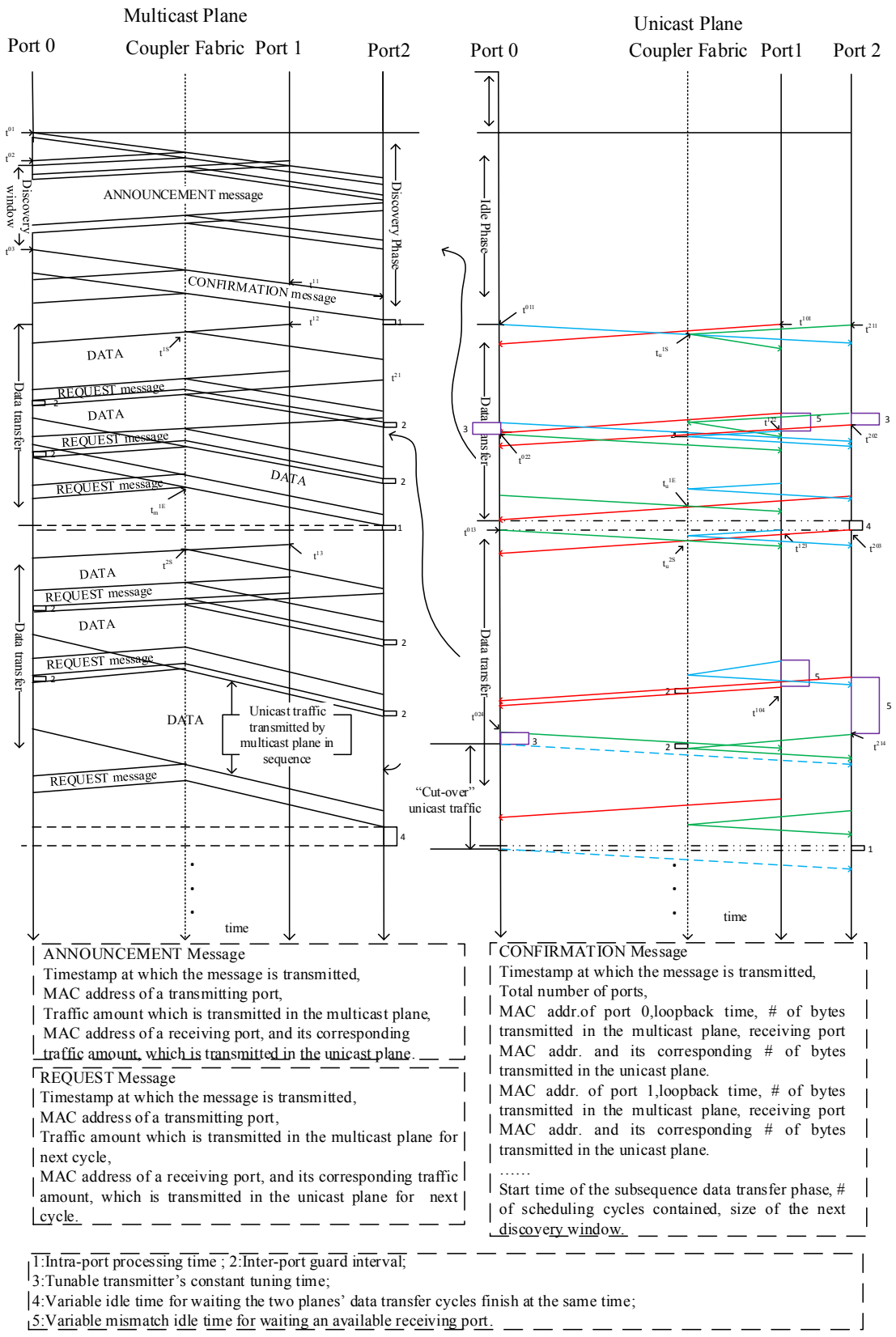

Figure 5.1 A sample message sequence chart for the MCDAP.

The discovery phase of the multicast plane is designed to achieve a plug-and-play function to reduce network operation costs. During this phase, ports in a POXN/MP will discover the other ports, set up a common reference clock, synchronize other ports' clocks to the reference clock, and calculate the round-trip and loopback time. Meanwhile, the unicast plane is in the idle phase until the end of the multicast plane's discovery phase. The data transfer phase of both the multicast and unicast planes follows immediately after the end of the discovery phase of the multicast plane, and it runs for a much longer time than the discovery phase. Every data transfer phase consists of multiple data transfer cycles. Each discovered transmitting port has a chance to send a burst of frames through the two planes simultaneously within each cycle. Moreover, the discovered transmitting ports are allowed to send their traffic bursts to different receiving ports through the unicast plane in parallel. At the end of the bursts, the transmitting ports also piggyback request messages through the multicast plane to compute the next data transfer cycle's schedules.

5.1 Discovery Phase for the Multicast Plane

There are two kinds of discovery phases for the multicast plane: the discovery phase at system boot and the discovery phase between data transfer phases.

5.1.1 Discovery Phase at System Boot

Upon system activation, all the ports begin sending an ANNOUNCEMENT message through the multicast plane after a random back-off time, which lets them be detected by the other ports in POXN/MP. The ANNOUNCEMENT message contains the mac address of the transmitting port and the time stamp of when the message was transmitted along with the discovery window time period. This message also contains information

regarding the amount of multicast traffic to be transmitted in the first data transfer cycle and a list containing the amount of unicast traffic as well as the corresponding receiving port mac addresses for the same cycle.

As a result of the broadcast property of coupler fabrics, every port hears the same information from the multicast plane. Thus, the first ANNOUNCEMENT message received at its own local receiver is also the first message successfully received at all the other ports from the multicast plane. The discovery window starts with the reception of the first ANNOUNCEMENT message. Once this period expires, no other ports are allowed to send messages. The first successfully discovered port will broadcast a CONFIRMATION message to summarize all the discovered ports. The detailed information the CONFIRMATION message carries is given in Figure 5.1.

After successfully receiving the CONFIRMATION message, each transmitting port generates two identical scheduling lists for the following data transfer cycle: one contains the amount of multicast traffic as well as its transmitting mac address for the multicast plane, and the other contains the amount of unicast traffic as well as the corresponding receiving and transmitting port mac addresses for the unicast plane. Based on these lists, each transmitting port locally runs a global identical multicast and unicast scheduling algorithm for each respective plane, which allocates time intervals for each transmitting port such that traffic collisions for both planes can be avoided.

5.1.2 Discovery Phase between Data Transfer Phases

After running the data transfer phase for the multicast plane long enough, a discovery phase follows to add new ports. During this phase, only the newly joined ports broadcast an ANNOUNCEMENT message to allow it to be discovered by the existing ports. Once

the discovery window ends, the current clock-reference port sends a CONFIRMATION message that contains information on all the existing and newly joined ports as well as information (e.g., the start time of the next discovery phase) regarding the following data transfer and discovery phases. Hence, a newly joined port must wait to receive the CONFIRMATION message, which means it must wait at least one data transfer phase before it can join the running network.

5.2 Data Transfer Phase for the Multicast Plane

An identical scheduling algorithm locally decides when a node can transmit its multicast traffic and how much it can transmit. All the discovered ports share the same wavelength for the multicast plane. Thus, each port can only send data in sequence during its assigned time interval. At the end of its transmission, every port broadcasts its traffic request for the next data transfer cycle for both the multicast and unicast planes via a REQUEST message. The information carried by a REQUEST message pertinent to the unicast plane is a list that contains the exact amount of unicast traffic for the next data transfer cycle and the corresponding transmitting and receiving port mac addresses. Moreover, to achieve load balance between the two planes, when the multicast plane has extra bandwidth capacity, transmitting ports in the MCDAP send parts of the unicast traffic through this plane in sequence.

5.3 Idle Phase for the Unicast Plane

The unicast plane begins by entering the idle phase at the same time as the multicast plane begins its discovery phase. During the idle phase, the ports must wait for the successful reception of the ANNOUNCEMENT and CONFIRMATION messages from the multicast plane. A transmitting port stores all the incoming unicast traffic from its

upper layer in the dedicated local unicast queues according to the corresponding receiving port mac addresses. The idle phase for the unicast plane ends at the same time as the discovery phase for the multicast plane, and the multicast plane eventually enters another discovery phase after running the data transfer phase for the multicast plane long enough. Meanwhile, the unicast plane enters another idle phase.

5.4 Data Transfer Phase for the Unicast Plane

Once the CONFIRMATION message has been successfully received by all the discovered ports through the multicast plane, every port uses the same CONFIRMATION message to generate two separate traffic request lists: one for the multicast plane and the other for the unicast plane. In this case, the multicast and unicast planes can compute their start times, scheduling orders, and assigned time intervals for the data transfer phase.

Examples of two 3-port POXN/MP system lists are depicted in Table 5.1 and Table 5.2. In Table 5.1, the first column represents the discovered port numbers (transmitting port addresses); the second column indicates the traffic amounts (in bits) that the corresponding port will transmit in the current data transfer cycle for the multicast plane.

Table 5.1 Traffic request list example for the multicast plane.

Transmitting Port Number	Traffic Amount (bits)
Port 0	X_0
Port 1	X_1
Port 2	X_2

Similarly, in Table 5.2, the first and third columns represent the discovered transmitting port numbers and unicast traffic amounts that must be sent in the current transfer cycle for the unicast plane, respectively. The second column represents the

corresponding receiving port addresses. Then, every port begins to run an identical algorithm locally for its multicast plane to decide when it can transmit traffic and how much traffic can be transmitted in the current data transfer cycle. At the same time, it also locally runs another identical algorithm for its unicast plane to decide when it can transmit, to which receiving port it can transmit, and how much traffic can be transmitted. Every port estimates its data transfer cycle length for the multicast and unicast planes, respectively. If there is extra bandwidth capacity for the multicast plane, the corresponding discovered ports will use the multicast plane to transmit parts of their unicast traffic in sequence to achieve load balance.

Table 5.2 Traffic request list example for the unicast plane.

Transmitting Port Number	Receiving Port Number	Traffic Amount (bits)
Port 0	Port 1	Y_{01}
Port 0	Port 2	Y_{02}
Port 1	Port 0	Y_{10}
Port 1	Port 2	Y_{12}
Port 2	Port 0	Y_{20}
Port 2	Port 1	Y_{21}

We illustrate the timer calculation of the MCDAP's working process through an example, which is depicted in Figure 5.1.

In Figure 5.1, we assume that port 1 is the first port to transmit traffic through the multicast plane. The order of transmitting traffic is decided by the scheduling algorithm. In this example, we assume that the scheduled data transmission order is port 1, port 2, and port 3 in the first data transfer cycle. The data transfer phase for the multicast plane begins with port 1 transmitting its first data burst traffic at time

$$t^{12} = t^{1S} - T_1^L / 2 \tag{5.1}$$

T_1^L is port 1's self-loopback time, which can be calculated at the discovery phase, and t^{1S} is specified in the CONFIRMATION message, which is the time when the first bit arrives at the coupler fabric for the multicast plane. At time t^{03}, when the CONFIRMATION message is transmitted, port 0 knows only its own loopback time. To allow all the discovered ports to successfully receive the CONFIRMATION message, port 0 assumes that the farthest port is 10 km from/to the coupler fabric. Here, we use T^P to denote the corresponding one-way propagation delay, where $T^P = 50$ μs. Thus, time t^{1S} can be written as

$$t^{1S} = t^{03} + 2T^P + T^C + T_0^L / 2 + T^D \tag{5.2}$$

where T^D is the transmission delay of the CONFIRMATION message, T_0^L is port 0's self-loopback time, and T^C is a constant time value set for message processing, algorithm running, etc.

Unlike the multicast plane, multiple transmitting ports may start sending their first bursts of frames to different receiving ports at the same time through the unicast plane. The start time for the unicast plane's data transfer phase can be computed as

$$t^{101} = t^{211} = t^{021} = t_u^{1S} - \frac{1}{2}\max\left\{T_0^L, T_1^L, T_2^L\right\} \tag{5.3}$$

where t_u^{1S} is equal to t^{1S}, which is specified by the CONFIRMATION message, and $\frac{1}{2}\max\{ T_0^L , T_1^L , T_2^L \}$ is the time for the last bit to propagate from the coupler fabric to the farthest port. All the discovered ports have an equal chance to transmit their data bursts to different receiving ports through different channels for the unicast plane, when the data transfer phase begins. Every port records the loopback time T_i^L and the amount of

multicast traffic and unicast traffic from REQUEST messages for the calculation of the next data transfer cycle. The ratio between the amount of multicast traffic and unicast traffic is dynamically changeable.

Thus, there may be extra bandwidth capacity for the multicast plane. To achieve load balance between the two planes, the corresponding transmitting ports will use the multicast plane to send their partial unicast traffic in sequence. However, it may be difficult to make the two planes finish their data transfer cycles simultaneously. To ensure that every discovered port has all the request information for the next data transfer cycle for each plane, an extra small guard time may be required if one of the two planes completes its data transfer cycle earlier.

The start time for port 1 to transmit traffic during the next data transfer cycle for the multicast plane can be calculated by

$$t^{13} = t^{2S} - T_1^L / 2 \tag{5.4}$$

where t^{2s} is computed locally for each port as

$$t^{2S} = t_m^{1E} + T^C + T_1^L / 2 + \frac{1}{2}\max\left\{T_0^L, T_1^L, T_2^L\right\} + t^G \tag{5.5}$$

where the time t^G is the guard time to ensure that the current data transfer cycle of each plane completes at the same time, and t_m^{1E} denotes the time at which the last bit of traffic of the last port arrives at the coupler fabric for the multicast plane. If t_m^{1E} is different from t_u^{1E}, t^G must be taken into consideration (Number 4 in Figure 5.1). The guard time can be computed locally by each port as

$$t^{G} = \left| t_{m}^{1E} - t_{u}^{1E} \right| \tag{5.6}$$

For the next data transfer cycle, multiple ports can begin transmitting traffic to different output ports at the same time through the unicast plane. The start time of the next data transfer cycle for the unicast plane can be computed as

$$t^{123} = t^{203} = t^{013} = t_{u}^{2S} - \frac{1}{2}\max\left\{T_{0}^{L}, T_{1}^{L}, T_{2}^{L}\right\} \tag{5.7}$$

where t_u^{2S} is equal to t^{2S}.

For the unicast plane, when a transmitting port finishes its data burst to a receiving port, the transmitting port may have to wait for an available wavelength before transmitting its next data burst. This variable waiting time is called mismatch idle time (Number 5 in Figure 5.1). Ideally, the first bit of the next data burst will arrive at the coupler fabric through a specific wavelength immediately following the last bit of the previous data burst so that collisions can be avoided and no extra time is wasted. This can be done because all the discovered ports know their propagation delays to the coupler fabric. If the clocks are not synchronized, an intra-port guard time may be necessary to avoid overlaps (Number 2 in Figure 5.1).

As opposed to the multicast plane, the unicast plane consists of multiple channels dedicated to different receiving ports. However, if multiple transmitting ports communicate with the same receiving port at the same time, collisions can occur. Thus, an algorithm must be designed for distributing traffic among different channels of the unicast plane as well as to optimize the bandwidth utilization.

Chapter 6: Algorithms for the Unicast Plane of the MCDAP

We now describe how the scheduling algorithms for the unicast plane of the MCDAP work. In this thesis, we assume that each transmitting port has multiple queues, with a dedicated queue for each receiving port. Every port stores its unicast traffic to a specific receiving port in its corresponding dedicated unicast queue. In addition, different unicast queues have different amounts of unicast traffic that must be sent to different receiving ports during a data transfer cycle. An example that presents the dedicated unicast queues of transmitting ports for a 3-port POXN/MP network is depicted in Figure 6.1. The vertical coordinates represent the corresponding unicast queue size, while the horizontal coordinates denote the transmitting port number. The rectangles stand for the different unicast queues, and the tags on them denote their corresponding receiving ports. For instance, transmitting port 0 has two unicast queues: one stores traffic dedicated to receiving port 1, while the other has traffic to be transmitted to receiving port 2.

In POXN/MP, a transmitting port sends traffic from different unicast queues through different wavelengths during different allocated time intervals. The wavelength used by one transmitting port for one time interval is recycled and assigned to another transmitting port during the next time interval. The receiving port then receives traffic through a single dedicated wavelength. This wavelength-port association is referred to as a run time decision.

As discussed above, today's tunable transponders can typically support tuning speeds on the 5-microsecond time scale, which is nearly four times greater than the transmission time of the maximum Ethernet frame size (1.2 μs) for a 10 Gbps link. If traffic

transmission is scheduled on a packet-by-packet basis, frequent tuning may lead to bandwidth waste of approximately 80%. Moreover, scheduling traffic on a packet level will require that packet level information be available to all transmitting ports within each cycle, which will introduce more overhead. Additionally, it costs more energy for a tunable transmitter to tune more frequently from one wavelength to another. Therefore, our top priority is to minimize tuning time.

One way to minimize tuning time is to send traffic to the same receiving port as a burst. In the following discussion, we always assume that a transmitting port will send traffic to the same receiving port continuously as a concatenated burst. It must finish sending one unicast queue completely before it can begin sending traffic from another unicast queue. When a sending port is sending to a receiving port, it will not be interrupted by other ports until it finishes. Using this approach, we minimize the impact of the tuning time and simplify our scheduling algorithms.

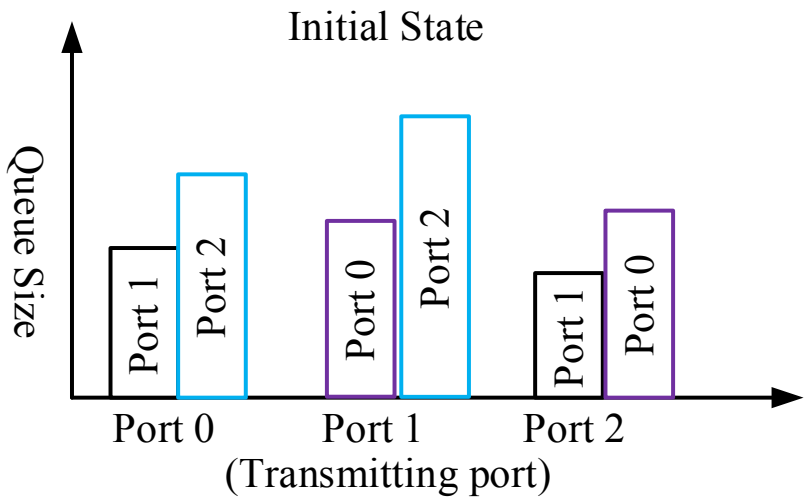

Figure 6.1 A sample unicast queue diagram of a 3-port POXN/MP.

Due to the randomness of traffic, there are several situations that may cause bandwidth to be wasted. In the first scenario, a transmitting port has finished sending all its unicast queues but must wait for other ports to finish all their unicast queues before a new cycle can start. This is called Type 1 mismatch. In the second scenario, called Type 2 mismatch, a transmitting port has unicast queues to be sent to receiving ports, but none of the receiving ports are available. Thus, the transmitting port has to wait until one of the receiving ports is available. In the third scenario, a transmitting port has traffic to send and the corresponding receiving port is available, but the transmitting port has to tune its wavelength to the receiving wavelength.

An illustrative transmission scheduling of a 3-port POXN/MP network is depicted in Figure 6.2 as an example. The rectangles marked with T_{ij}^{Q} describe different unicast queues belonging to different transmitting ports. Let T_{ij}^{Q} be the transmission time of packets in the unicast queue that will be sent from transmitting port *i* to receiving port *j* $(0 \leq i, j \leq 2)$. For example, T_{02}^{Q} denotes the transmission time of the packets in the unicast queue, which belongs to transmitting port 0 and will be sent to receiving port 2. Let t_{ik}^{Mx} indicate the *k*-th $(0 \leq k)$ Type x $(x = 1\ or\ 2)$ mismatch that transmitting port i $(0 \leq i \leq 2)$ experiences. $\sum_k t_{ik}^{Mx}$ is the total Type *x* mismatches that transmitting port *i* suffers for a data transfer cycle. The yellow rectangles marked with t_{ik}^{M1} denote the Type 1 mismatch. For instance, close to the end of the data transfer cycle, transmitting port 0 and port 2 have finished transmitting all their unicast queues, but port 1 still has data to be sent to a receiving port. In this case, transmitting port 0 and port 2 experience their first $(k = 0)$ Type 1 mismatch, t_{00}^{M1} and t_{20}^{M1}, respectively. The red rectangles marked with t_{ik}^{M2} represent the Type 2 mismatch. For example, transmitting port 2 suffers its first

Type 2 mismatch t_{20}^{M2} when it finishes its data burst to receiving port 1. In this case, transmitting port 2 becomes available and has data to be sent to receiving port 0; however, transmitting port 1 has not finished sending to receiving port 0, so transmitting port 2 has to wait until port 1 finishes. The third scenario is represented by the green rectangle, labeled T^T, which denotes a tunable transponder's constant tuning time. When transmitting port 0 finishes sending to receiving port 2, it still has data to be sent to receiving port 1. Although receiving port 1 is available to receive data, transmitting port 0 cannot start sending to receiving port 1 before it finishes tuning its wavelength to the receiving wavelength of port 1.

The Type 1 and Type 2 mismatches have to be addressed by designing scheduling algorithms that minimize the idle time caused by the mismatch of transmitting and receiving ports. The tuning time can be minimized by proactively scheduling and tuning the wavelength for a burst transmission whenever possible. Compared with the transmission time of a burst, the tuning time is typically small.

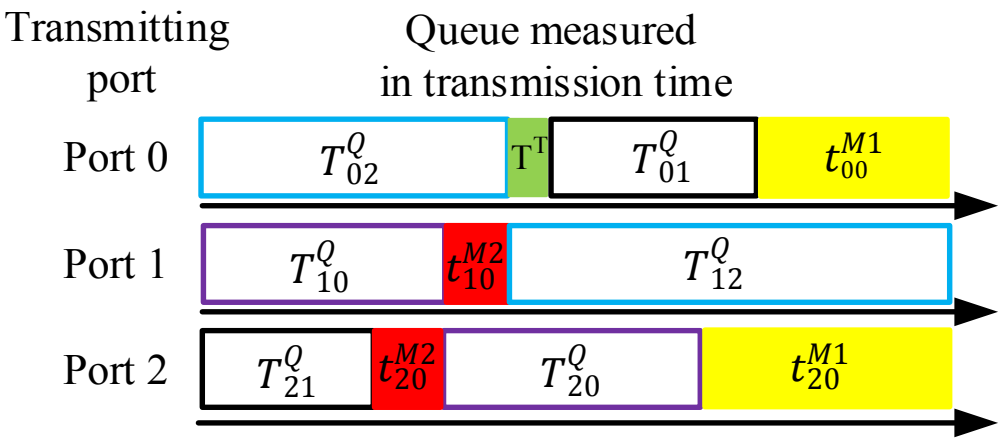

Figure 6.2 A sample transmission scheduling of a 3-port POXN/MP.

To minimize the aforementioned mismatches, we first formulate them as a mathematical programming problem and then propose several heuristic solutions in the following subsections. These solutions try to optimize traffic distribution among different wavelengths for the unicast plane and enable dynamic unicast traffic allocation.

6.1 Problem Descriptions and Formulation

The problem to be addressed can be summarized as follows: Maximize overall throughput under the constraints that traffic from one transmitting port to one receiving port must be sent in a burst, a receiving port can only receive traffic from one transmitting port and a transmitting port can only send traffic to one receiving port at any time.

We will focus on one data transfer cycle of a phase because the optimization will be performed on a cycle-by-cycle basis. For convenience of analysis, we assume that time starts from the beginning of a data transfer cycle. Given an *N* port POXN/MP network, assume that at the beginning of the data transfer cycle, the unicast queue sizes in each transmitting port are known by the MCDAP protocol described earlier. We denote these initial unicast queue sizes as T_{ij}^{Q}, where a unicast queue size is measured as the transmission time of the packets in a unicast queue, i is the index of a transmitting port and j is the index of a receiving port, and $0 \leq i, j \leq N-1$. Let $\delta_{ijt} = 1$ indicate that transmitting port i is sending to receiving port j at time t; otherwise, $\delta_{ijt} = 0$. Let t_{ij}^{S} denote the starting time of port i sending to port j and t_{ij}^{F}denote the corresponding finishing time. We assume the tuning time is negligible after proactive tuning. If the tuning time is not negligible, it can be counted as part of a data burst in the worst case. Our goal is to minimize the period of a data transfer cycle as below:

Objective:

$$\min\{\max_{i,j} t_{ij}^{F}\} \tag{6.1}$$

s.t.

$$\delta_{ijt} = 0, \forall t \le t_{ij}^{S}, i, j \le N-1, i \ne j \tag{6.2}$$

$$\delta_{ijt} = 0, \forall t \ge t_{ij}^{F}, i, j \le N-1, i \ne j \tag{6.3}$$

$$t_{ij}^{F} - t_{ij}^{S} = T_{ij}^{Q}, \forall i, j \le N-1, i \ne j \tag{6.4}$$

$$\delta_{ijt} = 1, \forall t : t_{ij}^{S} \le t \le t_{ij}^{F}, \forall i, j \le N-1, i \ne j \tag{6.5}$$

$$\sum_{j} \delta_{ijt} \le 1, \forall t \ge 0, i \le N-1, i \ne j \tag{6.6}$$

$$\sum_{i} \delta_{ijt} \le 1, \forall t \ge 0, j \le N-1, i \ne j \tag{6.7}$$

where equations (6.2) and (6.3) define the starting and ending times of bursts, equations (6.4) and (6.5) define burst lengths, equation (6.6) is the constraint imposed by a receiving port and (6.7) is the constraint by a transmitting port.

The above formulation is a complex programming problem [33, 59]. It typically takes some time for each transmitting port to process. A transmitting port can only start conducting the calculation after it receives all the requests for the next data transfer cycle. If it finishes processing before the end of the current data transfer cycle, it will not introduce any overhead. Otherwise, it will introduce extra overhead, leading to lower efficiency. Therefore, the time for this processing should be much smaller than a data transfer cycle, which is typically less than 1 ms for highly dynamic traffic patterns. In our numerical examples given later, the period of a data cycle can be as small as 150 μs. To complete the optimization process in such a brief time is almost impossible unless a

significant amount of computing power is deployed. Thus, it is necessary to look at heuristic solutions.

6.2 Shortest Queue First Algorithm

One of the main empirical studies of datacenter traffic characteristics shows that flow sizes of 80% of the datacenter traffic are considerably small (i.e., less than 10 KB). However, less than 20% of the total bytes are contributed by a few large flows [5]. Based on these observations, we assume that at the beginning of a data transfer cycle, the shorter unicast queues belonging to different transmitting ports may resemble each other in size. If each transmitting port tries to send its shortest unicast queue with an available receiving port first, a considerable number of transmitting ports will complete their data bursts at roughly the same time. This will allow these transmitting ports to recycle the wavelengths to other receiving ports at roughly the same time and, therefore, minimize Type 2 mismatch. To be consistent, when two transmitting ports try to send to the same receiving port, the transmitting port with the smaller unicast queue will send first. We name this algorithm Shortest Queue First (SQF).

As an example, we still use the aforementioned 3-port POXN/MP network to demonstrate the transmission process of the SQF algorithm. The high-level diagram of the transmission process of the SQF algorithm is depicted in Figure 6.3. To be consistent, we use the same notations as in Figure 6.2. Moreover, let t_{ijt}^{Q} denote the transmission time of the remaining packets in the unicast queue that will be sent from transmitting port i to receiving port j $(0 \leq i, j \leq 2)$ at time t. For instance, $t_{01t_1}^{Q}$ denotes the transmission time of the remaining packets in the unicast queue that will be sent from transmitting port 0 to receiving port 1 at time t_1.

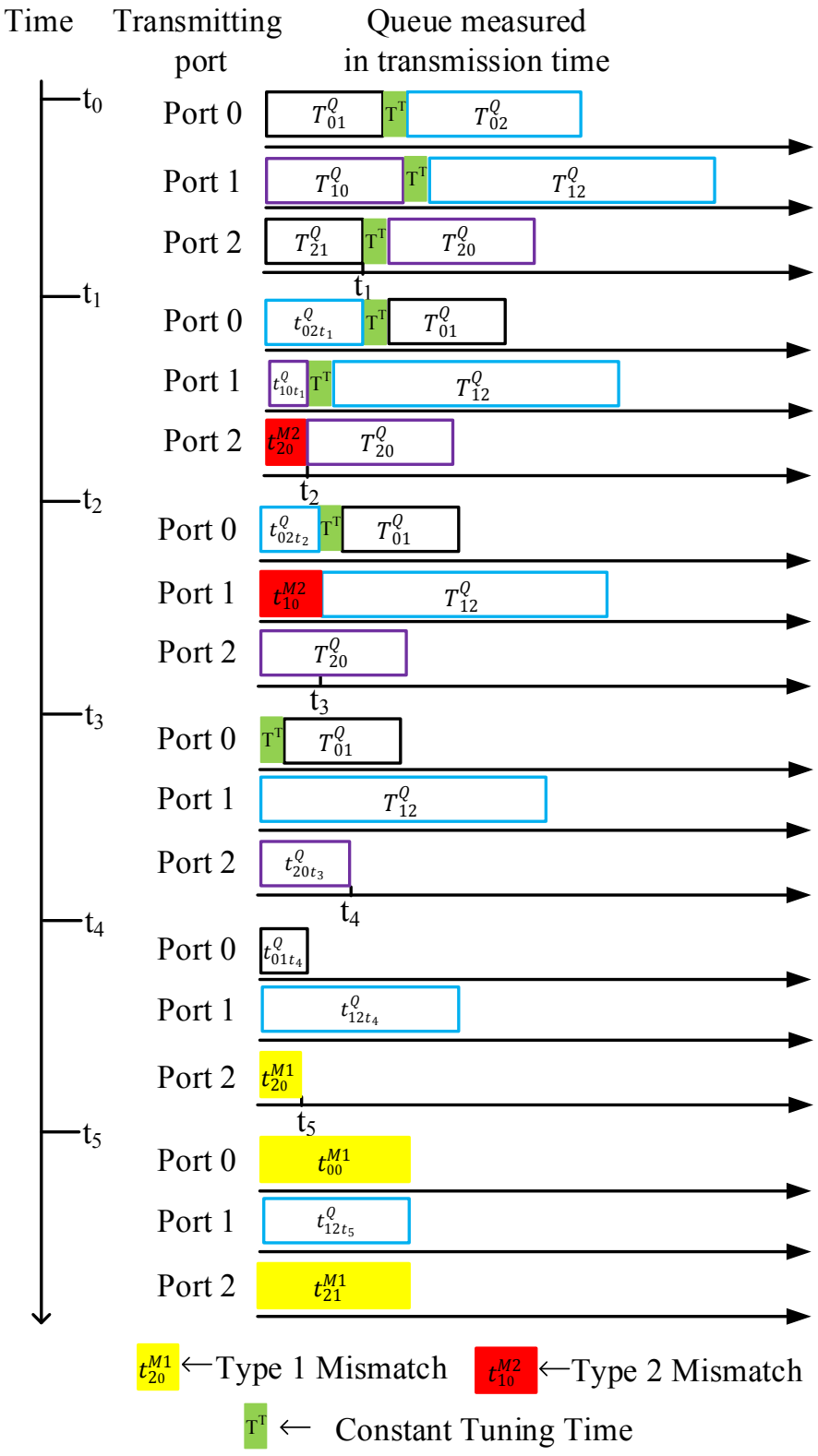

Figure 6.3 Transmission process of a 3-port POXN/MP using the SQF algorithm.

Figure 6.3 depicts snapshots of the remaining unicast queues of all transmitting ports at different times. The time coordinates on the left side of the figure denote the times when ports begin transmitting to the desired receiving ports (which correspond to events occurring in the event list). The unicast queue measured in transmission time coordinates describe the status of the remaining unicast queues for all transmitting ports corresponding to different event times. We assume that each transmitting port orders its unicast queues according to their sizes. The port coordinates denote the transmitting ports.

The SQF algorithm consists of the following steps:

Step 1: At the beginning of each cycle, each transmitting port will try to select the shortest unicast queue to send first. If there is a collision with other transmitting ports, the transmitting port with the shortest unicast queue size will win, and all other transmitting ports that have collisions will try their second-smallest unicast queues with different receiving ports. This process continues until a transmitting port finds the non-colliding shortest unicast queue; otherwise, the transmitting port will be idle and wait for a receiving port to become available. A transmitting port can figure out collisions beforehand because every transmitting port has the traffic burst information, learned through the MCDAP, about all other transmitting ports.

In this example, at the beginning of the data transfer cycle (t_0), transmitting port 2 selects its unicast queue 1 to send first. Simultaneously, transmitting port 0 selects its unicast queue 2, although unicast queue 1 is smaller. Meanwhile, transmitting port 1 selects its unicast queue 0 as its first data burst. It is assumed that transmitting port 2 will complete its data transmission of packets in unicast queue 1 at t_1 .

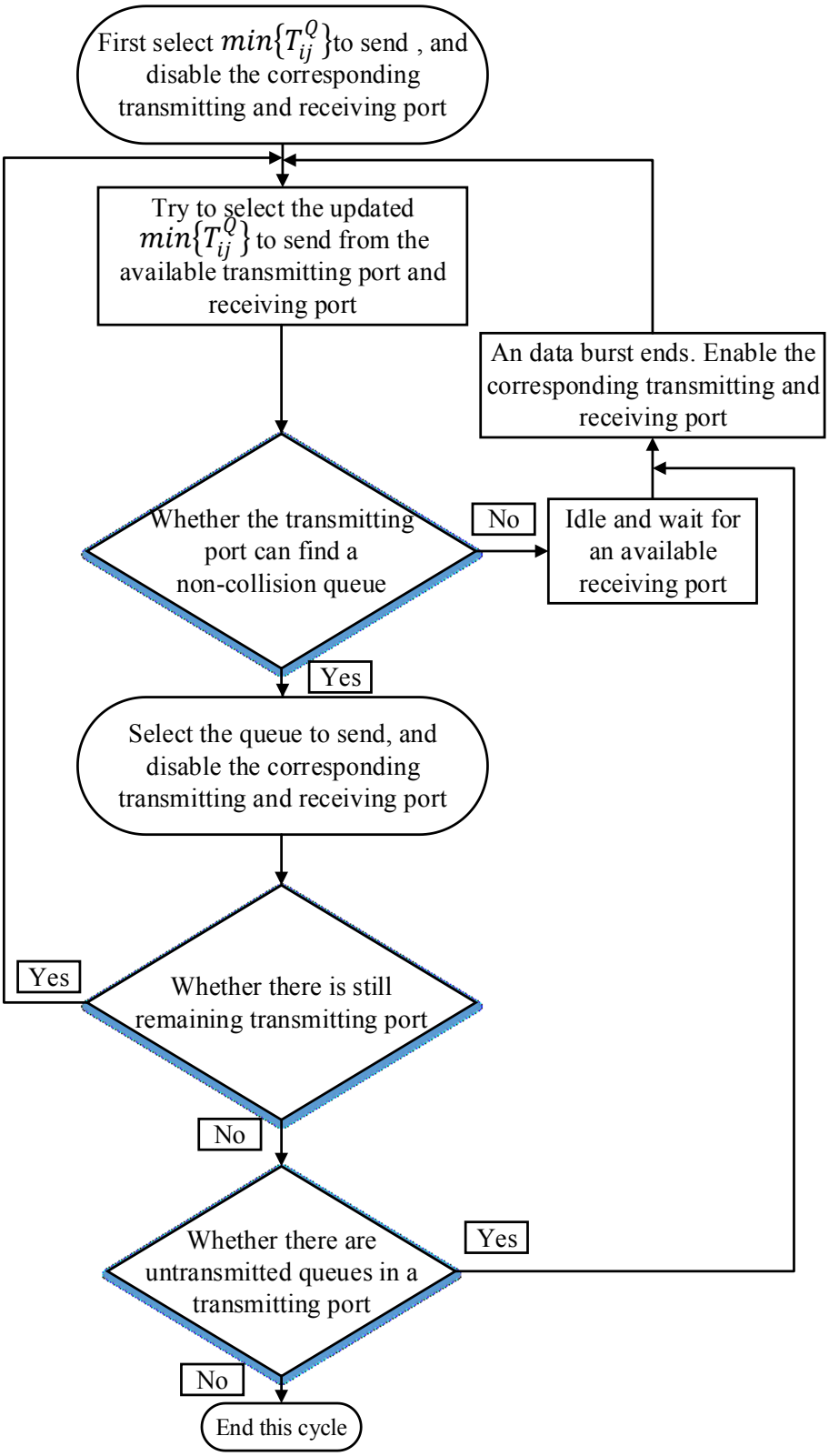

Figure 6.4 SQF algorithm flow chart.

Step 2: Once a transmitting port has completed the data transmission for its first selected unicast queue, it begins to select its next shortest unicast queue to an available receiving port as its second data burst. If there is no available receiving port at this moment, the transmitting port will be idle and wait for an available receiving port.

As depicted in Figure 6.3, transmitting port 2 will first finish sending traffic from its selected unicast queue 1 to receiving port 1 for this data transfer cycle at t_1. Then, transmitting port 2 only has unicast queue 0 that needs to be sent to receiving port 0. However, at this moment, receiving port 0 is not available. Therefore, transmitting port 2 will be idle and suffer from Type 2 mismatch from t_1 until transmitting port 1 completes its data burst to receiving port 0 at t_2, where $t_2 = t_1 + t^Q_{10t_1}$. During this idle time, transmitting port 2 can start tuning to the wavelength of receiving port 0 so that the tuning time does not incur extra overhead. This is what we call proactive tuning.

Step 3: Following the same rule as mentioned in step 2, all the transmitting ports complete their corresponding data bursts for the current data transfer cycle.

This step is illustrated from t_2 to t_5. At t_2, transmitting port 1 completes its remaining data burst to receiving port 0. As a result, receiving port 0 is available and transmitting port 2 can begin its next data burst to receiving port 0 immediately. Meanwhile, transmitting port 1 only has unicast queue 2 that needs to be sent to receiving port 2. However, at this moment, receiving port 2 is busy, so transmitting port 1 experiences Type 2 mismatch from t_2 until transmitting port 0 completes its data burst to receiving port 2 at t_3. Transmitting port 1 can certainly take this opportunity to tune its wavelength to receiving port 2 as an act of proactive tuning.

At t_3, transmitting port 0 finishes its data burst to receiving port 2, so receiving port

2 is available and transmitting port 1 can send its unicast queue 2 to receiving port 2. Meanwhile, receiving port 1 is also available; however, transmitting port 0 must wait for a constant tuning time T^T to tune its tunable transmitter to receiving port 1. Moreover, transmitting port 2 will complete its remaining data burst to receiving port 0 at t_4.

At t_4, transmitting port 2 completes all its data bursts for this data transfer cycle. Then, transmitting port 2 becomes idle. Thus, transmitting port 2 suffers from the Type 1 mismatch from t_4 until t_5. At t_5, transmitting port 0 completes its data burst and experiences Type 1 mismatch until transmitting port 1 finishes at the end of the current data transfer cycle. A flow chart of the corresponding SQF algorithm is depicted in Figure 6.4, where $min\{T_{ij}^Q\}$ denotes $\min_{i,j}\{T_{ij}^Q\}$.

6.3 Longest Queue First Algorithm

When we employed the SQF scheduling algorithm, we observed that the unicast queues belonging to the last completed transmitting ports tend to have a larger average unicast queue size at the end of a data transfer cycle. Moreover, during this period, some transmitting ports have already completed their data transmissions and are now waiting for the remaining un-finished transmitting ports. This will increase Type 1 mismatch. To minimize this mismatch, we propose another approach in which each transmitting port will try to send the longest unicast queue with an available receiving port first during the data transfer cycle. This algorithm is named Longest Queue First (LQF).

Again, we take the aforementioned 3-port POXN/MP network as an example to illustrate the transmission process of the LQF algorithm. The only difference with the SQF algorithm is that each transmitting port first tries to send traffic from the longest unicast queue to an available receiving port. Intuitively, compared with the SQF

algorithm, even though the LQF algorithm can reduce the Type 1 mismatch at the end of the data transfer cycle but tends to suffer from more Type 2 mismatches at the beginning of a cycle. Thus, there may not be a significant performance increase for the LQF algorithm. The high-level diagram of the transmission process for the LQF algorithm is depicted in Figure 6.5, which uses the same notations as Figure 6.3.

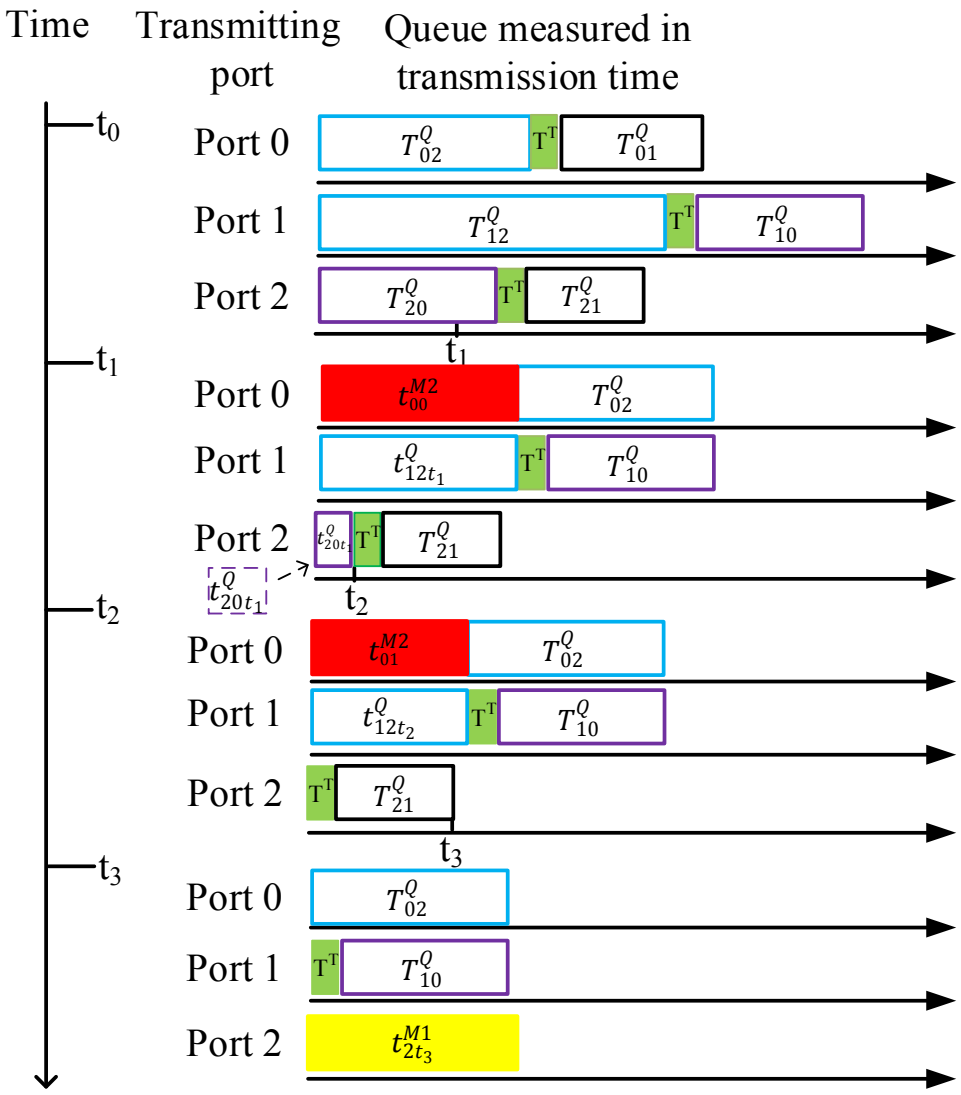

Figure 6.5 Transmission process of a 3-port POXN/MP using the LQF algorithm.

These two algorithms show that transmitting ports in the POXN/MP can, without collisions, send unicast traffic in parallel to different receiving ports through the unicast plane. The tuning time can be minimized through proactive scheduling. The SQF algorithm can reduce the impact of Type 2 mismatch. However, it experiences more Type 1 mismatches at the end of a data transfer cycle. In contrast, the LQF algorithm is helpful in minimizing the impact of Type 1 mismatch to some extent, but it suffers from more Type 2 mismatch. To minimize both types of mismatch and to achieve load balance between the two optical planes, we propose a third algorithm.

6.4 SQF with Cut-over Function

Another method to address the Type 1 mismatch toward the end of each cycle is to utilize the two optical planes dynamically. Before the end of each cycle, the last transmitting ports will use both the multicast plane and the unicast plane to send their unicast traffic from un-transmitted unicast queues whenever the multicast plane has extra bandwidth capacity. Therefore, the wasted bandwidth caused by the Type 1 mismatch can be minimized. Moreover, in the MCDAP, the two planes must start their next data transfer cycles at the same time. If the multicast plane finishes its data transfer cycle earlier, it will be idle and wait for the unicast plane to complete, wasting bandwidth for the multicast plane. Thus, this method can also optimize bandwidth utilization for the multicast plane by sending parts of unicast traffic to align the two planes when the multicast plane has extra capacity. We achieve this by designing a function named cut-over, which can dynamically distribute traffic between the two planes. The cut-over function is helpful to optimize overall system bandwidth utilization.

To elaborate on the cut-over function, it is necessary to consider the multicast plane.

We assume that each transmitting port also has a dedicated multicast queue, which is used to store multicast packets sent by the multicast plane (including broadcast packets). The cut-over function can be achieved according to the following operations:

First, every transmitting port estimates the period of its current data transfer cycle for both the multicast and unicast planes, which can be done by running the HEDAP's scheduling algorithm for the multicast plane based on the multicast traffic burst information and the SQF algorithm for the unicast plane based on the unicast traffic burst information. Then, each transmitting port will know its scheduling order and assigned time intervals during the current data transfer cycle for the unicast and multicast planes. Finally, based on the scheduling order, each transmitting port computes the data transfer cycle period for each of the two planes.

Second, each transmitting port compares these two data transfer cycle periods. When the data transfer cycle for the unicast plane is longer, the corresponding transmitting ports move a certain amount of unicast traffic from its unicast queues to the multicast queue. This may result in the order of data transmission for a unicast queue being different. Therefore, out-of-order delivery may occur for some packets. The out-of-order delivery problem can be addressed by running a hash function in every transmitting port, similar to Equal-Cost Multi-Path Routing (ECMP) [60]. The amount of unicast traffic should try to make the two planes finish their cycles at nearly the same time. Moreover, this unicast traffic is stored in the unicast queues that are scheduled to be sent during the end of the data transfer cycle. We name this amount of unicast traffic cut-over traffic.

Third, each transmitting port updates the scheduling order for the two planes. The cut-over unicast traffic will be sent by the multicast plane in sequence immediately after the completion of the original multicast traffic transmission.

After running the cut-over function, each transmitting port follows the new scheduling order so that the two planes will finish at nearly the same time. An example of the transmission process for the aforementioned SQF algorithm with the cut-over function is illustrated in Figure 6.6. As discussed in Chapter 4, the multicast plane works at a shared wavelength. Thus, in Figure 6.6, we add an extra port marked Multicast plane to denote the multicast plane. Let T_i^Q be the transmission time of multicast packets in the multicast queue that will be sent from transmitting port *i* to all the receiving ports through the multicast plane $(0 \leq i \leq 2)$. For instance, T_1^Q denotes the transmission time of multicast packets that will be sent from transmitting port 1 to all the receiving ports through the multicast plane.

Figure 6.3 shows that without the cut-over function, transmitting ports 0 and 2 finish their data bursts for this data transfer cycle at t_4 and t_5 , respectively. Then, they begin to suffer from Type 1 mismatch after their completion until transmitting port 1 completes its data burst to receiving port 2. This results in a waste of bandwidth for the unicast plane. Under the same circumstance, in Figure 6.6, we assume that the multicast plane completes its data transfer cycle before transmitting port 2 finishes its data burst to receiving port 1 through the unicast plane. According to the MCDAP, in this example, the multicast plane will be idle until transmitting port 1 completes its data burst to receiving port 2, leading to significant bandwidth wastage for the multicast plane.

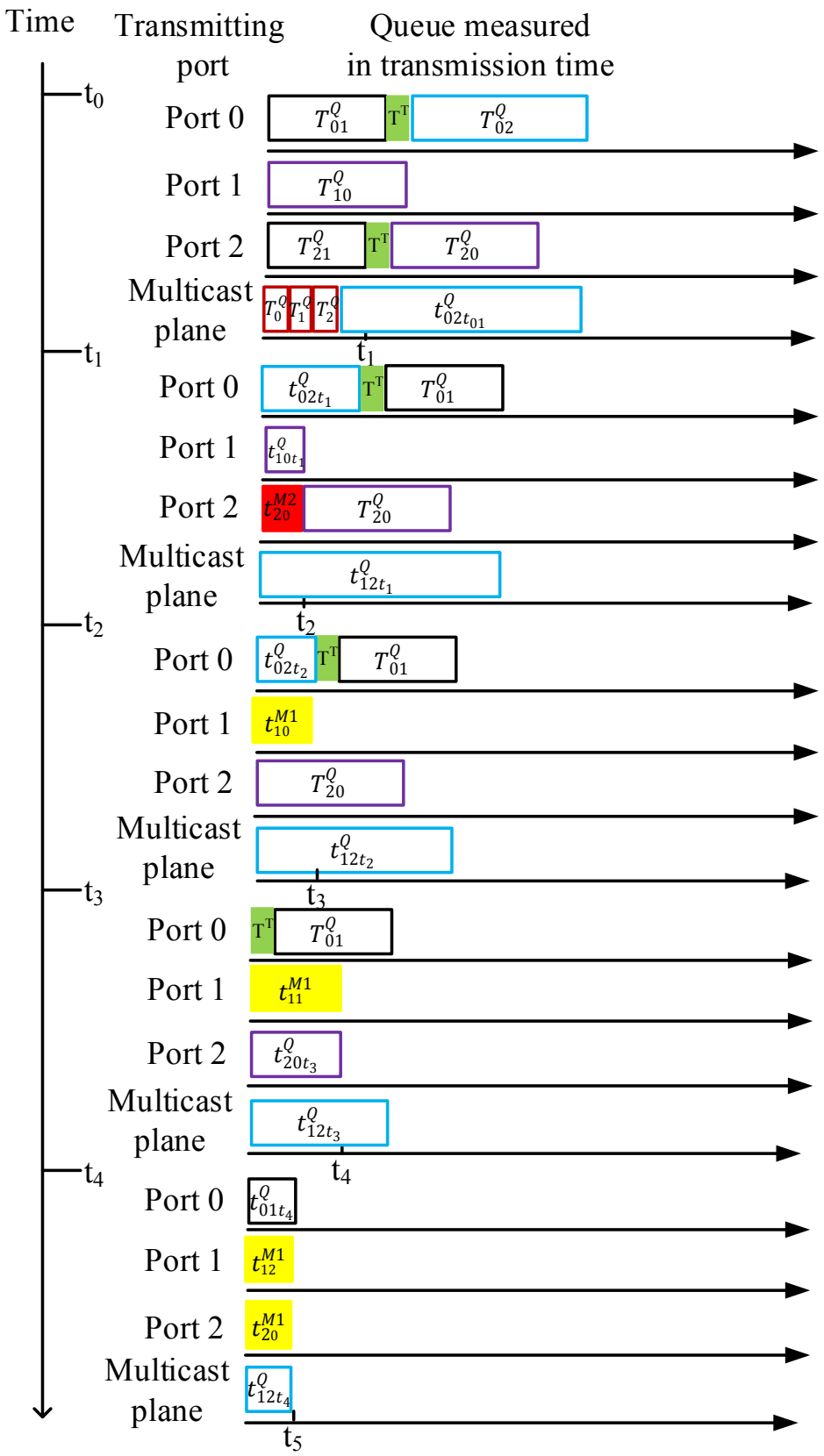

Figure 6.6 Transmission process of a 3-port POXN/MP using the SQF/CF algorithm.

To minimize the wasted bandwidth for the two planes, each transmitting port runs the abovementioned cut-over function locally. In this example, through the first step of the cut-over function, transmitting port 1 discovers the scheduling orders and assigned time intervals for all the transmitting ports. From the second step, transmitting port 1 finds that the multicast plane has extra capacity for this data transfer cycle. Thus, transmitting port 1 moves its desired unicast packets to receiving port 2 from the corresponding unicast queue to the multicast queue. Finally, instead of using the unicast plane to transmit the cut-over traffic from transmitting port 1 to receiving port 2, the cut-over traffic is sent through the multicast plane immediately after transmitting all the multicast traffic. In this example, the amount of cut-over traffic would try to make the multicast and unicast planes finish their data transfer cycles at t_5. Thus, it can be seen that the cut-cover function not only minimizes the Type 1 mismatch for transmitting ports 0 and 2 but also reduces the wasted bandwidth for the multicast plane. Additionally, the cut-over function enables dynamic traffic allocation between the two planes. We name this new algorithm Shortest Queue First with Cut-over Function (SQF/CF).

Chapter 7: Numerical Results

We evaluate the MCDAP in two parts through simulations. In the first part, we focus only on the efficiency of the proposed algorithms for the unicast plane (the numerical results are presented in section 7.1). In this thesis, efficiency refers to the ratio of the mean transmission time of unicast packets for a transmitting port divided by the period of a data transfer cycle for the unicast plane. In the second part, we implement the MCDAP protocol using the best suitable algorithm, and we test the performance of our protocol at a system level.

7.1 Algorithm Efficiency Analysis

We evaluate our proposed algorithms over a 48-port POXN/MP system through simulations. The port number 48 was chosen as an example because this port number is common to the widely used electronic switches in datacenters. In the setup environment, all 48 transmitting ports work at a 10 Gb/s wavelength line rate. For simplicity, we assume that the number of packets in a dedicated unicast queue follows a uniform distribution. Moreover, simulations in section 7.1 were achieved based on burst-by-burst basis as the descriptions of proposed algorithms in Chapter 6. In this thesis, the number of packets in a queue also denotes the corresponding queue length. Every packet has an equal size of 1024 bytes. This size of 1024 has been used to ensure consistency with the setup parameters in the HEDAP [14]. In addition, the tunable transmitter has a constant tuning time that is equal to the transmission time of 6 packets. We use 6 because the corresponding transmission time is nearly equivalent to 5 μs. All simulation results shown are with 95% confidence intervals.

An idealized result is calculated as the upper bound for the algorithm efficiency under the following assumptions: any transmitting port has the same amount of unicast traffic to be sent to all other receiving ports during a data transfer cycle; except for the constant tuning time, there are no Type 1 or Type 2 mismatches between any two adjacent scheduled data bursts; and all the transmitting ports start and finish their data bursts at the same time. Thus, for an *N*-port POXN/MP, the upper bound efficiency can be computed as

$$E = \frac{(N-1)\times T_{ij}^{Q}}{(N-1)\times T_{ij}^{Q} + (N-2)\times T^{T}} \tag{7.1}$$

where T^T denotes a tunable transponder's constant tuning time, T_{ij}^{Q} denotes the transmission time of packets in a dedicated unicast queue that will be sent from transmitting port *i* to receiving port *j*. In this case, all the unicast queues have the same queue length. When we compute the upper bound of each case, the value of T_{ij}^{Q} is set to the maximum queue length of a corresponding uniform distribution for simulations in section 7.1. Although this upper bound is looser than the result calculated using the formulation in section 6.1, it is more feasible in real system. As we will show later, the results of our best scheduling algorithm can be quite close to the upper bound.

From the above discussions, it can be shown that three factors mainly affect the efficiency of our proposed algorithms. The first is the relative overhead introduced by the tuning time, which is mainly affected by the mean of the unicast queue length. The second is the Type 2 mismatch, which is mainly influenced by the variation of the unicast queue length. The third factor is the Type 1 mismatch, which is mainly affected by the transmission time of the multicast traffic.

Following this, we study how the mean of the unicast queue length, the variation of the unicast queue length, and the transmission time of the multicast traffic affect the algorithms' efficiency. This can be tested through three different scenarios.

7.1.1 Mean of the Unicast Queue Length

In the first scenario, the unicast queue length follows a uniform distribution. Moreover, the mean of the unicast queue length is different, increasing from 45 to 65 in increments of 5. However, the standard deviation of each unicast queue length is set to 14.43. In addition, the multicast queue length for each transmitting port is set to 5. We will study the algorithm efficiency of different multicast queue lengths in a later subsection. We test the efficiency of the aforementioned algorithms and depict the results in Figure 7.1. Their results are compared with the upper bound.

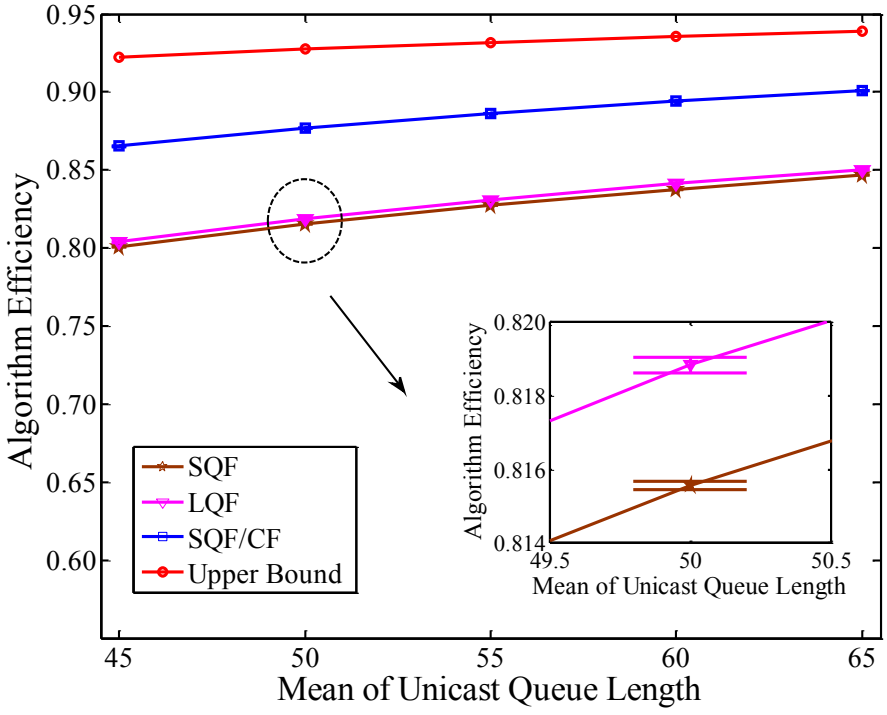

Figure 7.1 Algorithm efficiency for SQF, LQF and SQF/CF with different means of unicast queue length but a constant standard deviation of unicast queue length. Simulation results shown are with 95% confidence intervals.

In Figure 7.1, the horizontal coordinates denote the mean of the unicast queue length. The vertical coordinates denote algorithm efficiency. The integrated zoom-in figure in Figure 7.1 is used to show the confidence interval. To keep the other figures orderly, we only present the zoom-in figure in Figure 7.1.

Figure 7.1 shows how the algorithm efficiency changes with the mean of the unicast queue length. We see that when the mean of the unicast queue length becomes larger, the efficiency for all the proposed algorithms increases. This is due to the impact of the tuning time becoming smaller. Consequently, the algorithm efficiency will be higher. Figure 7.1 also shows that there is no significant difference between the efficiency of the LQF and SQF algorithms. This is most likely due to the tradeoff between Type 1 and Type 2 mismatches.

However, there is approximately an 8% relative increase in algorithm efficiency after adding the cut-over function compared with the LQF and SQF algorithms. This is a result of shortening the Type 1 mismatch at the end of a data transfer cycle. Moreover, Figure 7.1 shows that the efficiency of the SQF/CF algorithm under this simulation environment approaches the upper bound, with an approximately 7% relative difference when the mean of the unicast queue length is set to 45. Thus, there will be little practical gain on computing the bound using the optimization formulation in section 6.1.

7.1.2 Variation of the Unicast Queue Length

In the second scenario, the unicast queue length still follows a uniform distribution. Moreover, the mean of the unicast queue length is set to 65, which is the rightmost point in Figure 7.1. However, the standard deviation of each unicast queue length increases from 8.66 to 20.21 in increments of 2.88. In addition, the multicast queue length for each

transmitting port is set to 5. We test the efficiency of the aforementioned algorithms and depict the results in Figure 7.2. Their results are compared with the upper bound again.

In Figure 7.2, the horizontal coordinates represent the standard deviation of the unicast queue lengths. The vertical coordinates represent the algorithm efficiency.

Similar results have been observed in the second scenario. There is nearly no efficiency difference between the SQF and LQF algorithms. Moreover, the cut-over function can increase algorithm efficiency by nearly 9% when the standard deviation of the unicast queue length is 20.

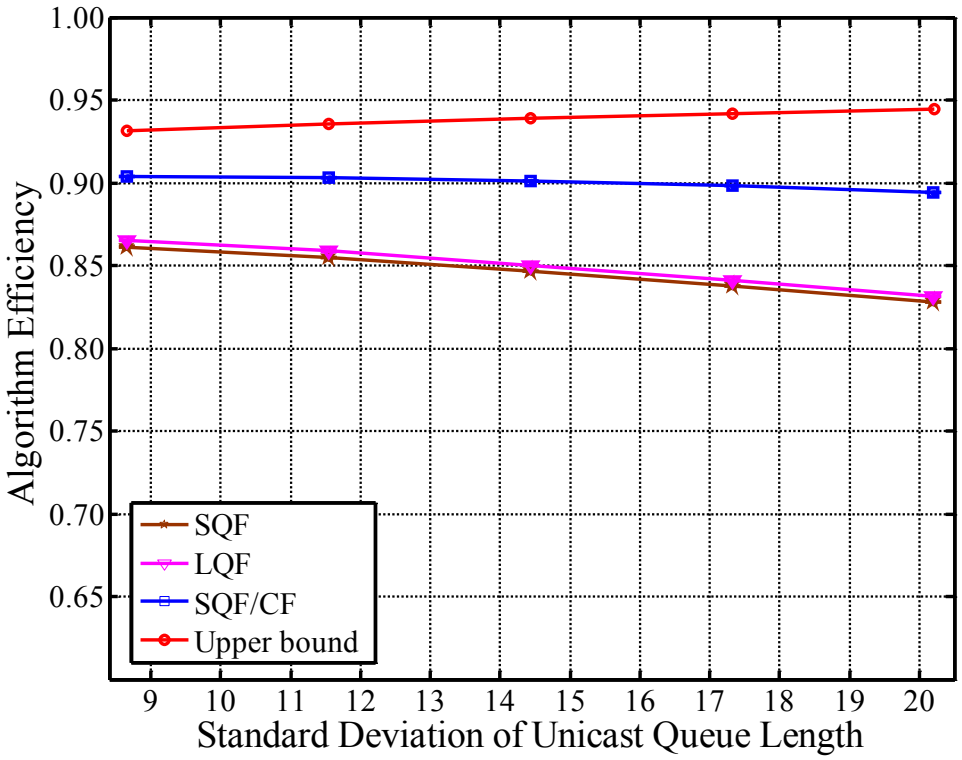

Figure 7.2 Algorithm efficiency for SQF, LQF, and SQF/CF with a constant mean of unicast queue length but an increasing standard deviation of unicast queue length. Simulation results shown are with 95% confidence intervals.

Figure 7.2 shows how the algorithm efficiency changes with the variation of the

unicast queue length. When the variation of the unicast queue length becomes larger, the efficiency for all the proposed algorithms decreases. This is because when the variation of the unicast queue length is growing larger, both Type 1 and Type 2 mismatches will increase. Therefore, the algorithm efficiency will decrease when the variation of the unicast queue length increases.

7.1.3 Transmission Time of Multicast Traffic

In the third scenario, the unicast queue length still follows a uniform distribution. We set the mean and standard deviation of the unicast queue length to 45 and 14.43, respectively. In addition, the multicast queue length of a transmitting port increases from 5 to 25 in increments of 5. In this case, we test the efficiency of the SQF/CF algorithm and depict the result in Figure 7.3. The result is compared with the upper bound again.

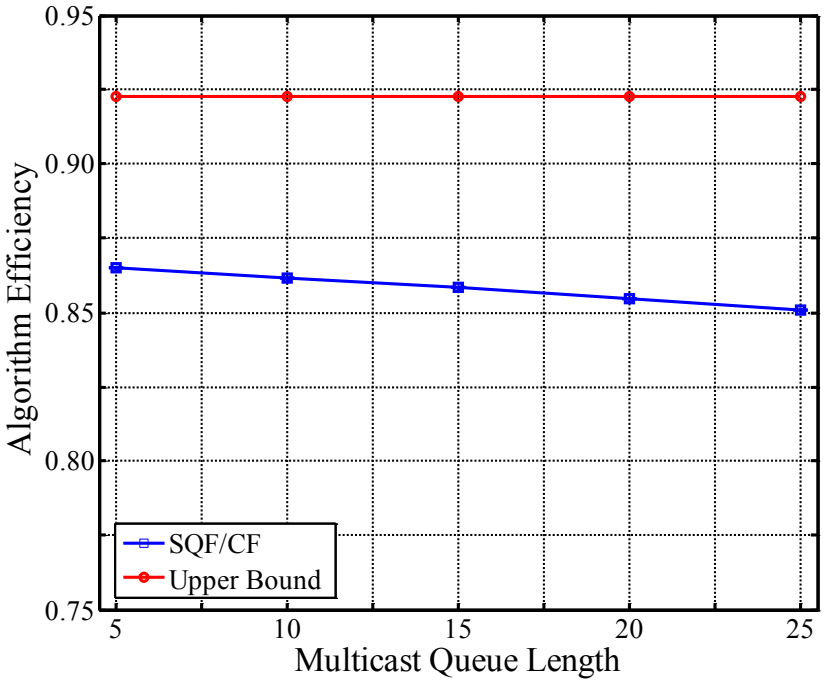

Figure 7.3 Algorithm efficiency for the SQF/CF algorithm. Simulation results shown are with 95% confidence intervals.

In Figure 7.3, the horizontal coordinates represent the multicast queue length for a transmitting port. The vertical coordinates represent the algorithm efficiency. Figure 7.3 shows how the algorithm efficiency changes with the transmission time of multicast traffic. When the multicast queue length is set to 25, the efficiency of the SQF/CF algorithm has an approximately 8% relative difference compared with the upper bound. We see that the efficiency of the SQF/CF algorithm decreases when the transmission time of the multicast traffic increases. This is because there will be less cut-over unicast traffic that can be sent by the multicast plane when the multicast plane has more multicast traffic to transmit. In this case, less reduction in Type 1 mismatch will occur, and the algorithm efficiency will decrease correspondingly.

7.2 System Throughput and Packet Delay Analysis

In this subsection, in order to analyze the MCDAP's performance at the system level (at which we take the MCDAP overhead into consideration.), we evaluate our protocol using the SQF/CF algorithm over an 8-port POXN/MP network using the OPNET Modeler. All the ports are connected to the same coupler fabric by a pair of 1 km fibers. We chose 8 to ensure consistency with the simulation environment in the HEDAP [14]. Moreover, simulations in this subsection were undertaken based on packet-by-packet basis. It would be rather complicated and infeasible to simulate a 48-port POXN/MP as in previous subsection. Thus, considering the simulation time, we chose 8-port POXN/MP in the following simulations.

We set up the values of the simulation parameters based on the existing EPON technology, and we employ the same setup environment as in the HEDAP [14]. It is assumed that all the transponders operate at 10 Gb/s. The values set for the system

simulation parameters are given in Table 7.1. In this simulation model, we assume that the frame arrival rates are the same for all the transmitting ports. All the control messages (i.e., ANNOUNCEMENT, CONFIRMATION, and REQUEST messages) are 128 bytes in length. All the aforementioned parameters are deterministic without variation. The discovery phase is triggered every 20000 scheduling cycles.

We first calculate the theoretical upper bounds for the efficiencies of both the HEDAP and the MCDAP. These upper bounds can be used as guidance to evaluate the performances of the two systems under more-realistic traffic scenarios. To this end, the upper bounds are calculated by considering overheads caused by protocols and physical constraints only. To be consistent with the computation in POXN [14], we set the maximum data transfer cycle to 144.96 μs for both the HEDAP and the MCDAP.

The overhead for the HEDAP consists of an inter-port guard interval, the transmission delay of REQUEST message T^D, twice 1-km propagation delays from the coupler fabric to a transmitting port, and a constant processing time T^C, as shown in Figure 5.1. Based on the parameters in Table 7.1, it can be calculated that the overhead time for the HEDAP within a data transfer cycle is $12.11\ \mu s$. This leads to a maximum system efficiency of 1-12.1114/144.96=0.916. Because ports can only send sequentially in the HEDAP, the maximum per-port efficiency will be 0.115.

The overhead for the unicast plane of the MCDAP is composed of an inter-port guard interval and twice 1-km propagation delays from the coupler fabric to a transmitting port, as shown in Figure 5.1. It can be calculated that the maximum per-port overhead for the unicast plane is $12\ \mu s$. Therefore, the per-port maximum efficiency for the unicast plane will be 0.917. Each port of the unicast plane of the MCDAP has approximately 8 times

the efficiency of the port in the HEDAP.

Table 7.1 Time Specification for each operation period.

Operation period	Time
Transmission time for control messages T^D	0.1014 μs
Inter-port guard interval T^I , which includes laser off and on, automatic gain control (AGC), clock and data recovery (CDR), and code-group alignment intervals [14]	2 μs
Intra-port processing time T^C	10 ns
Worst-case propagation delay from a port to the coupler fabric or from the coupler fabric to a port	5 μs
Inter-frame gap	9.6 ns
Second moment of the per-frame service time	0.3615 $(\mu s)^2$
Tunable transponder's constant tuning time	4.916 μs

We now consider the effects resulting from random traffic. We assume that both POXN and POXN/MP are under gated service with a maximum cycle period of 144.96 μs , which is consistent with the calculation of the upper bound. Frames arrive at each transmitting port following a Poisson process. Moreover, frames have an Ethernet format with minimum and maximum frame sizes of 64 and 1518 bytes, respectively. The frame size at each transmitting port follows the truncated geometrical distribution. The mean value of the distribution is 624.47 bytes. Thus, we obtain the mean value of the per-frame transmission time as $\overline{X} = 0.5092\ \mu s$. We use λ to denote the frame arrival rate of a transmitting port, which also represents its throughput when the system is stable. The ρ denotes the offered load to a transmitting port, which also represents the efficiency of

each port [61].

Intuitively, we can infer that the bandwidth efficiency for the MCDAP is determined by traffic load and the amount of multicast traffic. In what follows, we study how the traffic load and the amount of multicast traffic affect the mean packet delay.

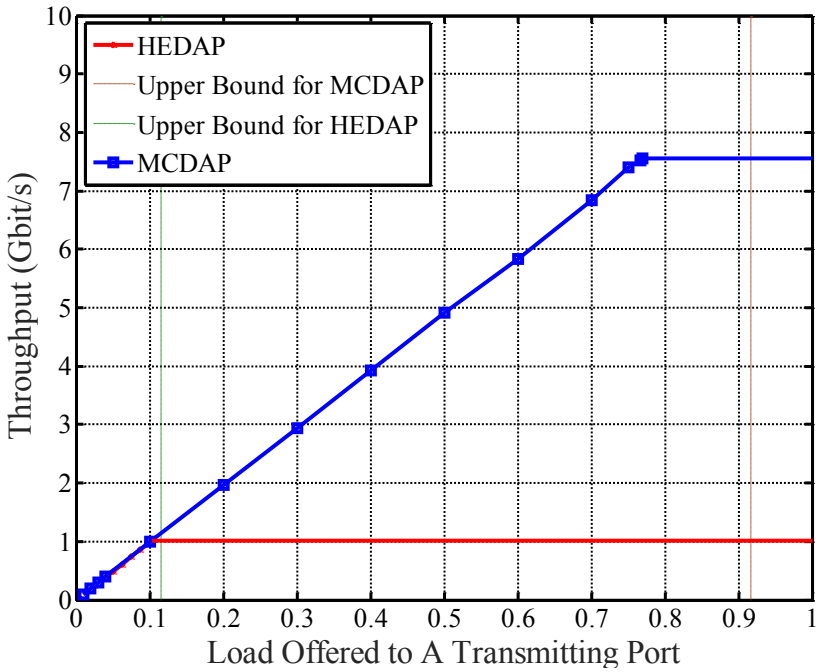

Figure 7.4 Throughput for the HEDAP and the MCDAP with different offered load ρ.

Figure 7.4 shows how the throughputs λ for both the HEDAP and the unicast plane of the MCDAP change with the offered load ρ to a transmitting port. Figure 7.4 shows that the λ of the HEDAP linearly increases with the increase of ρ until the λ of the HEDAP approaches 1.02 Gbit/s. Then, the λ of the HEDAP remains constant even though ρ increases. The maximum achievable λ of the HEDAP is achieved when ρ is 0.103, which nearly approaches its computed upper bound for maximum efficiency, with an approximately 10% relative difference. However, the maximum achievable λ of the unicast plane of the MCDAP is approximately 7.56 Gbit/s. This is achieved when ρ is

0.767, which is smaller than its upper bound for the maximum efficiency, with an approximately 16% relative difference. The maximum λ of the unicast plane of the MCDAP is nearly 7.4 times that of the HEDAP because the MCDAP supports parallel communications among different receiving ports. Furthermore, the transmitting ports in the MCDAP can send traffic through the two planes simultaneously, which can make its throughput even larger. Notably, although the theoretical per-port maximum efficiency of the unicast plane of the MCDAP is 8 times the maximum efficiency of the HEDAP for each port, the actual maximum efficiency that can be achieved is 7.4 times, which is an approximately 7% relative difference. This results because the three situations discussed at the beginning of Chapter 6 may cause efficiency loss.

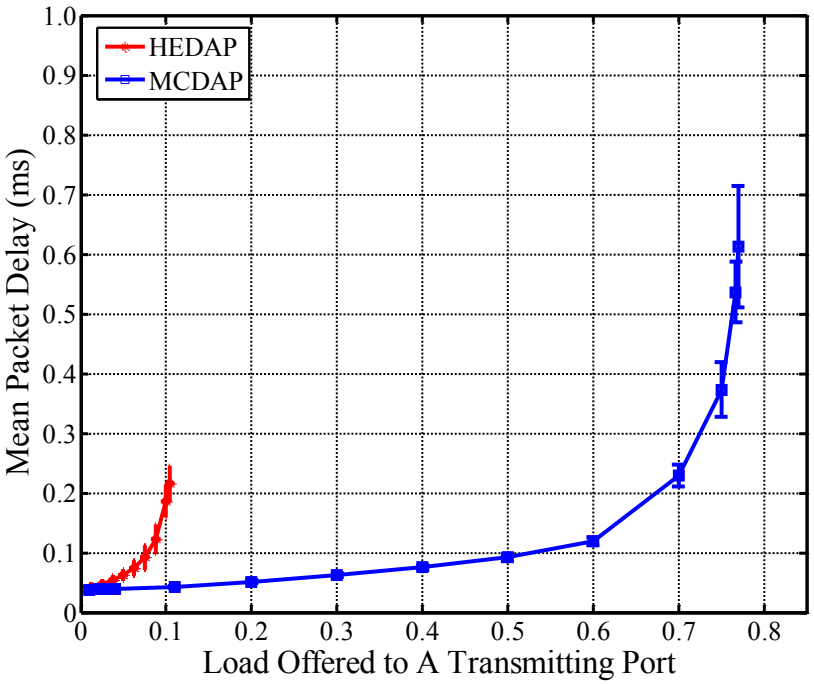

Figure 7.5 Mean packet delay for the HEDAP and the MCDAP with different offered load ρ. Simulation results shown are with 95% confidence intervals.

Figure 7.5 presents how the mean packet delay in the system changes with the offered

load ρ of both the HEDAP and MCDAP. Of the total load offered to a transmitting port in the MCDAP, 99% is set to be unicast traffic. The remaining 1% of the total load is multicast traffic. We will discuss the impact of the proportion of multicast traffic to total traffic on mean packet delay later. Figure 7.5 shows that the maximum offered load to a transmitting port for both the HEDAP and the MCDAP are no greater than 0.103 and 0.767, respectively.

When ρ approaches the load limit (0.767) for the MCDAP, the mean packet delay of the MCDAP sharply increases with a slight increase in ρ. Moreover, when the ρ of the HEDAP reaches its load limit, the mean packet delay of the MCDAP still increases slightly as ρ increases. This indicates that when packets in the HEDAP are experiencing a high average queuing delay, the proposed MCDAP still works under a fully gated service. Thus, packets experience less delay in the MCDAP compared with that of the HEDAP .

Next, we study the impact of different proportions of multicast to total traffic on the mean packet delay for a transmitting port. The incast and broadcast traffic are transmitted through the multicast plane in sequence in the MCDAP. Hence, in this simulation, we assume that the total traffic only consists of multicast and unicast traffic.

Figure 7.6 shows the impact of the proportion of the multicast traffic to total traffic on the mean packet delay. We set ρ to be 0.7, where mean packet delay begins to increase dramatically after this point as in Figure 7.5. The horizontal coordinates represent the proportion of the multicast traffic to total traffic. They reveal that when the proportion of the multicast to total traffic increases to a value larger than 11%, the mean packet delay will increase dramatically. This is because when the proportion of multicast traffic to

total traffic increases, the requested bandwidth during a data transfer cycle for the multicast plane increases accordingly. Hence, there will not be any extra bandwidth capacity remaining in the multicast plane. This will reduce the efficiency of the adopted SQF/CF algorithm.

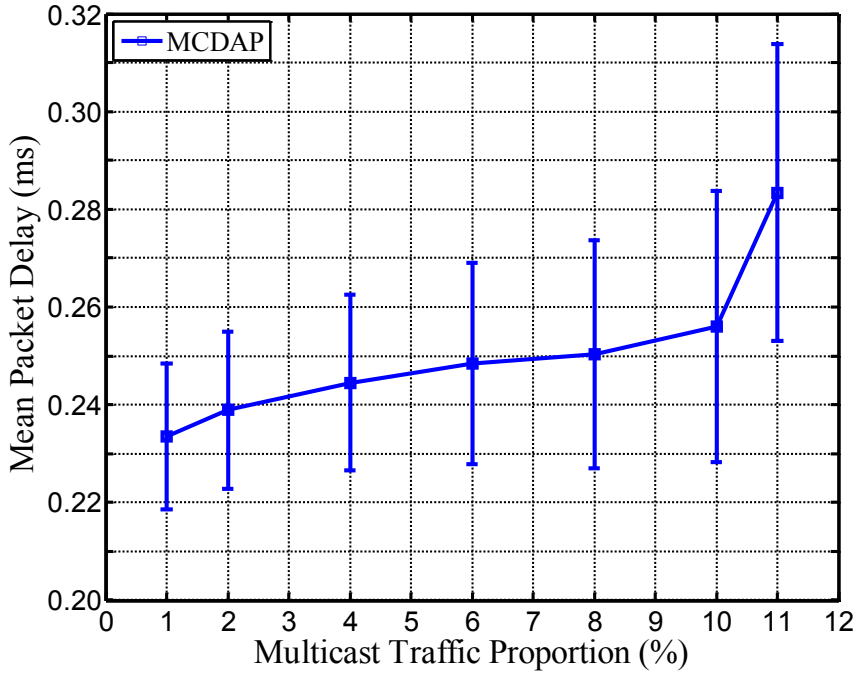

Figure 7.6 Mean packet delay time of the MCDAP with different proportions of multicast traffic to total traffic. Simulation results shown are with 95% confidence intervals.

Meanwhile, the unicast plane will finish the current data transfer cycle earlier because the amount of unicast traffic decreases accordingly. However, the unicast plane needs to wait until all the connected ports receive all the REQUEST messages through the multicast plane before it can start the next data transfer cycle. Therefore, the unicast plane needs to wait for the multicast plane to complete. This leads to an increase in the idle time between the two adjacent data transfer cycles for the unicast plane. This, in turn,

results in an increase in mean packet delay when the multicast traffic increases.

Moreover, Figure 7.6 shows that when the proportion of multicast to total traffic is less than 10%, the mean packet delay of the MCDAP remains nearly constant. This results because the multicast plane in this case has extra bandwidth capacity so that it can transmit the cut-over unicast traffic to achieve load balance. Thus, both the unicast and multicast planes in this case finish their data transfer cycles at nearly the same time, which helps reduce wasted bandwidth. Hence, it can be seen that the MCDAP enables dynamic traffic allocation to accommodate dynamic traffic patterns.

Chapter 8: Summary

We proposed a novel datacenter network called POXN/MP, an all-optical architecture with high energy efficiency that supports communication parallelism for unicast traffic to address the insufficient bandwidth problems in POXN [14]. Meanwhile, our network efficiently enables multicast/broadcast traffic transmission through the multicast plane, similar to [14]. In addition, it achieves dynamic traffic allocation to cater to the fluctuation of datacenter traffic patterns.

The tunable transponders deployed in POXN/MP have a switching time on the order of a few microseconds. Thus, the proposed scheme can be quickly reconfigured to meet traffic fluctuations, as opposed to the slow reconfiguration time of active optical circuit switch-based networks. When taking advantage of passive rather than active optical devices, we achieve low cost, high energy efficiency, and high bandwidth communication among interconnected ports. To solve the link layer collision problem, we propose the MCDAP, a fully distributed protocol that enables dynamic traffic allocation, which is suitable to the bursty nature of datacenter traffic patterns.

The scheduling problem for the unicast plane was formulated as a mixed integer programming problem. Three heuristic algorithms were proposed to speed up the computing process.

The simulation results show significant performance increases for both the system throughput and the mean packet delay compared with POXN. Thus, POXN/MP is more suitable for datacenter networks where traffic is bursty with high temporary peaks. The simulation results also show that the SQF/CF is the most promising scheduling algorithm

to take advantage of both the unicast and multicast planes and to achieve dynamic load distribution between the two planes.

POXN/MP efficiently supports communications with mixed traffic patterns. It is more suitable to be adopted in datacenter networks compared with previous POXN. Prevalent datacenter traffic studies show a bimodal packet size among some of the investigated datacenters. Some statistics show that larger flows vary from 1 MB to 50 MB [17]. POXN/MP is useful to cater to these larger flows. Thus, we can deploy POXN/MP near the end user. Moreover, we can deploy the POXN/MP near the core-switch tier to accommodate more aggregated bandwidth.

Our future work includes investigating how to fit the POXN/MP into different datacenter topologies.

Bibliography or References

[1] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107-113, 2008.

[2] S. Ghemawat, H. Gobioff, and S. T. Leung, "The Google file system," In *ACM SIGOPS operating systems review,* vol. 37, no. 5, pp. 29-43, 2003.

[3] L. A. Barroso, J. Dean, and U. Hölzle, "Web search for a planet: The Google cluster architecture," *Micro, Ieee*, vol.*23*, no. 2, pp. 22-28, 2003.

[4] H. Liu, C. F. Lam, and C. Johnson, "Scaling Optical Interconnects in Datacenter Networks Opportunities and Challenges for WDM," *18th IEEE Symposium on High Performance Interconnects,* 2010*,* pp. 113-116.

[5] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference,* 2009, pp. 202-208.

[6] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement,* 2010, pp. 267-280.

[7] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," *ACM SIGCOMM Computer Communication Review*, vol. 40, no 1, pp. 92-99, 2010.

[8] C. J. Sher Decusatis, A. Carranza, and C. M. DeCusatis, "Communication within clouds: open standards and proprietary protocols for data center networking," *Communications Magazine, IEEE*, vol. 50, no. 9, pp. 26-33, 2012.

[9] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, and S. Sengupta, "VL2: a scalable and flexible data center network," *ACM SIGCOMM computer communication review,* vol. 39, no. 4, pp. 51-62, 2009.

[10] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 63-74, 2008.

[11] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, and S. Lu, "BCube: a high performance, server-centric network architecture for modular data centers," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 63-74, 2009.

[12] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 75-86, 2008.

[13] H. Wang, Y. Xia, K. Bergman, T. S. Ng, S. Sahu, and K. Sripanidkulchai, "Rethinking the physical layer of data center networks of the next decade: Using optics to enable efficient*-cast connectivity," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 52-58, 2013.

[14] W. Ni, C. Huang, Y. L. Liu, W. Li, K. W. Leong, and J. Wu, "POXN: A new passive optical cross-connection network for low-cost power-efficient datacenters," *Journal of Lightwave Technology*, vol. 32, no. 8, pp. 1482-1500, 2014.

[15] C. Kachris and I. Tomkos, "A survey on optical interconnects for data centers," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 4, pp. 1021-1036, 2012.

[16] N. Farrington and A. Andreyev, "Facebook's data center network architecture," In *IEEE Optical Interconnects Conf.*, 2013.

[17] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren, "Inside the Social Network's (Datacenter) Network," In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 123-137.

[18] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, and V. Kurien, "Pingmesh: A Large-Scale System for Data Center Network Latency Measurement and Analysis," In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication,* 2015, pp. 139-152.

[19] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, and H. Zheng, "Packet-level telemetry in large datacenter networks," In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 479-491.

[20] Cisco Global Cloud Index: Forecast and Methodology, 2013–2018. Cisco Corp., [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.html

[21] T. Benson, A. An, A. Akella, and M. Zhang, "Microte: The case for fine-grained traffic engineering in data centers," *In ACM CoNEXT'11.*, 2011.

[22] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69-74, 2008.

[23] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 1-12, 2007.

[24] J. Perry, A. Ousterhout, H. Balakrishnan, D. Shah, and H. Fugal, "Fastpass: A centralized zero-queue datacenter network," In *Proceedings of the 2014 ACM conference on SIGCOMM*, 2014, pp. 307-318.

[25] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, "Helios: a hybrid electrical/optical switch architecture for modular data centers," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 339-350, 2011.

[26] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. S. Ng, M. Kozuch, and M. Ryan, "c-Through: Part-time optics in data centers," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 327-338, 2011.

[27] Z. Zhu, and S. Zhong, "Scalable and topology adaptive intra-data center networking enabled by wavelength selective switching," In *Optical Fiber Communication Conference*, 2014, pp. Th2A-60.

[28] P. Samadi, V. Gupta, B. Birand, H. Wang, G. Zussman, and K. Bergman, "Accelerating incast and multicast traffic delivery for data-intensive applications using physical layer optics," In *Proceedings of the 2014 ACM conference on SIGCOMM*, 2014, pp. 373-374.

[29] N. Hamedazimi, Z. Qazi, H. Gupta, V. Sekar, S. R. Das, J. P. Longtin, and A. Tanwer, "FireFly: a reconfigurable wireless data center fabric using free-space optics," In *Proceedings of the 2014 ACM conference on SIGCOMM*, 2014, pp. 319-330.

[30] C. F. Lam, C. H. Liu, B. Koley, X. Zhao, V. Kamalov, and V. Gill, "Fiber optic communication technologies: What's needed for datacenter network operations," *IEEE Communications Magazine*, vol.48, no. 7, pp. 32-39, 2010.

[31] N. Farrington, A. Forencich, P. C. Sun, S. Fainman, J. Ford, A. Vahdat, and G. C. Papen, "A 10 us Hybrid Optical-Circuit/Electrical-Packet Network for Datacenters," In *Optical Fiber Communication Conference*, 2013, pp. OW3H-3.

[32] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, and A. Vahdat, "Integrating microsecond circuit switching into the data center," *ACM SIGCOM Computer Communication Review*, vol. 43, no. 4, pp. 447-458, 2013.

[33] X. Ye, Y. Yin, S. B. Yoo, P. Mejia, R. Proietti, and V. Akella, "DOS: A scalable optical switch for datacenters," In *Proceedings of the 6th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, 2010, pp. 24.

[34] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: a topology malleable data center network," In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 2010, pp. 8.

[35] K. Xia, Y. H. Kaob, M. Yangb, H. J. Chao, "Petabit optical switch for data center networks," Polytechnic Institute of New York University, New York, Tech. Rep., 2010.

[36] J. Gripp, J. E. Simsarian, J. D. LeGrange, P. Bernasconi, and D. T. Neilson, "Photonic terabit routers: the IRIS project," In *Optical Fiber Communication Conference*, 2010, pp. OThP3.

[37] C. Nitta, R. Proietti, Y. Yin, S. B. Yoo, M. Farrens, and V. Akella, "*Leveraging AWGR-based Optical Packet Switches to Reduce Latency in Petascale Computing Systems*," University of California, Davis, Tech. Rep., 2012.

[38] R. Proietti, C. Nitta, Y. Yin, R. Yu, S. J. B. Yoo, and V. Akella, "Scalable and distributed contention resolution in AWGR-based data center switches using RSOA-based optical mutual exclusion," *Selected Topics in Quantum Electronics, IEEE Journal of*, vol. 19, no. 2, pp. 3600111-3600111, 2013.

[39] B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, and R. Rajamony, "The PERCS high-performance interconnect," In *High Performance Interconnects (HOTI), 2010 IEEE 18th Annual Symposium on*, 2010, pp. 75-82.

[40] Y. J. Liu, P. X. Gao, B. Wong, and S. Keshav, "Quartz: a new design element for low-latency DCNs," In *Proceedings of the 2014 ACM conference on SIGCOMM*, 2014, pp. 283-294.

[41] "The new Optical Datacenter," Polatis Data Sheet, Polatis Inc., 2009.

[42] C. Kachris, K. Kanonakis, and I. Tomkos, "Optical interconnection networks in data centers: recent trends and future challenges," *Communications Magazine, IEEE*, vol. 51, no. 9, pp. 39-45, 2013.

[43] G. Wang, D. G. Andersen, M. Kaminsky, M. Kozuch, T. S. Ng, K. Papagiannaki, "Your data center is a router: The case for reconfigurable optical circuit switched paths," Computer Science Department, Carnegie Mellon University, pp. 62, 2009.

[44] G. Kramer, B. Mukherjee, and G. Pesavento, "IPACT: A dynamic protocol for an Ethernet PON (EPON)," *IEEE Commun. Mag.,* vol. 40, no. 2, pp. 74-80, Feb. 2002.

[45] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 wireless local area networks," *Communications Magazine, IEEE*, vol. 35, no. 9, pp. 116-126, 1997.

[46] S. Bharati and P. Saengudomlert, "Analysis of the mean packet delay for dynamic bandwidth allocation algorithms in EPONs," *J. Lightw. Technol*., vol. 28, no. 23, pp. 3454-3462, Dec. 2010.

[47] I. M. I. Habbab, M. Kavehrad, and C.-E. W. Sundberg, "Protocols for every high-speed optical fiber local area networks using a passive star topology," *J. Lightw. Technol*., vol. 5, no. 12, pp. 1782-1794, Dec. 1987.

[48] H. Takagi, "Analysis and application of polling models," In *Performance Evaluation: Origins and Directions*, Springer Berlin Heidelberg, 2000, pp. 423-442.

[49] V. Chvátal, "Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete mathematics*," vol. 4, no. 4, pp. 305-337, 1973.

[50] O. A. Lavrova, G. Rossi, and D. J. Blumenthal, "Rapid tunable transmitter with large number of ITU channels accessible in less than 5 ns," In *Proc. ECOC*., vol. 2, pp. 169-170, 2000.

[51] C. F. Lam, (Ed.), *Passive optical networks: principles and practice*. Academic Press, 2011.

[52] A. Hasegawa and Y. Kodama, "Signal transmission by optical solitons in monomode fiber," *Proceedings of the IEEE*, vol. 69, no. 9, pp. 1145-1150, 1981.

[53] P. Dumon, W. Bogaerts, D. Van Thourhout, D. Taillaert, R. Baets, J. Wouters, "Compact wavelength router based on a silicon-on-insulator arrayed waveguide grating pigtailed to a fiber array," *Optics express*, vol. 14, no. 2, pp. 664-669, 2006.

[54] Classic Network Design Using Cisco Nexus 9000 Series Switches. Cisco Corp., [Online]. Available:http://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/guide-c07-730115.pdf

[55] Cisco Nexus 3548 and 3524 Switches Data Sheet. Cisco Corp., [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/switches/nexus-3548-switch/data_sheet_c78-707001.html

[56] www.fiberstore.com/c/10g-sfp-plus_63

[57] Cisco 10GBASE SFP+ Modules Data Sheet. Cisco Corp., [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/interfaces-modules/transceiver-modules/data_sheet_c78-455693.html

[58] Cisco 10GBASE Dense Wavelength-Division Multiplexing SFP+ Modules Data Sheet. Cisco Corp.

[59] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection." In *INFOCOM 2002. Twenty-First Annual Joint*

Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2002, vol. 3, pp. 1190-1199.

[60] C. E. Hopps, Analysis of an equal-cost multi-path algorithm. 2000.

[61] J. Banks, J. S. Carson, and B. L. Nelson, *DM Nicol, Discrete-Event System Simulation*. Englewood Cliffs, NJ, USA: Prentice hall, 2000.