國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

博士論文

光核心及光擷取網路之網路效能最佳化與頻寬控制技術

Optimization and Bandwidth Control for Optical Core and Access Networks

研　　究　　生：林士勛

指　導　教　授：楊啟瑞　博士

中 華 民 國 九 十 九 年 七 月

光核心及光擷取網路之網路效能最佳化與頻寬控制技術

Optimization and Bandwidth Control for Optical Core and Access Networks

研 究 生：林士勛　　**Student: Shih-Hsuan Lin**

指導教授：楊啟瑞　博士　　**Advisor: Dr. Maria C. Yuang**

國立交通大學　資訊學院

資訊科學與工程研究所

博士論文

A Thesis
Submitted to Department of Computer Science
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy
in
Computer Science
July 2010
Hsinchu, Taiwan, R.O.C.

中 華 民 國 九 十 九 年 七 月

光核心及光擷取網路之網路效能最佳化與頻寬控制技術

研究生：林士勛　　指導教授：楊啟瑞　博士

國立交通大學　資訊科學與工程研究所

Abstract in Chinese

在本論文中，主要研究三個最佳化問題分別應用於光核心、光都會，以及最後一哩的光存取分頻多工(wavelength division multiplexing; WDM)網路中。在第一部分的研究中，我們提出一高效能之最佳化逼近演算法，稱為 Lagrangean 放寬與探索式演算法(Lagrangean relaxation heuristics; LRH)。此方法主要用以解決光路徑與光波長之選擇(routing and wavelength assignment; RWA)於包含各式光交換元件(例如：全光纖交換、寬頻帶交換，窄頻帶交換)之光核心網路上。首先，我們將此光路徑與波長選擇問題有系統的公式化成整合性最佳化問題。此最佳化問題的目標為盡力降低最擁擠光纖路段之波長使用率。Lagrangean 放寬與探索演算法執行限制放寬的動作並經由次坡度疊代法(subgradient-based iterations)產生一組 Lagrangean 乘法參數用以計算出原問題之下限解。同時，此演算法使用這些 Lagrangean 乘法參數再搭配內部的新式探索演算法以得到接近最佳解之上限解。根據所得的下限解與上限解，我們可量化 Lagrangean 放寬與探索演算法之精確度與收斂速度。對於不同網路環境設定，可達到對 Lagrangean 演算法效能之評估。依照相同之量化標準，我們對此一新式演算法與傳統上採用線性放寬演算法(linear programming relaxation; LPR)進行效能之比較。此效能比較實測於各種廣為大眾熟知的網路以及電腦隨機產生的網路中。模擬結果證明 Lagrangean 放寬與探索演算法在精確度與收斂速度上皆優於線性放寬演算法。此結果對於較大網路環境尤其明顯。

接著，在第二部分的都會型光纖網路研究上，我們設計一可提供服務品質差異之仿效 Banyan 光封包交換系統(QoS-enabled pseudo-Banyan optical packet switching system; QBOPSS)，以應用於分頻多工光都會網路中。此系統由一群適當尺寸之光仿效 Banyan 封包交換器和少量光纖暫存器組成，以期達到高系統擴展性、高成本效益、以及低封包遺失率之目標。在此新式光封包交換系統中，封包排程機制由一考量服務品質差異之平行遞增排程演算法(QoS parallel incremental scheduling; QPIS)完成。此排程演算法在滿足兩個系統資源限制：封包交換器能力限制和光暫存器數量限制之下，極力達成系統效能之最佳化並優先滿足高優先權封包遺失率。值得注意的，我們證明此新式排程演算法具有“運算結果隨時間最佳化”的特性。換言之，每回合運算中搜尋到交換路徑的封包數量將隨時間漸增。接著，我們為此平行遞增排程演算法設計一套硬體實現，以真正達成平行運算之目標。根據此硬體實現，我們經由模擬與分析進一步顯示出此演算法可達到逼近最佳化之結果，並同時具有低計算複雜度之特性。假設 N 是系統輸入埠數量，W 和 M 分別表示外部與內部光波長數量，則此演算法的計算複雜度可表示為 $O(NW\times\log_2(NMW))$。最後，比較平行遞增排程演算法與其他演算法的模擬結果，我們可看出此新式排程演算法在封包遺失率、服務品質差異性提供、以及計算複雜度上皆優於其他演算法。

最後，論文的第三部分：最後一哩光存取分頻多工網路，我們提出一分散式控制之被動光纖網路(distributed control passive optical network; DCPON)架構。利用低速之額外控制頻道反射並廣播控制訊息的設計概念，分散式控制被動光纖網路可達到高系統頻寬使用率以及低資源要求/回應時間的目的。此新式架構中的動態資源分配(dynamic bandwidth allocation; DBA)由可調式封包串聯比例分配伺服演算法(adaptive packet-by-packet rate-proportional server; A-PRPS)完成。在新的分散式控制被動光纖網路中，我們依據使用者之流量需求變動程度和/或服務品質需求程度適當地調整可調式封包串聯比例分配伺服演算法內部之比重參數以及門檻參數。利用對應之參數調整，此新式動態資源分配演算法可有效率地降低

具有高流量變動率和/或高服務品質需求之使用者的平均封包延遲和延遲誤差。模擬結果進一步顯示可調式封包串聯比例分配伺服演算法在平均封包延遲上遠低於常見的週期可變式交叉輪詢演算法(interleaved polling with adaptive cycle time; IPACT)以及其他週期性頻寬分配演算法。

Optimization and Bandwidth Control for Optical Core and Access Networks

Student: Shih-Hsuan Lin *Advisor*: Dr. Maria C. Yuang

Department of Computer Science
National Chiao Tung University, Taiwan

Abstract

In this thesis, three optimization problems are investigated separately in the optical core, metropolitan, and first-mile access wavelength division multiplexing (WDM) networks. Firstly, in the core network, we propose an efficient approximation approach, called Lagrangean relaxation with heuristics (LRH), aimed to resolve routing and wavelength assignment (RWA) problem for multi-granularity WDM core networks facilitating fiber, waveband, and lambda switching capabilities. The RWA problem is first formulated as a combinatorial optimization problem in which the bottleneck link utilization is to be minimized. The LRH approach performs constraint relaxation and derives a lower-bound solution index according to a set of Lagrangean multipliers generated through subgradient-based iterations. In parallel, using the generated Lagrangean multipliers, the LRH approach employs a new heuristic algorithm to arrive at a near-optimal upper-bound solution. With lower and upper bounds, we delineate the performance of LRH with respect to accuracy and convergence speed under different parameter settings. We further draw comparisons between LRH and a typical linear programming relaxation (LPR) approach via experiments over several well-known networks and randomly generated networks. Numerical results demonstrate that LRH outperforms the LPR approach in both accuracy and computational time complexity particularly for large size networks.

In the second part of the thesis, we present the architecture of a QoS-enabled

pseudo-Banyan optical packet switching system (QBOPSS) for WDM metro networks. The QBOPSS system consists of a group of downsized optical pseudo-Banyan space switches and a handful of fiber-delay-line-based optical buffers, achieving high system scalability, cost-effectiveness, and low packet loss probability. Packet scheduling in QBOPSS is performed by a QoS parallel incremental scheduling (QPIS) algorithm. The algorithm minimizes the loss probability for high-priority packets while maximizing system throughput and satisfying two constraints (switch-contention free, and buffer-contention free). Most notably, we prove that QPIS is incremental, i.e., the computed packet sets within each time slot are monotonically non-decreasing. We then present a hardware system architecture to demonstrate the parallel implementation of QPIS. As will be shown, QPIS achieves a near-optimal solution with an exceptionally low complexity, $O(NW \times \log_2(NMW))$, where N is the number of input ports, and W and M the numbers of external and internal wavelengths, respectively. From simulation results that pit the QPIS algorithm against two other algorithms, we show that QPIS outperforms these algorithms on packet loss probability, QoS differentiation, and computational complexity.

Finally, in the last part– first-mile access network, we propose a distributed control passive optical network (DCPON). By using low-speed out-of-band control channel to reflect and distributed control information, DCPON achieves high utilization and low request/response time. Dynamic bandwidth allocation (DBA) in DCPON is done by an adaptable packet-by-packet rate-proportional server (A-RPRS). In DCPON, by adjusting weight and threshold parameters of A-PRPS according to bursty degrees and/or QoS requirements of users, A-PRPS dramatically downgrades mean delay and delay jitter for high bursty and/or high QoS requiring users. Simulation results show that A-PRPS outperforms the interleaved polling with

adaptive cycle time (IPACT) and other cycle-based DBAs on mean packet delay.

CONTENTS

ABSTRACT IN CHINESE III

ABSTRACT VI

LIST OF FIGURES X

LIST OF TABLES XII

ACRONYMS XIII

CHAPTER 1. INTRODUCTION 1

CHAPTER 2. ROUTING AND WAVELENGTH ASSIGNMENT IN WDM CORE NETWORK 4

- **2.1 PROBLEM FORMULATION** 6
- **2.2 LAGRANGEAN RELAXATION WITH HEURISTICS** 9
- **2.3 EXPERIMENTAL RESULTS** 16
- **2.4 SUMMARY OF PART I THESIS** 33

CHAPTER 3. OPTICAL PACKET SWITCHING SYSTEM AND SCHEDULING ALGORITHM DESIGN IN METROPOLITAN NETWORK 34

- **3.1 QBOPSS: NEW OPS SYSTEM DESIGN AND ASSESSMENT** 39
- **3.2 QOS PACKET SCHEDULING: THE QPIS ALGORITHM** 46
- **3.3 QPIS IMPLEMENTATION: HARDWARE PARALLEL SYSTEM ARCHITECTURE** 57
- **3.4 SIMULATION RESULTS** 62
- **3.5 SUMMARY OF PART II THESIS** 70

CHAPTER 4. DYNAMIC BANDWIDTH ALLOCATION IN FIRST-MILE PASSIVE OPTICAL NETWORK 72

- **4.1 DCPON SYSTEM ARCHITECTURE** 77
- **4.2 THE A-PRPS ALGORITHM** 81
- **4.3 SIMULATION RESULTS** 85
- **4.4 TESTBED EXPERIMENTATION** 90
- **4.5 SUMMARY OF PART III THESIS** 94

CHAPTER 5. CONCLUSIONS AND FUTURE WORKS 95

REFERENCES 97

BIOGRAPHY 102

List of Figures

Figure 2.1 Illustration of graph transformation (K = 4, B = 3).....6
Figure 2.2 Lagrangean Relaxation with Heuristics (LRH) 11
Figure 2.3 MSSP Algorithm 13
Figure 2.4 Primal Heuristic Algorithm.....15
Figure 2.5 Convergence speed versus accuracy on the basis of using termination requirement 18
Figure 2.6 Convergence speed versus accuracy on the basis of using fixed iteration number 19
Figure 2.7 Random Network topology.....20
Figure 2.8 Comparisons of accuracy and computation time for random network NET1 23
Figure 2.9 Comparisons of accuracy and computation time for random network NET2 with two FSC nodes 24
Figure 2.10 Comparisons of accuracy and computation time for random network NET3 with two FSC nodes 25
Figure 2.11 Comparisons of accuracy and computation time for random network NET2 with four FSC nodes.....26
Figure 2.12 Comparisons of accuracy and computation time for random network NET3 with four FSC nodes.....27
Figure 2.13 NSFNET Network 28
Figure 2.14 Comparisons of accuracy and computation time for NSFNET 29
Figure 2.15 Two large sized well-known networks (64 wavelengths).....30
Figure 2.16 The performance of LRH approach for USA network.....31
Figure 2.17 The performance of LRH approach for ARPA network 32
Figure 3.1 QBOPSS- system architecture 40
Figure 3.2 The QPIS Algorithm 51
Figure 3.3 Illustration for the proof of Theorem 1 53
Figure 3.4 The hardware parallel system architecture 58
Figure 3.5 Packet loss probabilities of QPIS under three practicable PBS sizes (in BP traffic model).....67
Figure 3.6 Packet loss probability comparisons for three algorithms under 8×8 PBS size (in BP traffic model) 68
Figure 3.7 Packet loss probability comparisons for three algorithms under 16×16 PBS size (in BP traffic model) 69
Figure 4.1 DCPON network and system architecture 77
Figure 4.2 DCPON operating concept (3 ONUs example) 79

Figure 4.3 The A-PRPS Algorithm.....84
Figure 4.4 Mean packet delay comparison for different schemes.....85
Figure 4.5 Maximum idle threshold (*T*) effect for A-PRPS under various traffic burstiness setting (*B*).....88
Figure 4.6 The impact of control message update frequency.....89
Figure 4.7 DCPON- experimental setup 91
Figure 4.8 DCPON- experimental results 93

List of Tables

Table 3.1 Component-count comparison of space switches with switch size *NW*×*NW* 43
Table 3.2 Signal-quality comparison of space switches with switch size *NW*×*NW* 44
Table 3.1 Comparison of computational complexity 64
Table 3.4 Loss probability comparisons between QPIS and Optimal /*At each table entry: BP value (IBP value)*/ 64
Table 3.5 Packet loss probability of QBOPSS using QPIS under different buffer sizes (in BP traffic model) 66

Acronyms

A-PRPS	Adaptable Packet-by-packet Rate-Proportional Server
AWG	Arrayed Waveguide Grating
BP	Bernoulli Process
CSC	Central Switch Controller
DBA	Dynamic Bandwidth Allocation
DCPON	Distributed Control Passive Optical Network
FDL	Fiber Delay Line
FOB	FDL Optical Buffer
FSC	Fiber Switch Capable
FWF	Fixed Wavelength Filter
FWM	Four-Wave Mixing
GPS	Generalized Processor Sharing
IBP	Interrupted Bernoulli Process
ILP	Integer Linear Programming
IPACT	Interleaved Polling with Adaptive Cycle Time
IPP	Interrupted Poisson Process
LB	Lower Bound
LP	Linear Programming
LPR	Linear Programming Relaxation
LR	Lagrangean Relaxation
LRH	Largangean Relaxation with Heuristics
LSC	Lambda Switch Capable
MSSP	Modified Successive Shortest Path
OLT	Optical Line Terminal
ONU	Optical Network Unit
OPS	Optical Packet Switching
PBS	Pseudo-Banyan Space Switch
PON	Passive Optical Network
QBOPSS	QoS-enabled Pseudo-Banyan Optical Packet Switching System
QoS	Quality-of-Service
QPIS	QoS Parallel Incremental Scheduling
QT	Quiescence_Threshold
RAM	Random Access Memory
RN	Remote Node
RWA	Routing and Wavelength Assignment
SASK	Superimposed Amplitude Shift Keying

SCFQ	Self-Clocked Fair Queueing
SD	Source-Destination
SFQ	Start-time Fair Queueing
SOA	Semiconductor Optical Amplifier
SPFQ	Starting Potential-based Fair Queueing
SSP	Successive Shortest Path
TDM	Time Division Multiplexing
TOWC	Tunable Optical Wavelength Converter
UB	Upper Bound
WC	Wavelength Converter
WDM	Wavelength Division Multiplexing
WFQ	Weighted Fair Queueing
WF^2Q	Worse-case Fair Weighted Fair Queueing
WSC	Waveband Switch Capable
XGM	Cross-Gain Modulation
XPM	Cross-Phase Modulation

Chapter 1. Introduction

Optical wavelength division multiplexing (WDM) has been shown successful in providing virtually unlimited bandwidth to support an ever-increasing amount of traffic for future optical networks. Future optical networks are expected to flexibly and cost-effectively satisfy various applications. Similar to traditional electronic networks, the global WDM network environment is constructed in hierarchism. By having different physical limitations between optical and electric, a lot of new problems and/or constraints occur. In this thesis, we focus on three important optimization problems separately in the optical WDM core, metropolitan, and first-mile networks. (I) Routing and wavelength assignment in WDM core network; (II) Optical packet switching system and scheduling algorithm design in metropolitan network; and (III) Dynamic bandwidth allocation in first-mile passive optical network.

In the WDM core network, a tremendous amount of data needs to be handled in short time, which brings the need of virtual circuit switching technique in core network. By WDM, the network not only decides routing paths for each source-destination user pair, but also needs to finely assign wavelength for achieving high system performance. It has been proved that the routing and wavelength assignment (RWA) problem is NP-complete. Therefore, in the first part of thesis, we propose an approximation approach, called Largangean relaxation with heuristics (LRH) algorithm, aimed to find optimal solution. Simulation results show that the LRH outperform other existing methods with respect to system performance and computation time.

In the metropolitan, the WDM networks are expected to fast switch incoming packets to their destinations. As we know, different from the core networks, the traffic

in WDM metro network is less smooth. Therefore, the optical packet switching (OPS) technique is much suitable for such a wide range of applications having time-varying bandwidth demands and diverse quality-of-service (QoS) requirements in WDM metro network. Thus, in this part of thesis, we mainly investigate the OPS system and the related packet scheduling algorithm. In order to address the limitations of OPS system, such as optical random access memory (RAM) and optical signal processing, we present a novel QoS-enabled pseudo-Banyan optical packet switching system (QBOPSS). The system consists of a group of downsized optical pseudo-Banyan space switches together with single-stage downsized fiber-delay-line-based optical buffer, achieving high system scalability, cost-effectiveness, and low packet loss probability. After that, the QoS parallel incremental scheduling (QPIS) algorithm is proposed to control the QBOPSS, aimed at minimizing the loss probability of high priority packets while maximizing system throughput. Simulation results show that QPIS achieves the near optimal solution on packet loss probability, QoS differentiation, and computation complexity.

In the first mile (or called last mile), the passive optical network (PON) is currently one of the most attractive local access network architectures. By only adopting passive optical components (such as slitter, coupler, circulator, and etc.) between optical line terminal (OLT) and optical network units (ONUs), PON can be widely deployed with effective maintaining cost on system equipment. However, while the upstream data channel is shared by all end users, how to fairly allocate upstream channel bandwidth becomes an important issue. Further, because of the high bursty phenomenon of user traffic in the first mile environment, the concept of generalized processor sharing (GPS) scheme seems unfair for making bandwidth allocation among such bursty users. As a result, in this part of investigation, we firstly design a new PON architecture, called distributed control passive optical network

(DCPON), which improves the upstream data channel utilization by frequently broadcasting user request/status messages on a cost-effective low-speed out-of-band control channel. Based on DCPON, a dynamic bandwidth allocation (DBA) scheme, called adaptable packet-by-packet rate-proportional server (A-PRPS) is then proposed to efficiently and fairly schedule bandwidth among users having bursty traffic. Simulation results show that A-PRPS outperforms the interleaved polling with adaptive cycle time (IPACT) and other cycle-based DBAs on mean packet access/queueing delay. Also, A-PRPS notably downgrades the packet mean delay and/or delay jitter for high bursty users.

The remainder of this thesis is organized as follow. In Chapter 2, the first optimization problem of the thesis is detailed. The second problem of the thesis is discussed in Chapter 3. Finally, in Chapter 4, we investigate the last problem of the thesis. Conclusion remarks are provided in Chapter 5.

Chapter 2. Routing and Wavelength Assignment in WDM Core Network

With advances in optical wavelength division multiplexing (WDM) technologies [1] and its potential of providing virtually unlimited bandwidth, optical WDM networks have been widely recognized as the dominant transport infrastructure for future Internet core networks. To maintain high scalability and flexibility at low cost, WDM networks often include switching devices with different wavelength conversion powers [2,3] (e.g., no, limited- or full-range), and multi-granularity switching capability [4,5,6]. In particular, examples of multi-granularity switching include switching on a single lambda, a waveband, i.e., multiple lambdas, or an entire fiber basis.

One major traffic engineering challenge in such WDM networks has been efficient routing and wavelength assignment (RWA) [3,7]. The problem deals with routing and wavelength assignment between source and destination nodes subject to the wavelength-continuity constraint [4] in the absence of wavelength converters. It has been shown that RWA is an NP-complete problem [4]. Numerous approximation algorithms [3,7] have been proposed with the aim of balancing the trade-off between accuracy and computational time complexity. In general, some algorithms [8,9] focused on the problem in the presence of sparse, limited, or full-range wavelength converters. Some others made an effort to either reduce computational complexity by solving the routing and wavelength assignment sub-problems separately [4], or increase accuracy by considering the two sub-problems [10] jointly. However, with the multi-granularity switching capability taken into consideration, most existing algorithms become functionally or economically unviable.

In the first problem of my thesis, our aim is to efficiently resolve RWA in

multi-granularity WDM networks with fiber switch capable (FSC), waveband switch capable (WSC), and lambda switch capable (LSC) nodes. The problem is in short referred to as RWA+. To tackle the problem, we propose an efficient approximation approach, called Lagrangean relaxation with heuristics (LRH). RWA+ is first formulated as a combinatorial optimization problem in which the bottleneck link utilization is to be minimized. The LRH approach performs constraint relaxation and derives a lower-bound solution index according to a set of Lagrangean multipliers generated through subgradient-based iterations. In parallel, using the generated Lagrangean multipliers, the LRH approach employs a new primal heuristic algorithm to arrive at a near-optimal upper-bound solution. With lower and upper bounds, we delineate the performance of LRH with respect to accuracy and convergence speed under different parameter settings and termination criteria. We further draw comparisons between LRH and a linear programming relaxation (LPR) approach [4] via experiments over the widely-used NSFNET and three randomly generated networks. Numerical results demonstrate that LRH outperforms the LPR approach in both accuracy and computational time complexity particularly for larger sized networks.

The first part of the thesis– Chapter 2 is organized as follow. In Chapter 2.1, we first give the RWA+ problem formulation. In Chapter 2.2, we present the LRH approach and its primal heuristic algorithm. In Chapter 2.3, we demonstrate numerical results of the performance study and comparisons under several well-known networks (NSFNET, USA, and ARPA) and randomly generated networks. Finally, major remarks are summarized in Chapter 2.4.

2.1 Problem Formulation

The RWA+ problem is formulated as a linear integer problem stated as follows. Given a physical topology (with FSC, WSC, and LSC nodes) and available wavelengths on each link, and requested lightpath demands between all source-destination pairs, determine the routes and wavelengths of lightpaths, such that the maximum number of lightpaths on the most congested link is minimized, subject to the wavelength continuity constraint. For ease of illustration, we assume in the sequel that the number of available wavelengths on each link is the same.

Due to the existence of FSC and WSC nodes, a graph transformation is first required. For each WSC node with K input (and output) fibers and B wavebands, it is replaced by a bipartite subgraph with K phantom nodes connecting to input fibers, and another K phantom nodes connecting to output fibers, as shown in Figure 2.1, where K=4 and B=3. Besides, there are additional $(K\times K)\times B$ phantom links connecting the $2K$ phantom nodes. These phantom links describe possible configuration combinations inside a WSC node. Notice that FSC is a special case of WSC node where B=1. We summarize the notation used in the formulation as follows:

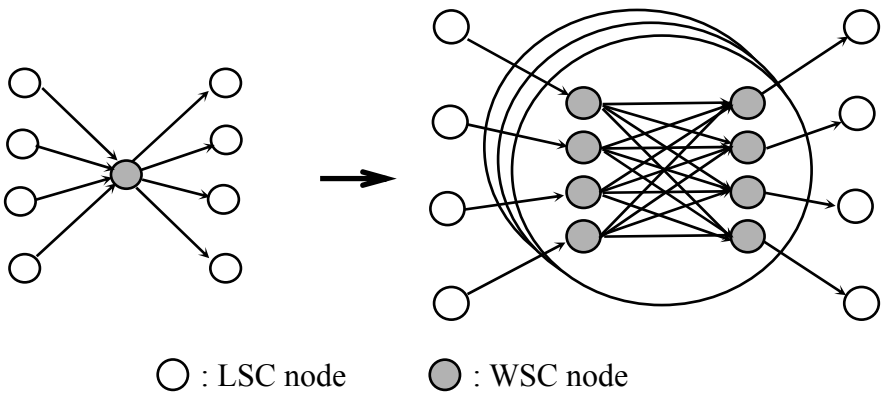


Figure 2.1 Illustration of graph transformation (K = 4, B = 3).

Input values:

N_{LSC} : the set of LSC nodes in the network;

N_{WSC} : the set of WSC nodes in the network;

L : the set of physical optical links;

L_{WSC} : the set of phantom links within WSC nodes;

V_n^{in} : the set of phantom input nodes for node n;

V_n^{out} : the set of phantom output nodes for node n;

W : the set of wavelengths on each link; (assumed to be the same for all links);
$|W|$= the number of wavelengths;

S : the set of source-destination (SD) pairs requesting lightpath set-up;

S_n : the set of SD pairs where node n is the source node;

B_n : the set of waveband of WSC node n;

P_s : candidate path set for SD pair s;

d_s : lightpath demand for SD pair s;

δ_{pl} : =1, if path p includes link l; =0, otherwise;

φ_{lw} : =1, if wavelength w belong to phantom waveband link l; =0, otherwise;

μ_{lb} : =1, if link l is belong to waveband b; =0, otherwise;

σ_{lv} : =1, if link l is incident to node v; =0, otherwise;

Decision variables:

α : most congested link utilization (number of lightpaths/$|W|$);

x_{pw} : =1, if lightpath p uses wavelength w; =0, otherwise;

y_l : =1, if phantom link l is selected; =0, otherwise;

[Definition 2.1]: Problem (P)

Objective min α

Subject to

$$\sum_{s\in S}\sum_{p\in P_s}\sum_{w\in W} x_{pw}\delta_{pl} \le \alpha |W| \qquad \forall l \in L \tag{2.1}$$

$$\sum_{p\in P_s}\sum_{w\in W} x_{pw} = d_s \qquad \forall s \in S \tag{2.2}$$

$$\sum_{s\in S}\sum_{p\in P_s} x_{pw}\delta_{pl} \le 1 \qquad \forall l \in L, w \in W \tag{2.3}$$

$$\sum_{s\in S}\sum_{p\in P_s} x_{pw}\delta_{pl} \quad \le \quad y_l \ \varphi_{lw} \qquad \forall w \in W, l \in L_{WSC} \tag{2.4}$$

$$\sum_{l\in L_{WSC}} y_l \mu_{lb}\sigma_{lv} \quad = \quad 1 \qquad \forall b \in B_n, v \in V_n^{in}, n \in N_{WSC} \tag{2.5}$$

$$\sum_{l\in L_{WSC}} y_l \mu_{lb}\sigma_{lv} \quad = \quad 1 \qquad \forall b \in B_n, v \in V_n^{out}, n \in N_{WSC} \tag{2.6}$$

$$x_{pw} \quad = \quad 0 \text{ or } 1 \qquad \forall p \in P_s, s \in S, w \in W \tag{2.7}$$

$$0 \ \le \ \alpha \ \le \ 1 \tag{2.8}$$

$$y_l \quad = \quad 0 \text{ or } 1 \qquad \forall l \in L_{WSC} \tag{2.9}$$

$$\sum_{s\in S_n}\sum_{p\in P_s} x_{pw}\delta_{pl} \quad \le \quad 1 \qquad \forall n \in N_{LSC}, l \in L \bigcup L_{WSC}, w \in W \tag{1.10}$$

■

The objective function is to minimize the highest utilization (α), namely the utilization on the most congested fiber link with the maximum number of lightpaths passing through. Constraint (2.1) requires that the number of wavelengths used on every link be less than that of the most congested link. Constraint (2.2) is the lightpath routing constraint, and restricts the lightpaths demands of all SD pairs to be satisfied. Constraint (2.3) indicates that for each link, there can be at most one lightpath using each wavelength. Constraints (2.3) and (2.7) jointly correspond to the wavelength continuity constraint. In particular, due to WSC nodes, Constraints (2.5), (2.6), and (2.9) delineate the possible configuration of WSC nodes. Constraint (2.4) states that paths can only pass through the phantom links determined by (1.5), (2.6), and (2.9). Finally, Constraint (2.10) is a redundant constraint [11] to Constraints (2.3) and (2.4), which is added for optimization purpose.

The problem is NP-complete [4], and is unlikely to obtain an exact solution for realistic networks in real-time. The problem is approximated using the LRH approach presented in the next sub-chapter.

2.2 Lagrangean Relaxation with Heuristics

Lagrangean relaxation (LR) [12-16] has been successfully employed to solve complex mathematical problems by means of constraint relaxation and problem decomposition. Particularly for solving linear integer problem, unlike the traditional linear programming approach that relaxes integer into non-integer constraints, the LR method generally leaves integer constraints in the constraint sets while relaxing complex constraints such that the relaxed problem can be decomposed into independent manageable subproblems. Through such relaxation and decomposition, the LR method as will be shown provides tighter bounds and shorter computation time on the optimal values of objective functions than those provided by the linear programming relaxation approach in many instances [14].

Essentially, the original primal problem is first simplified and transformed into a *dual* problem after some constraints are relaxed. If the objective of the primal problem is a minimization (maximization) function, the solution to the dual problem is a lower (upper) bound to the original problem. Such Lagrangean lower bound is a useful by-product in resolving the Lagrangean relaxation problem. Next, due to constraint relaxation, the lower bound solutions generated during the computation might be infeasible for the original primal problem. However, these solutions and the generated Lagrangean multipliers can serve as a base to develop efficient primal heuristic algorithms for achieving a near-optimal upper-bound solution to the original problem.

In the sequel, we first give the transformed dual problem and the derivation of the lower bound. We then present the primal heuristic algorithm for obtaining the upper-bound solution.

2.2.1 The Dual Problem and Lower Bound

In the relaxation process, Constraints (2.1), (2.3), and (2.4) are first relaxed from the constraint set. As shown in (2.11), the three expressions corresponding to the three constraints, are respectively multiplied by Lagrangean multipliers $\hat{s}$, $\hat{q}$, and $\hat{r}$, and then summed with the original objective function. Problem (P) is thus transformed into a dual problem, called Dual_P, given as follows:

[Definition 2.2]: Problem (Dual_P)

$$\text{Objective} \quad Z_{dual}(\boldsymbol{\rho}) = \min \left[\begin{array}{l} \alpha + \sum_{l \in L} \hat{s}_l \left(\sum_{s \in S} \sum_{p \in P_s} \sum_{w \in W} x_{pw} \delta_{pl} - \alpha \, |W| \right) \\ \quad + \sum_{l \in L} \sum_{w \in W} \hat{q}_{lw} \left(\sum_{s \in S} \sum_{p \in P_s} x_{pw} \delta_{pl} - 1 \right) \\ \quad + \sum_{l \in L_{WSC}} \sum_{w \in W} \hat{r}_{lw} \left(\sum_{s \in S} \sum_{p \in P_s} x_{pw} \delta_{pl} - y_l \varphi_{lw} \right) \end{array} \right]$$

$$= \min \left[\begin{array}{l} \left(1 - \sum_{l \in L} \hat{s}_l \, |W| \right) \alpha \\ + \sum_{s \in S} \sum_{p \in P_s} \sum_{w \in W} \left(\sum_{l \in L} (\hat{s}_l + \hat{q}_{lw}) \delta_{pl} + \sum_{l \in L_{WSC}} \hat{r}_{lw} \delta_{pl} \right) x_{pw} \\ - \sum_{l \in L_{WSC}} \sum_{w \in W} \hat{r}_{lw} \varphi_{lw} y_l - \sum_{l \in L} \sum_{w \in W} \hat{q}_{lw} \end{array} \right] \qquad (2.11)$$

Subject to Constraints (2.2), (2.5), (2.6), (2.7), (2.8), (2.9) and (2.10) where $\boldsymbol{\rho}$=($\hat{\boldsymbol{s}}$, $\hat{\boldsymbol{q}}$, $\hat{\boldsymbol{r}}$) is the non-negative Lagrangean multiplier vector. ■

To compute the Lagrangean multipliers, we adopt the subgradient method as delineated in the LRH algorithm outlined in Figure 2.2. By separating decision variable α, and decision variable vectors, $\boldsymbol{x}$, $\boldsymbol{y}$, Problem (Dual_P) in Equation (2.11) can be decomposed into three independent sub-problems- S1, S2 and S3. Specifically, we have

$$Z_{dual}(\boldsymbol{\rho}) = Z_{s1} + Z_{s2} + Z_{s3} - \sum_{l \in L} \sum_{w \in W} \hat{q}_{lw} \qquad (2.12)$$

where sub-problem S1 is given by

$$Z_{s1}(\hat{\boldsymbol{s}}) = \min \left(1 - \sum_{l \in L} \hat{s}_l \mid W \mid\right)\alpha ,$$

subject to Constraint (2.8);

sub-problem S2 is given by

$$Z_{s2}(\hat{\boldsymbol{q}}, \hat{\boldsymbol{r}}, \hat{\boldsymbol{s}}) = \min \sum_{s \in S} \sum_{p \in P_s} \sum_{w \in W} \left(\sum_{l \in L} (\hat{s}_l + \hat{q}_{lw}) \delta_{pl} + \sum_{l \in L_{WSC}} \hat{r}_{lw} \delta_{pl} \right) x_{pw} ,$$

subject to Constraints (2.2), (2.7) and (2.10);

and sub-problem S3 is given by

$$Z_{s3}(\hat{\boldsymbol{r}}) = \min \; - \sum_{l \in L_{WSC}} \sum_{w \in W} \hat{r}_{lw} \varphi_{lw} y_l ,$$

subject to Constraints (2.5), (2.6) and (2.9).

First, sub-problem S1 is to determine the decision variable, α. Clearly, α is set to 1 if the corresponding cost $1 - \sum_{l \in L} \hat{s}_l \mid W \mid$ is negative; otherwise α is set to 0. Thus, S1 requires O(L) computation time. Second, sub-problem S2 is to compute the decision

Algorithm LRH;
begin
  initialize the Lagrangean multiplier vector, $\hat{\boldsymbol{q}} \leftarrow \mathbf{0}$, $\hat{\boldsymbol{r}} \leftarrow \mathbf{0}$ and $\hat{\boldsymbol{s}} \leftarrow \mathbf{0}$;
  $UB \leftarrow 1$ and $LB \leftarrow 0$; /* upper and lower bounds on α */
  quiescence_age$\leftarrow$0;
  step size coefficient $\lambda \leftarrow 2$;
  for ($k \leftarrow 1$; $k \le$ *Iteration_Number*; $k \leftarrow k+1$) **do**
    run sub-problem S1;
    run sub-problem S2; /* by MSSP Algorithm */
    run sub-problem S3;
    $Z_{dual} = Z_{s1} + Z_{s2} + Z_{s3} - \sum_{l \in L} \sum_{w \in W} \hat{q}_{lw}$; /* Equation (2.12) */
    if ($Z_{dual} > LB$) $LB \leftarrow Z_{dual}$ and *quiescence_age*$\leftarrow$0;
    else *quiescence_age*$\leftarrow$*quiescence_age*+1; **endif**
    if (*quiescence_age* $\ge$ *Quiescence_Threshold*)
      $\lambda \leftarrow \lambda/2$ and *quiescence_age*$\leftarrow$0; **endif**
    run Primal Heuristic Algorithm;
    if ($ub < UB$) $UB \leftarrow ub$; **endif** /* ub is the newly computed upper bound */
    run update-step-size;
    run update-multiplier**;** **/*** by subgradient method ***/**
  endfor;
end

Figure 2.2 Lagrangean Relaxation with Heuristics (LRH).

variable vector, $\boldsymbol{x}$. There exist $|S_n|$ (one for each source node) independent problems, each of which is an edge-disjoint-path problem, starting from the given source node and destined to all destination nodes of the SD pairs with non-zero lightpath demands. Due to multiple wavelengths on each link, the network can be viewed as a layered graph with a total of $|W|$ layers, where each layer corresponds to each wavelength. Each layer then contains $(L+L_{wsc})$ links and $(N_{LSC}+N_{WSC})$ nodes. Notice that each link can be designated with unit flow capacity and a non-negative cost, for example, $\hat{s}_l + \hat{q}_{lw}$, for each non-phantom link.

Accordingly, the edge-disjoint-path problem for each source corresponds to a minimum-cost flow problem. Ultimately, with $|W|$ layers considered as a whole, the minimum-cost flow problem can be solved by the successive shortest path (SSP) algorithm [14]. However, the integrated problem requires high computational time complexity provided with a large value of $|W|$. To reduce the complexity, we exploit a modified successive shortest path (MSSP) algorithm as shown in Figure 2.3. In the algorithm, we treat each layer graph individually and performing incremental selection of minimum-cost edge-disjoint path (from one layer). The computational complexity of MSSP for each SD pair is $O(k(m+n\log_2 n))$, where $m=L+L_{WSC}$, $n=N_{LSC}+N_{WSC}$, and $k = max\{d_s, |W|\}$. All decision variables x's for S2 can be obtained by repeatedly applying the MSSP algorithm for all sources. Finally, sub-problem S3 is to resolve decision variable vector, $\boldsymbol{y}$. The problem can be further decomposed into $|N_{WSC}|$ (one for each WSC node) independent problems. Each of which, say WSC node i, further contains $|B_i|$ sub-problems. Each sub-problem can be optimally solved by a bipartite weighted matching algorithm. Thus for an $n \times n$ bipartite graph, the problem requires $O(n^3)$ computation time.

According to the weak Lagrangean duality theorem [14], Z_{dual} in Equation (2.12) is a lower bound of the original Problem (P) for any non-negative Lagrangean

multiplier vector $\rho = (\hat{s}, \hat{q}, \hat{r}) \geq 0$. Clearly, we are to determine the greatest lower bound. Equation (2.12) can be solved by the subgradient method, as shown as a part of the LRH approach delineated in Figure 2.2. As shown in Figure 2.2, the algorithm is run for a fixed number of iterations (i.e., *Iteration_Number*). (Notice that the algorithm can also be driven by given a termination requirement, as will be shown in the next sub-chapter). In every iteration, the three sub-problems (S1-S3) are solved (described above), resulting in the generation of a new Lagrangean multiplier vector

```
Algorithm MSSP;
begin
   for (each LSC node src∈N_LSC) do
      for (each wavelength w∈W) do       /* initialization */
         x←0;       /*decision variable vector*/
         π_w←0;       /*node potential vector*/
         for (each link l∈L_WSC) do cost_lw← r̂_lw ; endfor       /* phantom link cost */
         for (each link l∈L) do cost_lw← ŝ_l + q̂_lw ; endfor       /* physical link cost */
      endfor
      for (each sd-pair s∈|S_src|) do
         dest←destination(s);
         for (each w∈W) do
            ready_layer_w←"Unknown"; num-path-setup_s←0; endfor
         repeat
            for (each w∈W) do
               if (ready_layer_w="Unknown")
                  run Dijkstra's-s-shortest-path(cost, src, dest) on layer w;
                  if (the shortest path exists)
                     denote the path cost as k_w; ready_layer_w←"Yes";
                  else ready_layer_w←"No";    /* no more path on the layer for sd */
                  endif
               endif
            endfor
            if (there exists a layer w* with smallest cost k_w*
               and ready_layer_w*="Yes")
               update x_pw*, π_w*, cost_lw* ;       /* by SSP algorithm */
               num-path-setup_s←num-path-setup_s+1;
               ready_layer_w*←"Unknown";
            else return "infeasible"; endif       /* all ready_layer's are "No" */
         until num-path-setup_s=d_s;
      endfor
   endfor
end
```

Figure 2.3 MSSP Algorithm.

value. Then, according to Equation (2.12), a new lower bound is generated. If the new lower bound is tighter (greater) than the current best achievable lower bound (LB), the new lower bound is designated as the LB. Otherwise, the LB value remains unchanged.

Significantly, if the LB value remains unimproved for a number of iterations that exceeds a threshold, called *Quiescence_Threshold* (QT), the step size coefficient (λ) of the subgradient method is halved, in an attempt to reduce oscillation possibility. Specifically, in the "update-step-size" and "update-multiplier" procedures in Figure 2.2, the Lagrangean multiplier vector $\boldsymbol{\rho}$ is updated as $\boldsymbol{\rho}_{k+1} = \boldsymbol{\rho}_k + \theta_k \boldsymbol{b}_k$, where θ_k is the step size, determined by $\theta_k = \lambda_k (UB - Z_{dual}(\boldsymbol{\rho})) / \|\boldsymbol{b}_k\|^2$, in which λ_k is the step size coefficient, *UB* is the current achievable least upper bound obtained from the Primal Heuristic Algorithm described next, and $\boldsymbol{b}_k$ is a subgradient of $Z_{dual}(\boldsymbol{\rho})$ with vector size $|LW+L_{wsc}W+L|$.

2.2.2 The Primal Heuristic Algorithm and Upper Bound

The primal heuristic algorithm in the LRH approach is used to find an updated upper bound *ub*. Similar to the lower bound case, as given in Figure 2.2, if the new upper bound (*ub*) is tighter (smaller) than the current best achievable upper bound (UB), the new upper bound is designated as the UB.

As shown in Figure 2.4, the algorithm first settles the phantom links suggested by the solution to sub-problem S3 for all WSC nodes, reducing the problem complexity. The cost of each link is designated as the Lagrangean multipliers previously obtained. Clearly, the cost of unaccepted phantom links are set to ∞, excluding them from subsequent path consideration. The algorithm then repeatedly applies the Dijkstra's shortest path algorithm in an effort to satisfy the lightpath demands of all SD pairs.

At the end of the computation, the costs of those links associated with the selected wavelengths/paths are set to ∞ to prevent the links from being considered by other upcoming iterations. If the number of wavelengths (lightpaths) used on a link is greater than the current tightest lower bound multiplied by $|W|$, indicating potential congestion, the cost of the link is then scaled by multiplying by a constant, referred to as the penalty term. This is to avoid further lightpath set-up through this link. The process repeats until either the lightpath demands of all SD pairs are satisfied (i.e., feasible), or there is no remaining resource (i.e., infeasible) in the network.

```
Algorithm Primal Heuristic;
begin
   for (each wavelength w∈W) do
      for (each link l∈L_WSC ) do if (y_l=1) cost_lw← r̂_lw  else cost_lw←∞; endif endfor
      for (each link l∈L) do cost_lw← ŝ_l + q̂_lw ; endfor
   endfor
   for (each SD pair s←1; s ≤ |S|; s←s+1) do num-path-setup_s←0; endfor
   repeat
      for (each SD pair s←1; s ≤ |S|; s←s+1) do
         if (num-path-setup_s < d_s)
            src=source(s);
            dest=destination(s);
            run Dijkstra's-shortest-path(cost, src, dest) on each wavelength layer;
                  /* cost is vector of costs of all wavelengths and links */
            if (the shortest path exists)
               designate the wavelength associated with the shortest path as w*;
               for (all links l on the shortest path) do
                  cost_lw*:=∞;
                  if (the number of allocated paths on link l > LB×|W|)
                     for (each wavelength w∈W) do cost_lw←cost_lw×Penalty; endfor
                  else return "infeasible"; endif
               endfor
            endif
         endif
      endfor
   until all SD demand satisfied;
   update upper bound ub;
end
```

Figure 2.4 Primal Heuristic Algorithm.

2.3 Experimental Results

We have carried out a performance study on the LRH approach, and drawn comparisons between LRH and a linear programming relaxation (LPR)-based method [4] via experiments over randomly generated networks and the well-known NSFNET. Given the total number of nodes, say n, the greatest possible number of bi-directional links is $C(n, 2)$, where C is the combination operation. Then, for a randomly generated network with n nodes and connectivity ν, it is generated by randomly selecting $C(n,2)\times\nu$ out of the $C(n,2)$ bi-directional links of the network. In the experiments, we used 32 wavelengths on each fiber link (i.e., $|W|$=32) for all networks.

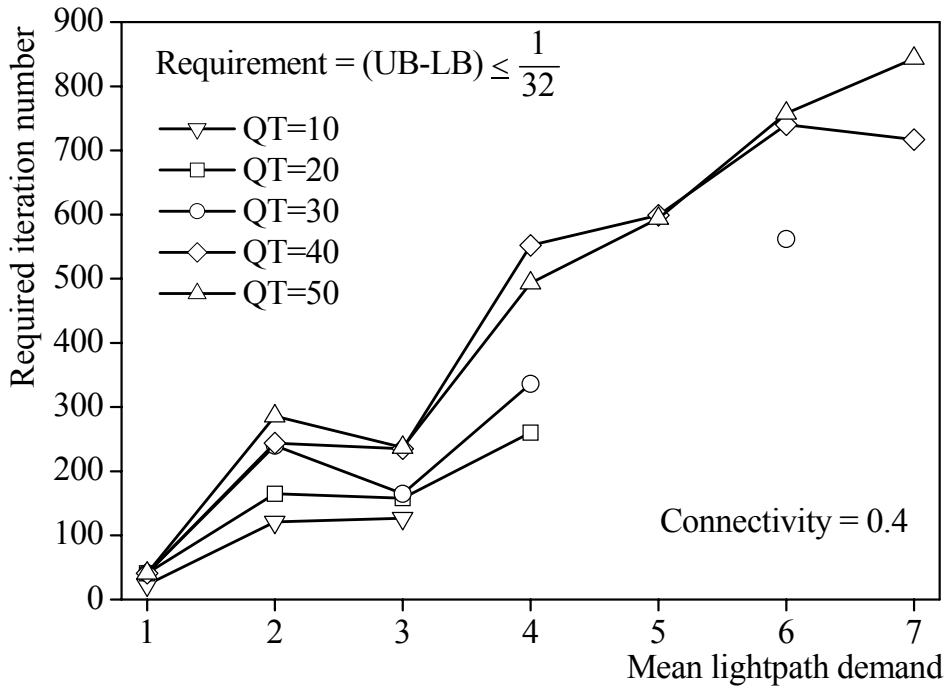
2.3.1 Performance Study

We carried out two sets of experiments over 15-node random networks with two connectivity ν = 0.4 and 0.8, which correspond to sparse and dense networks, respectively. In the first set of experiments, the LRH algorithm was terminated when the gap between the UB and the LB on α was less than or equal to one out of the maximum number of wavelengths, or the number of iterations exceeds 2000. While the former condition corresponds to reaching a near-optimal upper bound solution, the latter condition represents abnormal termination due to the failure of achieving such accuracy or solution infeasibility. We examine the total number of iterations required as a function of the mean lightpath demand under different QT values. Numerical results are plotted in Figure 2.5. Notice that the absence of data under certain demands corresponds to abnormal termination.

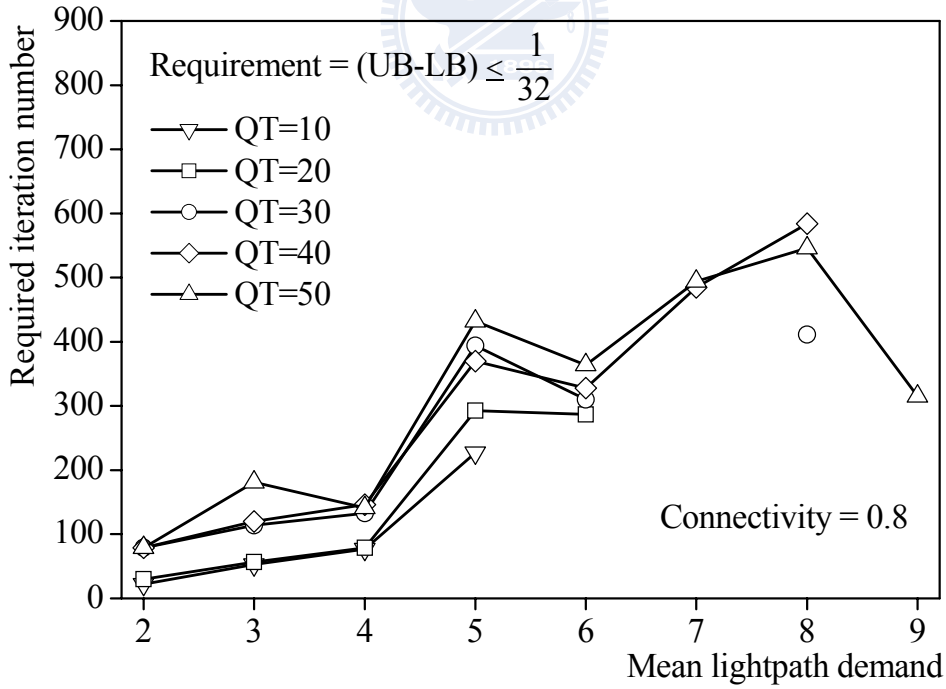
First, we observe that the dense network in general requires less number of iterations before reaching a near-optimal solution. Significantly, we discover from the figure that parameter QT plays a key role in the performance trade-off between

convergence speed and accuracy. Smaller values of QT, which imply frequent updates of the subgradient step-size coefficient, yield faster convergence to near-optimal solutions but at the cost of failing to reach accurate solutions under heavier lightpath demands. Greater QT values on the other hand result in completely opposite performance.

In the second set of experiments, the LRH algorithm was terminated when the number of iteration exceeded a pre-determined Iteration_Number, ranging from 0 to 1500. Numerical results are displayed in Figure 2.6. We study both the lower and upper bounds on α under different QT values. We observe that while the upper bound performance is irrelevant to QT, the lower bound performance is highly dependent on the QT setting in the same manner as above. Specifically, smaller QT values yield faster convergence but only to looser lower bounds, while larger QT values result in tighter lower bounds through gradual convergence over a larger number of iterations. This fact reveals that, by adjusting the QT value, the LRH approach is capable of balancing the trade-off between accuracy and efficiency for resolving various types of RWA problems.

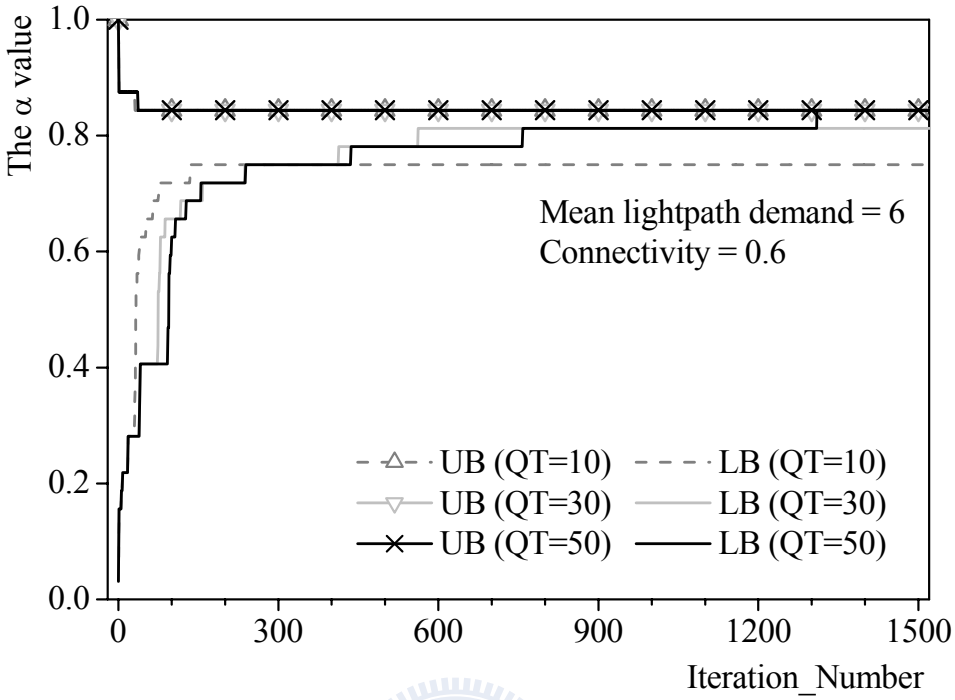


(a) Sparse network

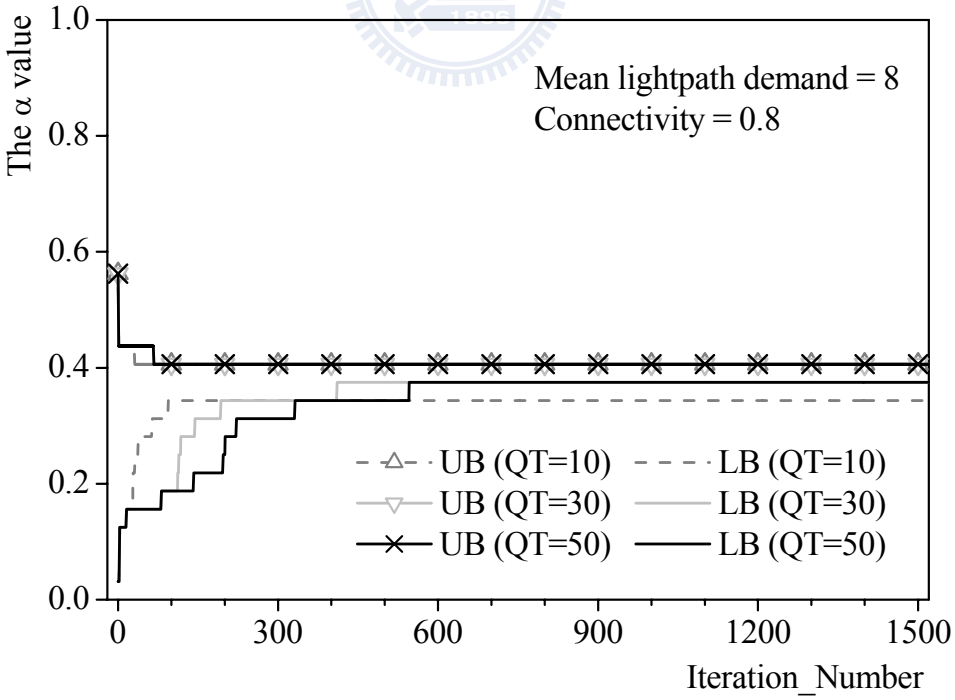


(b) Dense network

Figure 2.5 Convergence speed versus accuracy on the basis of using termination requirement.



(a) Sparse network



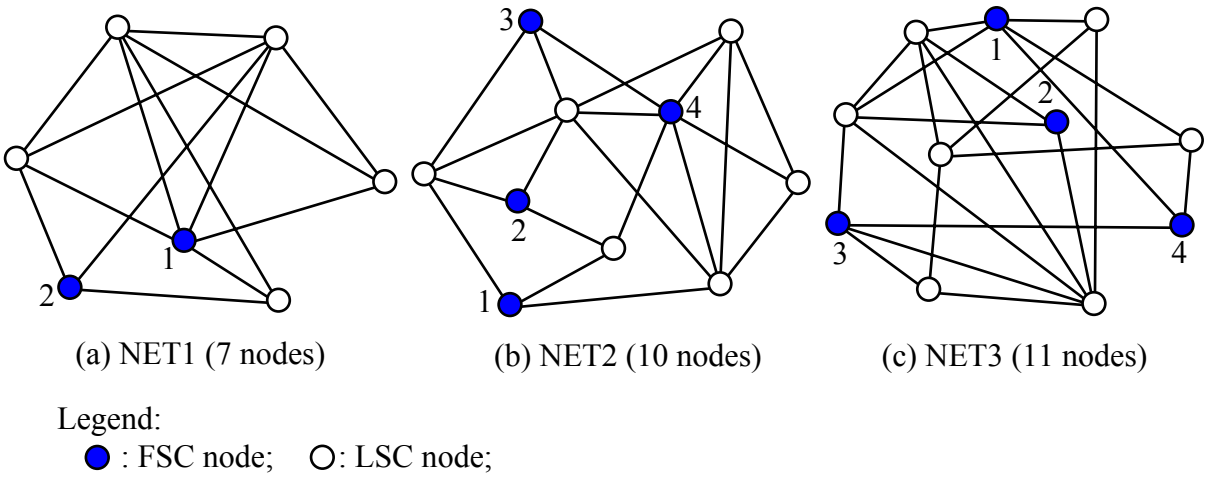
(b) Dense network

Figure 2.6 Convergence speed versus accuracy on the basis of using fixed iteration number.

2.3.2 Performance Comparisons

We further draw comparisons of accuracy and computation time between our LRH approach and a linear programming relaxation (LPR)-based method [4] over three randomly generated networks and the well-known NSFNET. For generating networks, it is impractical to experiment on networks with smaller numbers of nodes and links. However, for networks with greater than 11nodes, we experienced that the computation time using the LPR-based method became unmanageable. In addition, for simplicity, in the experiment we disregarded WSC nodes on randomly generated networks, and only considered WSC nodes on NSFNET.. Thus, we considered three random networks, NET1, NET2, and NET3, as shown in Figure 2.7. NET1 consists of seven nodes including two FSC nodes, and 14 bi-directional links, corresponding to a connectivity (v) of $0.6\overline{6}$. NET2 consists of ten nodes including two FSC (nodes 1-2) or four FSC (nodes 1-4) nodes, and 20 bi-directional links, corresponding to a connectivity (v) of $0.4\overline{4}$. Finally, NET3 consists of 11 nodes including two FSC (nodes 1-2) or four FSC (nodes 1-4) nodes, and 22 bi-directional links, corresponding to a connectivity (v) of 0.4. Numerical results are demonstrated in Figures 2.8-2.12.

In the computation using our LRH approach, we adopted QT=50 and three



(a) NET1 (7 nodes) (b) NET2 (10 nodes) (c) NET3 (11 nodes)

Legend:
● : FSC node; ○: LSC node;

Figure 2.7 Random Network topology.

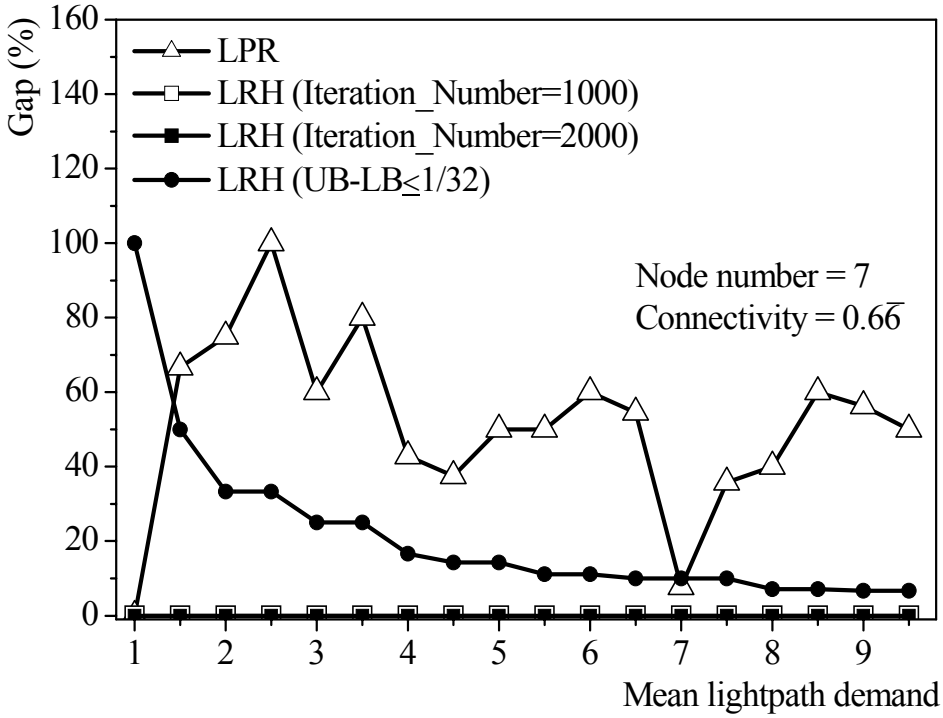
different termination criteria. The three criteria are: Iteration_Number=1000, 2000, and requirement (UB-LB)≤1/32. The algorithm was written in the C language and operated on a PC running Windows XP with a 2.53GHz CPU power. In the LPR-based method, by removing Constraints (2.7) and (2.9), the original integer linear programming (ILP) problem is relaxed to a linear programming (LP) problem. Thus, the solution to the relaxed problem is a legitimate lower bound of the original ILP problem. The upper bound is then obtained according to the randomization procedure proposed in [4]. In the experiment, the LP problem was solved using the *CPLEX* software, operating in the same PC environment. For both approaches, the accuracy is measured in terms of the Gap(%) which is defined as the ratio of the difference of the UB and LB values to the LB value in percentage.

First of all, we draw comparisons of accuracy and computation time between the LRH approach and the LPR method for random network NET1, as plotted in Figure 2.8. Notice that the LRH approach using fixed iteration numbers outperforms the LPR method in accuracy under all lightpath demands. However, it appears that the LRH method using the termination requirement yields a high gap under low demands. This is only due to the magnification of the gap resulting from being divided by a small LB value under low demands. In particular, under demand=1, the algorithm was terminated with UB=2/32 and LB=1/32, resulting a 100% gap. Surprisingly, we discover from part (b) of Figure 2.8 that the LPR method requires less computation time than that of the LRH approach using fixed iterations. This indicates that LPR is an efficient approach particularly for smaller sized networks.

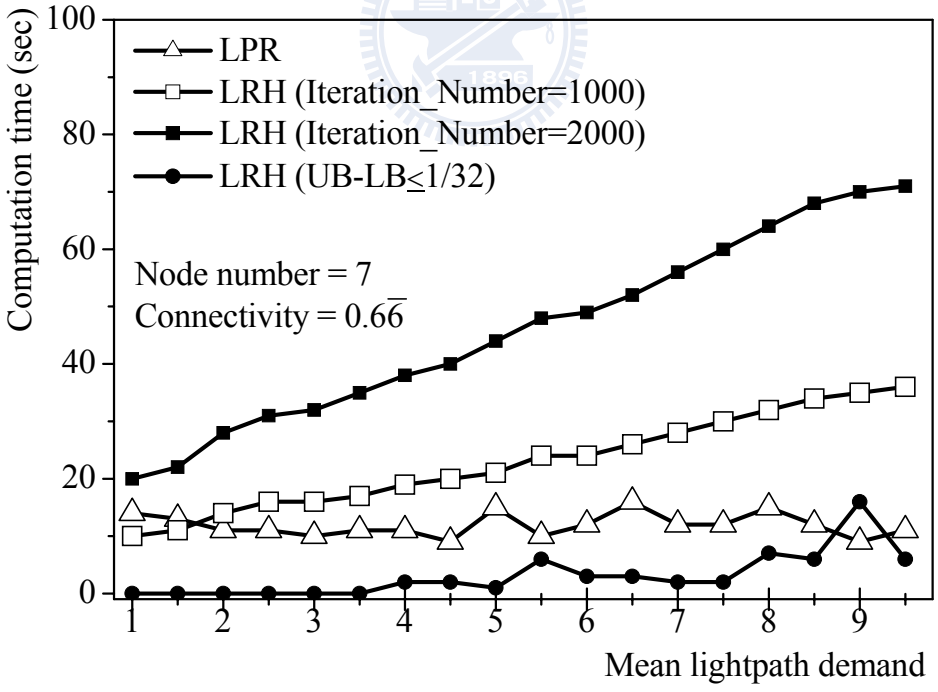
For random networks with size over ten nodes (NET2 and NET3) as shown in Figures 2.9-2.12, the LPR method yields larger gaps, namely poorer accuracy, and demands exponentially increasing computation time. In contrast, the LRH approach achieves identical lower and upper bounds, namely the optimal solutions under

several lightpath demand cases. In fact, we discover that, both LRH and LPR approaches achieve tight lower bounds. Significantly, the LRH heuristic algorithm arrives at much improved upper bounds due to the use of the Lagrangean multipliers derived upon seeking the Lagrangean relaxation solution. It is worth noticing that the results of the LRH approach using the termination requirement are not shown in Figures 2.9-2.12. This is due to its high accuracy and low computation time, yielding impossible plotting within the figures. Specifically, we discover from part (a) of Figures 2.9-2.12 that the LRH approach using the 1000 iterations achieves as high accuracy as that using the 2000 iterations under most demand cases. Significantly, the approach using the (UB-LB)≤1/32 requirement for NET2 reaches the small gap within only a total of (8,40,164,480,339,287,137,424) iterations for lightpath demands ranging from 1 to 8, respectively.

Furthermore, as shown in part (b) of Figures 2.9-2.12, the LRH approach outperforms the LPR method in computation time by at least one order of magnitude under all cases. Notice that, the LRH approach using the termination requirement incurs exceptionally low computation times that are equal to (0,1,7,24,18,17,9, 31) for eight lightpath demands, respectively. In this case, compared to the LPR method, the LRH approach offers an improvement of computation time by more than two orders of magnitude.

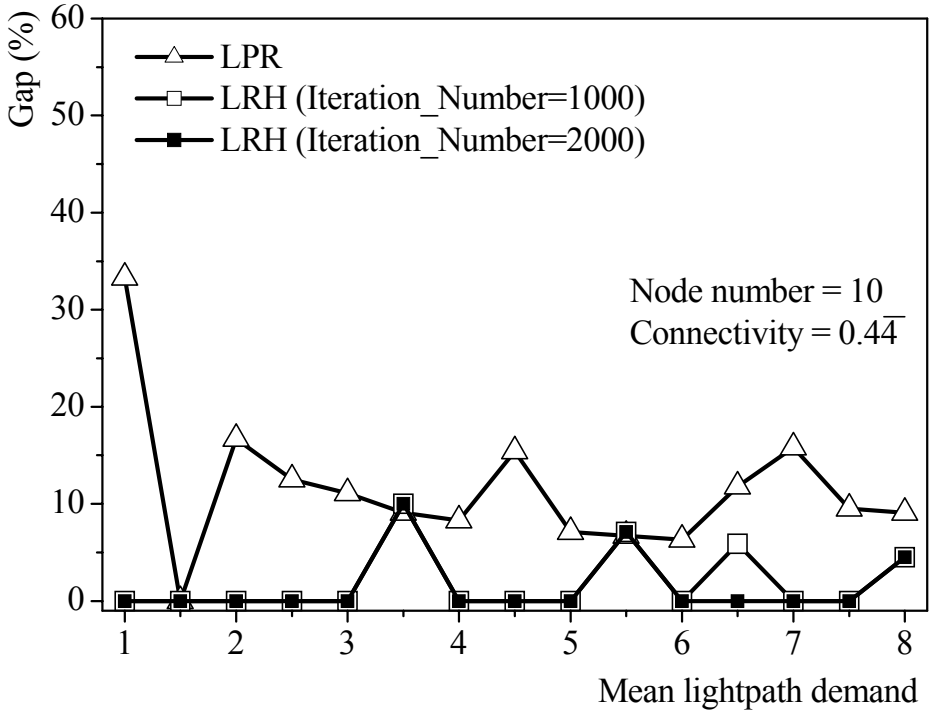


(a) Accuracy.

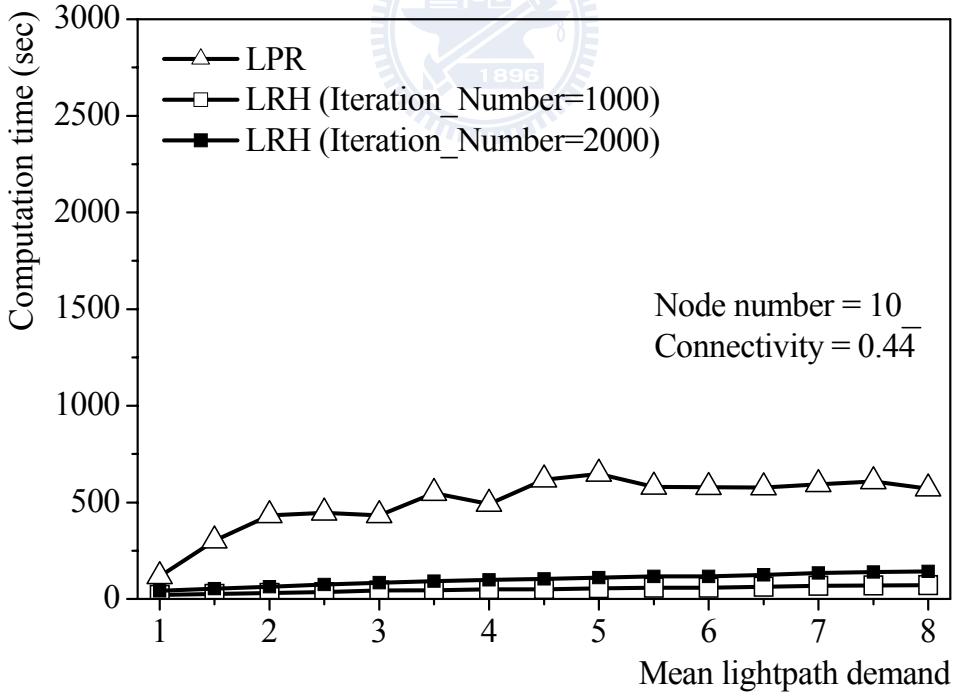


(b) Computation time.

Figure 2.8 Comparisons of accuracy and computation time for random network NET1.

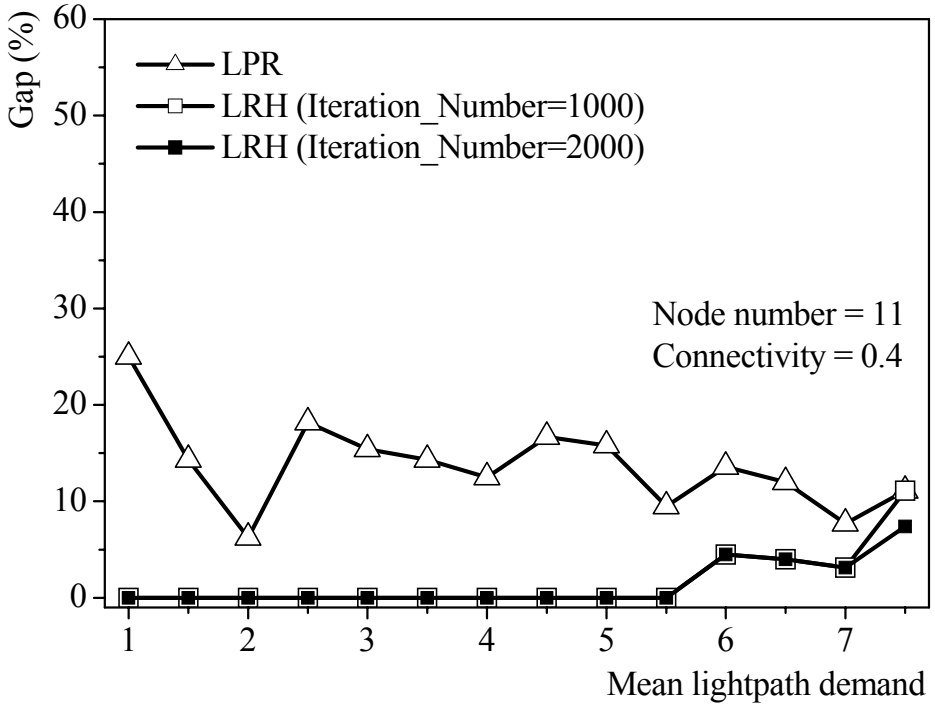


(a) Accuracy.

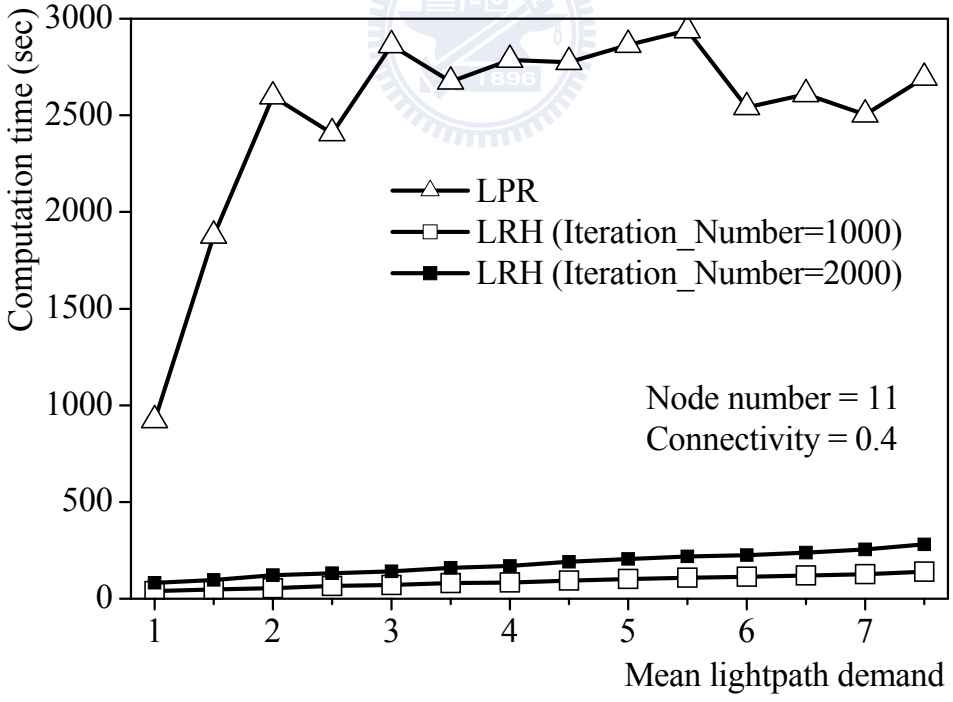


(b) Computation time.

Figure 2.9 Comparisons of accuracy and computation time for random network NET2 with two FSC nodes.

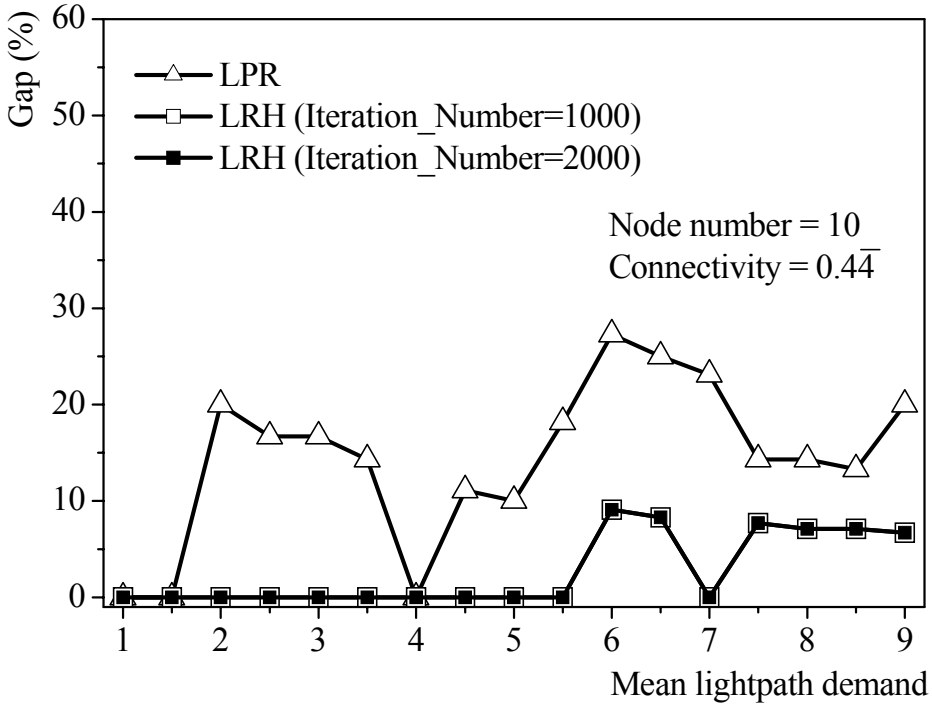


(a) Accuracy.

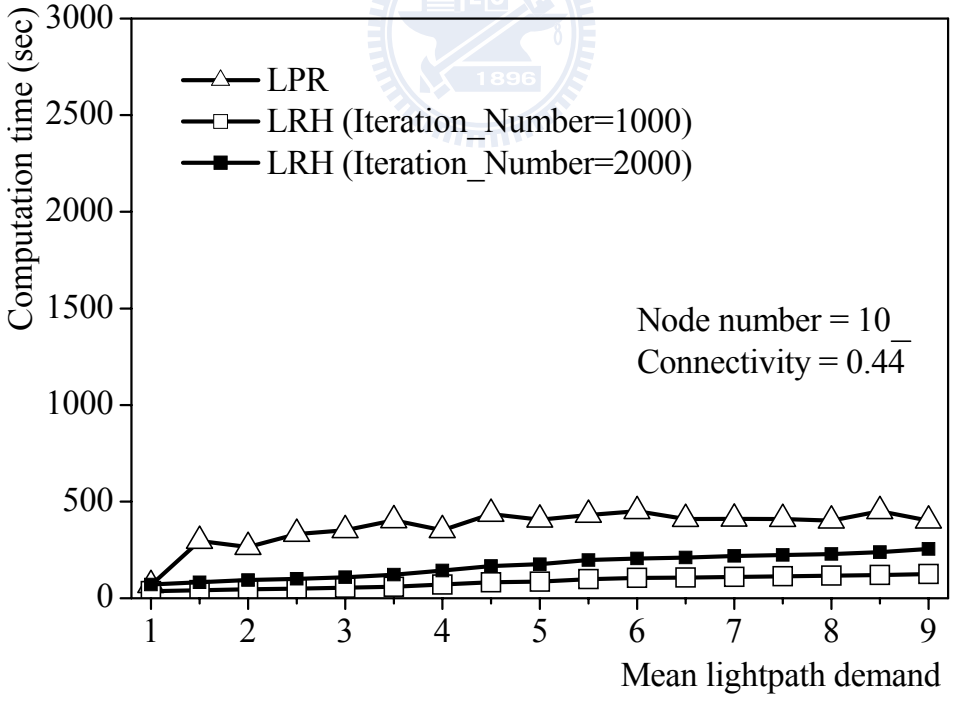


(b) Computation time.

Figure 2.10 Comparisons of accuracy and computation time for random network NET3 with two FSC nodes.

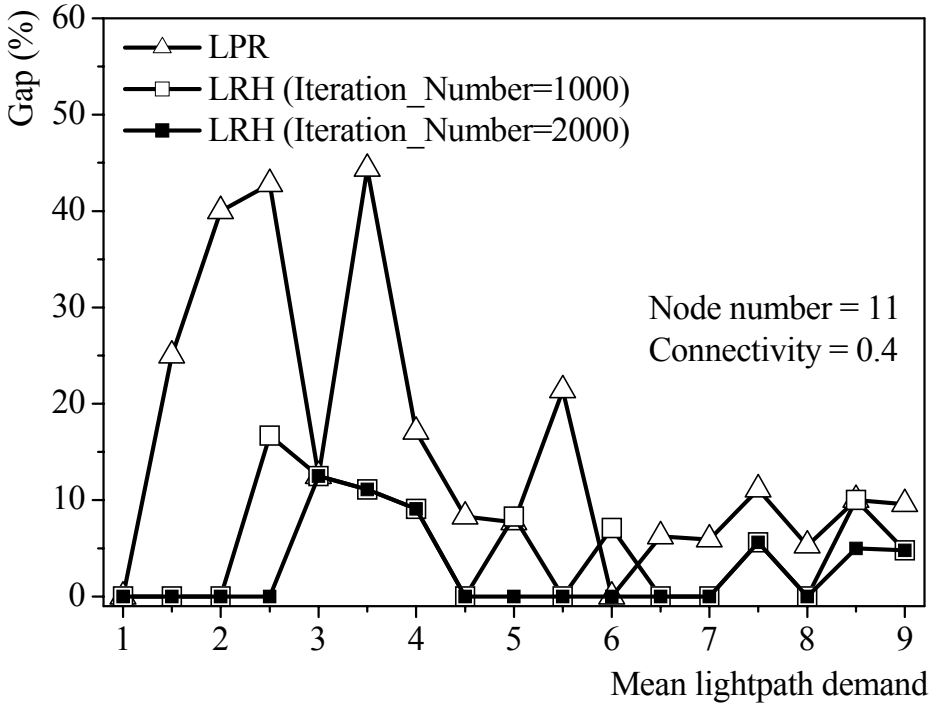


(a) Accuracy.

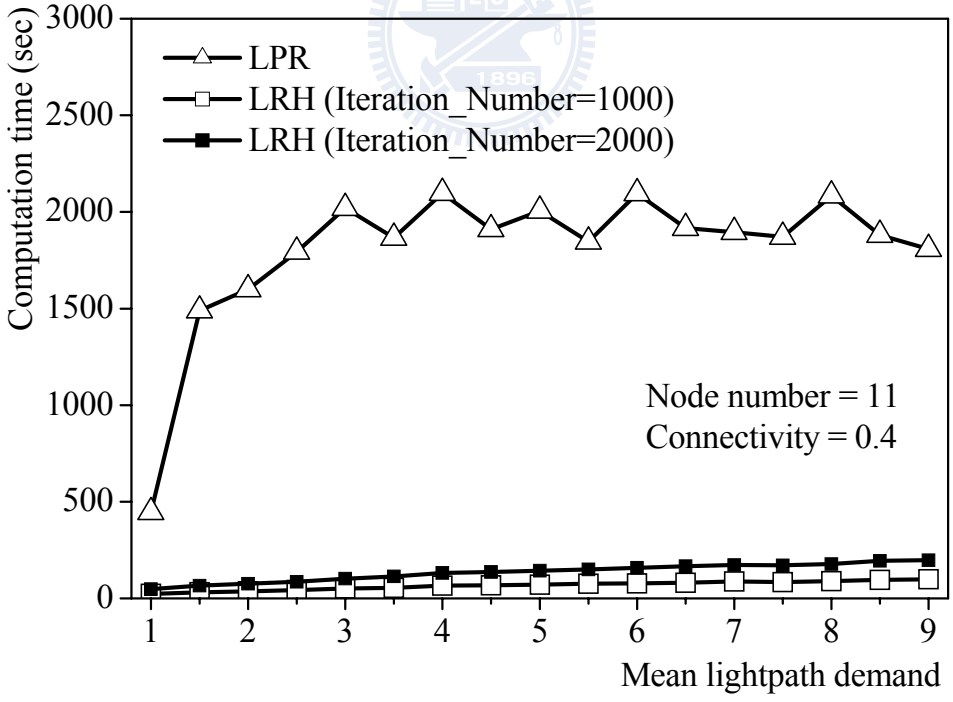


(b) Computation time.

Figure 2.11 Comparisons of accuracy and computation time for random network NET2 with four FSC nodes.



(a) Accuracy.



(b) Computation time.

Figure 2.12 Comparisons of accuracy and computation time for random network NET3 with four FSC nodes.

Finally, we draw comparisons of accuracy and computation time among three variants of the LRH approach and the LPR method over the well-known NSFNET [17]. The NSFNET we adopted in the experiment is displayed in Figure 2.13. Numerical results are plotted in Figure 2.14. The traffic demands (i.e., the number of lightpaths requested) for all SD pairs are randomly determined with their mean value given in the x-axis of all graphs. Moreover, the gap in the y-axis of the graphs in the figure is defined as the ratio of the difference of the upper and lower bounds to the lower bound in percentage. We observe that, compared to the three variants of LRH, the LPR method gives the poorest accuracy guarantee and incurs high CPU computation time.

To complete the simulation study, in order to further observe the performance of our LRH approach for large sized networks, we carried out experiments on two well-known networks, i.e., USA and ARPA, as shown in Figure 2.15, respectively. The USA network consists of 28 nodes including three FSC nodes and 90 bi-directional links, corresponding to a connectivity (v) of 0.12. The ARPA network has 61 nodes including four FSC nodes and 148 bi-directional links, which corresponds to a connectivity (v) of 0.04. There are 64 wavelengths on each fiber for both networks. Numerical results are displayed in Figure 2.16 and Figure 2.17.

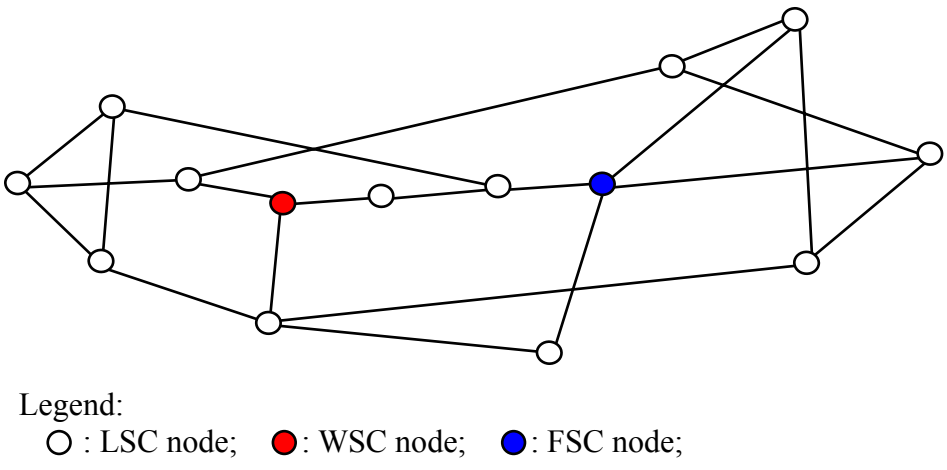
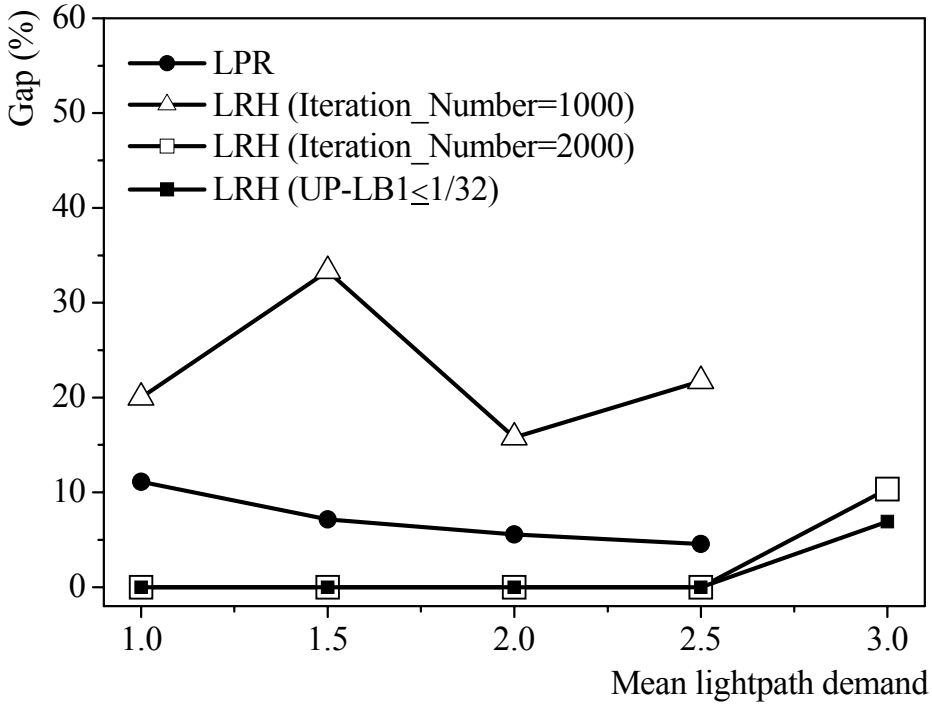
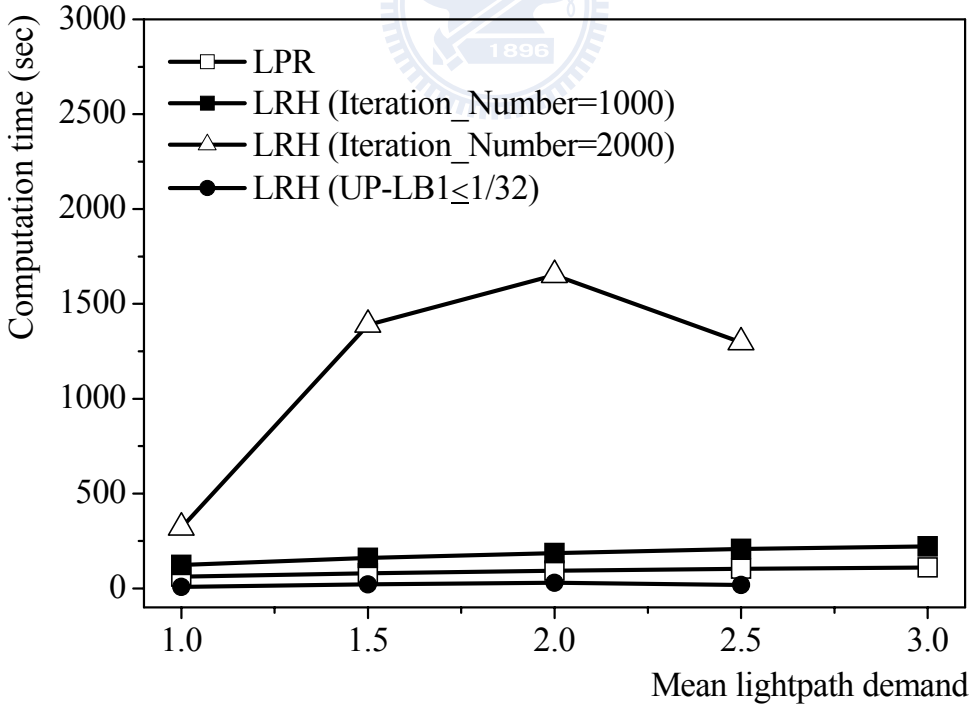


Figure 2.13 NSFNET Network.



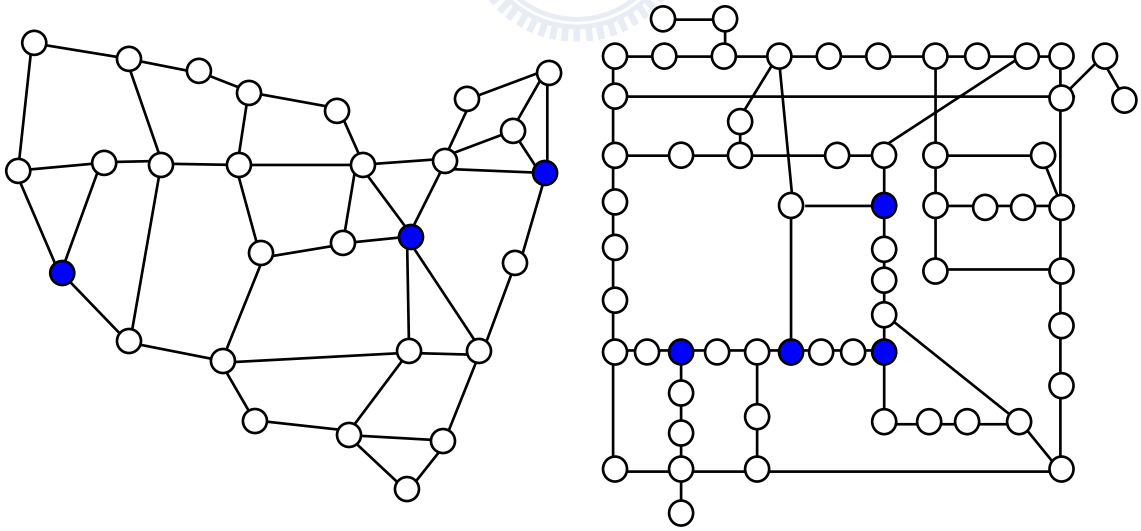
(a) Accuracy.



(b) Computation time.

Figure 2.14 Comparisons of accuracy and computation time for NSFNET.

In the experiment, we adopted QT=50 and two different termination criteria, namely Iteration_Number=500 and 1000. For the USA network, LRH achieves a guarantee of no more than 8% gap in less than 2800 seconds computation time between the upper and lower bounds under both termination criteria. For the ARPA network, the LRH achieves a guarantee of no more than 9.3% gap in less than 9400 seconds computation time. We particularly observe from Figure 2.17(a) that, in the ARPA network, the accuracy of the LRH approach based on the 500-iteration termination criterion is as high as that based on the 1000-iteration termination criterion under most lightpath demand cases. This again demonstrates the superiority of the LRH approach to the RWA+ problem with respect to both computation accuracy and time complexity for large sized networks.

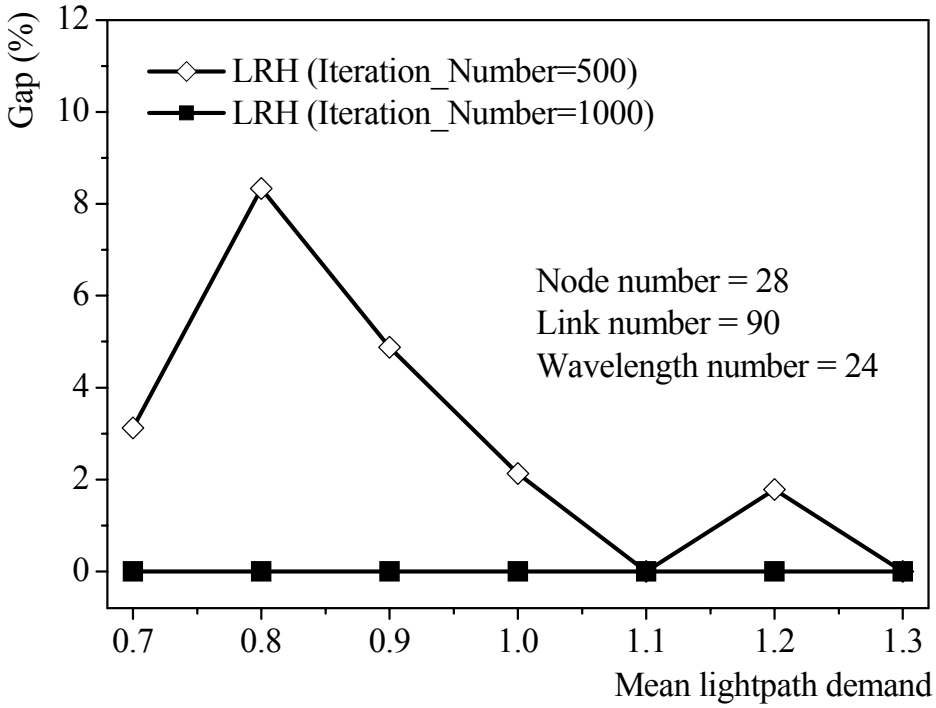


(a) USA network topology (28-node and 90-link)

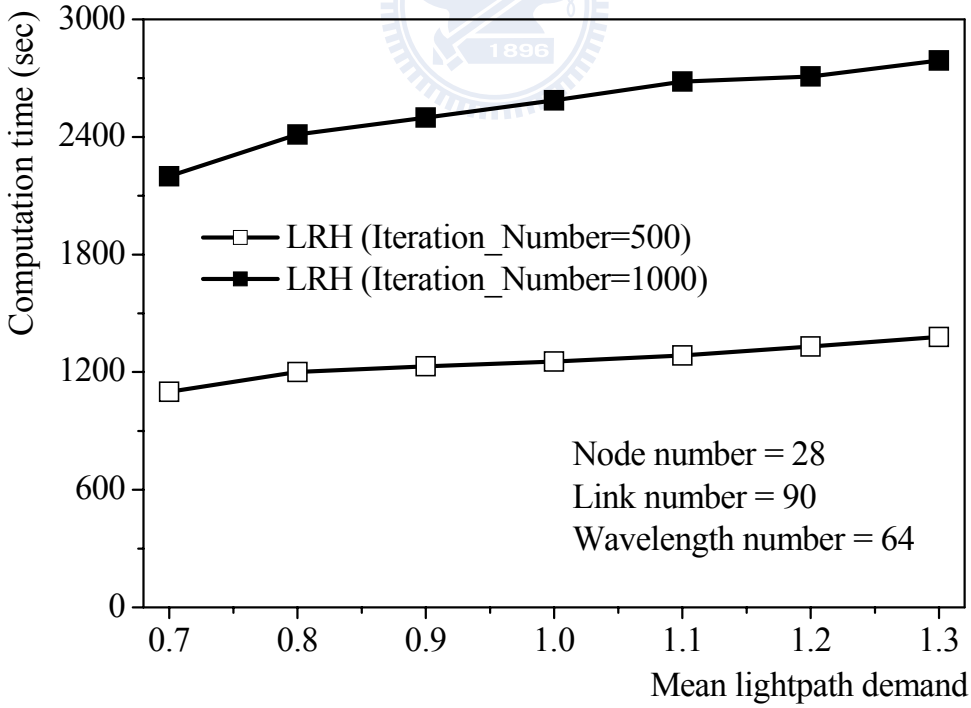
(b) ARPA network topology (61-node and 148-link)

Legend:
● : FSC node; ○: LSC node;

Figure 2.15 Two large sized well-known networks (64 wavelengths).

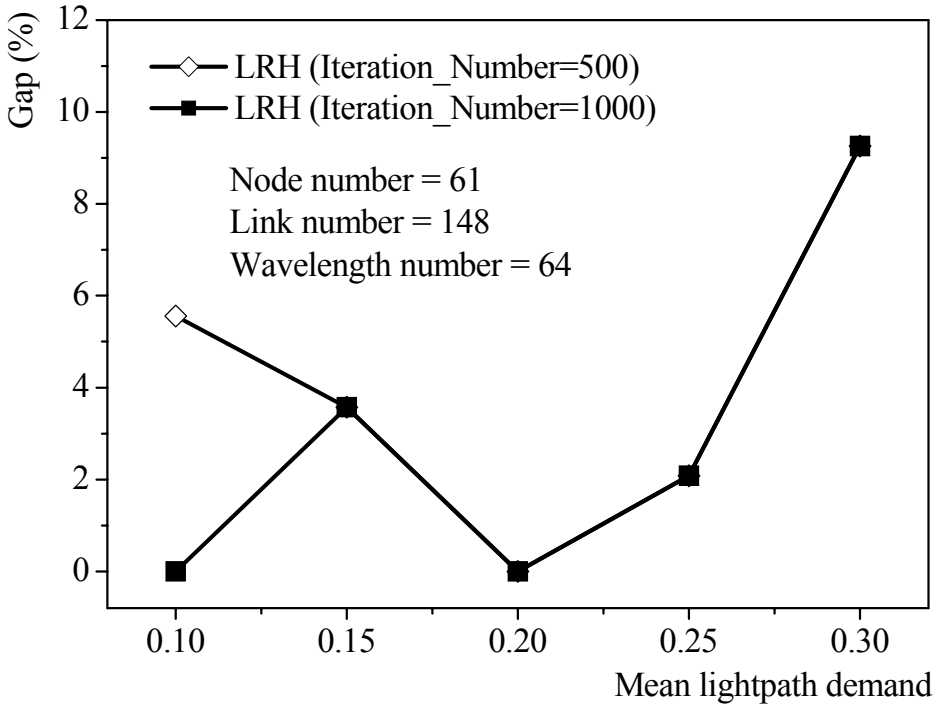


(a) Accuracy.

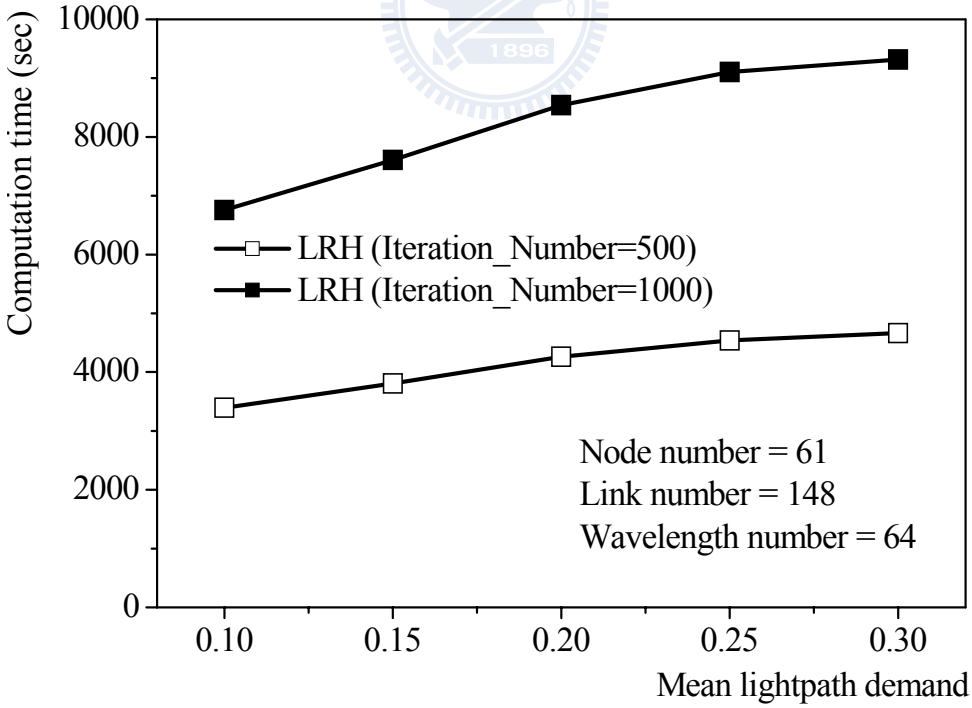


(b) Computation time.

Figure 2.16 The performance of LRH approach for USA network.



(a) Accuracy.



(b) Computation time.

Figure 2.17 The performance of LRH approach for ARPA network.

2.4 Summary of Part I Thesis

In the first part of the thesis, we have resolved a RWA+ problem using a Lagrangean Relaxation based approach augmented with an efficient primal heuristic algorithm, called LRH. With the aid of generated Lagrangean multipliers and lower bound indexes, the primal heuristic algorithm of LRH achieves a near-optimal upper-bound solution. A performance study delineated that the performance trade-off between accuracy and convergence speed can be manipulated via adjusting the Quiescence Threshold parameter in the algorithm. We have drawn comparisons of accuracy and computation time between LRH and the linear programming relaxation (LPR)-based method under various networks. Experimental results demonstrated that, particularly for small to medium sized networks, the LRH approach using a termination requirement is superior to the LPR method and fixed-iteration-based LRH, in both accuracy and computational time complexity, Furthermore, for large sized networks, i.e., the USA and ARPA networks, numerical results showed that LRH achieves a near optimal solution with acceptable computation time. The above numerical results justify that the LRH approach can be used as a dynamic RWA+ algorithm for large sized networks.

Chapter 3. Optical Packet Switching System and Scheduling Algorithm Design in Metropolitan Network

Future optical wavelength division multiplexing (WDM) [18] networks, especially the metropolitan and local network, are expected to flexibly and cost-effectively satisfy a multitude of applications with diverse quality of service (QoS) requirements. Such facts bring about the need of exploiting the optical packet-switching (OPS) [18,19,20] paradigm that advocates efficient sharing of wavelengths among multiple connections. Nevertheless, OPS systems still face several technological limitations, such as optical random access memory (RAM) and optical signal processing. Thus, the OPS system we study in this work employs fiber delay line (FDL) based-optical buffer, and *almost-all* optical switches in which packet payloads remain in the optical domain and only the control headers are processed electronically. It is known that such OPS systems have a stringent *time-bound requirement* [20] for the duration between the header removal/ insertion from/onto its payload.

Basically, a general OPS system consists of four components that are crucial to the performance and economy of the system. They are the packet scheduler, the space switch, the optical buffer, and the wavelength converter [21]. Due to the stringent time-bound requirement mentioned above, the packet scheduler demands incremental and exceedingly fast scheduling computation. The latter three components are discussed in more detail in the following. First, space switches can be categorized as being non-blocking or blocking [22,23]. Non-blocking switches are mostly used for traditional electronic circuit switching systems. Due to requiring a large number of switching elements, they are less scalable and economically unfeasible in the optical domain. There is another type of non-blocking switches, called rearrangeable switches

(e.g., Benes network), which route new input-output connections by rearranging all other existing connections. Rearrangeable switches are more scalable than their non-blocking counterparts, but they require more complicated scheduling (or rearranging) algorithms, which may fail to meet the time-bound requirement. Most work done on OPS systems considers non-blocking space switch architectures.

For blocking switches, Banyan switches enable self-routing and require the least number of switching elements. It is the most scalable and economic architecture. The price paid, however, is internal contention, also referred to as switch contention, when two packets attempt to access the same internal link in the switch [22]. Due to exceedingly high cost of fast optical switches (and switching elements), in our work we consider the blocking Banyan space switch to be a promising candidate for future optical networks. Traditionally, to resolve internal contentions in the electronic Banyan switch [22,24], numerous methods have been proposed. The most prevailing method, called buffered Banyan switch, queues contending packets at the input ports through using RAM-based buffers. Such a buffering strategy becomes impractical in the optical domain. In my thesis, the main goal of the second network problem has been to incorporate a scalable Banyan-like switch architecture together with a fast, QoS-enabled incremental packet scheduling algorithm to resolve internal and output contentions.

Second, optical buffers are currently achieved by either increasing the light path distance via fiber delay lines (FDL), or slowing down the light velocity. The slow-light technologies [25,26] have been shown to have limited capacity and delay-bandwidth product [26]. For this reason, the FDL-based optical buffer remains a practical option for resolving contentions in the time dimension. The FDL-based optical buffer can be applied under various buffering strategies [24]: input buffering, output buffering, and shared buffering. While input (output) buffering has a separate

buffer for each input (output) port, the shared buffering allows buffers to be shared among multiple inputs and/or outputs. There are two major FDL-based buffering structures: feedback or feed-forward [27]. The feedback structure can support dynamic buffering durations but at the expense of additional hardware to maintain signal quality [28,29,30]. By contrast, the feed-forward FDL structure only supports fixed buffering durations [31,32] but assures better signal quality. Thus, the feed-forward structure is generally preferred over the feedback-based counterpart.

Third, tunable optical wavelength converters (TOWCs) offer an alternative to contention resolution in the space (wavelength) dimension. TOWCs can be realized by three key methods [33]: cross-gain modulation (XGM) [34], cross-phase modulation (XPM) [35], and four-wave mixing (FWM) [36]. By taking advantage of the gain saturation phenomenon of the semiconductor optical amplifier (SOA), the XGM scheme inversely transfer the modulation of the input wavelength's amplitude to that of the output wavelength's amplitude. The scheme is simple and polarization insensitive due to the use of polarization insensitive SOA. However, its major drawback is the reduction of the extinction ratio for up-converted signals. In the XPM scheme, the refractive index of one arm of the SOA-integrated interferometer is modulated in accordance with the source signal's intensity. As a result, the two-arm XPM converter generates either constructive or destructive interference effect on the target wavelength. Such effect entails the encoding of the source bit stream onto the target wavelength. The scheme outperforms XGM in the extinction ratio and conversion range at the cost of more complex components. Nevertheless, the XPM converters are limited to amplitude modulation formats and require accurate control of the SOA bias [33]. Finally, an FWM converter is based on the four-wave-mixing nonlinear effect of SOA for converting wavelengths. The FWM scheme is attractive because of being able to convert a group of wavelengths simultaneously, and

transparent to the modulation format and data rate. With the method proposed in [36], the FWM converter can achieve high conversion efficiency for large conversion ranges. Owing to that TOWCs impose a high cost penalty on OPS systems, much research work has focused on the reduction of the cost via the sharing of TOWCs and/or the use of limited-range TOWCs [33,37-42]. In our work, we particularly focus on the exploitation of a FWM-based TOWC method to achieve packet preemption.

In the second part of my thesis, we present the architecture of a QoS-enabled pseudo-Banyan optical packet switching system (QBOPSS) for WDM networks. QBOPSS boasts three crucial features. First, it embodies a group of downsized pseudo-Banyan space switches (PBSs), each of which is of Banyan structure but made from unconventional two-by-two switching elements. Besides traditional cross and bar options, each switching element allows the merging of two packets in two different wavelengths from two inputs to the same output. Moreover, each PBS handles the switching solely for a cluster of wavelengths. Such cluster-based optical switch design aims at trading off limited statistical multiplexing gains for higher system scalability. Second, QBOPSS adopts a handful of feed-forward optical buffers that are applied based on the output and shared buffering strategies. Such design, as will be shown, yields drastic reduction in packet loss probability in an economical fashion. Third, by incorporating four-wave-mixing (FWM) [36] converters at the output section, QBOPSS supports *optical packet preemption* by permitting higher-priority packets to preempt lower-priority packets having already been in the FDLs, thereby achieving effectual QoS differentiation.

Packet scheduling in QBOPSS is performed by a QoS parallel incremental scheduling (QPIS) algorithm. Given a set of newly-arriving packets per time slot, QPIS minimizes the loss probability for high-priority packets while maximizing system throughput and satisfying two constraints, i.e., switch- contention free, and

buffer-contention free. Significantly, we prove that QPIS is incremental, i.e., the computed packet sets within each time slot are monotonically non-decreasing. We then present a hardware system architecture to demonstrate the parallel implementation of QPIS. As will be shown, QPIS achieves a near-optimal solution with an exceptionally low computational complexity, $O(NW \times \log_2(NMW))$, where N is the number of input ports, and W and M the numbers of external and internal wavelengths, respectively.. From simulation results that pit the QPIS algorithm against two other algorithms, we show that QPIS outperforms these algorithms on computational complexity, packet loss probability, and QoS differentiation.

The remainder of my second part thesis– Chapter 3 is organized as follows. In Chapter 3.1, we present the general architecture of QBOPSS, and draw comparisons between QBOPSS and several optical space switch structures with respect to hardware cost and signal quality. In Chapter 3.2, we describe the QPIS algorithm, and give the proof the incremental property. In Chapter 3.3, we present the hardware parallel system architecture and derive the computational complexity. Experimental results are then shown in Chapter 3.4. Finally, important remarks are summarized in Chapter 3.5.

3.1 QBOPSS: New OPS System Design and Assessment

In this sub-chapter, we first detail the new designed optical packet switching (OPS) system, QoS-enabled pseudo-Banyan optical packet switching system (QBOPSS). Then, we finish this sub-chapter by comparing QBOPSS and several prevailing OPS system structures with respect to component counts and output signal quality.

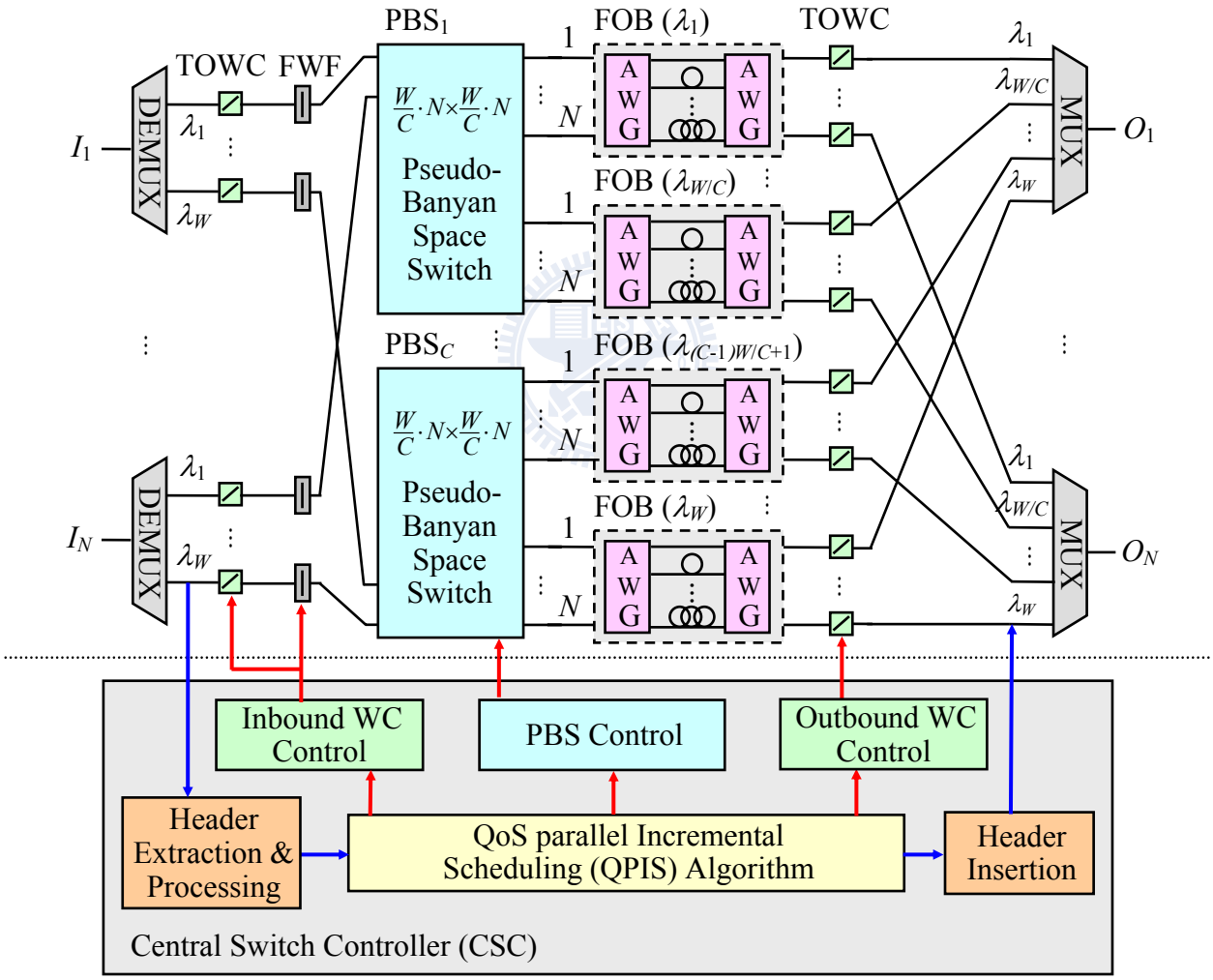
3.1.1 QBOPSS Architecture

QoS-enabled pseudo-Banyan optical packet switching system (QBOPSS) is a synchronous system that supports fixed-size packets of different priorities. As shown in Figure 3.1, it consists of two parts: the optical switch and central switch controller (CSC). First, before entering the optical switch, all packets are required to be time-aligned, namely synchronization. Such synchronization operation can be implemented by the use of cascaded switched fiber delay lines [43,44]. While each packet header that carries the label and QoS (priority) information is electronically processed by the CSC, the payload travels within the switch all-optically. The header is modulated with its payload based on the superimposed amplitude shift keying (SASK) technique [45].

The optical switch consists of four sections: input, space switch, output buffer, and output sections. In the input section, there are N input fibers each carrying W wavelengths, and $N \times W$ tunable optical wavelength converters (TOWCs). Each TOWC is followed by a fixed wavelength filter (FWF). After DEMUX, for each packet a TOWC converts its input wavelength to an internal wavelength that corresponds to a free space in the output optical buffer. If a newly-arriving packet observes a full system, it is converted to a dump wavelength causing the packet to be blocked by the

FWF. It is noted that the FWF is a broadband filter which passes whole internal wavelength band and only filters out the dump wavelength.

In the space switch section, there are C independent pseudo-Banyan space switches (PBSs) that are responsible for switching C clusters of wavelengths, respectively. More specifically, the k_{th} PBS, i.e., PBS_k, switches W/C input wavelengths (from $\lambda_{(k-1)W/C+1}$ to $\lambda_{kW/C}$) for each of N fibers, to W/C output wavelengths (from $\lambda_{(k-1)W/C+1}$ to $\lambda_{kW/C}$), leading the switch size to $N(W/C)\times N(W/C)$, where C is the



Legend:

Payload; Header; Control Signal;

TOWC: Tunable Optical Wavelength Converter; WC: Wavelength Converter;

FWF: Fixed Wavelength Filter; PBS: Pseudo-Banyan Space Switch;

FOB (λ_i): FDL Optical Buffers for λ_i; AWG: Arrayed Waveguide Grating;

Figure 3.1 QBOPSS- system architecture.

total number of clusters in the switch. Within each PBS, say of size $m \times m$, there are $(m/2) \times \log_2 m$ two-by-two switching elements, each of which can be constructed by four semiconductor optical amplifiers (SOAs) [18]. Each SOA can be considered to be an on/off switch to select/unselect the packet, thus forming four different switching decisions, namely cross, bar, and two merging options. Specifically, the merging option allows two packets with different internal wavelengths to be switched from two input ports to the same output port of PBS. These two packets ultimately will depart from the system through the same output wavelength and fiber, after receiving different delays. Moreover, like Banyan switches, the PBS maintains the self-routing property that allows packets to be uniquely switched according to their output port (N) and assigned output wavelength (W/C).

The output buffer section contains W FDL optical buffers (FOBs) for W wavelengths, respectively. Each FOB is shared by all N output ports. An FOB is composed of a pair of arrayed waveguide gratings (AWGs) and D optical FDLs connecting the AWGs, resulting in a total of B buffer positions, where $B=(D-1)\times M$, and M is the number of internal wavelengths. It is worth noting that, a packet entering the FOB at the i_{th} input port will exit the buffer from the i_{th} output port after receiving a certain delay time determined by the internal wavelength [32]. Thus, for any FOB, an internal wavelength of a packet uniquely determines the delay received by the packet. In the output section, there are $N \times W$ FWM-based TOWCs, and N output fibers each carrying W wavelengths. At each output port of the second AWG of an FOB, multiple packets that are carried by different internal wavelengths may have been scheduled to exit from the output port at the same time. This only occurs as a result of an attempt of preempting a lower-priority packet that has already been in the delay line by a higher-priority packet observing unavailable buffer space upon arrival [18]. The preemption is accomplished by tuning the FWM converter in such a way that,

upon converting the group of wavelengths, the target wavelength for the high-priority packet falls into the output wavelength. Then, all other packets outside of the output wavelength are automatically dropped after the MUX.

The CSC is composed of six modules. Headers of simultaneous arriving packets are first SASK-based demodulated [45] by the header extraction and processing module. Their labels and priority information is passed to the packet scheduler module, QoS parallel incremental scheduling (QPIS) algorithm. Serving as the brain of QBOPSS, QPIS performs QoS scheduling by determining for each packet the destined wavelength and the internal wavelength corresponding to the buffer delay that each packet is going to receive. The QPIS algorithm aims at minimizing the loss probability for high-priority packets under two constraints for newly arriving packets: (C1) they need to be free from switch- and buffer-contention; and (C2) they cannot contend with existing packets in FOBs, unless buffer preemption is permissible. Switch contention, also referred to as internal contention, occurs when more than one packet carried by the same wavelength attempts to access the same internal link in the PBSs. Buffer contention occurs when more than one packet competes for the same FOB space. And, buffer preemption permissible means that there exists a low-priority packet in the FOB which can be preempted by a newly-arriving high-priority packet. Notice that in SBOPSS, packets that are blocked in PBSs or fail to obtain an available FOB position will be inevitably dropped. The QPIS then passes the destined wavelength/output port, internal wavelength, and preemption information to the PBS control, inbound wavelength converter (WC) control, and outbound WC control modules, respectively. Finally, the header insertion module inserts and combines the new header with its payload before exiting from the optical switch.

3.1.2 Assessment of Optical Space Switches

We now draw comparisons between QBOPSS and several prevailing optical space switch structures with respect to component counts and output signal quality. It is noted that, in the comparisons, we only take the space switch section of QBOPSS into consideration. This is because the space switch section dominates the system performance while considering the component counts and output signal quality. Let N be the number of input fibers, M the number of wavelengths in each fiber, and C the wavelength clusters unique to our system, QBOPSS. Table 3.1 depicts the component counts of space switches under three space switch categories: non-blocking, rearrangeable, and blocking switches. Notice that, the broadcast-and-select space switch [46] can be constructed by simpler on-off SOA gates instead of the basic two-by-two switching elements. For comparing purpose, in this study we adopt the use of the two-by-two element as a basic element. We have observed in Table 3.1 that QBOPSS and Benes [47] outperform other structures with respect to the number of

Table 3.1 Component-count comparison of space switches with switch size $NW \times NW$

Category	Architecture	The number of 2×2 switching elements	Splitter number	Coupler number
Non-blocking Space Switches	CrossBar [21]	$(NW)^2$	0	0
	Broadcast-and-Select [46]	N^2W	NW	NW
	Cantor [22]	$NW/2\times(2\log_2(NW)-1)\times\log_2(NW)$	NW	NW
	Strictly Non-blocking Clos [23]	$2NW(2W-1)+(2W-1)N^2$	0	0
Rearrangeable Space Switches	Rearrangeable Clos [23]	$2NW^2+WN^2$	0	0
	Benes [47]	$NW/2\times(2\log_2(NW)-1)$	0	0
Blocking Space Switches	SBOPSS	$NW/2\times\log_2(NW/C)$	0	0

switching elements. However, for the Benes structure, the price paid is the requirement of complicated scheduling (rearranging) algorithms, causing slower switch processing. Significantly, QBOPSS equips C pseudo-Banyan space switches with size $N(W/C)\times N(W/C)$ by clustering wavelengths. Such wavelength-clustering design effectively reduces switch complexity to $O(NW\times\log_2 N)$ by selecting C as $W/4$ or $W/8$.

Optical signal suffers from signal impairment and power loss that are caused by passing a number of switching elements and splitters/couplers, respectively. Thereby, we perform three separate studies for broadcast-and-select, Cantor, and all remaining structures, respectively, due to their uses of different components. Results are summarized in Table 3.2. In general, with splitters and couplers, a 1-by-m splitter or an m-by-1 coupler causes severe power loss, yielding an output power being $1/m$ of the original signal power. Thus, for broadcast-and-select, the output signal power becomes $1/N^2$ after passing through one splitter (1-by-N), one basic two-by-two

Table 3.2 Signal-quality comparison of space switches with switch size $NW\times NW$

Category	Architecture	The number of 2×2 switching elements (Signal impairment)	Splitter and Coupler (Output/Input Power)
Non-blocking Space Switches	CrossBar [21]	$2NW$	N/A
	Broadcast-and-Select [46]	1	$1/N^2$
	Cantor [22]	$2\log_2(NW)-1$	$(1/\log_2(NW))^2$
	Strictly Non-blocking Clos [23]	$4W(2W-1)+2N^2$	N/A
Rearrangeable Space Switches	Rearrangeable Clos [23]	$4W^2+2N^2$	N/A
	Benes [47]	$2\log_2(NW)-1$	N/A
Blocking Space Switches	SBOPSS	$\log_2(NW/C)$	N/A

element (or on-off gate), and one coupler (N-by-1). For the Cantor switch, the optical signal passes through one 1-by-$\log_2(NW)$ splitter at the switch front and one $\log_2(NW)$-by-1 coupler at the switch end, resulting in an output power being $(1/\log_2(NW))^2$ of the input signal power. The Cantor switch also causes signal impairment from basic switching elements, where the number of basic elements is the same as that of the Benes switch. Finally, the signal quality for the remaining structures is solely anti-proportional to the number of basic switching elements [48,49]. Thus, we study the signal impairment by counting the number of two-by-two elements in the longest (worst) switching path. As a result, QBOPSS achieves the best signal performance among all switch structures.

3.2 QoS Packet Scheduling: The QPIS algorithm

3.2.1 Definitions and Notations

Before delving into the details of the QPIS algorithm, we first give notations and definitions that are used throughout the Chapter 3. Owing to that packet scheduling for different clusters is completely independent, we thereinafter discuss the scheduling problem for the system with one cluster. Let N denote the number of input/output fibers; W the number of input/output external wavelengths (λ_1^I to λ_W^I in the input fiber; and λ_1^O to λ_W^O in the output fiber); M the number of internal wavelengths (λ_1 to λ_M); FOB_i the optical buffer for wavelength i, where i=1 to W; and D the number of delay lines (including no delay) within each FOB. For the ease of illustration, we only consider two priorities of packets in this sub-chapter. The algorithm can easily be enhanced to support multiple priorities. Each newly-arriving-packet header is associated with the quartet information, (I_i, λ_j^I, O_k, Prt), where I_i is the input fiber, λ_j^I is the input external wavelength, O_k is the output fiber, and Prt is the priority of the packet. At the beginning of each time slot, the headers of all newly-arriving packets form a header set, $P=\{p_n | p_n=(I_i, \lambda_j^I, O_k, Prt)_n; 1\le n\le |P|; 1\le i\le N; 1\le j\le W; 1\le k\le N\}$.

To perform packet scheduling, the QPIS algorithm requires the constant update of the FOB states. The state of FOB_i is represented by an $N\times D$ matrix, denoted as $\text{FSTAT}_i[O_a, \text{FDL}_b]$, i=1 to W, where each row O_a (a=1 to N) corresponds to the an input/output port of the FOB, and each column FDL_b (b=0 to $D-1$) corresponds to a delay line in the FOB. Each entry of the matrix can be one of the three values: 0 if there is no packet; and "H" ("L") if a high-priority (low-priority) packet occupies the corresponding buffer position. Notice that the fact packets move forward in the delay

line after each time slot has elapsed is associated with the left shift of the entries of the matrix.

[Definition 3.1] A *valid path* for a packet with header (I_i, λ_j^I, O_k, *Prt*), denoted as (I_i, λ_j^I, O_k, λ_x^O, λ_y), is a route within the system that starts from an input port (I_i, λ_j^I) of the PBS, through the output port (O_k, λ_x^O) of the PBS, an inlet of an FOB for λ_x^O, a λ_y-corresponded delay line, and finally to the FOB outlet, and that is free from buffer contention (i.e., $FSTAT_x(O_k, FDL_{(y-k+M) \bmod M})=0$) with any packets currently in the buffer, or is buffer-preemption permissible (i.e., $FSTAT_x(O_k, FDL_{(y-k+M) \bmod M})<Prt$). ■

[Definition 3.2] A *sound-path set* for a group of newly-arriving packets is a set of valid paths $Q'=\{q_m|\ q_m=(I_i, \lambda_j^I, O_k, \lambda_x^O, \lambda_y)_m;\ 1\le m\le|P|;\ 1\le i\le N;\ 1\le j\le W;\ 1\le k\le N;\ 1\le x\le W;\ 1\le y\le M\}$, that satisfy the following two constraints: (C1) all paths in the set are mutually switch- and buffer-contention free; and (C2) all packets that the sound-path set corresponds are buffer-contention free from the packets in the buffer or buffer-preemption permissible. ■

Notice that a packet may have many valid paths associated with paths lending different delays. Finally, the packet-scheduling problem is formally defined as follows.

[Definition 3.3]: Packet-Scheduling Problem

Consider a number of simultaneously arriving packets, with the header set $P=\{p_n|\ p_n=(I_i, \lambda_j^I, O_k, Prt)_n;\ 1\le n\le|P|;\ 1\le i\le N;\ 1\le j\le W;\ 1\le k\le N\}$.

(i) With the time-bound constraint lifted, the packet-scheduling problem is to find the largest sound-path set that also contains a maximal number of valid paths for high-priority packets. The set is referred to as the *target sound-path set* $Q=\{q_m|\ q_m=(I_i,$

λ_j^I, O_k, λ_x^O, $\lambda_y)_m$; $1 \le m \le |P|$; $1 \le i \le N$; $1 \le j \le W$; $1 \le k \le N$; $1 \le x \le W$; $1 \le y \le M\}$;

(ii) Given a time-bound constraint, T, the packet-scheduling problem is to obtain within time T the largest sound-path set that also contains a maximal number of valid paths for high-priority packets. The set is referred to as the *transient sound-path set* $Q(T)=\{q_m |\ q_m=(I_i, \lambda_j^I, O_k, \lambda_x^O, \lambda_y)_m;\ 1 \le m \le |P|;\ 1 \le i \le N;\ 1 \le j \le W;\ 1 \le k \le N;\ 1 \le x \le W;\ 1 \le y \le M\}$. ■

A packet will be discarded if its valid path is not included in the target or transient sound-path set. A discarded packet is converted to wavelength λ_0 that in turn will be discarded through a filter before entering the PBS.

3.2.2 The QPIS Algorithm

The packet-scheduling problem can be proved to be *NP*-complete. In this sequel, we present our QPIS heuristic algorithm that finds a near-optimal solution, or incrementally returns a feasible solution within a given time constraint. As shown in Figure 3.2, the QPIS algorithm operates in three phases- the graph transformation, directed-graph construction, and iterative self-marking phases, on a slot basis. In the first phase, the algorithm transforms the packet-scheduling problem into a graph problem according to the following rule. Consider all valid paths for all newly-arriving packets, for each valid path (I_i, λ_j^I, O_k, λ_x^O, λ_y), a vertex, $v_{(I_i, \lambda_j^I, O_k, \lambda_x^O, \lambda_y)}$, is created and drawn into the undirected graph G_u. Afterward, an edge, CT_edge(v_σ, v_τ), is drawn between two vertices v_σ and v_τ, if their corresponding valid paths are in (switch or buffer) contention with each other. Notice that, although each packet may have more than one valid path, there is at most one valid path for each packet to be included in any sound-path set. Hence, a special edge, OR_edge(v_σ, v_τ), is

drawn between two vertices if their corresponding valid paths belong to the same packet, i.e., share the same I_i and λ_j^I, but using different delays and/or external wavelengths. Thus, the packet-scheduling problem thus becomes to find a maximal set of disconnected vertices that also contains a maximal number of high-priority-packet paths (vertices) in G_u without edges connecting any two of them.

In the second phase, the algorithm converts the undirected graph G_u into a directed graph G_d. The edge directions are assigned based on two priority-based QoS rules and two general heuristic rules. The two QoS rules attempt to schedule a maximal number of high-priority vertices, while two heuristic rules are in attempt to maximize the sound-path set. The two priority-based QoS rules are: (QoS-rule one) assigning the edge directions from high-priority to low-priority vertices; and (QoS-rule two) assigning the edge directions from buffer-contention-free to buffer-preemption-permissible vertices. Notice that the QoS-rule two is to avoid preemption should there be empty space in the buffer, so that system throughput can be maximized. The two general heuristics pertain to contention and delay. For contention, we define the total number of CT_edges (but not OR_edges) connecting to the vertex as the *Degree* of a vertex. Thus, the higher the degree of a vertex, the more paths the vertex (path) is in contention with. For delay, the longer the FDL delay of a vertex, the more system resources (buffers) are occupied, resulting in greater possibility that the future arriving packets are blocked. Namely, for the edge assigning process, lower-degree vertices are preferred because they contend with fewer vertices (paths), and shorter-delay vertices (paths) are preferred because they leave the system and release resources more quickly. Accordingly, the two general heuristics are: (Heuristic-rule one) assigning the edge directions from the lower-degree to higher-degree vertices; and (Heuristic-rule two) assigning the edge directions from the

shorter-delay to longer-delay vertices. Importantly, QoS rules take precedence over Heuristic rules, and rule one takes precedence over rule two.

In the last iterative self-marking phase, each vertex in graph G_d iteratively updates the status (*Tag*) by selecting or deselecting itself according to the status of its neighboring vertices on a round basis. All vertices are initially marked *Tag*=ON as being selected. In each round for any vertex, say v, if there exists one neighboring vertex that has an edge directing to vertex v and is selected (*Tag*=ON), vertex v must deselect itself (Tag_v=OFF) to prevent from potential contention. Otherwise, vertex v will select itself (Tag_v=ON). Essentially, as asserted by Theorem 3.1 (next sub-chapter) for proving the incremental property, if the status of a vertex remains unchanged for two consecutive rounds (the vertex is said to be "stable"), the status of the vertex will no longer be changed. The iteration stops either when all vertices' status remains unchanged within the entire round by the end of a slot time; or the requested time constraint (T) expires. In the former case, the target sound-path set Q is given as the set of valid paths for the selected vertices. In the latter case, the transient sound-path set $Q(T)$ is given as the set of valid paths for the vertices that are selected and "stable".

Algorithm QPIS**;**
While (given a newly-arriving packet set P at the beginning of a time slot) **do**
1. Shift each entry in FSTAT's one position left;
   /* Update FDL positions for packets currently in the buffer */

Phase I: Graph transformation phase /* Problem transformation */

2. **for** (each p_n) **do**
   Search all valid paths; For each valid path, add a vertex into G_u; **endfor**
3. **for** (each vertex pair v_σ and $v_\tau \in G_u$) **do**
   if (v_σ and v_τ belong to the same packet) Add an OR_edge(v_σ, v_τ) into G_u;
   elseif (v_σ and v_τ yield an switch/buffer contention)
   Add a CT_edge(v_σ, v_τ) into G_u; **endif endfor**

Phase II: Directed-graph construction phase **/*** Edge direction assign rules */

4. **for** (each $v_\sigma \in G_u$) **do** Calculate $Degree_\sigma$; **endfor**
5. **for** (each CT_edge(v_σ, v_τ) or OR_edge(v_σ, v_τ) $\in G_u$) **do**
   if ($Prt_\sigma > Prt_\tau$) Set edge direction from v_σ to v_τ; /* QoS-rule one */
   elseif (v_σ is buffer-contention free and v_τ is buffer-preemption permissible)
   Set direction from v_σ to v_τ; /* QoS-rule two */
   elseif ($Degree_\sigma < Degree_\tau$)
   Set direction from v_σ to v_τ; /* Heuristic-rule one */
   elseif (delay of v_σ < delay of v_τ)
   Set direction from v_σ to v_τ; /* Heuristic-rule two */
   elseif (vertex id of v_σ < vertex id of v_τ)
   Set direction from v_σ to v_τ; **endif endfor**

Phase III: Iterative self-marking phase /* Parallelism: all vertices $v_\sigma \in G_d$ run in parallel */

6. *rounds*←1;
7. **for** (each $v_\sigma \in G_d$) **do** Initialize Tag_σ←ON; **endfor**
8. **for** (each $v_\sigma \in G_d$) **do**
   if ($\exists \overrightarrow{\text{CT\_edge}}(v_\tau, v_\sigma)$ or $\exists \overrightarrow{\text{OR\_edge}}(v_\tau, v_\sigma) \in G_d$, and Tag_τ=ON) Tag_σ←OFF;
   else Tag_σ←ON; **endif endfor**
9. **for** (each $v_\sigma \in G_d$) **do**
   if (Tag_σ is unchanged for two consecutive rounds) $Flag_\sigma$←"stable";
   else $Flag_\sigma$←"unstable"; **endif endfor**
10. **if** (all *Tag*'s remain unchanged in this round) /* near-optimal solution Q */
    for (each $v_\sigma \in G_d$, where Tag_σ=ON) **do**
    Add its corresponding valid path into Q; **endfor**
    The target sound-path set Q is found;
    Schedule packets according to Q;
    elseif (*rounds*=T) /* Incremental property: feasible solution $Q(T)$ */
    for (each $v_\sigma \in G_d$, where Tag_σ=ON and $Flag_\sigma$="stable") **do**
    Add its valid path into $Q(T)$; **endfor**
    The transient sound-path set $Q(T)$ is found;
    Schedule packets according to $Q(T)$;
    else *rounds*← *rounds* +1; **goto** 8; **endif**

Figure 3.2 The QPIS Algorithm.

3.2.3 The Incremental Property of QPIS

In Theorem 3.1, we first assert and prove that a vertex's status will no longer change once it is "stable". Accordingly, the incremental property is then given and proved in Theorem 3.2.

Lemma 3.1: The directed graph, G_d, does not contain cycles. ■

Proof (of Lemma 3.1): In the directed-graph construction phase, after the four edge assigning rules are applied, all vertices of G_d are sorted in an absolute order. (Notice that if the vertices have the same priority, degree, and delay, they are sorted by the designated id, as described in the algorithm in Figure 3.1). Accordingly, all edges are directed in the same direction, allowing the Lemma to hold. ■

Theorem 3.1: If the status (*Tag*) of a vertex remains unchanged for two consecutive rounds in the iterative self-marking phase, i.e., the vertex is "stable", the vertex will not change its status in the following iteration rounds. ■

Proof (of Theorem 3.1): The proof is performed via mathematical induction on the number of vertices in the directed graph G_d. Assume that V is the vertex set of G_d. The basic condition states that the theorem holds for $|V|=1$. If the theorem also holds for $|V|=k$, we are to prove that the theorem holds for $|V|=k+1$. Without loss of generality, v_{k+1} is chosen as the vertex that only has inward edges. By Lemma 3.1, the vertex must exist. Also due to the inductive assumption, vertices v_1~v_k must obey this theorem because the vertex v_{k+1} will not influence them obviously. Therefore, the proof can be completed by proving that Tag_{k+1} does not produce traces OFF→OFF→ON and ON→ON→OFF during the iterative self-marking phase.

Part 1: First, we show that Tag_{k+1} will never produce a trace of OFF→OFF→ON. By contradiction, assume that Tag_{k+1} does indeed produce the trace OFF→OFF→ON from iteration round r to $r+2$. In this case, there are only two possibilities that can realize such a trace, which are illustrated as Case 1 and Case 2 in Figure 3.3. That is,

for making Tag_{k+1}=OFF in round r, at the end of round $r-1$, there must have at least one vertex whose *Tag* is ON. The main difference between Case 1 and Case 2 is that, there exists a non-empty set of vertices that direct to v_{k+1} with *Tag*=OFF in Case 2, whereas all vertices are of status ON in Case 1.

Case 1:

1. Round $r-1$: Assume that Tag_{k+1}=ON/OFF arbitrarily, and *Tag*=ON for every vertex with an edge directing to v_{k+1}. This will imply Tag_{k+1}=OFF in round r.

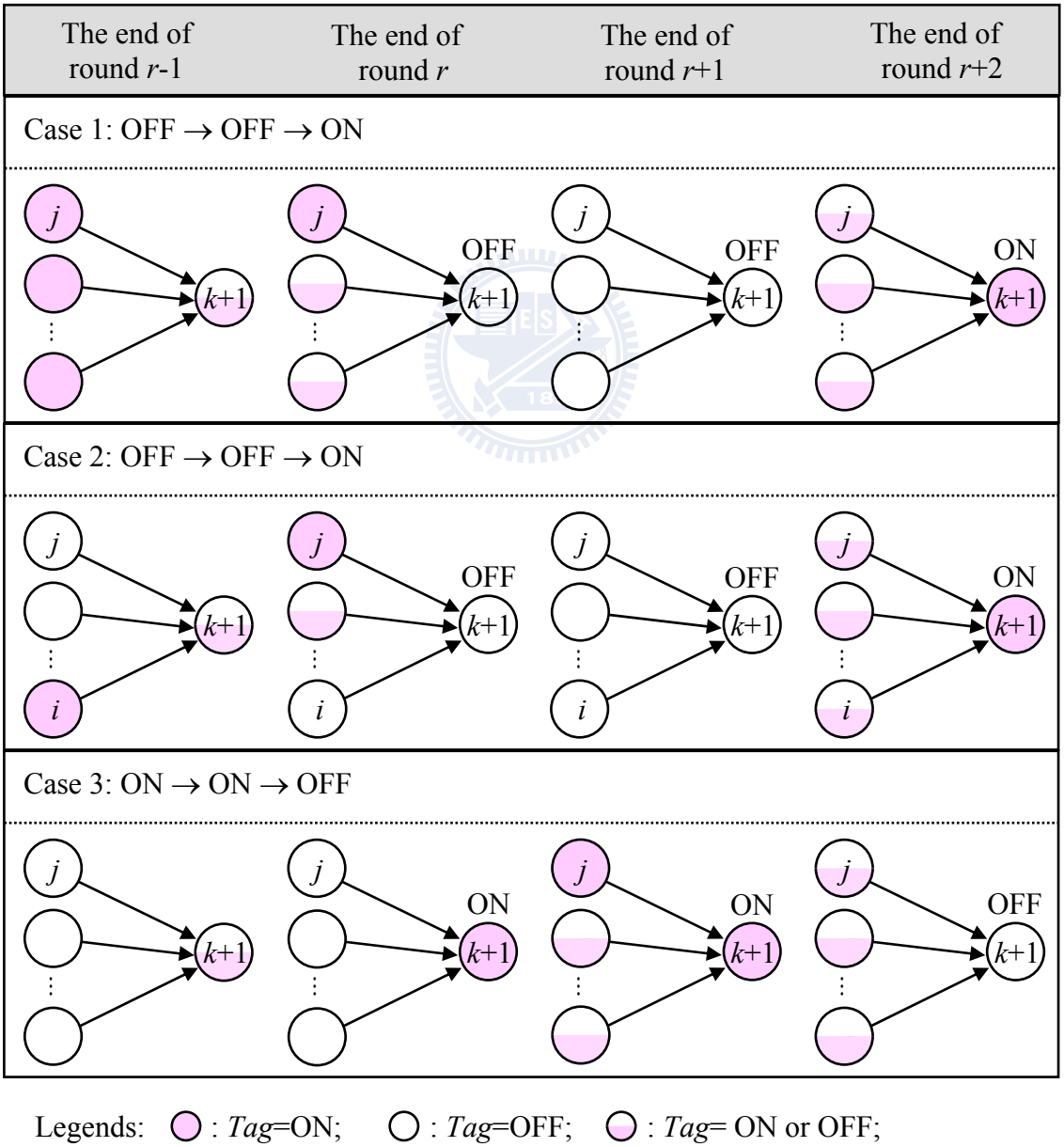


Figure 3.3 Illustration for the proof of Theorem 1.

2. Round r: To keep Tag_{k+1}=OFF in the next round, r+1, there must have a vertex, say v_j in Figure 3.3, directing to v_{k+1} and Tag_j is set to ON within this round.
3. Round r+1: Set Tag_{k+1}=OFF because Tag_j=ON at the end of round r. To have Tag_{k+1}=ON in the next round, r+2, all vertices directing to v_{k+1} must set *Tag* to OFF within round r+1.
4. Round r+2: Set Tag_{k+1}=ON because all *Tags* directing to v_{k+1} are OFF at the end of round r+1.

Now, a contradiction occurs because vertex v_j experiences a Tag_j trace of ON→ON→OFF from iteration round r−1 to r+1, violating the inductive assumption we made in the proof.

Case 2:

1. Round r−1: Assume that Tag_{k+1}=ON/OFF arbitrarily. There must be a non-empty set of vertices with edges directing to v_{k+1} and *Tag*=ON, which trigger Tag_{k+1}=OFF in round r. For simplicity, in the sequel (and in Figure 3.3), our illustration includes only one vertex (v_i) in this set.
2. Round r: In this round, vertex v_i must change its *Tag* to OFF. Otherwise, it becomes stable by the inductive hypothesis, and the stability makes itself remain Tag_i=ON in the following rounds, resulting that Tag_{k+1} can never be set to ON in round r+2. However, since it must hold that Tag_{k+1}=OFF in round r+1, there must be a vertex (say vertex v_j in Figure 3.3) that was OFF at the end of round r−1 but is updated to ON in this round. (Note that a situation of having no such a vertex is what Case 1 discusses).
3. Round r+1: Set Tag_{k+1}=OFF since Tag_j=ON at the end of round r. In this round, the *Tag* of each vertex directing to v_{k+1} must be changed to OFF to allow Tag_{k+1}=ON in round r+2.
4. Round r+2: Set Tag_{k+1}=ON because all *Tags* directing to v_{k+1} are OFF at the end

of round r+1.

Now, by the inductive hypothesis made in the proof, all vertices that are unstable (except v_{k+1}) must have the *Tag* switched continuously between OFF and ON. Recall that all vertices belonging to G_d are initialized with *Tag*=ON. Thereby, the unstable vertices must have their *Tag*s changed in a synchronous manner. A contradiction occurs since v_i and v_j are unstable but with different *Tag* values at the end of round $r-1$. Combining Case 1 and Case 2, we have proved that Tag_{k+1} does not produce a trace of OFF→OFF→ON.

Part 2: We now show that Tag_{k+1} will never produce a trace of ON→ON→OFF. Again by contradiction, assume that Tag_{k+1} produces such a trace from iteration round r to r+2. Under this scenario, there is only one possibility that is illustrated as Case 3 in Figure 3.3.

Case 3:

1. Round $r-1$: Assume that Tag_{k+1}=ON/OFF arbitrarily. To trigger Tag_{k+1}=ON in round r, all vertices directing to v_{k+1} must have *Tag*=OFF in this round.
2. Round r: Set Tag_{k+1}=ON. In order to keep Tag_{k+1}=ON in the next round, the *Tag* of each vertex directing to v_{k+1} must retain OFF in this round.
3. Round r+1: Set Tag_{k+1}=ON. In this round, there must have a vertex, say v_j, which is directing to v_{k+1}, converting its Tag_j to ON. This action then triggers Tag_{k+1}=OFF in round r+2.
4. Round r+2: Set Tag_{k+1}=OFF because Tag_j=ON at the end of round r+1.

By the inductive hypothesis, this case arrives at a contradiction because Tag_j produces a trace of OFF→OFF→ON from iteration round $r-1$ to r+1.

With all three cases reasoned, by mathematical induction, the theorem holds for all $|V|$. ■

Theorem 3.2: Transient sound-path sets $Q(T_1) \subseteq Q(T_2)$ if $T_1 \leq T_2$, i.e., $Q(T)$ is monotonically non-decreasing over time T. This assertion is referred to as the incremental property of the QPIS algorithm. ■

Proof (of Theorem 3.2): Assume a vertex $v \in Q(T_1)$ is selected (Tag_v=ON) and stable at time constraint T_1. According to Theorem 3.1, vertex v is also selected and stable under time constraint T_2 if $T_2 \geq T_1$. In other words, $v \in Q(T_2)$ and the theorem holds. ■

3.3 QPIS Implementation: Hardware Parallel System Architecture

In this sub-chapter, we present the hardware parallel system architecture for the efficient implementation of the QPIS algorithm. We then derive the upper-bound computational complexity of the algorithm.

3.3.1 Hardware System Architecture

Given a QBOPSS, we can pre-construct the hardware for all legitimate paths, i.e., vertices $v_{(I_i,\ \lambda_j^I,\ O_k,\ \lambda_x^O,\ \lambda_y)}$, where $1 \le i \le N$; $1 \le j \le W$; $1 \le k \le N$; $1 \le x \le W$; $1 \le y \le M$, and all CT_edges and OR_edges connecting these vertices. As depicted in Figure 8.1, each vertex is implemented by a hardware subsystem consisting of three modules- graph transformation, directed-graph construction, and iterative self-marking, which correspond to the three phases of the QPIS algorithm, respectively. The internal interfaces between modules and external interfaces between subsystems are made through control signals (binary), control bus (non-binary), and data signal (binary), as shown in Figure 3.4.

Initially, all subsystems are inactive. Upon the arrival of a set of packets, P, the graph transformation module of a vertex determines if the vertex belongs to G_u (a valid path) for packet set P by matching its (I_i, λ_j^I, O_k), and checking if the entry $\mathrm{FSTAT}_x(O_k,\ \mathrm{FDL}_{(y-k+M)\bmod M})$ is empty (i.e., $\mathrm{FSTAT}_x(O_k,\ \mathrm{FDL}_{(y-k+M)\bmod M})=0$) or preemption permissible (i.e., $\mathrm{FSTAT}_x(O_k, \mathrm{FDL}_{(y-k+M)\bmod M})<Prt$), with packets in P. If the matching/checking succeeds, the vertex is included in G_u, and its corresponding subsystem becomes activated with *active* signals sent to two remaining modules. Otherwise, the subsystem remains inactive. After becoming active, the packet priority and the corresponding entry value (buffer state) in FSTATs are then sent to the

directed-graph construction module (as shown in Figure 3.4).

Upon having received an active signal, the phase-two directed-graph construction module broadcasts the active signal to its neighboring subsystems. The *Degree* value can be computed as the total number of received active signals from the neighboring subsystems that are connected via CT_edges. It can be derived that, for a QBOPSS with N input/output ports, M internal wavelengths, and W external wavelengths, there are at most $NMW \times \log_2(NW)$ edges connecting to a directed-graph construction module. As a result, the module contains $(1/2) \times NMW \times \log_2(NW)$ two-to-one adders, and the *Degree* can be computed in parallel in $O(\log_2(NMW \times \log_2(NW)))$. Then, the

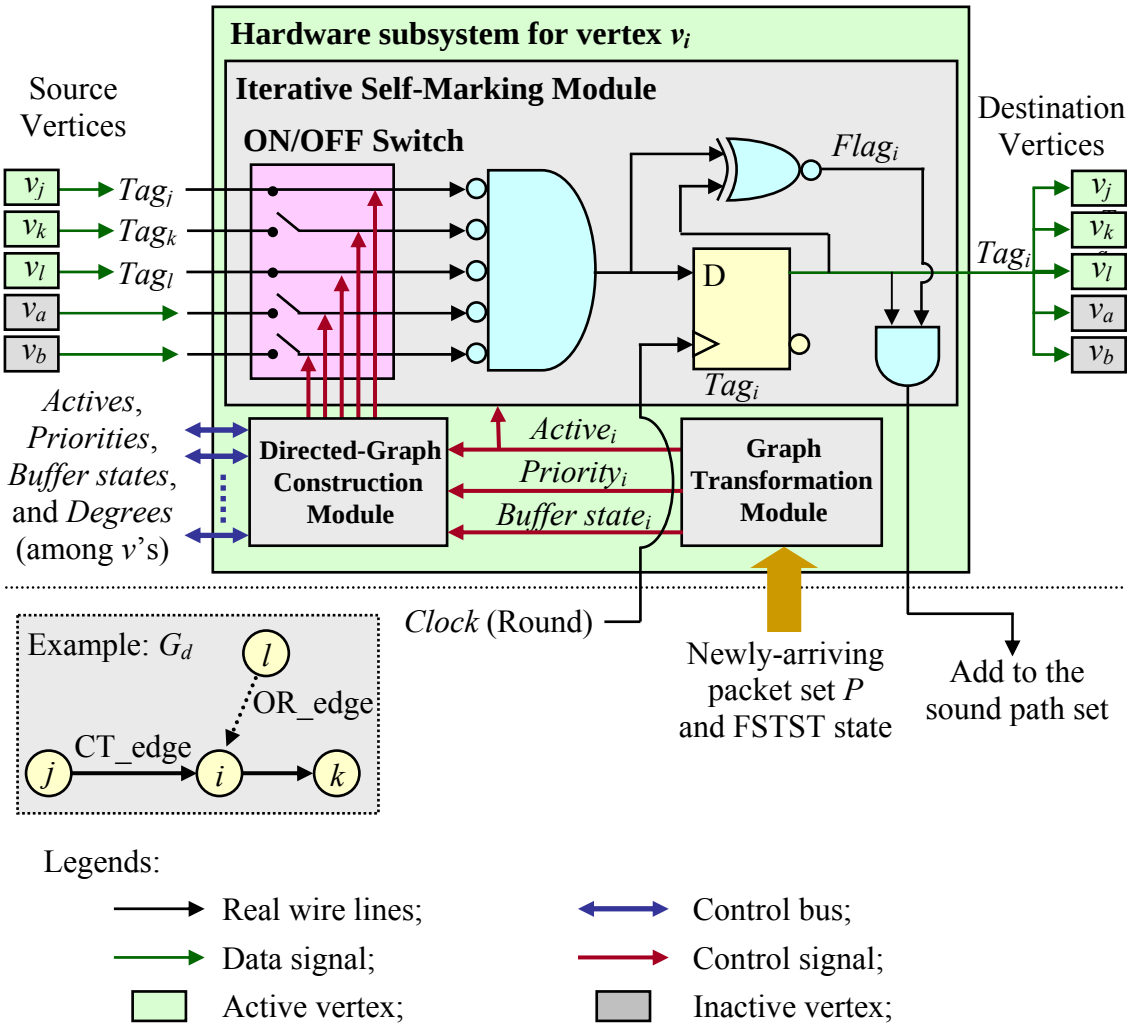


Figure 3.4 The hardware parallel system architecture.

phase-two module informs the neighboring active subsystems of its priority, emptiness/preemption-permissible of the buffer space (buffer state), and the *Degree* via control buses. It then determines the edge directions for G_u by comparing priorities, buffer states and/or *Degree* values with its neighboring subsystems in parallel. Once these steps are performed, the directed graph G_d is formed. The phase-two module finally triggers the ON/OFF switch in the iterative self-marking module as shown in Figure 3.4. The ON/OFF switch comprises a number of uni-directional wires, each of which stands for a directed edge pointing to this vertex. The "ON" state indicates that the directed edge belongs to G_d, while the "OFF" state means the contrary.

Finally, the phase-three module performs the iterative update of *Tag* on a round basis. The *Tag* is initialized to be ON, and is stored in a D flip-flop shown in Figure 3.4. It is noted that both *Tag*=ON and *Flag*="stable" correspond to a hardware value of 1; and both *Tag*=OFF and *Flag*="unstable" a value of 0. To update the *Tag* in the subsequent round, this module passes neighboring *Tag* values from uni-directional wires through an AND gate after the inversion. The new *Tag* value is updated and in turn recorded in the D flip-flop. The *Flag* of a vertex can be determined by logically XNOR-ing two consecutive *Tags* from the inlet and outlet of the D flip-flop. By logically AND-ing the *Flag* and *Tag*, one can determine whether the vertex belongs to the sound-path set or not at the end of each round.

3.3.2 Computational Complexity of QPIS

In this subchapter, we derive the computational complexity of the QPIS algorithm. Prior to it, we first assert two crucial properties of G_d, followed by proving in Theorem 3.3 that the maximum number of iteration rounds to finish QPIS computing

for any packet set P is O(NW).

Lemma 3.2: The following two properties hold for vertices in G_d:

(a) A vertex becomes stable only at the end of *odd* (*even*) rounds by having *Tag*=ON (OFF) in two consecutive rounds.

(b) Each iteration round generates at least one new stable vertex. ■

Proof (of Lemma 3.2): (a) Recall that *Tag* is initialized to be ON prior to the first iteration round. Unstable vertices switch their *Tags* from ON to OFF in odd rounds, and OFF to ON in even rounds. Therefore, the second consecutive ON always appears in odd rounds, and OFF in even rounds. (b) Notice that, as Lemma 3.1 indicates there exists no cycle in G_d. The statement certainly holds by only considering unstable vertices in G_d. Thus, before executing the update of the r_{th} round, one can always find one unstable vertex v_i that no other unstable vertices direct to it with an edge due to the cycle-free assertion. Therefore, all vertices directing to v_i are stable before executing the r_{th} round update. These stable vertices keep the same *Tag* values in $r-2_{nd}$ and $r-1_{st}$ rounds. As a result, Tag_i must repeat the same *Tag* value in the r_{th} round as that in the $r-1_{st}$ round. Such update makes vertex v_i a new stable vertex by the end of the r_{th} round, proving that the lemma holds. ■

Theorem 3.3: Given a new packet set P, without time constraint, the iterative self-marking phase of the QPIS algorithm completes the computing in O(NW) iteration rounds. ■

Proof (of Theorem 3.3): Given a packet set P, there are at most $N \times W$ newly-arriving packets. Therefore, due to the incremental property of QPIS (Theorem 3.1 and 3.2), the maximum number of selected paths in the target sound-path set, i.e., the maximum number of stable vertices with *Tag*=ON, is $N \times W$ after completing the QPIS algorithm. By Lemma 3.2, we know that all vertices with *Tag*=ON will be stable after $2NW-1$ rounds. Also, at the end of round $2NW$, all vertices in G_d must be stable and the QPIS

algorithm terminates. If not, a contradiction occurs by having a new stable vertex with *Tag*=ON at the end of the next round 2*NW*+1. Thereby, the theorem holds. ■

Now, the first step of the QPIS algorithm in Figure 3.2, that involves simultaneous left shifting of all entries, requires an O(1) computation. In the graph transformation phase, as shown in the hardware system in Figure 3.4, each vertex tests if it belongs to G_u by matching the newly-arriving packet set P and checking its related entry of FSTATs, resulting in an O(NW) computation, where $|P|$=O(NW). In the directed-graph construction phase, a vertex broadcasts informing signals to its edge-connected neighbors, and also receives signals from them if it is a valid path. Therefore, the *Degree* calculation for each vertex can be carried out in $O(\log_2(NMW \times \log_2(NW)))$, and the edge direction can be assigned in O(1) by triggering the ON/OFF switch. The iterative self-marking module performs one round update (step 8 of Figure 3.2) by logically AND-ing the neighboring *Tag*s after the inversion. Similar to calculating the *Degree*, this AND-ing action can be performed in $O(\log_2(NMW \times \log_2(NW)))$. By Theorem 3.3, this iterative phase can be finished in $O(NW \times \log_2(NMW \times \log_2(NW)))$. Finally, the near-optimal solution Q is updated into FSTATs in O(NW), where $|Q|$=O(NW). Accordingly, the computational complexity ($\mathcal{T}$) can be derived as follows:

$$\begin{aligned}\mathcal{T}(N, M, W) &= O(1) + O(NW) + O(\log_2(NMW \times \log_2(NW))) \\ &\quad + O(NW \times \log_2(NMW \times \log_2(NW))) + O(NW) \\ &= O(NW \times (\log_2(NMW) + \log_2\log_2(NW))) \\ &= O(NW \times \log_2(NMW)). \end{aligned} \tag{3.1}$$

3.4 Simulation Results

We now demonstrate and compare the performance of QPIS and three packet scheduling algorithms with respect to computational complexity, packet loss probability, and QoS differentiation, via simulation results. The three packet scheduling algorithms are: exhaustive Optimal method, SMinB, and SMinD. The Optimal method returns an optimal solution by testing all of the path combinations for the newly-arriving packet set P. With M internal and W external wavelengths, there are a total of MW+1 path choices for each packet, where the additional one corresponds to the discard of the packet. While QPIS considers constraints (C1) and (C2) simultaneously, SMinB and SMinD consider the two constraints separately. SMinB aims at minimal blocking within the PBS by searching all valid paths that satisfy constraint (C1) first. All candidate paths are then tested and inserted only if constraint (C2) is satisfied. On the other hand, SMinD aims at minimal delay by testing constraint (C2) before constraint (C1).

Since packet scheduling for different clusters of QBOPSS is independent, therefore without loss of generality, we assume there is only one cluster in QBOPSS. In the simulations, we assume that there are a total of $N\times W$ i.i.d. (independent and identically distributed) input traffic flows entering into the QBOPSS simultaneously. We also experiment with two different traffic arrival distributions- Bernoulli process (BP) and interrupted Bernoulli process (IBP), to model smooth and bursty traffic, respectively. Specifically for the IBP, we adopt a ratio of mean idle to busy period being equal to 1/20 corresponding to a highly bursty traffic arrival. The traffic load is defined as the mean number of newly-arriving packets $|P|$ divided by the total channel capacity $N\times W$, i.e., $E[|P|]/(N\times W)$. The destination of each packet is uniformly distributed among all output ports.

We first summarize in Table 3.3 the computational complexity of the QPIS and three other algorithms. Due to considering all path combinations, the exhaustive Optimal method results in exceptionally high complexity, $O((MW+1)^{NW})$, where $|P|=O(NW)$. For SMinB, the switching process requires $O((NW)^2MW)$ and $O(NW)$ to perform scheduling satisfying constraints (C1) and (C2), respectively, yielding a complexity of $O((NW)^2MW)$. For all packets in P, SMinD requires $O((NW)W)$ for examining W output external wavelengths in order to assign minimal-delay available entries of FOBs (constraint (C2)). To satisfy constraint (C1), SMinD sequentially tests if each packet contends internally with pre-scheduled packets, yielding a complexity of $O((NW)^2)$. Thus, SMinD requires a complexity of $O((NW)^2)$. We can conclude that QPIS requires much lower complexity than all remaining three algorithms.

In Table 3.4, we draw a comparison of packet loss probability between QPIS and the exhaustive Optimal method under four traffic settings in both BP and IBP traffic models. These four traffic settings are fix high-priority (H) load at light load 0.2, fix H load at heavy load 0.5, fix low-priority (L) load at light load 0.2, and fix L load at heavy load 0.5. Due to unmanageable complexity of the Optimal method, we can only attain packet loss probability for the QBOPSS that is of small size, i.e., PBS size 8×8 (N=2, W=4). As shown in Table 3.4, QPIS achieves as profoundly low loss probability as that of the exhaustive Optimal method. In the cases under heavier high-priority load or lighter low-priority load, the exhaustive Optimal method yields higher low-priority loss than QPIS for further minimizing high-priority loss. One can perceive from the results that QPIS returns a near-optimal solution with exceptional low complexity.

As shown in Table 3.4, we have observed much similar system performance (the packet loss probabilities) between more bursty traffic setting (IBP distribution) and smooth traffic setting (BP distribution). Therefore, in the following simulation, we

take BP distribution as incoming traffic flow model for further observing and comparing the performance among QPIS, SMinB, and SminD.

In Figure 3.5, we display the packet loss probability of QPIS using three practicable PBS sizes, 8×8 (N=2, W=4), 16×16 (N=4, W=4), and 32×32 (N=8, W=4), under two different traffic settings (fix H=0.2 and fix L=0.5). In the simulation, we fix D=4, yielding buffer sizes B=12, B=12, and B=24 for the three PBS cases, respectively. Notice that, as N becomes larger, owing to the switch clustering design,

Table 3.3 Comparison of computational complexity

Method	QPIS	Optimal	SMinB	SMinD
Complexity	$O(NW\times\log_2(NMW))$	$O((MW+1)^{NW})$	$O((NW)^2MW)$	$O((NW)^2)$

Table 3.4 Loss probability comparisons between QPIS and Optimal**.**

/* At each table entry: BP value (IBP value) */

PBS size 8×8 and D=4		Total load=0.75		Total load=0.85		Total load=0.95	
		Optimal	QPIS	Optimal	QPIS	Optimal	QPIS
Fix H load =0.2	H	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	6.25×10^{-8} (7.52×10^{-8})
	L	1.05×10^{-4} (2.58×10^{-4})	1.17×10^{-4} (2.77×10^{-4})	2.58×10^{-3} (5.50×10^{-3})	2.82×10^{-3} (5.61×10^{-3})	2.40×10^{-2} (3.39×10^{-2})	2.38×10^{-2} (3.28×10^{-2})
Fix H load =0.5	H	0 (0)	2.00×10^{-6} (4.99×10^{-6})	2.50×10^{-7} (2.25×10^{-6})	1.52×10^{-5} (3.51×10^{-5})	3.25×10^{-6} (4.01×10^{-6})	7.75×10^{-5} (1.35×10^{-4})
	L	3.81×10^{-4} (8.41×10^{-4})	3.29×10^{-4} (6.65×10^{-4})	5.96×10^{-3} (1.22×10^{-2})	4.74×10^{-3} (9.72×10^{-3})	4.64×10^{-2} (6.49×10^{-2})	3.76×10^{-2} (5.35×10^{-2})
Fix L load =0.2	H	6.82×10^{-7} (9.09×10^{-7})	6.14×10^{-6} (1.38×10^{-5})	2.50×10^{-5} (7.23×10^{-5})	2.52×10^{-4} (5.45×10^{-4})	7.64×10^{-4} (1.30×10^{-3})	3.68×10^{-3} (5.78×10^{-3})
	L	4.62×10^{-4} (1.04×10^{-3})	4.09×10^{-4} (8.92×10^{-4})	1.09×10^{-2} (2.15×10^{-2})	9.08×10^{-3} (1.78×10^{-2})	1.03×10^{-1} (1.38×10^{-1})	8.34×10^{-2} (1.11×10^{-1})
Fix L load =0.5	H	0 (0)	0 (3.03×10^{-7})	0 (0)	7.14×10^{-7} (2.50×10^{-6})	5.56×10^{-7} (1.11×10^{-6})	2.75×10^{-5} (5.43×10^{-5})
	L	1.19×10^{-4} (3.04×10^{-4})	1.20×10^{-4} (2.91×10^{-4})	3.83×10^{-3} (8.02×10^{-3})	3.27×10^{-3} (6.70×10^{-3})	4.11×10^{-2} (5.78×10^{-2})	3.36×10^{-2} (4.78×10^{-2})

QBOPSS retains a manageable size of PBSs by increasing the number of clusters. As shown in Figure 3.5, QPIS achieves high QoS differentiation under all traffic settings. For example, for the PBS size of 8×8, high-priority packets experience a packet loss probability that is three to five orders of magnitude lower than low-priority packets.

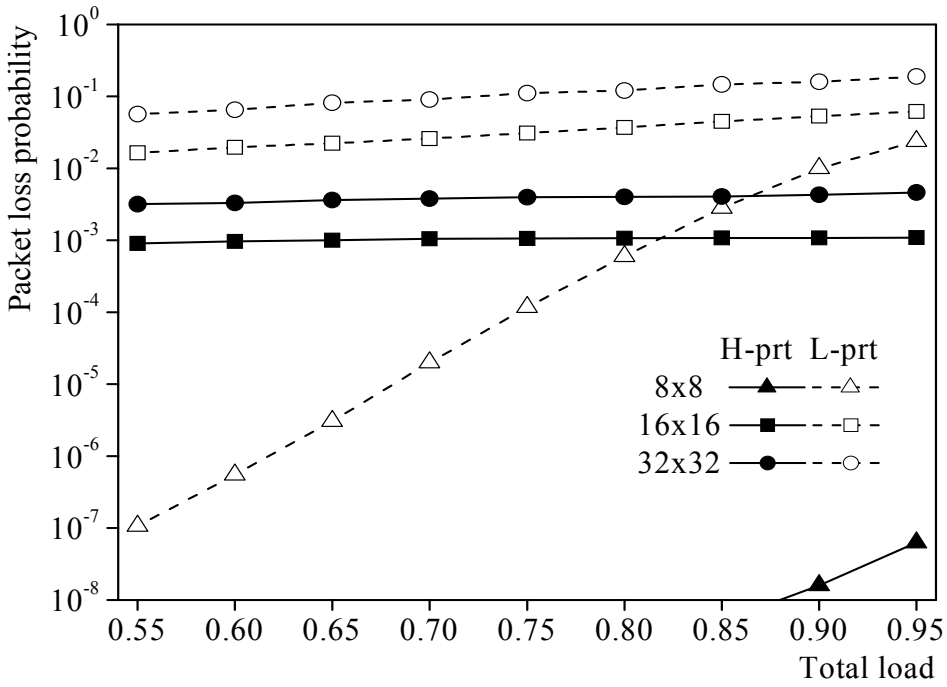
Furthermore, we draw a comparison of packet loss probability among QPIS, SMinB, and SMinD under two PBS sizes, 8×8 and 16×16, where the number of delay lines is also given as 4. Simulation results are shown in Figures 3.6 and 3.7. We observe that QPIS greatly outperforms SMinB and SMinD due to considering constraints (C1) and (C2) at the same time. Among these algorithms, SMinD undergoes the worst loss probability. This implies that constraint (C1) has greater impact on loss probability than constraint (C2), and explains the rationale behind the design that the degree-based heuristic rule (Heuristic-rule one) takes precedence over delay-based rule (Heuristic-rule two). As particularly shown in Figure 3.6(a) under lighter high-priority load (H=0.2), QPIS outperforms SMinB and SMinD on QoS differentiation (in terms of the ratio of high-priority to low-priority loss probability) by three and four orders of magnitude, respectively.

Finally, we demonstrate the impact of the optical buffer size on packet loss probability of QBOPSS using the QPIS scheduling algorithm. For simplicity, traffic flows are assumed to follow the BP model, and the packets are set to the same priority. Here, we adopt three different sizes of PBS, i.e., 8-by-8 (N=2, W=4), 16-by-16 (N=4, W=4), and 32-by-32 (N=8, W=4). In each case, we use three different buffer sizes-buffer-less (D=1, which results to B=0), a smaller buffer size (D=4), and a larger buffer size (D=8), as indicated in Table 3.5. We observe a crucial fact in all cases that, compared to the buffer-less system, QBOPSS achieves drastic improvement in throughput by applying only a handful of optical buffers (D=4, which results in B=12 and D=24). However, as the buffer size grows (D=8 resulting in B=56), the

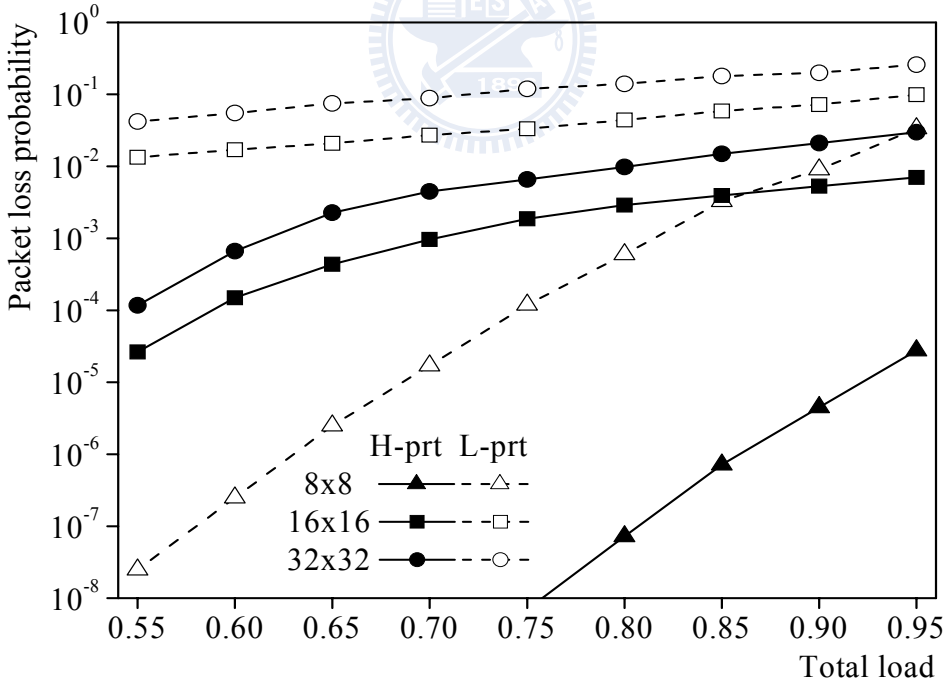
effectiveness of the improvement is decreased especially for large sized BPS. This fact justifies our economic use of downsized optical buffers.

Table 3.5 Packet loss probability of QBOPSS using QPIS under different buffer sizes (in BP traffic model)

PBS Size	Buffer Size	Total load				
		0.55	0.65	0.75	0.85	0.95
8×8 (*N*=2 *W*=4)	*D*=1, *M*=2 (*B*=0)	3.48×10^{-2}	5.01×10^{-2}	6.99×10^{-2}	9.42×10^{-2}	1.23×10^{-1}
	D=4, *M*=4 (*B*=12)	0	6.34×10^{-6}	1.97×10^{-4}	3.66×10^{-3}	2.41×10^{-2}
	D=8, *M*=8 (*B*=56)	0	0	1.66×10^{-7}	2.86×10^{-5}	4.87×10^{-3}
16×16 (*N*=4 *W*=4)	*D*=1, *M*=4 (*B*=0)	9.41×10^{-2}	1.25×10^{-1}	1.57×10^{-1}	1.88×10^{-1}	2.16×10^{-1}
	D=4, *M*=4 (*B*=12)	1.26×10^{-2}	1.69×10^{-2}	2.39×10^{-2}	4.06×10^{-2}	6.84×10^{-2}
	D=8, *M*=8 (*B*=56)	1.26×10^{-2}	1.68×10^{-2}	1.94×10^{-2}	3.89×10^{-2}	6.08×10^{-2}
32×32 (*N*=8 *W*=4)	*D*=1, *M*=8 (*B*=0)	1.36×10^{-1}	1.79×10^{-1}	2.21×10^{-1}	2.60×10^{-1}	2.96×10^{-1}
	D=4, *M*=8 (*B*=24)	3.85×10^{-2}	5.84×10^{-2}	8.29×10^{-2}	1.16×10^{-1}	1.55×10^{-1}
	D=8, *M*=8 (*B*=56)	3.83×10^{-2}	5.71×10^{-2}	8.20×10^{-2}	8.78×10^{-2}	1.53×10^{-1}

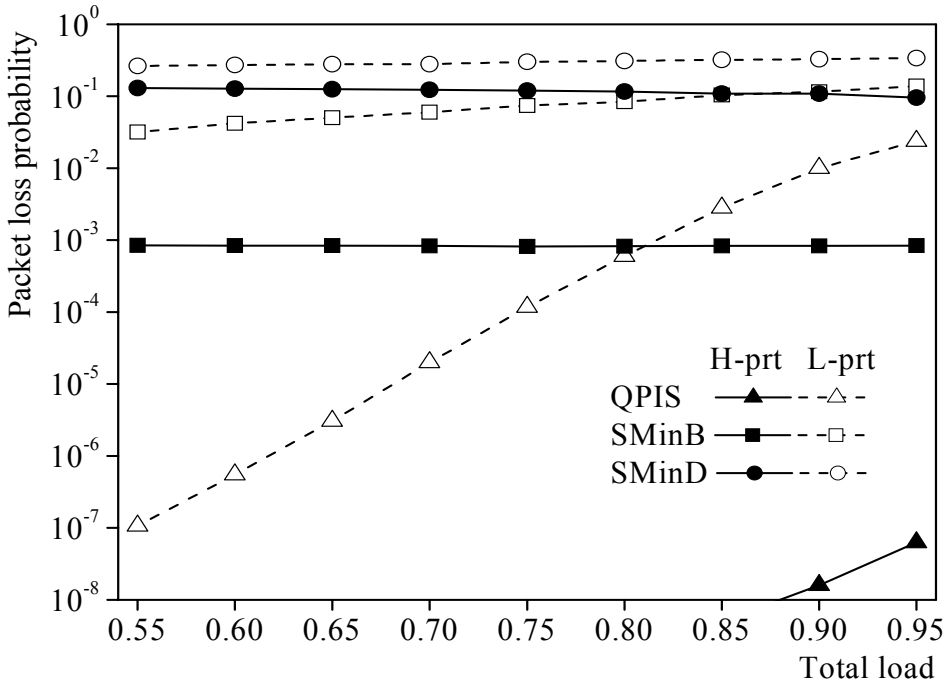


(a) Fix high-priority load H=0.2

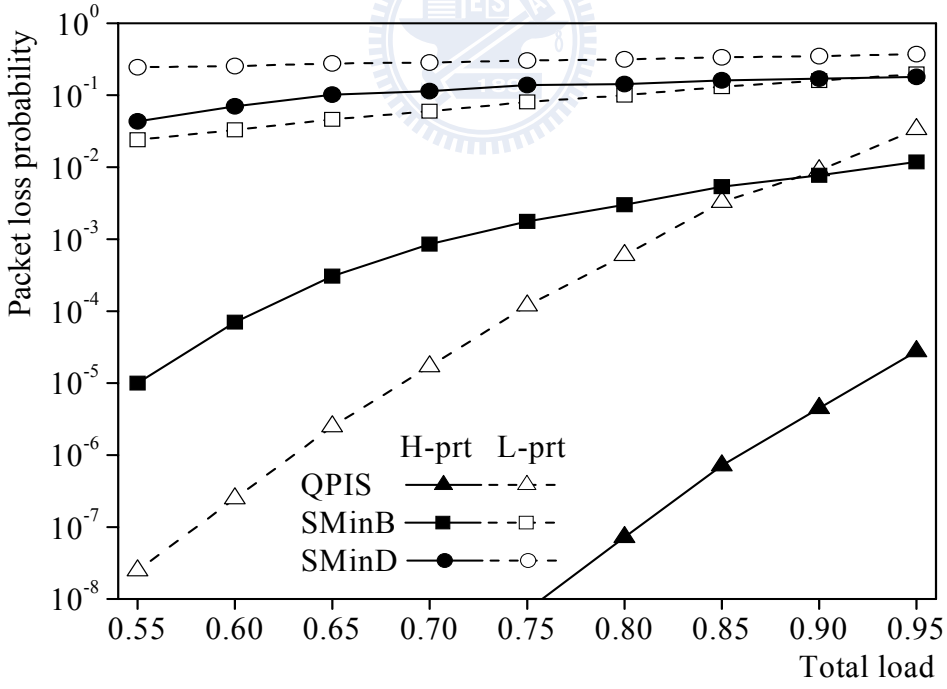


(b) Fix low-priority load L=0.5

Figure 3.5 Packet loss probabilities of QPIS under three practicable PBS sizes (in BP traffic model).

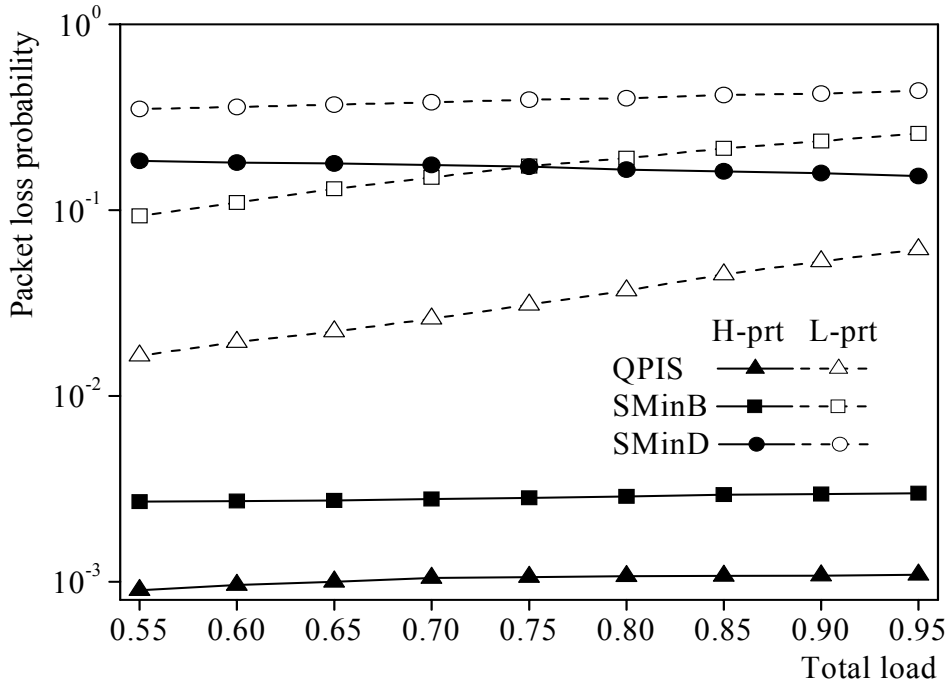


(a) Fix high-priority load H=0.2

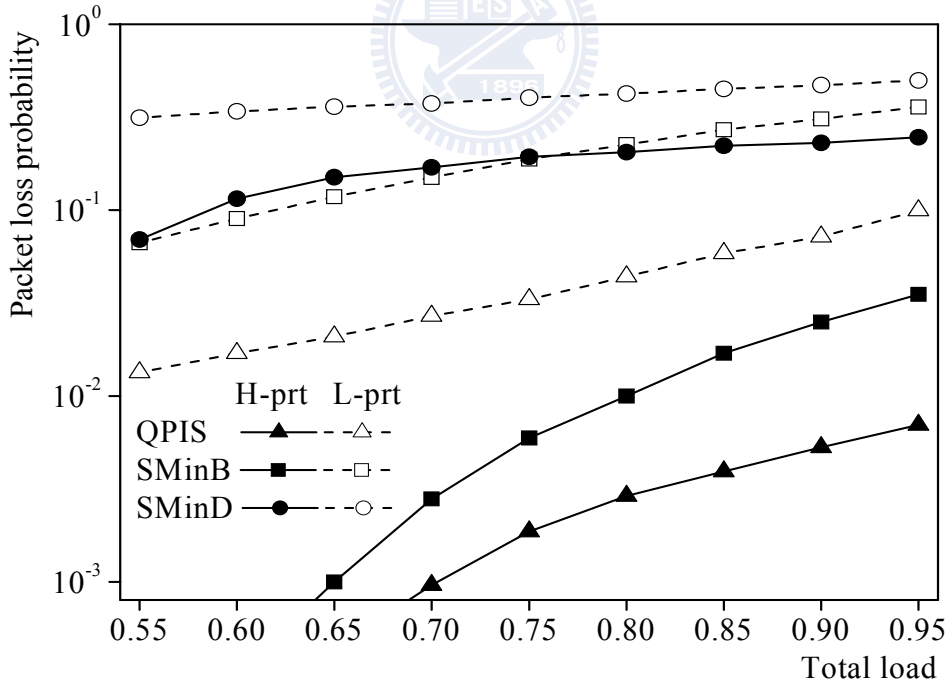


(b) Fix low-priority load L=0.5

Figure 3.6 Packet loss probability comparisons for three algorithms under 8×8 PBS size (in BP traffic model).



(a) Fix high-priority load H=0.2



(b) Fix low-priority load L=0.5

Figure 3.7 Packet loss probability comparisons for three algorithms under 16×16 PBS size (in BP traffic model).

3.5 Summary of Part II Thesis

We have proposed a QoS-enabled scalable almost-all optical packet switching system, QBOPSS. The system incorporates cluster-based downsized pseudo-Banyan optical switches and a small amount of feed-forward FDL buffers for the optical switching of packet payloads. Compared with prevailing space switches structures, QBOPSS was shown to include the least number of optical component counts and schieve the best output signal integrity. While packet payloads are switched all optically in QBOPSS, packet headers are electrically processed by a central switch controller, including a QoS parallel incremental scheduling algorithm, QPIS. Packet scheduling in QBOPSS is governed by QPIS. Through a three-phase algorithm, for newly-arriving packets per time slot, QPIS determines the target sound-path set, aiming at minimizing the loss probability for high-priority packets while maximizing system throughput subject to satisfying constraints C1 and C2. QPIS was proved to be incremental in the sense that transient sound-path sets are monotonically non-decreasing throughout each iteration round, which is a crucial feature for almost-all OPS systems with time-bound requirement. Moreover, we presented a hardware system architecture for the parallel implementation of QPIS. As a result, QPIS requires an exceedingly low computational complexity, $O(NW \times \log_2(NMW))$. We drew comparisons of system performance (packet loss probability, QoS differentiation, and computational complexity) between QPIS and three other scheduling algorithms, i.e., the exhaustive Optimal method, SMinB, and SMinD. The QPIS algorithm was shown to attain a near-optimal solution and invariably outperform SMinB and SMinD on packet loss probability, QoS differentiation, and computational complexity. Finally, we showed that, for the QBOPSS with 16-by-16 PBS under a high total load 0.95, packet loss probability is greatly improved form 2.16×10^{-1} with no buffer to 6.84×10^{-2} with B=12. However, the throughput no longer

improves as the buffer size is increased to B=56. The results justify our economic use of downsized optical buffers.

Chapter 4. Dynamic Bandwidth Allocation in First-Mile Passive Optical Network

In the first-mile environment, various applications bring the need of high bandwidth and cost effective access network design. In decades, the passive optical network (PON) is expected to be the most attractive architecture in the first-mile access network. PON is a point-to-multipoint structure with one optical line terminal (OLT) at the central end and multiple optical network units (ONUs) at the user end. The optical components from source to destination are all passive, such as splitter/coupler, for reducing the maintaining cost. While the downstream data (from OLT to ONUs) is simply broadcasted to all ONUs, the upstream data (from ONUs to OLT) is multiplexed in wavelength and/or time domains. While the wavelength division multiplexing (WDM) shares upstream bandwidth by allocating a fixed wavelength to one ONU which results in inflexibility of bandwidth sharing, the time division multiplexing (TDM) provides much dynamic share of upstream channel by allocating a varying time duration to each ONU.

Several variations of TDM-based PON have been proposed. Basically, they can be categorized as being centralized PON [50,51], distributed PON with in-band control channel [53], and distributed PON with out-of-band control channel [54]. The centralized PON makes bandwidth arbitration at the central end, i.e., OLT, by collecting requesting and/or status messages from ONUs. The two distributed PON architectures arbitrate bandwidth by all ONUs at the user end, where the requesting/status messages are distributed to all ONUs by in-band or out-of-band control channel. In order to efficiently and fairly share the system bandwidth to end users, in TDM-PON the design of dynamic bandwidth allocation (DBA) plays an important role in multiplexing upstream traffic of ONUs.

In general, DBA schemes can be divided into two classes: imprecise DBAs and precise DBAs. For the first class, the imprecise DBAs act like frame-based or cycle-based bandwidth allocating schemes, which update ONU request/status information occasionally at the begin of each frame, and distribute bandwidth of a frame to ONUs according their requests. The imprecise DBA is suitable for centralized PON or distributed PON with in-band control channel. This is because the two kinds of architectures broadcast request messages (also called control messages) by using the upstream channel. If we frequently update request messages, the utilization of upstream channel will be greatly downgraded. As a result, by imprecise DBAs, longer frame size brings higher upstream channel utilization while receiving out-of-date ONU request/status information. Many studies focus on finding a suitable frame size and/or fairly allocating the bandwidth of a frame to ONUs. The interleaved polling with adaptive cycle time (IPACT) algorithm [55] increases channel utilization by interleaving downstream polling signals from the OLT and upstream data from ONUs which is widely used in the centralized PON architecture. For fairly allocating bandwidth, IPACT further suggests several bandwidth allocating strategies, such as gated, fixed, limited, constant/linear credit, and elastic services. However, by adopting IPACT, the OLT allocates bandwidth to a particular ONU by only considering the request/status information of its own. In fact, for IPACT, taking local information into consideration will cause the unfair share of bandwidth among ONUs. Thereafter, many studies start to allocate bandwidth after receiving global request/status information from all ONUs. In [56,53], two-round allocating schemes are proposed. After receiving global request information, OLT allocates bandwidth to underloaded ONUs in the first round. After that, the excess bandwidth is than shared by overloaded ONUs according to their weight [56] or request [53] in the second round. However, in these two-round DBAs, channel bandwidth wastes by having that some slightly

overloaded ONUs may receive much more bandwidth than their request. In order to alleviate the bandwidth waste, [57] iteratively computes excess bandwidth and re-allocates them to overloaded ONUs. While [57] improves bandwidth allocating efficiency for these two-round DBAs, as a trade-off, it suffers from complicated computational complexity.

In the second class, the precise DBAs can be viewed as packet-by-packet or slot-based scheduling algorithms with the assumption that ONU status is frequently updated. By precise DBAs, at each time slot, the bandwidth of one packet is scheduled to one ONU after the pre-scheduled packet finishes transmission. As a result, the precise DBAs outperform imprecise DBAs in fairly allocating bandwidth because precise DBAs take highly accurate global status information into consideration, and make packet-by-packet allocating decision. Due to the requirement of up-to-date status information, the precise DBA is suitable for distributed PON with out-of-band control channel. In this kind of PON architecture, because of the request messages being sent in an extra control channel, frequently updating requests does not downgrade the utilization of upstream data channel while providing highly accurate ONU status for DBA decision.

For scheduling algorithms, to my best knowledge, the generalized processor sharing (GPS) scheme is viewed as the fairest one. However, because of the bit-by-bit scheduled character, the GPS is unrealizable in practice, and many packet-by-packet GPS-approximating scheduling methods are proposed. Weighted fair queueing (WFQ) [58,59] approximates GPS by maintaining a virtual time which emulates GPS operation in background, and sends packets in increasing order of virtual finish time stamp of packets. In order to improve the WFQ, the worse-case fair weighted fair queueing (WF^2Q) [60] is proposed to deliver packets by further considering virtual start time stamp of packets in addition to virtual finish time stamp. Because of the

high computational complexity of WFQ and WF^2Q, which result from exact GPS emulation in background, a lot of schemes which tolerate inexact virtual time are studied. Self-clocked fair queueing (SCFQ) [61] estimates the virtual time by setting it to virtual finish time stamp of the packet under transmission, and again sends packets in increasing order of finish time stamp. On the other way, the two schemes, start-time fair queueing (SFQ) [62] and starting potential-based fair queueing (SPFQ) [63], estimates the virtual time by setting to virtual start time stamp of the transmitting packet. While SFQ sends packets in increasing order of start time stamp, SPFQ sends in increasing order of finish time stamp. However, it is noted that the GPS is designed and suitable for groomed and/or smooth data traffic [59,64]. Because of the bursty phenomenon of data traffic in the first-mile network, we discover that the GPS-based approximating algorithms are not suitable for users in such network environment.

In the last part of this thesis, we propose a distributed control passive optical network (DCPON) by using an out-of-band controlled channel to increase system performance and also to provide up-to-date request/status information of users. In DCPON, we also design a distributed precise DBA algorithm, called adaptable packet-by-packet rate-proportional server (A-PRPS) algorithm for dynamically allocating bandwidths among end users according to their requirement. From simulation results, we show that A-PRPS outperforms interleaved polling with adaptive cycle time (IPACT) and the cycle-based DBA on mean packet delay. Finally, we make a testbed experimentation to realize the DCPON system with A-PRPS algorithm.

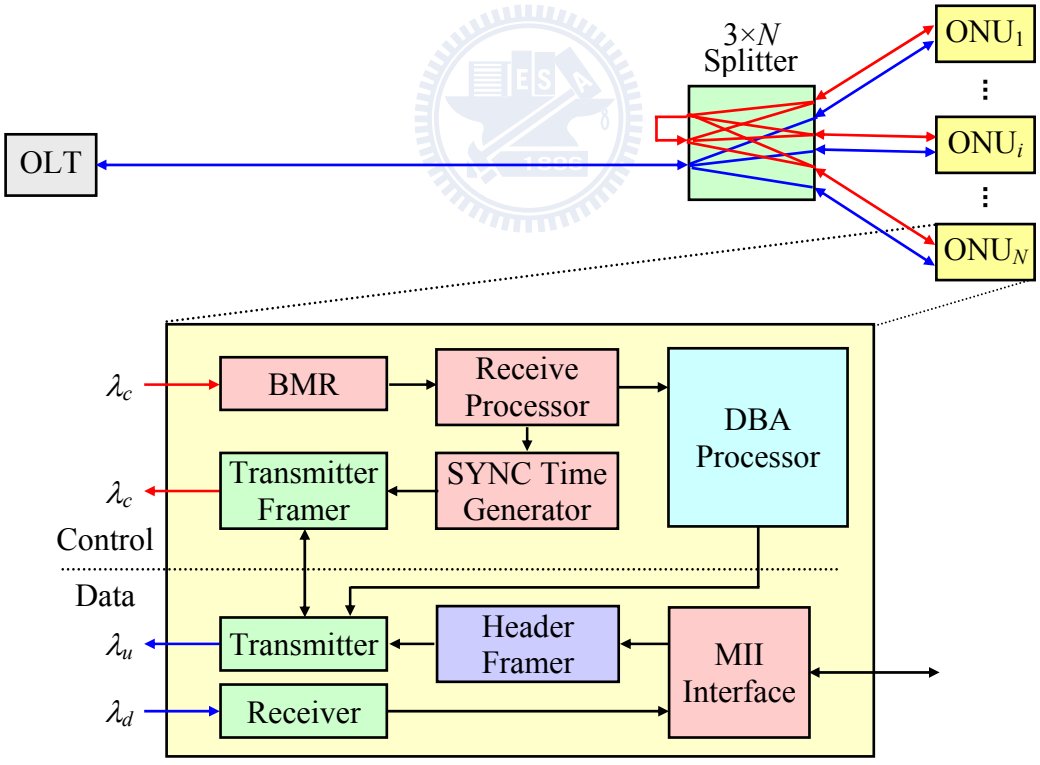
The last part of my thesis is organized as follows. In Chapter 4.1, we present the general architecture of DCPON system. In Chapter 4.2, we specifically describe the A-PRPS algorithm. In Chapter 4.3, we show simulation results. To realize DCPON, experimental results are given in Chapter 4.4. Finally, conclusion remarks are

summarized in Chapter 4.5.

4.1 DCPON System Architecture

As shown in Figure 4.1 the distributed control passive optical network (DCPON) uses an out-of-band control channel (λ_c) to distribute control messages of optical network units (ONUs), while the upstream/downstream data traffic is transmitted in upstream/downstream data channel (λ_u/λ_d) [54]. In DCPON with a total of N ONUs, a 3-by-N optical splitter is equipped in the remote node (RN), instead of the common 1-by-N splitter in PON architecture, so that a control message which contains request and/or status information of an ONU will be loop backed and broadcasted to all other ONUs.

In order to achieve high bandwidth utilization and accurate/precise bandwidth allocation for upstream channel, DCPON is designed to be a slotted-based PON



Legend:

OLT: optical line terminal;
ONU: optical network unit;
BMR: Burst Mode Receiver;
MII: Media Independent Interface;
λ_c: Control channel;
λ_u: Upstream data channel;
λ_d: Downstream data channel;

Figure 4.1 DCPON network and system architecture.

system, where both the control channel and upstream data channel are divided into fixed-sized time slots. (Notice that, the downstream data channel is not necessary to be slotted. This is because the downstream data channel is only used by OLT to broadcast information, and no bandwidth allocation is required.) As depicted in Figure 4.1, an ONU consists of two sections: data section and control section. In the data section, the downstream Ethernet data frame is simply received by ONUs without additional processing. For the upstream transmission, incoming Ethernet data frames of ONUs are segmented into fixed sized data packets, where each packet takes exactly one time slot long. After segmentation, each data packet is added a PON header and then stored in the buffer queue for transmission.

On the other hand, the control section transmits/receives control messages to/from ONUs through the out-of-band control channel. Further, each time slot of the control channel (also referred as a control slot) is decomposed into several mini-control slots, where a mini-control slot can be used to carry one control message of an ONU for announcing its request and/or current status. ONUs broadcast their control messages in the mini-control slot in a round-robin fashion, offering frequent update of ONU status. It is noted that initial ranging has to be performed at the beginning when ONU powered up. It guarantees that all ONUs are fully synchronized, and no collision occurs during updating control information. After receiving up-to-date request/status information (control messages) from all ONUs, the dynamic bandwidth allocation (DBA) algorithm distributedly operates at each ONU to arbitrate bandwidth arrangement by themselves. Accordingly, depending on how many bits the request/status information needed for the adopted DBA, the length of one mini-control slot is positively proportional to a control message size. Smaller control message size leads to shorter mini-control slot resulting in the increase of updating rate of ONU status, however, the DBA may make bandwidth decision by fewer status

information. In this thesis, for simplifying the discussion, we assume that all control messages can be broadcasted in one control slot time, and also the DBA processing can be finished before the end of this control slot as shown in Figure 4.2. Later, in the simulation results, we will observe the influence of performance under various mini-control slot lengths.

In DCPON, by the frequent update of all ONU status within one control slot, the bandwidth allocation can be performed precisely. Therefore, a lot of precise DBAs are much suitable for the DCPON. As described, the precise DBA works like a packet-by-packet version scheduling algorithm which makes bandwidth allocation by taking up-to-date ONU request/status information into consideration. Considering the scheduling algorithms, to my best knowledge, the generalized processor sharing (GPS) algorithm is conceptually viewed as the fairest method. Due to the infeasibility of bit-by-bit scheduling property in GPS, many packet-by-packet GPS-based approximating scheduling algorithms are proposed (ex: WFQ, SFQ, SPFQ). Recall that, the GPS shares bandwidth round-robinly for all backlogged ONUs in bit-by-bit version. Here, we say an ONU is backlogged if there are packets being queued in such

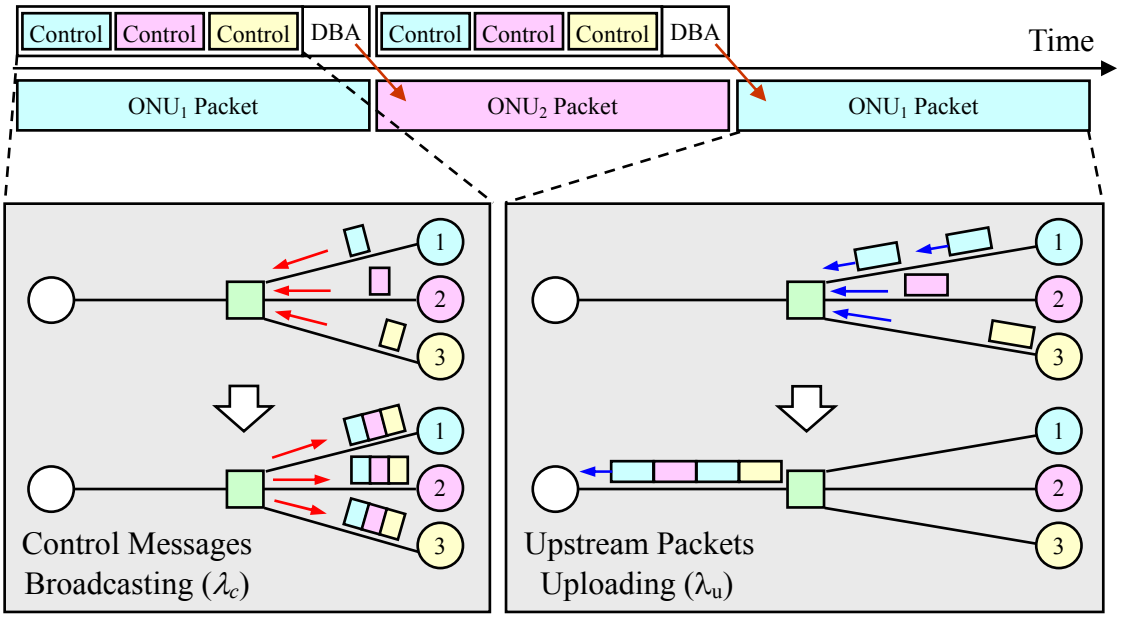
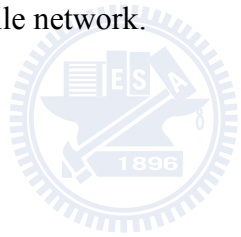


Figure 4.2 DCPON operating concept (3 ONUs example).

ONU; otherwise, we say this ONU is idle. While an ONU becomes idle, the ONU is punished during the idle period. This is because by GPS such ONU does not retrieve any bandwidth which the ONU loses during its idle period. As a result, the GPS scheme seems much suitable for arbitrating bandwidth among users with groomed or smooth traffic. However, the first mile PON network is close to end users which results in highly bursty phenomenon of user traffic, i.e., the traffic rapidly switches between idle periods and backlogged periods. Therefore, by using GPS, the ONU with high bursty traffic will suffer from a lot of punishment which brings unfair share of bandwidth among ONUs. As a result, in the following investigations, we propose a precise distributed DBA algorithm, called adaptable packet-by-packet rate-proportional server (A-PRPS) for DCPON to address the bandwidth scheduling among bursty users in the first-mile network.



4.2 The A-PRPS Algorithm

Before delving into the detail of the adaptable packet-by-packet rate-proportional server (A-PRPS) algorithm, again we summarize notations that are used throughput this and following discussions. Assume a set of N ONUs share the upstream link with total channel bandwidth R bits/sec in DCPON. The upstream data is divided into fixed sized packets which each packet contains L bits. For each ONU_i, two system parameters: the weight ρ_i and the maximum idle threshold T_i are given according to its bandwidth requirement and bursty degree of traffic. For simplicity, we assume that $\sum_{j=1}^{N} \rho_j \le R$. Let $W_i[t_1, t_2]$ denotes the bandwidth received for ONU_i between t_1 and t_2, and $Q_i(t)$ represents the amount of packets queued in the ONU_i at time t. It is noted that ONU_i is backlogged (or in the backlogged period) at time t if $Q_i(t)>0$. We also denote that $ONU_i \in B(t)$ where $B(t)$ is the set of backlogged ONUs at time t. Otherwise, ONU_i is idle (or in the idle period), denoted as $ONU_i \in I(t)$ where $I(t)$ is the idle ONU set at time t.

We now specifically describe the A-PRPS algorithm that schedules bandwidth among ONUs by taking (ρ_i, T_i) and precise queue information $Q_i(t)$ into consideration. Further, in A-PRPS, we introduce an ONU potential function $P_i(t)$ for each ONU_i and a system potential function (also called global potential function) $P(t)$ to help memorize the status of bandwidth allocation up to time t. Conceptually, the ONU potential function $P_i(t)$ is a non-decreasing function which keeps track of the amount of weight-normalized bandwidth received by ONU_i until time t. As detailed in Figure 4.3, if the ONU_i keeps backlogged during interval $[t_1, t_2]$, $P_i(t)$ increases by adding the amount of normalized bandwidth $W_i[t_1, t_2]/\rho_i$ from t_1 to t_2, that is $P_i(t_2)=P_i(t_1)+W_i[t_1, t_2]/\rho_i$. Otherwise, $P_i(t)$ is keeping un-changed during the idle period because of no bandwidth being received. It is worth to notice that, if an idle ONU_i wakes up at time t

(i.e., becomes backlogged from idle at time t), the potential $P_i(t)$ has to be reasonably updated according to its length of related idle period ($\widetilde{I}_i$: which is defined to be a random variable of idle duration) up to time t and bandwidth allocating status of other ONUs, which are information that the ONU$_i$ misses during its idle period.

In order to simplify the $P_i(t)$ updating process while ONU$_i$ transits into backlogged from idle at time t, in A-PRPS, the system potential function $P(t)$ is then introduced to retain the bandwidth allocating status for ONUs. Similar to ONU potentials, $P(t)$ is defined to be a non-decreasing function, which is updated by taking

Algorithm Adaptive Packet-by-Packet Rate-Proportional Server (A-PRPS);

Input parameters: Given (ρ_i, T_i) for each ONU$_i$, and $\sum_{j=1}^{N} \rho_j \le R$

begin

/* the following four procedures 1~4 can be run independently and parallelly */

1. Update ONU potential function $P_i(t)$ as follows:
   if (ONU$_i$ is idle at time t) $P_i(t)=P_i(t\text{-})$;
   elseif (ONU$_i$ becomes backlogged from idle at time t)
      if (the related idle period $\widetilde{I}_i \le T_i$, i.e., experiencing pseudo-idle period)
         $P_i(t)=P_i(t\text{-})$;
      elseif (the related idle period $\widetilde{I}_i > T_i$, i.e., experiencing normal-idle period)
         $P_i(t)=\max\{P_i(t\text{-}), P(t)\}$; **endif**
   elseif (ONU$_i$ remains backlogged during interval $[\tau, t]$)
      $P_i(t)= P_i(\tau)+W_i[\tau, t]/\rho_i$; **endif**
2. Update system potential $P(t)$ as follows:
   if (during serving a packet) $P(t)=P(t\text{-})$;
   elseif (a packet departs at time t) $P(t)= \max\left\{P(t-), \min_{j\in B(t-)}\left\{P_j(t-)\right\}\right\}$; **endif**
3. Schedule the ONU to send packets:
   if (a packet finishes transmission at time t)
      Schedule the backlogged ONU$_i$ with $P_i(t)= \min_{j\in B(t)}\left\{P_j(t)+\frac{L}{\rho_j}\right\}$ as the candidate to send a packet;
      If more than one ONU with minimal potential is found, randomly choose one among minimal ONUs; **endif**
4. System reset condition checking:
   if (each ONU$_i$ keeps idle exceeding T_i at time t)
      Reset $P(t)=P_i(t)=0$ $\forall i$; **endif**

end

Figure 4.3 The A-PRPS Algorithm.

all backlogged ONU potentials into consideration. As described in Figure 4.3, the system potential is updated as $P(t)=\max\left\{P(t-),\ \min_{j\in B(t-)}\left\{P_j(t-)\right\}\right\}$ at each time a packet departs. By exploiting $P(t)$, after an ONU$_i$ transits into backlogged from idle at time t, its potential $P_i(t)$ is updated by firstly checking whether the related idle length ($\widetilde{I}_i$) until to time t exceeds the maximum idle threshold (T_i) or not. If not (i.e., $\widetilde{I}_i \le T_i$), the related idle period is viewed as a pseudo-idle period, where the "pseudo" marks that the idle period is high-probably caused by bursty phenomenon. Therefore, $P_i(t)$ hold its value without any update at time t. And, the holding action leads ONU$_i$ to be scheduled more bandwidth after time t for retrieving the lost bandwidth during pseudo-idle period. Otherwise, if yes (i.e., $\widetilde{I}_i > T_i$), the related idle period is denoted as a normal-idle period, where the "normal" means less relation between the idle period and bursty phenomenon. Under such condition, at time t, A-PRPS update $P_i(t)$ to be the maximum of the current ONU potential $P_i(t\text{-})$ and the system potential $P(t)$. This helps ONU$_i$ to catch up the current status of global system.

By keeping track of ONU potentials, A-PRPS schedules the backlogged ONU with minimal potential plus one normalized packet work ($P_i(t)+L/\rho_i$) as a candidate to send its packet each time when the pre-scheduled packet finishes transmission. If more than one minimal ONU is found, one among these ONUs are randomly selected. To implement A-PRPS in DCPON, $Q_i(t)$ and $P_i(t)$ are the only control messages which is needed to be broadcasted by ONU$_i$ in the mini-control slot [54]. Further, for preventing from overflow, the system potential $P(t)$ and all ONU potentials $P_i(t)$ can be reset to zero at time t^*, if each ONU$_i$ is keeping idle with the related idle period longer than its maximum idle threshold at time t^* (i.e., $t^* \in \widetilde{I}_i$ and $\widetilde{I}_i > T_i$ for all i). It is noted that, if some ONUs experience pseudo-idle period instead of normal-idle up

to time t^*, A-PRPS still has to maintain current values of system and all ONU potentials in order to memorize information of lost bandwidth for such pseudo-idle ONUs.

4.3 Simulation Results

In this sub-chapter, we first draw comparisons of mean packet access/queueing delay among A-PRPS and two DBA schemes, interleaved polling with adaptive cycle time (IPACT) algorithm [55] and cycle-based dynamic bandwidth allocating (cycle-based DBA) algorithm [56]. Then, we focus on investigating the relationshop between traffic bursiness and maximum idle threshold. Finally, the impact of control information update frequency for DCPON is evaluated.

The first part comparison mainly justifies that the architecture of distributed PON with out-of-band control channel, i.e., DCPON structure, brings remarkable system performance with respect to packet access/queueing delay. Recall that the IPACT is the current standard DBA applied for centralized PON which centralizedly arranges bandwidth sharing by OLT. After receiving bandwidth requests from ONUs, OLT allocates and polls each ONU in round robin fashion. Importantly, for increasing upstream data channel utilization, in IPACT, the downstream polling and upstream data transmission work interleavingly. While there are many different bandwidth allocation schemes, such as fixed, limited, constant credit, linear credit, and elastic services, in this simulation we adopt limited service in IPACT. The cycle-based DBA is adoptted for distributed PON with in-band control channel. By reflecting request messages in upstream data channel, the two-round allocating scheme is operated distributedly in ONUs to share bandwidth according to their weights.

In this simulation, there are 32 ONUs which are assigned equal weight (ρ_i=1/32), and the distance between the OLT and ONUs is 20 km which results in 0.1 ms propagation delay (round trip time=twise of propagation delay=0.2 ms). Each ONU generates the same traffic demand following interrupted Poisson process (IPP) with burstiness B=50, where the burstiness of an ONU is measured as the peak rate divided by the mean rate of arriving packets. As shown in Figure 4.4, for the cycle-based DBA, the setting of one cycle length highly influences the system performance. While shorter cycle length makes request announcment much faster, as described, the channel bandwidth is reduced by control message broadcasting which results in longer mean packet delay. The mean delay time gets worse as the traffic load increases. As a proof, while the cycle-based DBA with cycle=4 ms receives better packet delay than IPACT under all traffic loads, the middle cycle length (2 ms) has longer delay than IPACT under traffic load=0.98, and the short cycle length (1 ms) is worse than IPACT under traffic load=0.96. Obviously, by simply setting all threshold values to 0, A-PRPS outperforms these two DBAs under various traffic loads and

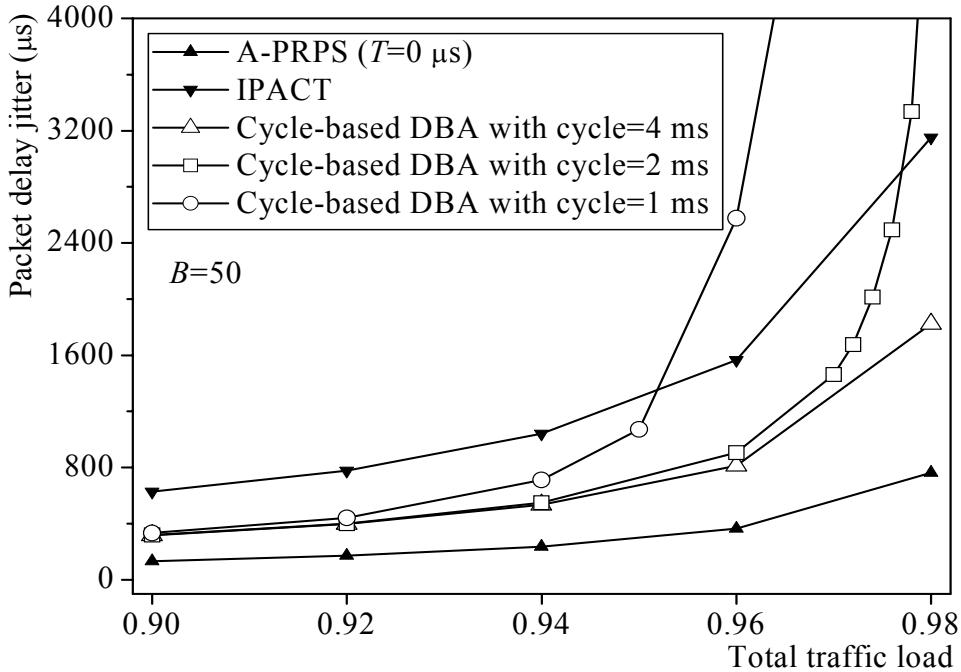
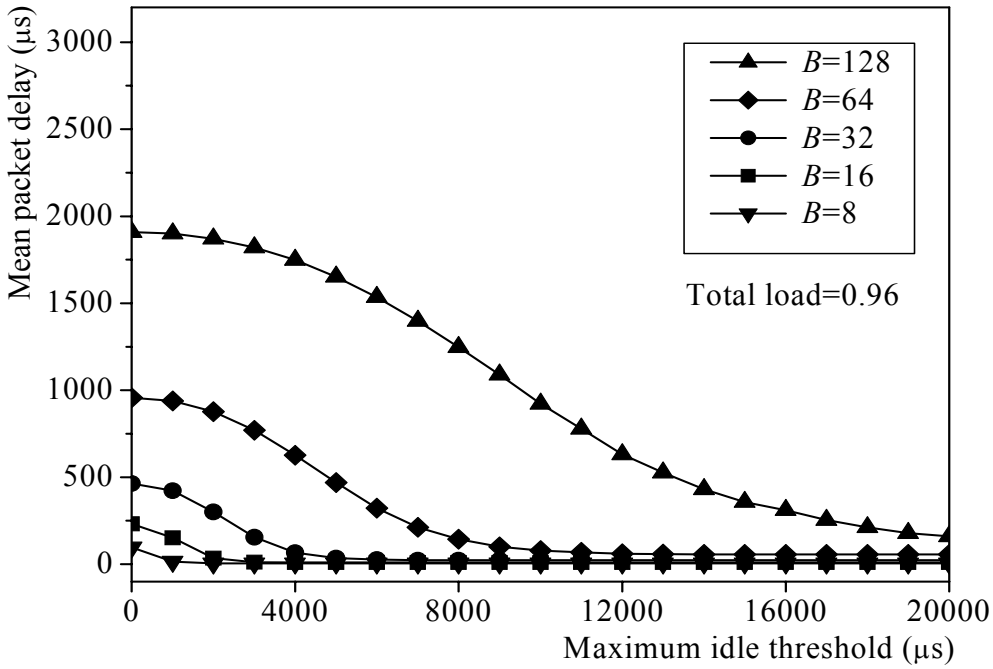


Figure 4.4 Mean packet delay comparison for different schemes.

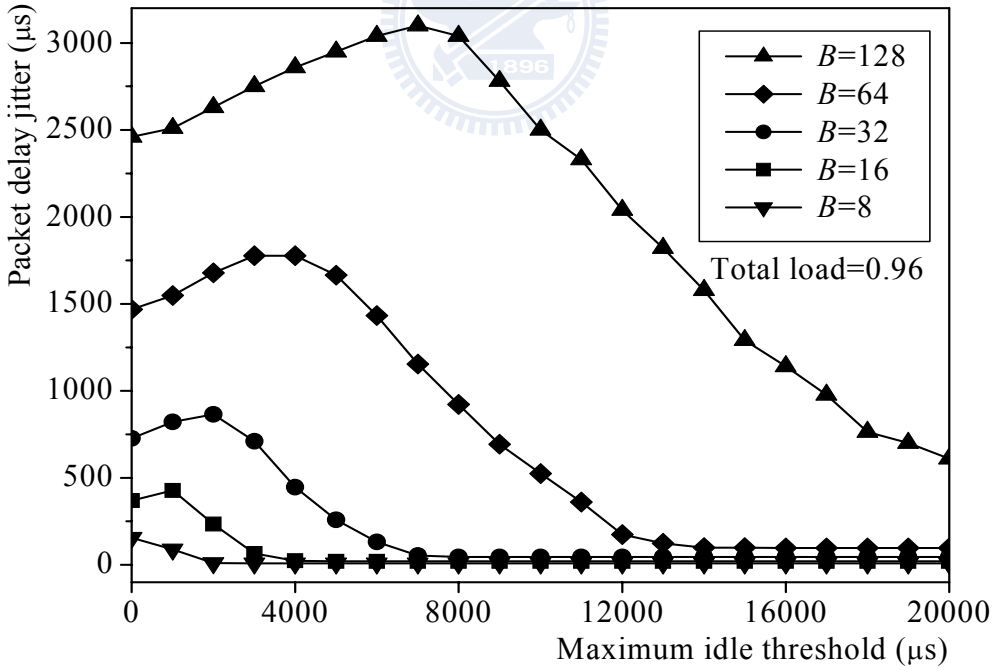
cycle settings. In fact, the setting of threshold will not influerence the mean packet delay of the DCPON system because of the work conserving property of A-PRPS. The simulation results justify that the DCPON architecture can receive much better performance by using low-speed out-of-band control channel and a precise DBA, such as A-PRPS.

In the sequal, we begin at discussing how the setting of threshold parameters affects the packet delay for various bursty traffic settings. Similar to previous setting, all ONUs have the same weight and traffic load. Within 32 ONUs, 16 of them (ONU_0~ONU_{15}) are fixed with smooth traffic (B=1) and zero threshold (T=0). The remaining ONUs are varing their traffic burstiness and related threshold values as shown in Figure 4.5. As we can see, in order to keep the mean and variance/jitter of packet delay at a low level, the threshold needs to be increased for ONUs with larger traffic burstiness. This also justifies that suitably adjusting the threshold value can efficiently downgrade packet mean delay and/or jitter to achieve QoS-level guarantee for bursty users.

Finally, we evaluate the impact of control message updating frequency under various mini-control slot lengths. We define C is the number of mini-control slots inside one control slot. As shown in Figure 4.6, high updating rate of control messages (i.e., large C value) leads to lower packet delay time. It is noted that, performance improvement almost diminishes as C goes over 4. As a result, for the same size of control message in bits, the control channel can be implemented with cost-effective low-speed lasers (i.e., decrease C) for largely enhancing the performance of upstream data channel.



(a) Mean packet delay



(b) Packet delay jitter

Figure 4.5 Maximum idle threshold (*T*) effect for A-PRPS under various traffic burstiness setting (*B*).

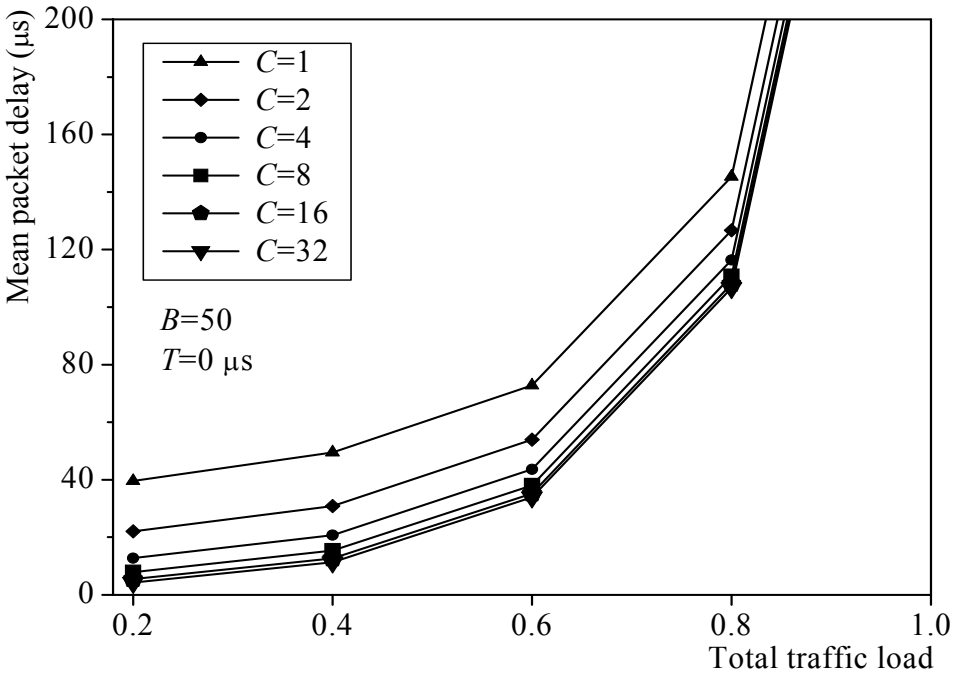


Figure 4.6 The impact of control message update frequency.

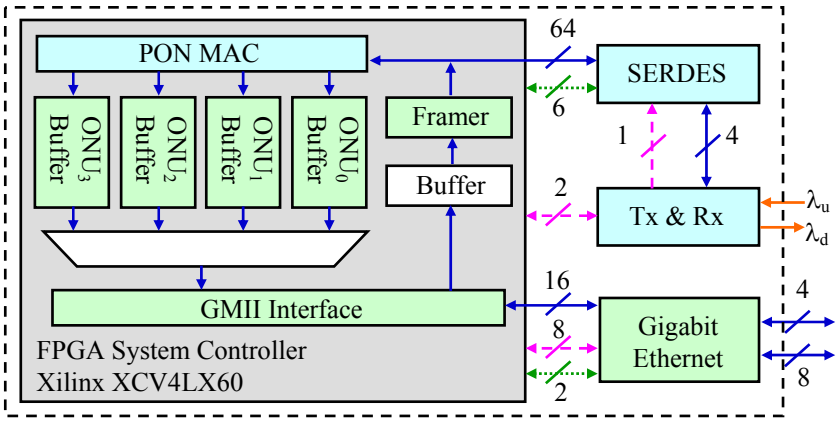
4.4 Testbed Experimentation

We constructed an experimental testbed for DCPON shown in Figure 4.7. The testbed consists of one OLT and four ONUs. The upstream data channel operated at rates of 1.25 Gbps on wavelength 1310 nm and the downstream data channel operated at rate of 2.5 Gbps on wavelength 1490 nm. In addition, we used 1550 nm to support 125 Mbps out-of-band control channel. The proposed A-PRPS algorithm was implemented on a Xilinx XCV4LX60 FPGA board (Figure 4.7(b)). Queue size in each ONU was 228 frames. Data channels and control channel were time-slotted with each slot being 2 μsec. Figure 4.7(c) depicts the control frame structure (one control message). Fields were duplicated for error detection. In the testbed, $Q_i(k)$ was simplified to one bit to indicate if there was any packet waiting in queue at k-th time slot. Each control slot was further divided into two control mini-slots, each one containing 16 bytes control message.

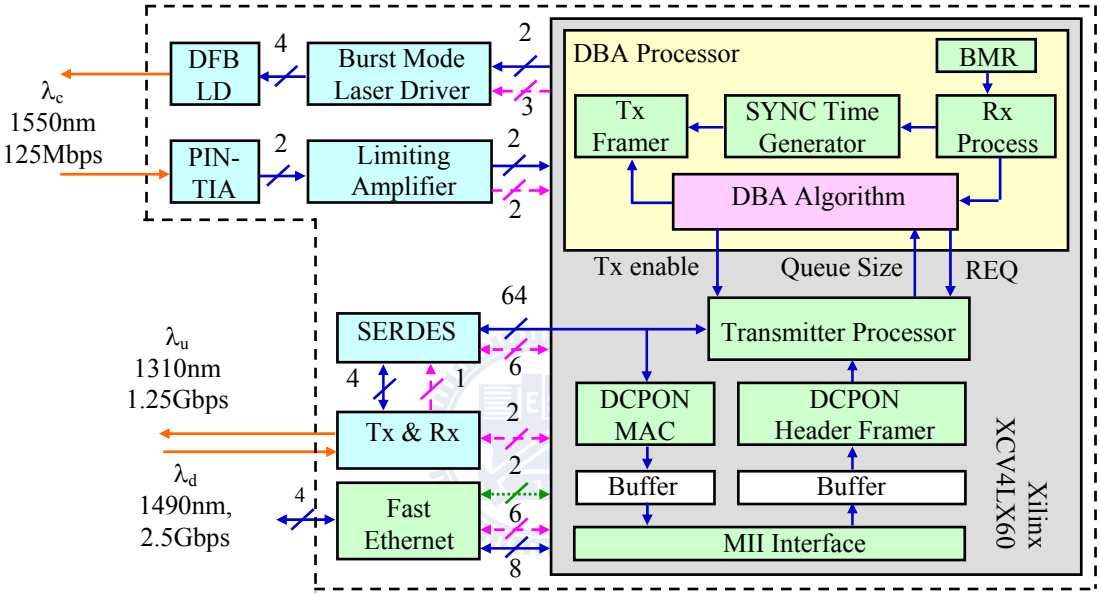
For the control channel, we implemented a 125 Mbps burst mode receiver. We used a low time constant AC-couple amplifier in the front-end circuit and adopted an oversampling-based burst-mode CDR circuitry inside the FPGA. In this configuration, among 56 bits preamble overhead, 35 bits was allocated to front-end amplifier and remaining 21 bits were used for clock extraction. Figure 4.8(a) displays snapshots of the signal waveforms. We intentionally adjusted the output power of ONUs' data channels to identify their sources. For simplicity, equal weight (ρ_i=0.25) was set for all ONUs, and packet length (L) is fixed to 1. A-PRPS determined the ONU that had non-empty queue and minimum $P_i(k)+1/\rho_i$ value to be the winner. The waveform shows two control slots, where slot k= [(Q_2, P_2)=(1,4), (Q_3, P_3)=(1,0)] and slot k+1=[(Q_0, P_0)=(0,4), (Q_1, P_1)=(1,4)]. Slot k-1 and slot k jointly determined ONU$_3$ to send packet. After sending the packet, every ONU countdowns Q_3=0, and increases P_3

by 1/0.25=4. As a result, ONU_1 became the next one to send packet.

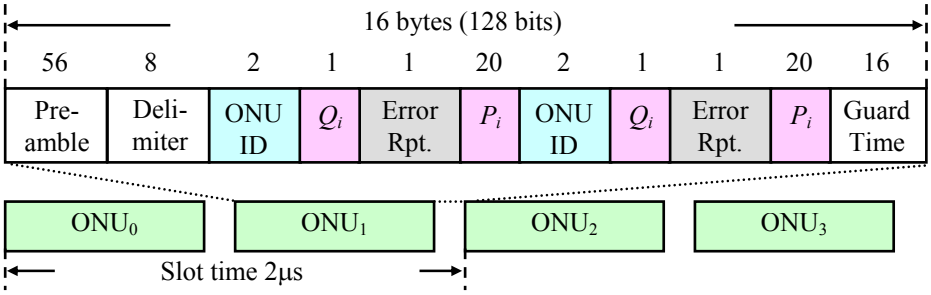
We further used Smartbit to generate input traffic. The arrival traffic followed Poisson process and each Ethernet packet had the same size of 1500 bytes. To evaluate the system under unbalanced load, the mean rate was 100 Mbps for ONU_0~ONU_2, and was 900Mbps for ONU_3. ONU_3 acted as a malicious node trying to seize a lot of network bandwidth. Figure 4.8(b) depicts the mean packet access/queueing delay on cumulative distribution function (CDF). We discover that the proposed A-PRPS algorithm outperforms IPACT. For 99% of the input traffic, the delay time of A-PRPS is less than 6.8 μsec. The gap between A-PRPS and IPACT is up to 3 orders of magnitude.



(a) OLT setup



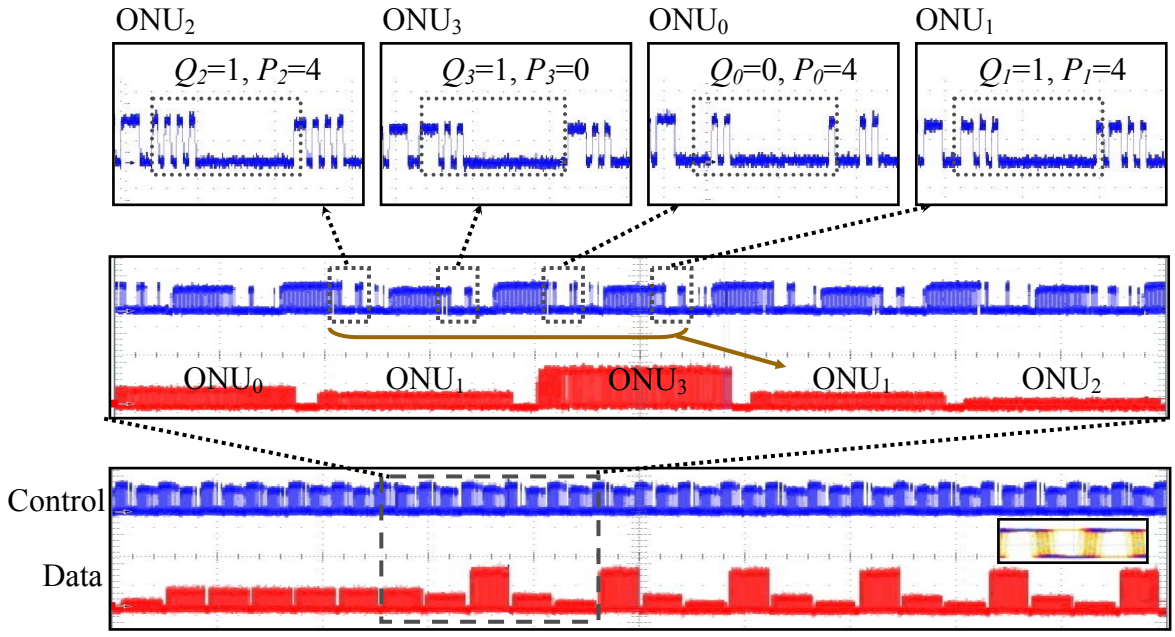
(b) ONU setup



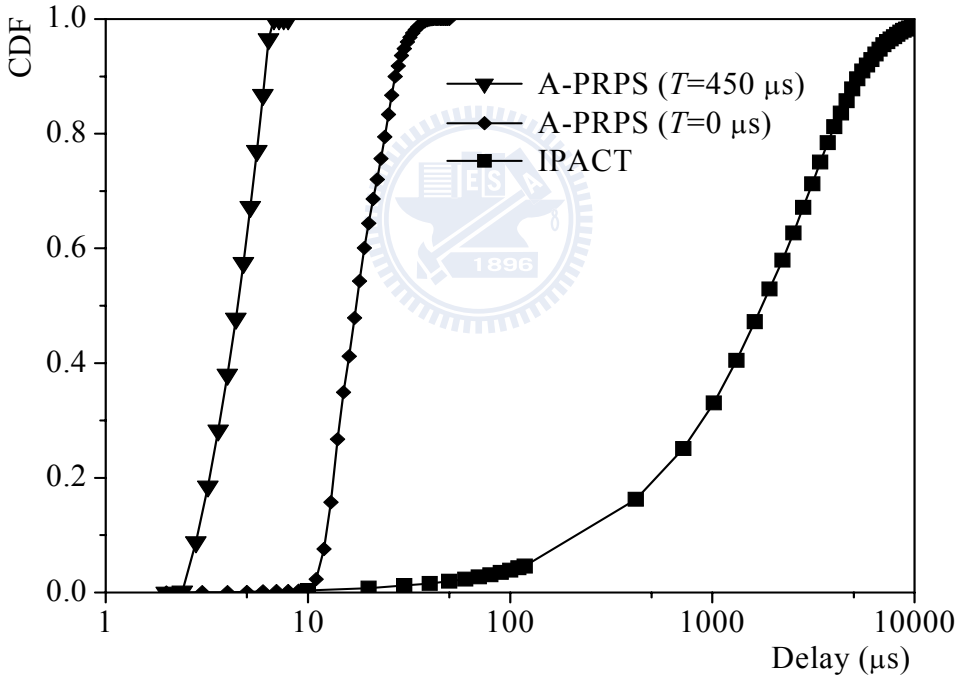
(c) Control frame format

Data & Control line (optical); Data line (electric);
Control line (electric); Clock line (electric);

Figure 4.7 DCPON- experimental setup.



(a) Experimental results



(b) Access/queueing delay comparisons

Figure 4.8 DCPON- experimental results.

4.5 Summary of Part III Thesis

In the last part of this thesis, we have proposed a distributed control passive optical network (DCPON) by adopting an out-of-band control channel to efficiently increase the upstream data channel performance. Also, for efficiently allocating bandwidth for users with bursty traffic in the first-mile environment, a precise DBA, called adaptable packet-by-packet rate-proportional server (A-PRPS), is designed in DCPON. We draw comparisons between A-PRPS and two other DBAs, IPACT and cycle-based DBA which separately operate on centralized PON and distributed PON with in-band control. The A-PRPS is shown to have much lower mean packet access/queueing delay than two other DBAs. For A-PRPS, to fairly share bandwidth among ONUs with various burstiness, simulation results show that the maximum idle thresholds are expected to be set large for high bursty ONUs. Finally, we show that the upstream channel performance is improved as the length of mini-control slot decreases, i.e., C increases. However, the improvement diminisges as C goes over 4. The result means that, for implementing A-PRPS in DCPON, cost-effective low speed out-of-band control channel is sufficient for achieving high system performance. The experimental results are then given as a hardware implementation of DCPON.

Chapter 5. Conclusions and Future Works

In my thesis, we have investigated three main topics. In the first topic, we resolve the routing and wavelength assignment (RWA) problem by adopting a Lagrangean relaxation with heuristics (LRH) algorithm in the optical core WDM networks. From simulation results, the LRH algorithm achieves higher accuracy and lower computational complexity than linear programming relaxation (LPR) based schemes. Furthermore, for large sized networks, LRH is shown achieving a near optimal solution within acceptable computation time. For the future works in this topic, we are going to further investigate the converge speed of LRH scheme in various network environments. For example, the increase of FSC and/or LSC nodes may highly influence the converge speed. On the other hand, considering the subgradient method for updating the Lagrangean multipliers, the updating rule is highly related to converge speed also. Various updating rules can be investigated and applied for enhancing the performance.

In the second topic, we focus on the construction of optical packet switching (OPS) system in the optical metropolitan WDM networks. We present a flexible and cost-effective OPS system, called QoS-enabled pseudo-Banyan optical packet switching system (QBOPSS), by using a group of downsized optical pseudo-Banyan space switches (PBSs) and a handful of fiber-delay-line (FDL) optical buffers. For achieving QoS differentiation and high system performance, the QoS parallel incremental scheduling (QPIS) algorithm is proposed to deal with the packet scheduling problem. We prove that the QPIS possesses incremental property and parallel running feature. Simulation results show that QPIS achieves near-optimal system performance, and also outperforms two other schemes on packet loss probability, QoS differentiation, and computational complexity. In the second part of

future work, while the theoretical computational complexity of QPIS has been proposed, the practical limitation of QPIS computing time related to actual system time bound constraint should be further investigated. Also, to realize the QBOPSS system, the size of pseudo-Banyan space switch or the number of clusters should be well determined because it highly influences the QPIS computing speed, statistical multiplexing gain (system throughput), and system scalability.

Finally, in the last topic, we design a distributed control passive optical network (DCPON) to achieve high upstream data channel performance and suppert precise dynamic bandwidth allocation (DBA) to end users. For first-mile users with bursty traffic, the adaptable packet-by-packet rate-proportional server (A-PRPS) is proposed to weighted fairly arbitrate bandwidth among users while taking burty phenomenon of user traffic into consideration. The following work of the last topic is to investigate the relationship between threshold parameters (T) and the measurement of burstiness degrees (B) for achieving specific QoS guarantee among end users with respect to mean packet delay and delay jitter.

References

[1] B. Mukherjee, "WDM Optical Communication Networks: Progess and Challenges," *IEEE J. Select. Areas Comm.*, vol. 18, no. 10, Oct. 2000, pp. 1810-1824.

[2] R. Ramaswami, and G. Sasaki, "Multiwavelength Optical Networks with Limited Wavelength Conversion," *IEEE/ACM Trans. Networking*, vol. 6, no. 6, Dec. 1998, pp. 744-754.

[3] R. Ramaswami, and K. Sivarajan, *Optical Networks- A Practical Perspective*, 2nd Edition, Morgan Kaufmann, 2002.

[4] D. Banerjee, and B. Mukherjee, "A Practical Approach for Routing and Wavelength Assignment in Large Wavelength-Routed Optical Networks," *IEEE J. Select. Areas Comm.*, vol. 14, no. 5, Sep. 1996, pp. 903-908.

[5] P. Ho, and H. Mouftah, "Routing and Wavelength Assignment with Multigranularity Traffic in Optical Networks," *J. Lightw. Techno.*, vol. 20, no. 8, Aug. 2002, pp. 1292-1303.

[6] K. Zhu, H. Zhu, and B. Mukherjee, "Traffic Engineering in Multigranularity Heterogeneous Optical WDM Mesh Networks Through Dynamic Traffic Grooming," *IEEE Networks*, vol. 17, no. 2, March/April 2003, pp. 8-15.

[7] B. Mukherjee, *Optical Communication Networks*, McGraw-Hill, USA, 1997.

[8] C. Xiaowen, L. Bo, and I. Chlamtac, "Wavelength Converter Placement under Different RWA Algorithms in Wavelength-Routed All-Optical Networks," *IEEE Trans. Comm.*, vol. 51, no. 4, April 2003, pp. 607-617.

[9] H. Qin, S. Zhang, and Z. Liu, "Dynamic Routing and Wavelength Assignment for Limited-Range Wavelength Conversion," *IEEE Comm. Lett.*, vol. 7, no. 3, March 2003, pp. 136-138.

[10] A. Mokhtar, and M. Azizoglu, "Adaptive Wavelength Routing in All-Optical Networks," *IEEE/ACM Trans. Networking*, vol. 6, no. 2, April 1998, pp. 197-206.

[11] B. Gavish, P. Trudeau, M. Dror, M. Gendreau, and L. Mason, "Fiberoptic Circuit Network Design under Reliability Constraints," *IEEE J. Select. Areas Comm.*, vol. 7, no. 8, Oct. 1989, pp. 1181-1187.

[12] H. Chen, C. Chu, and J. Proth, "An Improvement of The Lagrangean Relaxation Approach for Job Shop Scheduling: A Dynamic Programming Method," *IEEE Trans. Robotics and Automation*, vol. 14, no. 5, Oct. 1998, pp. 786 -795.

[13] M. Guignard, "On Solving Structured Integer Programming Problems with Lagrangean Relaxation and/or Decomposition," in *Proc. IEEE Decision and Control*, Dec. 1989, pp. 1136 -1141.

[14] R. Ahuja, T. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, USA, 1993.

[15] M. Saad, and Z. Luo, "A Lagrangean Decomposition Approach for the Routing and Wavelength Assignment in Multifiber WDM Networks", *IEEE*

GLOBECOM, *2002*.

[16] Y. Zhang, O. Yang, and H. Liu, "A Lagrangean Relaxation Approach to the Maximizing-Number- Of-Connection Problem in WDM Networks," *IEEE Workshop on High Performance Switching and Routing*, 2003.

[17] O. Crochat, and J. Boudec "Design Protection for WDM Optical Networks," *IEEE J. Select. Areas Comm.*, vol. 16, no. 7, Sep. 1998, pp. 1158-1165.

[18] M. Yuang, Y. Lin, J. Shih, P. Tien, J. Chen S. Lee, and S. Lin, "A QoS Optical Packet Switching System Architectural Design and Experimental Demonstration," to be published on *IEEE Comm. Magazine*, May 2010.

[19] M. Yuang, S. Lee, P. Tien, Y. Lin, J. Shih, F. Tsai, and A. Chen, "Optical Coarse Packet-Swtiched IP-over-WDM Network (OPSINET): Technologies and Experiments," *IEEE J. Select. Areas Comm.*, vol. 24, no. 8, Aug. 2006, pp. 117-127.

[20] M. Yuang, P. Tien, and S. Lin, "Pseudo-Banyan Optical WDM Packet Switching System with Near-Optical Packet Scheduling," accepted by *IEEE J. Select. Areas Comm. and Networking series*, and published in *IEEE Optical Comm. and Networking*, vol. 1, no. 3, Aug. 2009, pp. B1-B14.

[21] G. Papadimitriou, C. Papazoglou, and A. Pomportsis, "Optical Switching: Switch Fabrics, Techniques, and Architectures," *J. Lightw. Technol.*, vol. 21, no. 2, Feb. 2003, pp. 384-405.

[22] S. Li, *Algebraic Switching Theory and Broadband Applications*, Academic Press, USA, 2001.

[23] A. Jajszczyk, "Nonblocking, Repackable, and Rearrangable Clos Networks: Fifty Years of the Theory Evolution," *IEEE Comm. Magazine*, vol. 41, no. 10, Oct. 2003, pp. 28-33.

[24] S. Dixit, *IP OVER WDM Building the Next Generation Optical Internet*, Wiley, USA, 2004.

[25] G. Chang, J. Yu, Y. Yeo, A. Chowdhuay, and Z. Jia, "Enabling Technologies for Next-Generation Optical Packet-Switching Networks," *Proceedings of the IEEE*, vol. 94, no. 5, May 2006, pp. 892-910.

[26] R. Tucker, P. Ku, and C. Hasnain, "Slow-Light Optical Buffers: Capabilities and Fundamental Limitations," *J. Lightw. Technol.*, vol. 23, no. 12, Sep. 2005, pp. 4046-4066.

[27] M. Chia, D. Hunter, I. Andonovic, P. Ball, I. Wright, S. Ferguson, K. Guild, and M. O'Mahony, "Packet Loss and Delay Performance of Feedback and Feed-Forward Arrayed-Waveguide Gratings-Based Optical Packet Switches With WDM Inputs-Outputs," *J. Lightw. Technol.*, vol. 19, no. 9, Sep. 2001, pp. 1241-1254.

[28] Z. Zhang and Y. Yang, "Low-Loss Switching Fabric Design for Recirculating Buffer in WDM Optical Packet Switching Networks Using Arrayed Waveguide Grating Routers," *IEEE Trans. Comm.*, vol. 54, no. 8, Aug. 2006, pp. 1469-1472.

[29] S. Liew, G. Hu, and H. Chao, "Scheduling Algorithms for Shared

Fiber-Delay-Line Optical Packet Switches—Part I: The Single-State Case," *J. Lightw. Technol.*, vol. 23, no. 4, April 2005, pp. 1586-1600.

[30] F. Choa, X. Zhao, X. Yu, J. Lin, J. Zhang, Y. Gu, G. Ru, G. Zhang, L. Li, H. Xiang, H. Hadimioglu, and H. Chao, "An Optical Packet Switch Based on WDM Technologies," *J. Lightw. Technol.*, vol. 23, no. 3, March 2005, pp. 994-1014.

[31] T. Zhang, K. Lu, and J. Jue, "Shared Fiber Delay Line Buffers in Asynchronous Optical Packet Switches," *IEEE J. Select. Areas Comm.*, vol. 24, no. 4, April 2006, pp. 118-127.

[32] W. Zhong and R. Turker, "Wavelength Routing-Based Photonic Packet Buffers and Their Applications in Photonic Packet Switching Systems," *J. Lightw. Technol.*, vol. 16, no. 10, Oct. 1998, pp. 1737-1745.

[33] V. Eramo, M. Listanti, and A. Germoni, "Cost Evaluation of Optical Packet Switches Equipped With Limited-Range and Full-Range Converters for Contention Resolution," *J. Lightw. Technol.*, vol. 26, no. 4, Feb. 2008, pp. 390-407.

[34] K. Obermann, S. Kindt, D. Breuer, and K. Petermann, "Performance Analysis of Wavelength Converters Based on Cross-Gain Modulation in Semiconductor-Optical Amplifiers," *J. Lightw. Technol.*, vol. 16, no. 1, Jan. 1998, pp. 78-85.

[35] B. Sarker, T. Yoshino, and S. Majumder, "All-Optical Wavelength Conversion Based on Cross-Phase Modulation (XPM) in a Single-Mode Fiber and a Mach-Zehnder Interferometer," *IEEE Photon. Technol. Lett.*, vol. 14, no. 3, March 2002, pp. 340-342.

[36] D. Hsu, S. Lee, P. Gong, Y. Lin, S. Lee, and M. Yuang, "High-Efficiency Wide-Band SOA-based Wavelength Converters by using Dual-Pumped Four-Wave Mixing and an Assist Beam," *IEEE Photon. Technol. Lett.*, vol. 16, no. 8, Aug. 2004, pp. 1903-1905.

[37] V. Eramo, M. Listanti, and M. Spaziani, "Resources Sharing in Optical Packet Switches With Limited-Range Wavelength Converters," *J. Lightw. Technol.*, vol. 23, no. 2, Feb. 2005, pp. 671-687.

[38] F. Yan, W. Hu, W. Sun, W. Guo, Y. Jin, H. He, and Y. Dong, "Placements of Shared Wavelength Converter Groups Inside a Cost-Effective Permuted Clos Network," *IEEE Photon. Technol. Lett.*, vol. 19, no. 13, July 2007, pp. 981-983.

[39] V. Eramo and M. Listanti, "Pakcet Loss in a Bufferless Optical WDM Switch Employing Shared Tunable Wavelength Converters," *J. Lightw. Technol.*, vol. 18, no. 12, Dec. 2000, pp. 1818-1833.

[40] V. Eramo, M. Listanti, and M. Donato, "Performance Evaluation of a Bufferless Optical Packet Switch With Limited-Range Wavelength Converters," *IEEE Photon. Technol. Lett.*, vol. 16, no. 2, Feb. 2004, pp. 644-646.

[41] G. shen, S. Bose, T. Cheng, C. Lu, and T. Chai, "Performance Study on a WDM Packet Switch With Limited-Range Wavelength Converters," *IEEE Comm. Lett.*, vol. 5, no. 10, Oct. 2001, pp. 432-434.

[42] H. Li and I. Thng, "Cost-Saving Two-Layer Wavelength Conversion in Optical Switching Network," *J. Lightw. Technol.*, vol. 24, no. 2, Feb. 2006, pp. 705-712.

[43] J. P. Mack, H. N. Poulsen, and D. J. Blumenthal, "Variable Length Optical Packet Synchronizer," *IEEE Photon. Technol. Lett.*, vol. 20, no. 14, July 2008, pp. 1252-1254.

[44] W. Feng and W. D. Zhong, "Noise Analysis of Photonics Packet Synchronizer," *J. Lightw. Technol.*, vol. 22, no. 2, Feb. 2004, pp. 3431-350.

[45] Y. Lin, M. Yuang, S. Lee, and W. Way, "Using Superimposed ASK Label in a 10-Gb/s Multihop All-Optical Label Swapping System," *J. Lightw. Technol.*, vol. 22, no. 2, Feb. 2004, pp. 351-361.

[46] Z. Zhang and Y. Yang, "Optical Scheduling in Buffered WDM Interconnects with Limited Range Wavelength Conversion Capability," *IEEE Trans. Computers*, vol. 55, no. 1, Jan. 2006, pp. 71-82.

[47] Z. Hass, "The Staggering Switch: An Electronically Controlled Optical Packet Switch," *J. Lightw. Technol.*, vol. 11, no. 6, June 1993, pp. 925-936.

[48] O. Ladouceur, B. Small, and K. Bergman, "Physical Layer Scalability of WDM Optical Packet Interconnection Networks," *J. Lightw. Technol.*, vol. 24, no. 1, Jan. 2006, pp. 262-270.

[49] J. Yu and P. Jeppesen, "Improvement of Cascaded Semiconductor Optical Amplifier Gates by Using Holding Light Injection," *J. Lightw. Technol.*, vol. 19, no. 5, May 2001, pp. 614-623.

[50] B. Skubic, J. Chen, J. Ahmed, L. Wosinska, and B. Mukherjee, "A Comparison of Dynamic Bandwidth Allocation for EPON, GPON, and Next-Generation TDM PON," *IEEE Comm. Magazine*, vol. 47, no. 3, March 2009, pp. S40-S48.

[51] M. Mcgarry, M. Reisslein, and M. Maier, "Ethernet Passive Optical Network Architectures and Dynamic Bandwidth Allocation Algorithms," *IEEE Comm. Surveys & Tutorials*, vol. 10, no. 3, Aug. 2008, pp. 46-60.

[52] A. Dhaini, C. Assi, M. Maier, and A. Shami, "Dynamic Wavelength and Bandwidth Allocation in Hybrid TDM/WDM EPON Networks," *J. Lightw. Technol.*, vol. 25, no. 1, Jan. 2007, pp. 277-286.

[53] S. Sherif, A. Hadjiantonis, G. Ellinas, C. Assi, and M. Ali, "A Novel Decentralizewd Ethernet-Based PON Access Architecture for Provisioning Differentated QoS," *J. Lightw. Technol.*, vol. 22, no. 11, Nov. 2004, pp. 2483-2497.

[54] M. Yuang, S. Lin, S. Lee, J. Shih, Y. Lin, C. Hsu, P. Tien, and J. Chen, "Demonstration of a Novel PON System with Distributed Real-Time Bandwidth Allocation," *IEEE/OAC OFC 2009*.

[55] G. Karmer, B. Mukherjee, and G. Pesavento, "IPACT a Dynamic Protocol for an Ethernet PON (EPON)," *IEEE Comm. Magazine*, vol. 40, no. 5, Aug. 2002, pp. 74-80.

[56] B. Chen, J. Chen, and S. He, "Efficient and Fine Scheduling Algorithm for Bandwidth Allocation in Ethernet Passive Optical Networks," *IEEE J. Select.*

Topics in Quantum Elect., vol. 12, no. 4, July/Aug. 2006, pp. 653-660.

[57] P. Choudhury and P. Saengudomlert, "Efficient Queue Based Dynamic Bandwidth Allocation Scheme for Ethernet PONs," *IEEE GLOBECOM*, 2007.

[58] A. Demers, S. Keshav, and S. Shenkar, "Analysis and simulation of a fair queueing algorithm," *Internet. Res. and Exper.*, vol. 1, 1990.

[59] A. Parekh and R. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case," *IEEE/ACM Trans. Networking*, vol. 1, no. 3, June 1993, pp. 344-357.

[60] J. Bennett and H. Zhang, "WF^2Q: Worst-case Fair Weighted Fair Queueing," *IEEE INFCOM*, 1996.

[61] S. Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," *IEEE INFCOM*, 1994.

[62] P. Goyal, H. Vin, and H. Cheng, "Start-Time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, Oct. 1997, pp. 690-704.

[63] D. Stiliadis and A. Varma, "Efficient Fair Queueing Algorithms for Packet-Switched Networks," *IEEE/ACM Trans. Networking*, vol. 6, no. 2, April 1998, pp. 175-185.

[64] D. Stiliadis and A. Varma, "Rate-Proportional Servers: A Design Methodology for Fair Queueing Algorithms," *IEEE/ACM Trans. Networking*, vol. 6, no. 2, April 1998, pp. 164-174.

Biography

Shih-Hsuan Lin received the B.S. degree from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, in 2002, and the M.S. degree from the Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, in 2004. He is currently working toward the Ph.D. degree in the Department of Computer Science, National Chiao Tung University, Hsinchu. Taiwan. His research interests include optical networks, wireless data networks, performance modeling and analysis, and optimization theory.