Bandwidth Allocation and Scheduling in Photonic Networks

Nahid Saberi

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

February 2007

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-32236-9
Our file Notre référence
ISBN: 978-0-494-32236-9

Abstract

This thesis describes a framework for bandwidth allocation and scheduling in the Agile All-Photonic Network (AAPN). This framework is also applicable to any single-hop communication network with significant signalling delay (such as satellite-TDMA systems). Slot-by-slot scheduling approaches do not provide adequate performance for wide-area networks, so we focus on frame-based scheduling. We propose three novel fixed-length frame scheduling algorithms (Minimum Cost Search, Fair Matching and Minimum Rejection) and a feedback control system for stabilization.

MCS is a greedy algorithm, which allocates time-slots sequentially using a cost function. This function is defined such that the time-slots with higher blocking probability are assigned first. MCS does not guarantee 100% throughput, thought it has a low blocking percentage. Our optimum scheduling approach is based on modifying the demand matrix such that the network resources are fully utilized, while the requests are optimally served. The Fair Matching Algorithm (FMA) uses the weighted max-min fairness criterion to achieve a fair share of resources amongst the connections in the network. When rejection is inevitable, FMA selects rejections such that the maximum percentage rejection experienced in the network is minimized. In another approach we formulate the rejection task as an optimization problem and propose the Minimum Rejection Algorithm (MRA), which minimizes total rejection. The minimum rejection problem is a special case of maximum flow problem. Due to the complexity of the algorithms that solve the max-flow problem we propose a heuristic algorithm with lower complexity.

Scheduling in wide-area networks must be based on predictions of traffic demand and the resultant errors can lead to instability and unfairness. We design a feedback control system based on Smith's principle, which removes the destabilizing delays from the feedback loop by using a "loop cancelation" technique. The feedback control system we propose reduces the effect of prediction errors, increasing the speed of the response to sudden changes in traffic arrival rates and improving the fairness in the network through equalization of queue-lengths.

Résumé

Cette thèse présente une méthode permettant l'attribution de la bande passante et la synchronisation d'un réseau de communication de type réseau agile tout-photonique (AAPN). Cette méthode est également applicable aux réseaux de communication à simple saut comportant des délais de transmission importants (tels les systèmes satellites TDMA). Etant donné que les approches de synchronisation tranche de temps par tranche de temps ne fournissent pas des performances adéquates pour les grands réseaux, nous nous intéressons ici à la synchronisation à partir des trames. Nous proposons trois nouveaux algorithmes de synchronisation se fondant sur des trames à longueur fixe (recherche par moindre coût: MCS, mise en correspondance équitable: FMA, et rejet minimal: MRA) et un système de contrôle en boucle fermée pour garantir la stabilité du système.

L'algorithme MCS, gourmand en ressources, alloue la durée des tranches de temps à l'aide d'une fonction de coût. Cette fonction est définie de façon à assigner en premier les tranches de temps dont la probabilité de blocage est élevée. L'algorithme MCS ne garantit pas 100% de débit car il comporte un faible pourcentage de risque de blocage. Notre approche de synchronisation optimale est fondée sur la modification de la matrice de demande de façon à ce que les ressources du réseau soient pleinement utilisées, tout en servant optimalement les requêtes. L'algorithme de mise en correspondance équitable (FMA) utilise le critère de ressemblance pondérée maximisée pour obtenir un partage équitable des ressources parmi les connexions du réseau. Lorsque le rejet est inévitable, l'algorithme FMA sélectionne les rejets de manière à minimiser le pourcentage maximum de rejet obtenu dans le réseau. Dans une autre approche, nous formulons la tâche de rejet comme un problème d'optimisation en appliquant l'algorithme de rejet minimal (MRA) qui minimise le rejet total. Le problème de rejet minimal est le cas particulier d'un problème de flot maximal. Du fait de la complexité des algorithmes existants qui traitent le problème de flot maximal, nous proposons ici un algorithme heuristique à plus faible complexité.

La synchronisation dans les grands réseaux doit être fondée sur une prédiction du trafic. Néanmoins, les erreurs qui en résultent peuvent conduire à l'instabilité et à la non-équité. C'est pourquoi nous avons développé un système de contrôle en boucle fermée fondé sur un prédicteur de Smith afin de compenser les délais déstabilisants de la boucle de retour, ceci en utilisant une technique de boucle d'annulation. Le système de contrôle par retour réduit l'effet des erreurs de prédiction, augmente la vitesse de la réponse au changement soudain du débit du trafic et améliore l'équité dans le réseau en nivelant les longueurs des queues d'attente.

Acknowledgments

For his continued guidance, support and encouragement throughout my PhD, I would like to thank my adviser Professor Mark Coates. I greatly appreciate the respect and the freedom in research that you have given me. This is an opportunity for me to thank you for the time you spent discussing problems with me, suggesting proofs and solutions, and writing numerous letters during my stay in the US.

I am grateful to Professor Richard Vickers and Professor Lorne Mason for discussions about photonic networks and bandwidth sharing methods during the first two years of my PhD, and also for their thoughtful reviews of my thesis and very helpful suggestions.

Thanks to all my colleagues at the computer networks research lab and photonic systems group at McGill University. I thank Dr. Anton Vinokurov for donating his AAPN model implemented in OPNET which was used in many of my simulations. I wish to acknowledge implementation of frame-based scheduling framework in OPNET by Xiao Liu, which was the result of her hard work for several weeks. I also wish to acknowledge the flow-based traffic prediction method proposed by Tarem Ahmed that was used in some of my experiments. I am thankful to Madeleine Mony for being kind enough to answer my numerous questions regarding photonic switches.

I would like to thank Dr. Yvan Pointurier for his help in correcting the French language translation of my abstract and for reviewing my thesis and giving valuable feedbacks.

My sincere gratitude goes to all my old friends from IUT who have always been ready to help and discus problems with me. I would like to thank Fariba and Zohreh for always being supportive friends and for helping me in submitting my thesis. I am thankful to Farzaneh and Maryam for editing Chapter 5 of my thesis.

Words cannot express my thanks to my parents. I can only say thank you for your love, encouragement, and all of those unforgettable moments and the unique happiness you gave to me.

I would like to thank my sisters for their company, concern, and generously sharing their experiences with me throughout my studies.

I would like to thank my husband Mohsen for his patience, unfailing support and belief in me. All of these years you have been helping me in your very own way.

And above all, thanks to God for everything. I thank You for creating space and time, and giving me the opportunity to exist. Thank You for making me understand that this is all about You.

Contents

List of Figures

List of Acronyms

AAPN	Agile All-Photonic Network
AWG	Array Waveguide Grating
BALSAM	BALanced Scheduling AlgorithM
BFS	Best Fit Search
CG-SOA	Clamped Gain Semiconductor Optical Amplifiers
ESA	Equal Share Matching Algorithm
FMA	Fair Matching Algorithm
IBS	Interval-Based Scheduling Algorithm
IBT	In-Band-Terminator
IWS	Individual wavelength switching
JET	Just-Enough-Time
LIST	List scheduling
MCBM	Maximum Cardinality Bipartite Matching
MCS	Minimum Cost Search
MINSWT	Minimum Switching Time
MMFT	Modified Multi-FiT
MRA	Minimum Rejection Algorithm
NP	Nondeterministic polynomial-time
OBS	Optical Burst Switching
ODL	Optical Delay Lines
O/E/O	Optical-Electronic-Optical
OPS	Optical Packet Switching
OTDM	Optical Time Division Multiplexing
OTSI	Optical Time Slot Interchangers
PI	Proportional Integral
PIM	Parallel Iterative Matching

PSR	Photonic Slot Routing
RFD	Reserve-a-Fixed-Duration
SEQSAM	SEQuential Scheduling AlgorithM
SOA	Semiconductor Optical Amplifiers
SS	Sequential Search
TAG	Tell-And-Go
TDM	Time Division Multiplexing
TOF	Tunable Optical Filters
TSOBS	Time Sliced Optical Burst Switching
TWC	Tunable Wavelength Convertors
WDM	Wavelength Division Multiplexing

Chapter 1

Introduction

Optical fiber has been used as the physical medium for high rate data transmission since the late 1960s. Following the development of SONET and SDH standards in the mid-1980s it has been extensively used for long-haul transmissions of telephone signals. However the high cost of optical devices such as transmitters, receivers, and switches has hindered fiber optics for short-haul transmission for over two decades [10].

With the growth of the demand for higher speeds and the development of optical devices, optical fibers will soon replace the traditional copper wires in access networks. This in turn will require the future technology to overcome the problem of bottleneck in the backbone networks, which is the speed of electronics in optical/electronic switches. With the advent of fast all-optical switches that are reconfigurable on the order of sub-microseconds, all-optical switching technologies for core network operations are becoming feasible. All-optical cross-connect switches are naturally protocol-independent and bit-rate independent, thus they are able to carry services in their native format, the property that is referred to as *transparency.*

Early applications of optical fibers modulated data onto a single optical carrier frequency that is commonly referred to as a wavelength. The amount of data transferred was therefore limited by the speed of the electronics that generated the original signal, thereby wasting the tens of THz of optical capacity. Wavelength division multiplexing (WDM) now allows multiplexing several carriers on the same fiber, utilizing the available bandwidth. Achieving packet switching in all-photonic networks requires switching speeds in the order of nanoseconds. Although optical switching in the order of picoseconds is available, the lack of optical buffering and very limited optical processing abilities prohibit the use of optical packet switching in today's transmissions. Instead the use of bandwidth reservation methods and combining packets of variable length in optical time slots of fixed length provide an acceptable granularity as well as technological simplicity. In addition, this enables time division multiplexing (TDM) of slots on every wavelength for more efficient use of bandwidth by sharing each wavelength by several traffic flows.

1.1 Agile All-Photonic Network

Transparency of all-optical switches offers simplicity and lower cost to the network as it eliminates the use of wavelength convertors and O/E/O conversion. However, bandwidth provisioning is a more challenging issue in all-photonic networks since the transparent optical switches do not change the format (e.g. wavelength) of the packets arriving at their ports to resolve the contention to some extent. In addition, the photonic switches do not perform queueing, so there is a need for additional control functionality to reduce or eliminate the potential of contention for egress ports. Burst switching and just-in-time reservation approaches [11], and routing and wavelength assignment techniques [12], are some of the many approaches that have been used in general mesh topologies. An alternative approach is to focus on a simpler architecture that reduces the complexity of the control challenge. In this research, we focus on the overlaid star topology, as specified in the design for the agile all-photonic network (AAPN) architecture of [1,2,13]. This design overcomes some of the difficulties of the all-photonic networks by introducing agility to the network. The term "agility" refers to the ability of all-photonic networks to adapt to the variations of the traffic flows through fast reconfigurable optical switches [1].

The star topology makes accurate synchronization much more feasible [14], and this enables the application of a range of Optical Time Division Multiplexing (OTDM) techniques for sharing link and switch capacity. A source edge-node must be aware of when it has ownership of a given time-slot and is allowed to transmit to a specific destination edge node. By suitably allowing for the differing propagation delays between various edge nodes and the core, time slots arrive at the core crossbar switch at the same time and can be switched to their appropriate destinations without output port collisions.

The design of efficient scheduling methods for all photonic switched networks is challenging because no effective optical buffer devices exist. Once an optical signal is launched into the network, the arrival time at junction points or switches is determined exclusively by the length of the fiber link and the signal's propagation speed. For synchronous time slot switching, the slots arriving on the input ports of the optical space switch must be phase aligned and separated by a guard time sufficient for switch reconfiguration, in order for the slots to traverse the switch without corruption. Phase alignment can be accomplished for star network topologies as well as more general tree network topologies by buffering the inbound traffic in electronic buffers at the edge nodes, and launching the signals at the appropriate offset time in order that all slots arrive in phase at the photonic switch. The underlying assumption is that the core switch and all edge nodes of the star are synchronized relative to a single clock. In general slot phase alignment is much more difficult and often impossible to achieve in general network topologies such as mesh networks. Keslassy et al. [14] have employed delay graph models of networks to examine the class of topologies which admit efficient scheduling methods. While star and tree network topologies have the desired delay graph properties to allow efficient link utilization in all cases, general mesh networks do not.

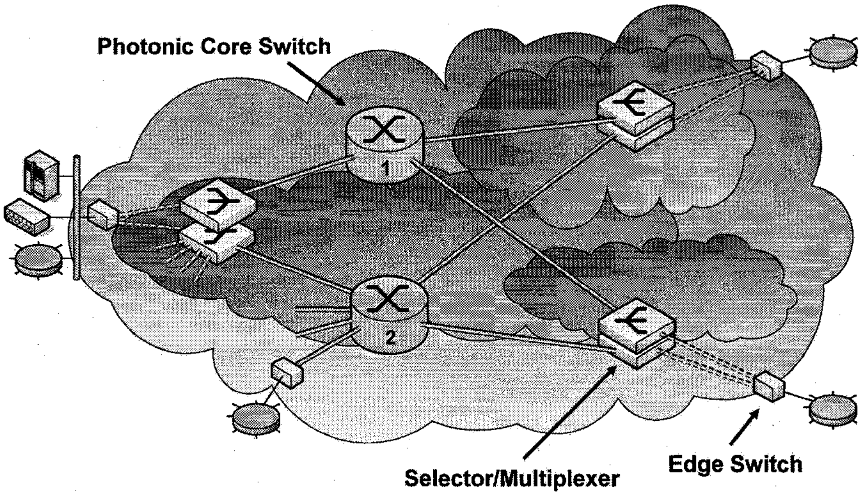


Fig. 1.1 Architecture of the Agile All-Photonic Network described in [1,2]. Edge nodes perform electronic-to-optical conversion and transmit scheduling requests to the photonic core node(s). Selector/multiplexor devices are used to merge traffic from multiple sources onto single fibers and to extract traffic targeted to a specific destination. The structure forms an overlaid star topology (see Figure 1.2).

The AAPN network architecture (see Figure 1.2) consists of edge nodes, where the optical electronic conversion takes place, connected via selector/multiplexor devices to photonic core crossbar switches, which act independently of one another. It permits each node to transmit to one destination node and receive from one source node simultaneously *on each wavelength.* Each edge node constructs a separate queue for the traffic destined to each

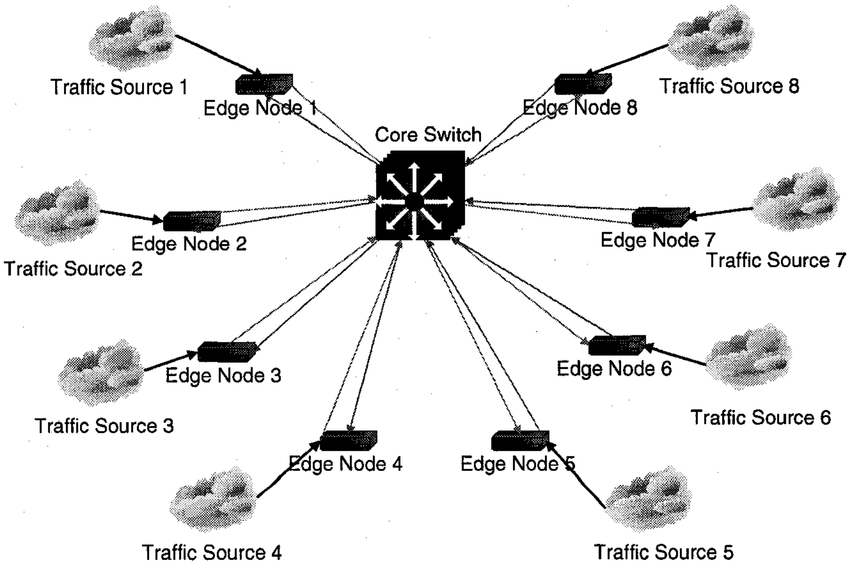


Fig. 1.2 The star topology induced by the agile all-photonic network architecture.

of the other edge nodes referred to as a virtual output queue (VOQ). Traffic aggregation is performed in these queues. Packets are collected together and then transmitted as optical packets across the network. The core node is reconfigured in response to traffic load variations reported by the edge nodes. The core switches act independently, so the control problem becomes one of scheduling the switch configurations to achieve a good match with the traffic arrival pattern at the edge nodes.

The slot allocation can be fixed and deterministic, or it can adapt to the traffic arrivals through signalling between the edge nodes and the core switch. In the latter case, adaptation can be performed on a per frame basis (a block of slots) or per time-slot basis. Frame-based scheduling is more appropriate for wide-area networks since the impact of propagation delay is reduced (bandwidth is reserved for predicted traffic demand in advance of the traffic arrivals) [15]. We focus on fixed-length frames, because this simplifies protocol design and implementation of control functions.

The general objectives of bandwidth sharing are to achieve minimum loss (rejected requests) or maximum throughput, and minimum end-to-end delay, whilst maintaining fairness in the network. Minimizing the number of rejected requests (those not accommodated by the scheduled frame) has highest priority. A secondary objective is to minimize the number of switching operations in a frame in order to reduce the power consumed by the core switch.

1.2 State-of-the-Art

Scheduling in AAPN is similar to scheduling of an input queued switch with the difference that there is a large propagation delay between the input buffers (at the egress edge nodes) and the switch (at the optical core). The major problem with input queued switches is head of line (HOL) blocking which limits the throughput of the switch to 58.6% of the maximum capacity [16]. HOL blocking can be eliminated using virtual output queueing (VOQ) by implementing a separate queue for each output at the input buffers [17–21]. It has been shown that with a suitable centralized scheduling algorithm VOQ can emulate the performance of output queueing (100% throughput) [22]. Maximum size bipartite algorithms perform well when the arriving traffic is uniformly distributed over all the switch outputs, but perform poorly when traffic is non-uniform [19].

McKeown et al. have proposed a number of slot-based solutions for achieving 100% throughput in input queued switches [19, 20, 22, 23]. Longest queue first (LQF) and oldest cell first (OCF) achieve 100% throughput for all independent arrival processes [20]. LQF gives priority to long queues by using a maximum weight matching algorithm, where each weight is set to the corresponding queue length. But LQF is very difficult to implement in hardware at high speeds and has a very high time complexity. OCF on the other hand gives priority to the queues bearing packets with larger waiting times. The time complexity of these two algorithms is $O(N^3 \log N)$. The longest port first (LPF) algorithm proposed by the same authors [22] combines the benefit of a maximum size matching algorithm (acceptable complexity) with that of a maximum weight matching algorithm (high throughput). LPF finds the set of maximum size matchings and chooses the match with the largest weight, where the weights are functions of queue lengths. It has been shown that LPF achieves 100% throughput for both uniform and non-uniform traffic. LPF has a time complexity of $O(N^{2.5})$. Furthermore, there are a variety of heuristic approximations to perform maximum size matching algorithm [17, 18, 21].

In [15] we have shown that scheduling AAPN with large propagation delay should be performed using a frame-based algorithm rather than using a slot-based algorithm. The frame-based algorithms can be implemented using any of the slot-based algorithms used for input queued switches. However depending on the type of the frame (fixed or variable-length) different modifications should be applied. Most of the frame-based scheduling algorithms have a linear worst-case time complexity in the frame length. Since frame-based algorithms update the schedule once during a frame, they do not need to be as fast as slot-based algorithms. Bianco et al. [24] proposed a frame-based scheduling algorithm for a network of input-queued switches. The approach is based on a differential frame-based matching scheme, which combines the frame-based advantages and the features of differential schemes. In differential schemes the estimated traffic is used to modify the matchings determined in the previous frame, thereby reducing the algorithmic complexity. This scheme is applicable to a scenario where traffic rates do not vary dramatically within a frame duration. This is a reasonable assumption especially for core switches aggregating thousands of flows. Using a cost function the rate of every flow is estimated. Rate estimation is inspired by the concepts of the game theory. The game theory framework is used to prove the properties of the scheduling algorithm proposed in this article. It has been proved that this scheduling policy achieves 100% throughput under a large class of input traffic patterns. In this approach the extra bandwidth is evenly shared among competing flows.

Scheduling in networks with large propagation delays is usually based on estimating the traffic arrivals. Scheduling in these networks such as AAPN is modeled as an open-loop system, which does not compensate for the errors of traffic prediction entered into the system. Consequently designing a feedback control system is desirable to avoid system's instability and unfairness as the results of prediction error. Mascolo [25] combines classical control theory and Smith's principle to design a simple congestion control law that guarantees no packet loss and efficient use of bandwidth . The use of Smith's principle, which alleviates the stability difficulties of control systems with large delays, makes Mascolo's design applicable to network paths with a wide range of propagation delays. The control system proposed by Mascolo exploits an approach for controlling the transmission rates of the best effort traffic in the networks. The problem we wish to address differs significantly. We assume that we have no control over arrival rates; instead we can adjust, through scheduling, the resources allocated within the network. This results in an inverted version of the standard congestion control problem: switch resources are controlled rather than source rates.

1.3 Thesis Contribution

We investigate the problem of bandwidth allocation and scheduling in single-hop all-photonic networks with cross-connect switches and large propagation delays. The scheduling problem has been investigated in depth for the past forty years, so there are naturally many approaches to generating the schedule once the demand matrix is available. The majority of frame-based scheduling algorithms have focused on *variable-length* frames (for example, see [26–28] and the references therein). There has been some literature on designing schedules with fixed lengths, but there has been no discussion of the NP-hardness of an optimum solution for this problem. The authors of [29–32] have considered the problem of scheduling a frame of fixed length for star-coupled networks with tunable transmitters/receivers, but do not address the problem of utilizing the frame capacity in case of admissible traffic or the problem of rejecting some of the requests based on an appropriate criterion when the traffic is inadmissible. Algorithms designed for the variable-length frame scheduling problem can be applied to the case of fixed-length frames, but there must be rate adjustment when demand is low or inadmissible. When the predicted demand is insufficient to fill the schedule completely, we need a policy to divide the extra time slots amongst active connections (these extra slots alleviate the effect of potential underestimates in the demand matrix, which is merely a prediction of future traffic arrivals). On the other hand, when the demand is inadmissible, some of the predicted demand must be rejected. The choice of which requests to reject depends on whether the goal is to minimize the total amount of rejection or to achieve some form of fair rejection.

The focus of this thesis is to explore the fixed-length frame scheduling problem and its complexity, and propose optimal solutions with respect to the objectives that we follow in all-photonic networks bandwidth allocation. The main contributions of this research are:

- Formulation of the fixed-length frame scheduling as an optimization problem and investigate the NP-hardness of this problem.

- Proposal of a new scheduling algorithm called the *Fair Matching Algorithm (FMA)* based on the weighted max-min fairness criterion, and examination of its properties.

- Development of the Minimum Rejection Algorithm (MRA) which minimizes the number of generated rejections while scheduling a given inadmissible traffic matrix.

- Comparison of the proposed algorithms with other scheduling techniques, and establishment of general rules for the scheduling problem in photonic networks [15, 33].

- Design of a feedback control system for the networks comprising fixed-length frame schedules to compensate for traffic prediction and scheduling errors as well as to increase the speed of response to the variations of traffic.

1.4 Thesis Organization

An extensive literature survey of several topics related to bandwidth reservation in WDM networks is presented in Chapter 2. In Chapter 3 we formulate the fixed-length frame scheduling problem as an optimization problem and investigate its NP-hardness and propose approximate solutions for this problem. The chapter begins with a review on the variable-length frame scheduling problem and its Open Shop formulation. Then we extend the problem to fixed-length frame scheduling. Chapter 4 introduces our novel algorithms for the scheduling problem in the AAPN network with an emphasis on fairness and minimizing rejection. In Chapter 5 we describe an approach for designing a feedback control system to address the problem of prediction and algorithm errors in fixed-length frame scheduling and to increase the speed of reaction to traffic variations in wide area networks. This chapter begins with a review of Smith's principle and the control systems that are based on it. We then present a novel adaptation of the Smith controller to the AAPN network and our scheduling algorithm. Finally, Chapter 6 provides a concluding discussion and topics for future work.

1.5 Published Work

The content presented in this thesis has been partly published in the following articles. Parts of the literature review presented in Chapter 2 and some part of Chapter 3 have been published as technical reports. The results presented in Section 4.2.1-A were published in IEEE International Conference on Information, Communication and Signal Processing (ICICS) 2005, Bangkok, Thailand. Xiao Liu and Professor Lorne Mason presented a slot-based algorithm for the Agile All-Photonic Network. Xiao Liu also implemented a framework for simulating frame-based scheduling with the OPNET simulator, using which we implemented our frame-based scheduling approach. In the ICICS paper we compared the performance of our frame-based scheduling algorithm with that of the slot-based scheduling algorithm.

- N. Saberi and M.J. Coates, Minimum rejection scheduling in all-photonic networks, in Proc. IEEE BROADNETS, San Jose, CA, Oct. 2006.
- N. Saberi and M.J. Coates, Fair matching algorithm: fixed-length frame scheduling in all-photonic networks, in Proc. IASTED Int. Conf. Optical Comm. Sys. and Networks, Banff, AB, Canada, July 2006.

- N. Saberi and M. J. Coates, Bandwidth reservation in Optical WDM/TDM star networks, in Proc. 22nd Biennial Symposium on Communications, Kingston, ON, Canada, June 2004.

- X. Liu, N. Saberi, M.J. Coates and L.G. Mason, A comparison between time slot scheduling approaches for all-photonic networks, in Proc. IEEE Int. Conf. on Information, Comm. and Signal Proc. (ICICS), Bangkok, Thailand, Dec. 2005.

- N. Saberi and M.J. Coates, Scheduling and Control in Wide-Area All-Photonic Networks, submitted for review.

- N. Saberi and M.J. Coates, WDM Bandwidth Allocation, AAPN Tech. Report, Department of Electrical and Computer Engineering, McGill University, Montreal, Canada, June 2004.

- N. Saberi and M.J. Coates, Fair matching algorithm: an optimal scheduling algorithm for the AAPN, Technical Report, Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada, Sept. 2005.

Chapter 2

Photonic Networks

Traditional communication systems send and receive information through copper wires, coaxial cables or air. With the increasing demand for higher communication speeds, the traditional communication media are not able to provide sufficient capacity. The use of optical fibers permits data transmission up to several Tbps (theoretically, a single mode optical fiber can support a data rate of 50 Tbps [34]). If a transmission network uses optical fibers as its transmission media, it is called an *photonic network* [35]. However, it is difficult to fully utilize the capacity of optical fibers for two reasons. First, the speed of electronic hardware is much lower than the capacity of optical fibers, and hence the speed of electronic switching limits the rate of optical transmission. Second, optical transmitters/receivers operate at lower speeds than several Tbps [35]. To overcome these difficulties, wavelength division multiplexing (WDM) was proposed. Partitioning the enormous bandwidth in photonic networks into WDM channels makes the optical bandwidth compatible with the speed of electronic components. In WDM systems the end users communicate via optical channels called lightpaths. In networks with a *continuity constraint* a lightpath is required to occupy the same wavelength on all fiber links, while in *wavelength convertible* networks a lightpath can occupy different wavelengths on different fiber links along the path. Wavelength conversion allows networks to support more lightpaths, but it is expensive and difficult to implement [36]. Therefore in this thesis we mainly consider networks with a continuity constraint except where otherwise indicated.

In these networks, resource allocation is a key problem which is addressed in many different contexts, including routing and wavelength assignment (RWA), optical burst/packet switching, photonic slot routing, TDM/WDM assignment, and broadcast-and-select technology.

Optical backbone networks multiplex a large amount of traffic coming from numerous users on circuit-switched wavelength paths. This technology (the so-called wavelength routing approach) has been widely studied in the literature [36]. In contrast, wavelength routing in access and metro networks, which have a reduced level of traffic aggregation, is not an adequate solution [37]. In these domains, other approaches such as optical burst/packet switching and time-division multiplexing over WDM channels provide a more dynamic bandwidth allocation. Based on TDM/WDM technology several different approaches have been developed such as photonic slot routing and TDM/WDM bandwidth reservation in broadcast and select networks.

In this chapter we survey the background literature and recent developments in WDM resource allocation techniques. Section 2.1 introduces routing and wavelength assignment in WDM networks. Section 2.2 presents an overview of recent optical switching deployments. Section 2.3 provides a background of bandwidth reservation in WDM networks with point-to-point and broadcast-and-select transmissions.

2.1 Routing and Wavelength Assignment Schemes

The problem of assigning wavelengths and paths to a set of requests for bandwidth between source-destination pairs in WDM networks is referred to as the routing and wavelength assignment (RWA) problem. It has been proved that the RWA problem is *NP*[1]-complete[2] [39], and partitioning this problem into two subproblems, (i) routing, and (ii) wavelength assignment, makes it more tractable. Numerous solutions for each of the problems have been proposed; these can be classified under static and dynamic approaches. Depending on the traffic pattern the problem can be formulated differently. When the traffic pattern is static, the requests are fixed for a long duration of time as the connections are long-lived. Therefore, the entire demand is known in advance (off-line information). Once the connections are established there is no need for further operations. In a dynamic traffic pattern the requests arrive on-line and may depart after a while. Therefore the RWA problem has to consider the dynamics of the related parameters for assigning paths and wavelengths to such traffic. For the routing subproblem there are three basic approaches

[1] Nondeterministic polynomial-time problem; a problem is said to be *NP* if there exists a nondeterministic polynomial-time algorithm that recognizes the elements of this problem (i.e. that we can test in polynomial-time whether a candidate solution is a solution).

[2] A problem in *NP* is said to be *NP*-complete if finding a deterministic polynomial-time algorithm for solving this problem allows us to solve any problem in *NP* in polynomial-time (for more information regarding the theory of *NP*-completeness refer to [38]).

known as fixed routing, fixed alternate routing and adaptive routing. For the wavelength assignment component many approaches have been proposed, including First-Fit, Least-Used, Most-Used, Min-product, Least loaded, Max-SUM and Wavelength Reservation. We now present a review of various routing and wavelength assignment approaches for both off-line and on-line cases; the review is partially derived from [36].

2.1.1 Routing Schemes

Routing algorithms can be classified in three basic groups:

Fixed Routing

Fixed routing, the simplest way of performing routing in a network, chooses a fixed path from a node to each destination. Fixed shortest-path routing is an instance of the fixed routing approach, which can be implemented by using standard shortest path algorithms, such as Dijkstra's algorithm or the Bellman-Ford algorithm. The major problem with this approach arises when the resources are insufficient to meet the demand, which leads to a high blocking probability [40].

Fixed-Alternate Routing

Fixed alternate routing constructs a list of options for routing a path between each source-destination pair. The algorithm may use different criteria for constructing and sorting the list, such as a shortest-path criterion or the load at each link. For example, the list may include the first, the second and the third shortest paths between each source-destination pair. This routing approach reduces the blocking probability compared to fixed routing. In addition, in the case of path failure, it can provide some degree of flexibility for rerouting the connections [41].

Adaptive Routing

Adaptive routing determines the available paths across a network and evaluates them based on the state of the network. It then chooses the one that will provide the best path for a connection. This approach works well for networks with dynamic states. Examples of adaptive routing include:

Adaptive Shortest-Cost-Path. This approach assigns a cost to each link in the network. The cost is determined based on the state of the link in the network. Each unused link is assigned a cost of 1 unit, each used link is assigned a cost of ∞, and each used link with a wavelength convertor is assigned a cost of c units, where the value of c is defined in such a way as to avoid the use of wavelength convertors as long as any direct path is available. If the wavelength convertor is full (i.e., it is occupied by the other lightpaths) $c = \infty$ is considered. When a connection request arrives, the algorithm calculates the cost for each possible path, and chooses the one with the lowest cost. A random selection is performed for breaking the ties [42].

Least-Congestion-Path. This method determines a list of paths for each source-destination pair. When a connection request arrives, the path with the lowest load amongst the corresponding pre-determined paths is chosen [43]. In case of a tie, other algorithms such as the shortest path routing algorithm may be used. Since this algorithm considers all links on all pre-determined paths the computational complexity is high. For reducing the complexity of this method, Li et al. [44] proposed to only consider the first k links on each path (referred to as source neighborhood information) where k is defined appropriately.

2.1.2 Fault Tolerant Routing

A common approach to protect connections against link (node) failure is to consider at least two link-disjoint (node-disjoint) paths for each source-destination pair. In case of failure the backup path can be used. Fixed alternate routing directly provides some reservations for each connection. In the case of adaptive routing, backups should be identified by a protection scheme, which determines the alternate routes immediately after the primary connections have been established. The protection scheme can be the same as routing scheme except that the cost of ∞ is assigned to undesirable links to promote link-disjoint (node-disjoint) paths [7].

2.1.3 Wavelength Assignment

Wavelength assignment is the second sub-problem within the RWA problem, and is usually addressed as an independent problem. The main objective is to assign a wavelength to each connection in an efficient way such that no two lightpaths on a link share a common wavelength. Wavelength assignment can be resolved for either dynamic traffic or static traffic.

Static Wavelength Assignment

In static wavelength assignment the lightpaths and their routes are known in advance and we need to assign wavelengths to each lightpath, such that each of the lightpaths on a given link occupies a unique wavelength. This problem can be presented in two forms: (i) for a given number of wavelengths, maximize the number of connections which can be established, (ii) for a given topology (without wavelength convertors), minimize the number of wavelengths needed for a set of connection requests under the wavelength continuity constraint. The latter problem can be reduced to a sequential graph coloring problem [45].

Graph Coloring Problem. The problem of static wavelength assignment can be reduced to a graph coloring problem which is NP-complete; among the NP problems it is the least likely to be solved by any polynomial-time algorithm, because if there is any algorithm which solves this problem quickly, any problem in NP can be solved quickly by the same algorithm [38]. This proves that the wavelength assignment problem is NP-complete itself.

The graph coloring problem constructs a graph G of nodes, such that each node corresponds to a lightpath in the network. Each common fiber between two or more lightpaths in the actual system is represented by a link between their corresponding nodes in the graph. The next step is to color the nodes of the graph in such a way that no two adjacent nodes have the same color [45]. The minimum number of colors required by the process for coloring the graph G (the minimum number of wavelengths) is not easy to determine. Therefore heuristics such as the sequential graph-coloring algorithm have been introduced which are reasonably efficient in practice [46].

Dynamic Wavelength Assignment

Dynamic wavelength assignment considers the more realistic case where connection requests arrive dynamically. The connection requests are accepted if sufficient resources are available for the complete path and are blocked otherwise. Dynamic algorithms that consider a fixed number of wavelengths try to minimize the blocking probability. We now summarize some of the proposed heuristics, drawing from the review material in [36].

Random Wavelength Assignment. When a connection request arrives, a search procedure determines the available wavelengths on the appropriate path. One of the available wavelengths is randomly selected .

First-Fit(FF). The FF algorithm numbers all of the wavelengths in the network. When a connection request arrives, the first available wavelength is assigned to that connection. This approach has a very low computational complexity.

Most-Used(MU)/Pack. This method has been proposed for networks with fixed routes between each source destination pair [47]. The idea is to pack connections into a fewer number of wavelengths by selecting the most-used wavelength in the network. Packing connections is valuable in networks with the wavelength continuity constraint. In these networks there is no wavelength conversion and reserving the least-used wavelengths reduces the blocking probability in the network.

Least-Loaded(LL). LL has been designed as a wavelength assignment technique for multi-fiber networks [48]. For each connection request the most-loaded link of multi-fibers along the required path is examined by the algorithm to determine the least-loaded wavelength on this link. The least-loaded wavelength on a multi-fiber network is the one with the largest residual capacity. The residual capacity of each wavelength may vary between 0 and N, where N is the number of fibers on each link.

MAX-SUM(MΣ). This method also has been proposed for multi-fiber networks [47,49]. MΣ considers all the routes in the network that might be used by any of the connections. Then for a request the algorithm selects a wavelength on the selected path in such a way that the maximum available capacity on the remaining paths is obtained.

Wavelength Reservation(RSV). A wavelength reservation scheme reserves a specific wavelength on the links along a path of a multi-hop stream. This approach reduces the blocking probability for multi-hop streams, although the blocking probability for single-hop streams may increase.

2.2 Switching Schemes in WDM Systems

In addition to routing and wavelength assignment several different technologies have been developed for the transfer of optical data over WDM networks, such as optical packet switching, optical burst switching, and photonic slot routing. In this section we review these switching technologies.

2.2.1 Photonic Slot Routing

Photonic Slot Routing (PSR) is a time division multiplexing approach for all-optical access and metro networks. PSR attempts to reduce complexity by eliminating the use of individual wavelength switching (IWS) components. The time-shared nature of this approach provides a sufficient level of traffic aggregation in networks for which the wavelength routing solution is inefficient. We now examine a photonic slot routed network which has been designed by Zang et al. [3].

Photonic Slot Routing in All-Optical WDM Mesh Networks

In PSR, time is slotted into fixed spans, each comprising a photonic slot. A photonic slot includes all wavelengths in a network. The packets of data destined for the same node are loaded into the photonic slots and are sent as a single integrated unit. Therefore there is no need for individual wavelength routing along a path and wavelength insensitive components are adequate for routing the photonic slots. Eliminating the use of wavelength demultiplexers results in faster switching, less complexity and lower cost [3].

Network Architecture. In this design a mesh network of wavelength insensitive nodes (Figure 2.1) is considered. At the source end, each node considers a separate electronic buffer space for each destination. The photonic slots for each destination consist of several data packets on a number of wavelengths and a header on a different wavelength. At each intermediate node headers are extracted from slots. During the header processing period the data slots travel along delay lines. To avoid the need for delay lines a solution is to send the header of a slot a fixed period of time before the payload, long enough for processing the header. The headers contain information about the wavelengths being used by the slot and the destination addresses. When a header of a slot arrives at a node which has some packets headed for the same destination, the node may insert its packets to the free wavelengths of the arriving slot. Inserting the packets can be performed by using couplers.

When two or more arriving slots contend for the same output port several techniques may be used such as optical buffering, deflection routing, or dropping the slots randomly.

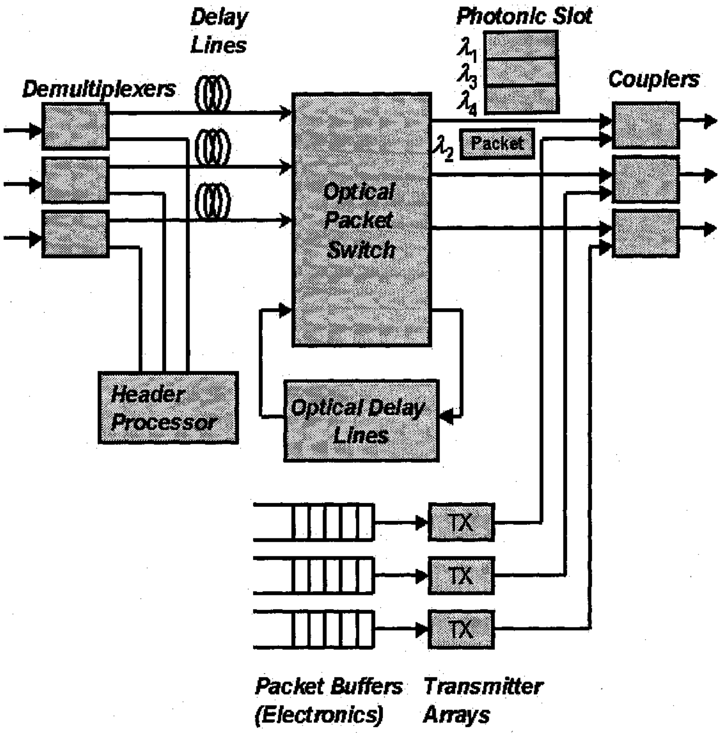


Fig. 2.1 Photonic Slot Routing (PSR) node architecture; each node is capable of buffering, header processing, and packet insertion. A packet on wavelength λ_2 is inserted into an arriving photonic slot which is free at λ_2 [3].

PSR Protocol. When a node has several packets to send, it may choose to add its packet to an arriving slot headed for the same destination or it may place the packets in an empty slot and sign the slot for its destination. There are a number of policies for slot assignment and adding packets to the existing slots. A slot may be occupied entirely by a node or may be left with some free spaces to be used by intermediate nodes. We now review two types of slot assignment policies.

1. Packet Arrival Based Assignment: Upon receiving an empty slot a node randomly chooses one of its queues and inserts a number of packets into the slot [3]. This policy implies that as long as there are empty slots or free space in arriving slots with the appropriate destination the node is allowed to insert its packets. However this approach results in unfairness in resource allocation as well as high probability of blocking. Nodes located in the internal regions of the network usually receive

assigned or full slots, while nodes located towards the edges of the network usually receive empty slots.

2. Capacity Allocation: A straightforward method for capacity allocation is the slot preassignment approach proposed by Chlamtac et al. [50]. In this approach a TDMA frame consisting of L slots is considered, in which a fixed number of slots is assigned to each source-destination pair. The number of slots for each pair is determined by using a network-wide TDMA schedule to achieve fairness and contention free slot routing at intermediate nodes. However this approach is impractical since the length of the frame is fixed and the number of slots per pair is always an integer. An alternative method [3] is to assign destinations probabilistically to arriving slots based on the capacity allocation results. This approach consists of two steps: the "Capacity-Allocation" step in which the fraction of the capacity of a link for each source-destination pair is determined, and the "Slot-Assignment" step in which a destination is assigned to each slot based on the results of the first step.

2.2.2 Optical Packet Switching

Optical packet switching (OPS) offers a better bandwidth granularity compared to the circuit switched networks, which results in a finer transmission, a bandwidth-efficient design and a more flexible all-photonic network. However, this technology faces a number of limitations, in that optical packet switching requires optical buffering and packet-level processing [11].

An optical packet network consists of interconnected optical packet switches which are usually composed of four parts: the input/output interfaces, switching fabric and control unit. The input interfaces align the packets, extract the header information from the packets and remove the headers. The control unit performs the control functions based on the header information. The switch fabric switches the packets optically based on the control information. The output interfaces are responsible for optical signal regeneration and header insertion [11]. Header processing must be performed in the electrical domain, while the payload remains in the optical domain. Therefore packets have to be stored in the switch (e.g., using Optical Delay Lines (ODLs)) to be forwarded to the next stage when the header processing is complete, referred to as store-and-forward nature of packet switching. However this process limits the speed of the network. A typical IP packet would pass through a switch every 1 nanosecond. Given that using traditional design techniques and current electronics, a packet will take about 100 nanoseconds to process using fast lookup techniques such as MPLS, and hence traditional electrical methods cannot keep up with the optical capacity.

A new generation of optical packet switched networks employ all-optical label swapping (AOLS) by using optical labels for routing the packets [51]. AOLS uses optical encapsulation to route packets independently of their length, bit-rate or format. Therefore, AOLS is not limited to IP packets, and the other forms of packets such as ATM cells can be directed using this approach. To illustrate the AOLS procedure, suppose that an IP packet enters the network through an "ingress" node (which transforms the electronic signals to optical signals). The ingress node determines an optical label and a wavelength by reading the packet's IP header and using information stored in a pre-established local lookup table. The packet is then encapsulated with the optical label and retransmitted on the determined wavelength.

As long as the packet is inside the all-optical network, only its optical label is used to make routing decisions. At the internal all-optical nodes, labels are read and optically erased, then a new label is attached to the packet and the optically labeled packet is converted to a new wavelength using all-optical wavelength conversion. During this process the IP packet header and payload are not passed through electronics until the packet exits the all-optical network through the "egress" node. At an egress node the optical label is removed and the original packet is passed to the electronic routing hardware.

Contention Resolution in Optical Packet Switching

Whenever two or more packets are destined to the same output port at the same wavelength, external blocking occurs. Optical packet switches are mostly non-blocking designs, therefore internal blocking is not an issue. To resolve contention in optical packet switches several approaches have been introduced including optical buffering, wavelength convertors, and deflection routing.

Optical Buffering. Optical delay lines are currently the only way of implementing optical buffers. Using electrical buffering is not acceptable, since electronic components cannot match optical speeds. Optical buffers can be implemented by several ODLs with variable lengths to provide different delay lines. A counter is used to keep track of the number of packets in the buffer. Using a separate counter for each ODL adds flexibility to the use of optical buffers and reduces the length of the delay lines (i.e., a packet may be circulated through several ODLs to achieve the desired delay). Implementing optical buffers needs an enormous amount of fiber and a complex electronic control. In addition, optical signals traveling through delay lines may experience a considerable power loss so that optical amplifiers are necessary. The accumulated noise due to cascaded amplifiers limits the network size or necessitates signal regeneration, which is expensive. For more details about recent advances in optical buffering see the papers by Shun et al. [52] and Harai et al. [53].

Wavelength Convertors. Wavelength convertors can be used to reduce the number of delay lines, by converting the wavelength of a contending packet to a free wavelength at the output port. In a switch capable of optical buffering and wavelength conversion, the input is demultiplexed and each packet is sent to an appropriate destination (e.g., an output port). In the case of contention, a non-blocking space switch may send a packet on an available wavelength at the output port by using wavelength convertors or it may delay the packet. However wavelength convertors are expensive and full conversion is not easy to achieve [52].

Deflection Routing. Deflection routing resolves the contention by sending one of the contending packets to the desired link and passing the rest through any available link. The deflected packets are routed at the other nodes to their destination. This way packets of the same source-destination may experience different routes with different number of hops, which affects the network performance. Deflection routing usually is used in conjunction with optical buffering to reduce the need for buffering and to avoid too many recirculations at delay lines, which gives rise to signal to noise ratio degradation [52]. In the simplest method of deflection routing, delay lines are not used at all (see the hot-potato routing approach [54]).

Synchronous and Asynchronous Optical Packet Networks

Optical packet networks can be divided into two main categories: synchronous (slotted) and asynchronous (unslotted) frameworks. In the slotted case all the packets are aligned before they enter the switch. Whether or not the network is synchronous, bit-level synchronization and fast clock recovery are necessary at the switching stage for header recognition and packet delineation [52].

Synchronous Optical Packet Networks. In a slotted network the packet size is fixed.

Each packet and a fixed guard time should fit into a fixed length time-slot. Therefore the packets arriving at the switch are aligned in phase with a local clock reference. In such a network contention is mostly resolved by using optical delay lines with propagation delays equal to a multiple of the time-slot duration [52].

Asynchronous Optical Packet Networks. In an unslotted network the packets are not aligned and switching the packets may be performed at any time. Also the packets are not required to have the same size. Therefore the general behavior of the network is more unpredictable and the chance of contention is higher than that of slotted networks. On the other hand such a network is easier and cheaper to implement [52].

A Switch with KEOPS Architecture and Broadcast-and-select fabric

The KEOPS (KEys to Optical Packet Switching) switch was proposed as one of the key architectures in the European ACTS KEOPS project [4]. This $N \times N$ switch includes a wavelength encoder, a cell/packet buffer and broadcast section, and a wavelength selector block (Figure 2.2). At each input the wavelength convertor encodes its packet on a fixed wavelength. The cell/packet buffer block contains a splitter, K optical delay lines and a space switch stage. The space switch stage consists of a splitter, optical gates or Clamped-Gain Semiconductor Optical Amplifiers (CG-SOA) and combiners. CG-SOAs select the correct packets from delay lines and send them to their corresponding outputs based on the control unit information. The wavelength selector block is responsible for amplifying the signals before retransmission. The block corresponding to each output selects the correct packets from the inputs (i.e., the packets destined to that output port) and sends them to appropriate Semiconductor Optical Amplifiers (SOAs), which are wavelength dependent amplifiers. At the output ports the signals with different wavelengths are recombined and sent out. In this switch each input and output carries only one wavelength but the wavelength for an output port is not fixed (i.e., it can vary from packet to packet).

A Switch with a Broadcast-and-Select Fabric and Recirculation Buffer

A switch with a broadcast-and-select fabric proposed in [55] is shown in Figure 2.3. As in the architecture described in Section 2.2.2 the wavelength at each output port depends on the incoming packets' wavelengths. In this switch fabric, at the node inputs, tunable wavelength convertors (TWCs) convert the wavelengths of the incoming packets to new

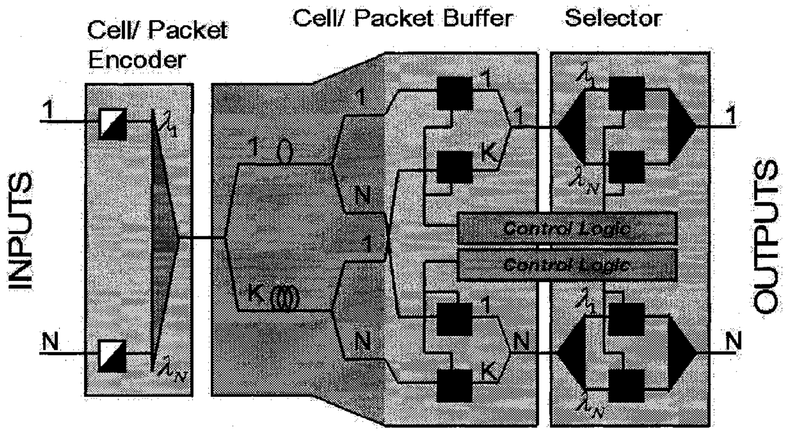


Fig. 2.2 Schematic of Broadcast-and-Select Switch including a wavelength encoder, a cell/packet buffer and broadcast section, and a wavelength selector block (see Section 2.2.2) [4].

wavelengths under the control unit information. At the beginning of each time-slot up to M input wavelengths enter the coupler (up to N input wavelengths plus the feedback from the 1 time-slot delay line). The coupler combines the input wavelengths and splits them into N tunable optical filters (TOFs) ($N < M$) and M fixed optical filters. According to the packets' destination the control unit determines up to N packets to be sent out through N TOFs and the remaining packets are delayed and recirculated through 1 time-slot delay line. The recirculated packets form part of the input to the coupler at the beginning of the following slot.

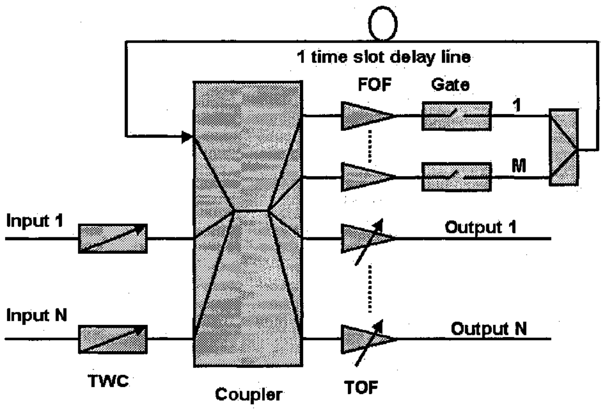


Fig. 2.3 A switch with a broadcast-and-select fabric and recirculation buffer (see Section 2.2.2) [4].

Architecture with a Wavelength Routing Switch Fabric

Many switch architectures based on wavelength routing have been proposed in the literature [56]. The switching procedure in most of these switches is based on using ODLs for contention resolution, and routing the packets to the correct output ports through the wavelength switch fabric. Here we review an architecture with an input-buffered switch fabric [5]. As Figure 2.4 shows this switch is composed of scheduling and switching sections. In the scheduling section each incoming wavelength is passed through a TWC and then the two $K \times K$ arrayed waveguide gratings (AWGs), where $K = \max(N, M)$. The AWGs are responsible for switching the optical signals from input ports to output ports. Each incoming and outgoing port carries a single wavelength. At each output port the wavelength varies with the packet. Between the two AWGs, M ODLs are used to resolve either internal or external contention. This architecture implements N individual buffers (corresponding to N different incoming wavelengths), each of which incorporates M positions. Each TWC converts the wavelength of the incoming packet to another wavelength in such a way to meet an ODL with an appropriate delay (each delay line has a different fixed wavelength). The ODLs are selected so that no two packets appear at the output of any ODL or the switch at the same slot. After the packets experience appropriate delays they are forwarded to their destination output ports through TWCs and AWGs. TWCs assign the appropriate wavelength to each outgoing packet corresponding to its destination port. Using wavelength convertors reduces complexity in the switching section.

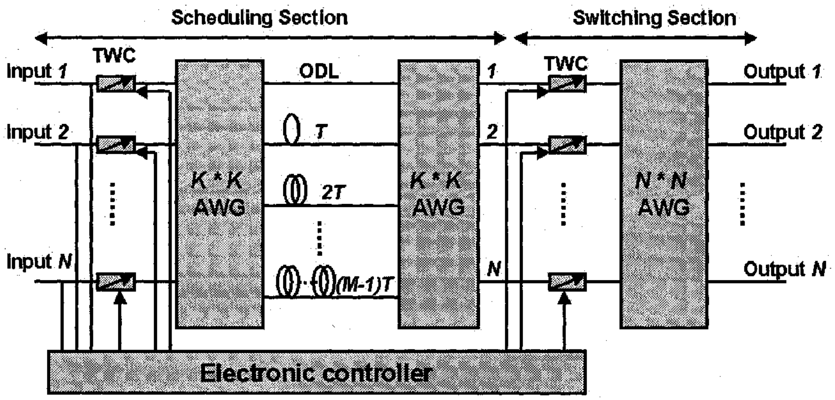


Fig. 2.4 An input-buffered switch including Array Waveguide Gratings (AWGs) for switching the optical signals and Tunable Wavelength Convertors (TWCs) [5].

2.2.3 Optical Burst Switching

Optical Burst Switching (OBS) tries to combine circuit and packet switching while avoiding the shortcomings of each. OBS is based on a one-way reservation scheme in which bursts of data follow control packets without waiting for acknowledgment [6]. Using OBS there is no need for optical-electrical-optical (O/E/O) conversion as in optical circuit switching, and similar to packet switching techniques it enables the network to share the resources among a number of users and increase the bandwidth utilization. In OBS the control packets are sent before the burst of data, so they are processed before the data arrives at the intermediate nodes. Consequently there is no need for data buffering as in packet switching. Moreover aggregating the packets as a burst reduces the overhead due to control packets. However, OBS is subject to a significant packet loss and burst retransmission. In the following we describe some proposed approaches for this switching method.

Optical Burst switching for Service Differentiation in the Next-Generation Optical Internet

Yoo et al. [6] survey various designs for an optical burst switch network. In OBS the control packets representing a burst of data are processed at each node along a predetermined route to establish a lightpath by configuring the WDM switches. Then the corresponding burst passes through the pre-configured lightpath. There are three different techniques for burst switching, called, tell-and-go (TAG), in-band-terminator (IBT), and reserve-a-fixed-duration (RFD). Among these techniques the third has been studied for all-photonic networks [6]. RFD is the basis for an optical burst switching protocol called Just-Enough-Time (JET) [57], which is described as follows:

An OBS Protocol using Offset-Time and Delayed-Reservation (Just-Enough-Time). The basic functionality of this protocol is shown in Figure 2.5. The header of each burst is sent before the payload by a "base" offset time, $T \geq \Sigma_{h=1}^{H} \delta(h)$, where $\delta(h)$ is the expected control delay at hop $1 \leq h \leq H$. In Figure 2.5, H=3 and $\delta(h) = \delta$. In the JET-based control protocol the bandwidth at each hop is reserved from burst arrival time, t_s, until burst departure time, $t_s + l$, where l is the length of the burst. At hop i, the burst arrival time is the summation of t_a, the time at which the control packet processing has been finished, and $T(i) = T - \Sigma_{h=1}^{i} \delta(h)$ which is the remainder of the offset time at hop i (i.e. $t_s = t_a + T(i)$). If the reservation fails due to contention with other requests, the

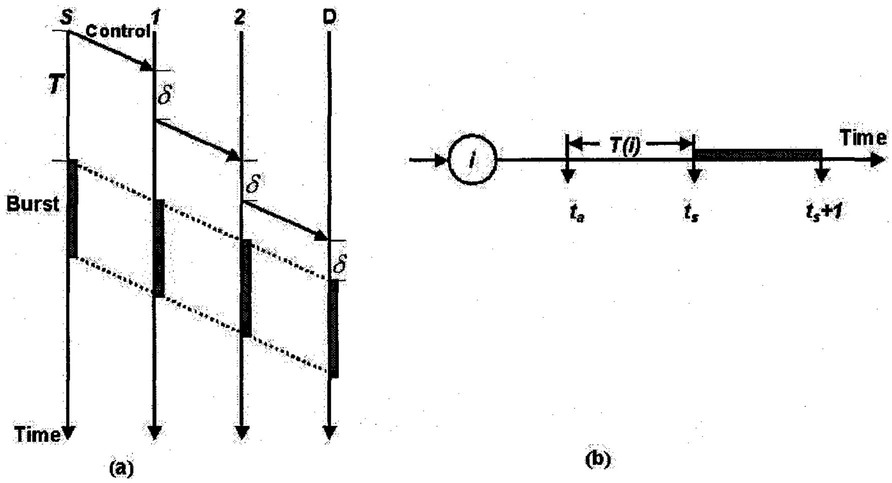


Fig. 2.5 The use of offset time, and delayed reservation in JET-based OBS: (a) the burst follows the control packet after a base offset time, T; (b) the bandwidth is reserved from the burst arrival time, t_s [6].

burst will be blocked. Blocking may be resolved by using ODLs. If ODLs are not available then the burst will be dropped [6].

An Offset-Time-Based QoS Scheme. This scheme uses an "extra" offset-time for supporting class isolation (or service differentiation) with or without ODLs [58]. Suppose that there are only two classes of traffic, class 0 and class 1, where class 1 has the highest priority. For the class 1 bursts, an extra offset time, t_o^1 is considered. The base offset time is considered negligible compared to this extra time. Let t_a^i and t_s^i be the arriving time and the service start time for a class i request, respectively, and l_i be the length of the request. To show how the class differentiation scheme works, we consider the following cases without ODLs (Figure 2.6):

Case 1, where the request 1 has arrived before request 0, $t_a^0 > t_a^1$ (Figure 2.6-(a)):
if $t_a^0 + l_0 \geq t_s^1 \geq t_a^0$ or $t_s^1 \leq t_a^0 \leq t_s^1 + l_1$ the class 0 request will be dropped
else, the request 0 will succeed.

Case 2, where request 0 has arrived before the request 1, $t_a^1 > t_a^0$ (Figure 2.6-(b)):
if $t_a^1 + t_o^1 < t_a^0 + l_0$ request 1 will be dropped.

To avoid blocking of class 1 requests, t_o^1 needs to be larger than the maximum burst length in class 0. When ODLs are available the problem is more complicated but by choosing an appropriate offset time, class 1 can be isolated from class 0 requests in reserving both bandwidth and ODLs [6].

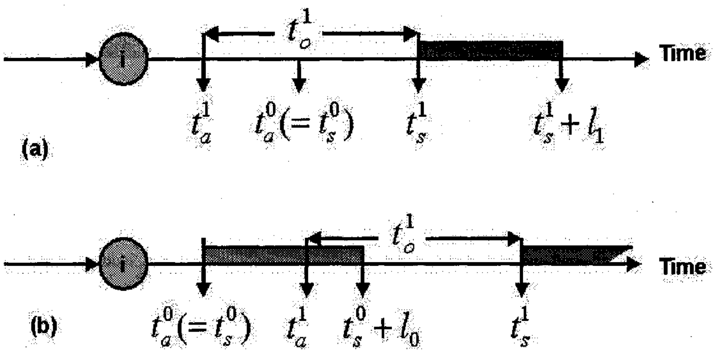


Fig. 2.6 Class isolation at an optical switch without ODLs: (a) shows the case where request 1 has arrived before request 0; and (b) shows the case where request 0 has arrived before request 1 [6].

Time Sliced Optical Burst Switching

Time Sliced Optical Burst Switching (TSOBS) is an approach proposed by Ramamirtham et al. [7], which replaces wavelength domain switching with time domain switching. This eliminates the need for wavelength converters and results in a lower cost. TSOBS also reduces the number of delay lines required for supporting burst switching by using blocking Optical Time-Slot Interchangers (OTSI) and Optical Crossbars.

TSOBS Networks. Figure 2.7 shows the architecture of a time-sliced optical burst switched network. Terminals or other networks are connected to the TSOBS network by concentrators. WDM links connect the network of time-sliced optical burst switches. The information on each wavelength is organized into a series of frames of fixed length time-slots. Concentrators transmit user data packets (packet switching) or aggregated user data (burst switching) in time-division channels. The Burst Header Cells (BHC) are carried on separate control wavelengths. If the ratio of the average burst length to the BHC length is L, each control wavelength can support $L-1$ data wavelengths. Each BHC carries information about the length of the burst, source-destination addresses, wavelength, and the identification of the first frame in which the data burst appears. It also includes a field in which the distance traveled by a burst and the number of optical operations since its last regeneration are recorded. This information is used to regenerate the burst before too much signal degradation arises due to noise and signal attenuation at each switching stage [7].

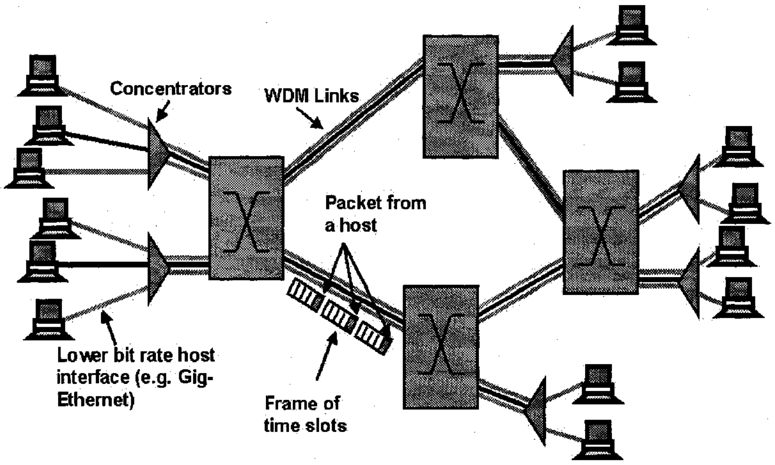


Fig. 2.7 Time-sliced packet switched network architecture, composed of a network of TSOBSs and concentrators; concentrators transmit user data bursts in time-division channels. Data is carried on the WDM link within frames comprised of fixed length time-slots [7].

Optical time-slot Interchangers. OTSIs perform time domain switching in the TSOBS architecture. They can be used in either non-blocking or blocking designs. The simplest form of a non-blocking OTSI has N delay lines and an $(N+1) \times (N+1)$ optical crossbar switch (see Figure 2.8), where N is the number of time-slots in each frame. Delay lines provide $1, 2, \cdots, N$ time-slots interval delay. A more practical design for a non-blocking switch has been introduced in [7]; the design uses $2\sqrt{N}-2$ delay lines and a $(2\sqrt{N}-1)\times(2\sqrt{N}-1)$ crossbar switch. Blocking OTSIs are an alternative to nonblocking OTSIs. These designs have lower cost and complexity, at the cost of some small non-zero blocking probability. For this design $N/2$ delay lines provide $1, 2, 4, \cdots, N/2$ time-slot delays. A small crossbar switch $(\sqrt{N} \times \sqrt{N})$ is needed to support switching operations between the delay lines as well as the incoming time-slots. Thus the number of switching operations increases as the switch is subject to blocking. When a time-slot arrives at the switch, a search procedure is performed to find an available sequence of delay lines for creating the desired delay for that slot. For this task the state of the delay lines must be tracked and recorded in a scheduling array. To find the best delay path, a shortest path tree algorithm is performed. This algorithm finds the path with the smallest number of switching operations which is needed to obtain the desired delay. Blocking OTSIs can be designed with different number of delay lines and crossbar switches. Each design has a different cost and blocking probability [7].

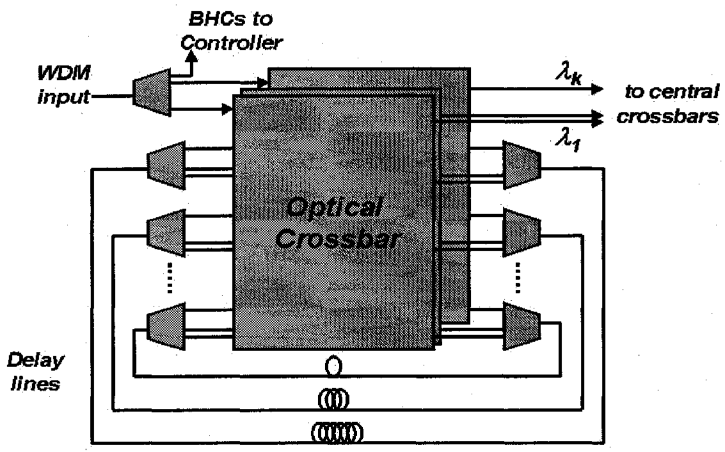


Fig. 2.8 Optical time-slot Interchanger, which performs time-domain switching; each OTSI contains a set of optical crossbars for switching time-slots among the inputs, outputs and a set of delay lines. The signals are demultiplexed to perform the switching operations and remultiplexed onto the delay lines, allowing the cost of the delay lines to be shared by the different wavelengths [7].

Switch Architecture. Figure 2.9 shows the overall design for a TSOBS router. The router consists of synchronizers, OTSIs, a central optical crossbar, a controller and WDM multiplexors. The synchronizers can be implemented using a space-division optical switch and a finely calibrated set of delay lines to synchronize the incoming frames to the local clock. Performance evaluations show that the blocking design performs almost as well as non-blocking switches while the cost is reduced significantly due to a small number of delay lines needed to provide an acceptable blocking probability. An OTSI separates the control wavelengths and forwards them to the controller. The controller processes the BHC information and passes the results to optical crossbar switch as well as the OTSIs. When a BHC arrives at the controller, the controller determines the appropriate outgoing link for that burst. Then on the outgoing links and the output of the corresponding OTSI a look-up procedure determines the available time-slots on the wavelength being used by the burst. OTSI then provides time domain switching for all wavelengths available on each link based on the information about the available time-slots on the outgoing links. It separates the data wavelengths and sends each one on a different fiber to the optical crossbar after the appropriate time domain switching is performed. The number of delay

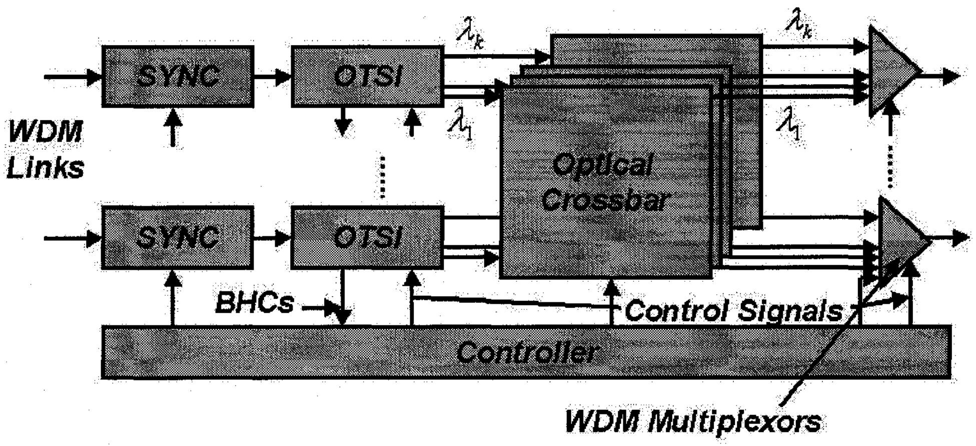


Fig. 2.9 The overall Time-Sliced Optical Burst Switch Design; each incoming WDM link terminates on a Synchronizer (SYNC) which synchronizes the incoming frame boundaries to the local timing reference. Optical Time Slot Interchangers (OTSI) provide the required time domain timing of switching operations on the data wavelengths [7].

lines used in OTSIs is the key parameter affecting both the performance and the cost of the TSOBS switches. The optical crossbar performs space switching on individual wavelengths and WDM multiplexors combine the data wavelengths with control wavelengths on the outgoing links.

2.3 Bandwidth Reservation in WDM systems

In an all-optical communication system, wavelength division multiplexing (WDM) has been introduced to use the substantial optical bandwidth efficiently by partitioning the optical bandwidth into WDM channels compatible with the speed of electronic processing. WDM systems can incorporate time-division multiplexing (TDM), which provides WDM access to many end-users sharing the optical channels in the time domain, and therefore makes photonic networking practical.

In order to effectively facilitate multiple access to the network resources, bandwidth reservation can be used as an admission control mechanism. Bandwidth reservation uses at least one channel for reservation and the rest are allocated to data transmissions [59]. Reservation can be performed in several forms including burst switching and fixed time-slot assignment. In a burst switching model the bandwidth is requested before transmitting the burst of data, while the data is usually transmitted after a period of time, without waiting for acknowledgment [6]. The waiting time is considered just long enough for processing the request of bandwidth on an appropriate outgoing link, which should be done in relation to the other contending requests. In fixed time-slot scheduling the time is divided into small partitions called slots, which are assigned to the requests ahead of time. Therefore in this model the transmission is performed after receiving the approval from the controller. In time-slot scheduling, the assignment can be accomplished using many varieties of scheduling algorithms.

Scheduling algorithms can be designed in an *on-line* [59–61] or *off-line* [62–64] manner, each of which involves a different set of requirements and techniques. An on-line (or incremental) scheduling algorithm computes a schedule based on the available partial information for each arriving request. During the reservation phase, as soon as the first request arrives the scheduler starts computation. This technique has a relatively low computation time, but the result is not always optimal [59]. On the other hand the off-line scheduling algorithms need the entire traffic demand matrix to start computing the schedule. The demand matrix whose (i, j)-th element represents the number of (fixed-size) packets that must be transmitted from source i to destination j, can be collected during the reservation phase (in a dynamic traffic scenario) or the algorithm might consider an *a priori* traffic distribution pattern (in a static traffic scenario) [60]. In the latter case referred to as *static* scheduling, the traffic demand changes do not affect the schedule [59].

Depending on the traffic pattern the scheduling problem can be solved for a fixed frame length or a variable frame length. The variable frame length structure does not suit connections which need reservations for several consecutive frames. In long term transmissions the users usually specify their requests in bits per second. Therefore changing the frame length changes the user's throughput if the number of slots per frame allocated to a user is not changed. Consequently the variable frame length formalization is more suitable for one-shot transmissions [65]. In addition, when there is a large propagation delay between the elements in the network, the length of the next frame should be estimated which adds computational complexity to the control plane. Also with variable-length frames the running time is not fixed, and the scheduling calculation time is limited by a variable amount (i.e. the length of the frame).

2.4 Summary

In this chapter we presented a summary of the proposed resource allocation techniques for photonic networks. Routing and wavelength assignment deal with the problem of sharing optical links and wavelengths between circuit-switched optical connections. It has been proved that this problem is *NP*-complete and the problem is divided to two subproblems, routing problem and wavelength assignment problem. Due to the limited number of wavelengths in a network, circuit switching is not a practical solution for large networks. For the case of packet-switched photonic networks several strategies have been proposed such as photonic slot routing, packet switching and optical burst switching.

Photonic slot routing is a time slotted approach that uses a special type of packet switching. A group of packets heading for the same destination travels over a photonic slot that includes all wavelengths in a network. Compared to a packet-switched network, a photonic slot (instead of each individual packet per wavelength) is switched at each intermediate node.

Optical burst switching is a combination of circuit switching and packet switching in the sense that the bandwidth is requested for a burst of data, while the burst is transmitted without waiting for an acknowledgment.

The main problem with all of the packet switching techniques stated above is contention. Bandwidth reservation in a TDM basis provides a more flexible tool for both avoiding contention and utilizing the optical bandwidth. This chapter provided a brief introduction to TDM/WDM bandwidth reservation, which will be discussed in more detail in following chapters.

Chapter 3

Scheduling

This chapter reviews basic concept about scheduling and its application to telecommunication networks. The chapter starts by an introduction on scheduling of the systems that can be represented by bipartite graphs. Then an adaptation of bipartite graph scheduling to photonic networks is presented. This provides background material for studying different types of scheduling (i.e. slot-based, variable-length and fixed-length frame scheduling) for point-to-point and broadcast transmissions in WDM networks, and their relationship.

Scheduling is the problem of allocating resources over time to a set of processors to perform a number of tasks. In order to study the traditional scheduling algorithms we describe a scheme for classifying scheduling problems developed by Graham et al. [66]. Suppose that M machines or processors P_k $(k = 1, ..., M)$ have to process N jobs J_i $(i = 1, ..., N)$. A schedule is an allocation of one or more machines to each job. A schedule is *feasible* if at any time, there is at most one job on each machine, each job is run on at most one machine, and it satisfies a number of requirements concerning the machine environment and the job characteristics. A schedule is optimum if it minimizes (or maximizes) a given optimality criterion.

A shop scheduling problem consists of a set of M processors. Each of these processors performs a different task. There are N jobs, each consisting of M tasks. Each task j of job i denoted by $t_{i,j}$ is to be processed on processor j for a total duration of $dur(t_{i,j})$. In each time the following restrictions must be satisfied for the machines and the jobs: (i) each machine can execute at most one task at any given time, and (ii) for each job at most one task is to be assigned. Depending on the orders by which the jobs and the tasks should be performed the shop scheduling problem is usually classified to three basic groups:

1. When there is no ordering constraints on operations of the jobs the scheduling is

described as an *open* shop scheduling.

2. When operations of the tasks of each job should follow a specific order the shop scheduling is called *job* shop.
3. When every job goes through all M machines in a unidirectional order the shop scheduling is *flow* shop. Each job has exactly M tasks. The first task of every job is done on machine 1, second task on machine 2 and so on. However, the processing time each task spends on a machine varies depending on the job that the task belongs to.

When the shop is *open* the jobs and tasks can be executed in any order. The scheduling algorithms can be designed for two different categories based on how the tasks deal with interruption: *preemptive* and *nonpreemptive* schedules. A preemptive schedule is the one which does not restrict the tasks to be executed continuously. A nonpreemptive schedule is the one in which the tasks can not be interrupted once they have begun execution [26].

For a given open shop problem, we try to obtain an optimal finish time (OFT). An OFT minimizes the *makespan* or the time required to complete all the jobs. In general a *preemptive open shop* problem can be solved in polynomial time, while a nonpreemptive open shop problem is shown to be NP-hard [1] [67] for more than three machines [68,69], but many heuristics exist to obtain near-optimal finish time for this case [70,71]. The open shop scheduling problem of N jobs and M machines is denoted by $N \mid M \mid openshop \mid OFT$ [26].

The shop scheduling problem can be represented by means of graphs. We now review some terminology and fundamental definitions concerning bipartite graphs. The following definitions are presented from [68, 72].

Definition 1. *Graph: A graph G is a pair $G = (Y, E)$ where Y is a finite set of nodes or vertices and the elements in E consist of subsets of Y of cardinality two called edges. If $e = [v_1, v_2] \in E$, then we say that e is incident upon v_1 (and v_2). The degree of a vertex v of G is the number of edges incident upon v.*

Definition 2. *Weighted graph: A graph G in which each edge has been assigned a number, or a weight, is called a weighted graph. In a weighted graph, the weight of a vertex v is*

[1] A problem is *NP*-hard if solving it in polynomial time would make it possible to solve all problems in class NP in polynomial time.

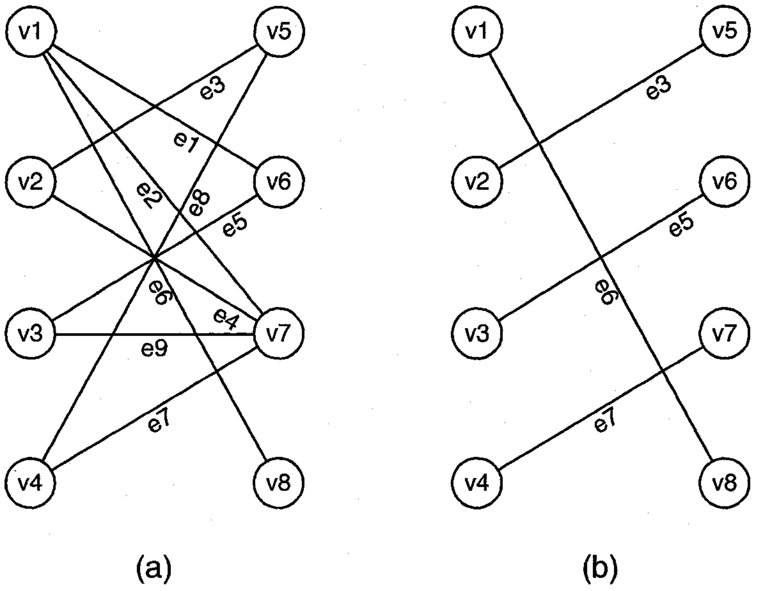


Fig. 3.1 (a) A bipartite graph of 8 vertices and 9 edges. (b) A matching of maximum cardinality obtained from the bipartite graph. This matching is a perfect matching.

defined as the sum of the edge weights of all edges incident to v in graph G.

Definition 3. *Bipartite graph: Let $G = (Y, E)$ is a graph that has the following property: the set of vertices Y can be partitioned into two sets, V and U, and each edge in E has one vertex in V and one vertex in U. Then G is called a bipartite graph and is usually denoted by $G = (V \cup U, E)$. Figure 3.1-(a) shows a bipartite graph of 8 vertices and 9 edges.*

Definition 4. *Degree of a graph: The degree of a graph $G = (Y, E)$ is defined as the maximum degree of all its vertices. For example in Figure 3.1-(a) the degree of the graph is 3.*

Definition 5. *Bipartite matching: Let $G = (V \cup U, E)$ be a bipartite graph with vertex sets V and U, and edge set E. A set $I \subseteq E$ is a matching if no vertex $w \in V \cup U$ is incident with more than one edge in I. A matching of maximum cardinality is called a maximum matching. A matching is called a complete matching (or perfect matching) of V into U if the cardinality (size) of I equals the number of vertices in U. Figure 3.1-(b)* *shows a matching of maximum cardinality obtained from the graph of Figure 3.1-(a).*

Definition 6. *Augmenting path: An augmenting path P relative to a matching M in G is a path in G such that the first and the last vertices in P are not covered by M, and the edges in P alternate between being in M and not being in M.*

Claim 1. *If there exists such an augmenting path it can be proved that a matching of greater cardinality can be found in G by augmenting M with augmenting path P. Consequently, a matching is of maximum cardinality if and only if it permits no augmenting path [72].*

Maximum Cardinality Matching Algorithm for Bipartite Graphs (MCB): Based on *Claim 1* we can construct maximum-cardinality matchings by searching for augmenting paths and stopping when none exist. The algorithm starts with an arbitrary matching Q in graph G, and search for a possible augmenting path. If an augmenting path P with respect to Q is found, the matching Q is augmented which means a new matching is constructed by taking those edges of Q or P that are not in both Q and P. The process is repeated and the matching is maximal when no augmenting path is found.

3.1 Variable-Length Frame Scheduling for Point-to-Point Transmission

The problem of finding an optimum schedule for a variable-length frame, has been extensively studied in input-queue switching, WDM and satellite systems [26,28,67,73,74]. We are presented with a demand matrix $D = [D_{ij}]$, where D_{ij} is the number of slots requested by source node i for destination j during next frame. The goal is to minimize the overall transmission time T for a schedule S:

$$T(S) = T_x(S) + \tau \, . \, N_s(S), \tag{3.1}$$

where N_s is the number of switch reconfigurations, τ is the switching time, and T_x is the time spent transmitting the traffic [26, 67]. All times are measured in slots. The schedule can be considered as a two dimensional matrix with time and destination (or source) ID dimensions. The elements of this matrix show the source (or destination) ID for which the corresponding time slots are reserved (see Figure 3.2). For the case of multiple optical channels and distinct transmitters and receivers, each wavelength can be scheduled separately. For a given demand matrix we define:

$$\begin{aligned} r_i(D) &\triangleq \sum_{1\leq j\leq N} D_{ij}, \\ c_j(D) &\triangleq \sum_{1\leq i\leq N} D_{ij}. \end{aligned} \tag{3.2}$$

In other words, r_i, is the amount of the demand at input i, and c_j is the total demand for output j.

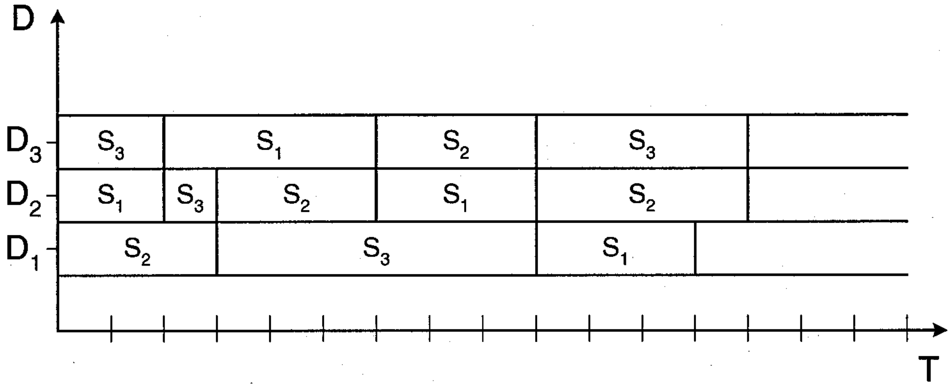


Fig. 3.2 Destination-Time scheduling matrix: D and S, and T correspond to destination, source and time-slot respectively. In Point-to-point transmission at a given time slot on each wavelength each source node transmits to at most one destination.

PROBLEM 1: Solve the following optimization problem for a frame of variable length with total transmission time $T(S)$ defined by (3.1), observing the constraint that $S \in \mathcal{S}$, the set of schedules that satisfy the demand matrix.

$$S_1^* = \arg\min_{S\in\mathcal{S}} T(S) \tag{3.3}$$

Depending on the value of τ the problem of finding a minimum schedule length given by equation 3.1 is usually reduced to one of the three following cases.

1. When τ is negligible compared to the duration of a time slot, *PROBLEM 1* can be closely approximated by the minimization of T_x, which is solvable in polynomial time [75–77]. The minimum traffic transmission time is [68]:

   $$T_{x_{min}} = \max\{\max_i\{r_i(D)\}, \max_j\{c_j(D)\}\}.$$

   We can then establish:

   Claim 2. *A schedule S_x that minimizes the traffic transmission time, i.e., $T_x(S_x) = T_{x_{min}}$, solves PROBLEM 1 to within an approximation factor of* $1 + \tau$.

   Proof. The number of switch reconfigurations $N_s(S) < T_x(S)$ (note that T_x is measured in time slots and so is comparable with N_s) and $T_{min}(S_1) = T_x(S_1) + \tau N_s(S_1) > T_{x_{min}}$. Hence if S_x minimizes the traffic transmission time, it satisfies $T(S_x) < T_{x_{min}}(1 + \tau) < T_{min}(S_1)(1 + \tau)$. □

   For the special case of small τ, approximate algorithms that attempt to minimize N_s subject to the constraint that T_x is minimum have been proposed in [26–28, 73]. The algorithms achieve minimum traffic transmission time, $T_{x_{min}}$, but do not guarantee minimum *total* transmission time, $T_{min}(S_1)$, unless the switching overhead is completely neglected. The *EXACT* algorithm, presented in [27, 28], achieves a minimum traffic transmission time, $T_{x_{min}}$ and the derived schedule has at most $N_s = N^2 - 2N + 2$ switch configurations [28]. In the case of an admissible demand matrix, the *EXACT* algorithm generates a schedule S that has length less than L and therefore zero rejection. The *EXACT* algorithm is an iterative procedure that repeatedly performs maximum cardinality bipartite matching (MCBM) to obtain the schedule.

2. When τ is very large (on the order of minimum transmission time), *PROBLEM 1* is reduced to minimizing $T(S)$ subject to the constraint that N_s is minimum which is an *NP*-hard [27, 67] problem. Approximate algorithms for this special case have been proposed in [26, 67].

3. When τ is moderately large, it is more desirable to obtain near minimum solutions for both the number of switchings and the traffic transmission time [28]. Crescenzi et al. demonstrate that *PROBLEM 1* cannot be approximated by a polynomial algorithm within a factor less than $\frac{7}{6}$ [27].

The three problems stated above have been formulated as variants of the open shop scheduling problem in the literature [28, 67, 68, 74, 78].

3.1.1 Analogy

The scheduling problem for an $N \times N$ optical switch can be translated into an open shop scheduling problem with $M = N$ processors and N jobs. The jobs correspond to the inputs of the switch and the processors correspond to the outputs. Each input-output traffic demand, D_{ij}, is represented by a task $t_{i,j}$. Similar to the open shop formulation each task, D_{ij}, belongs to a specific job (input i) and is to be processed by a specific processor (output j) [26]. The scheduling problem obtained is $N \mid N \mid$ *open shop* $\mid$ *OFT*. The scheduling constraint in an optical switch operating on one wavelength can be translated directly to the open shop scheduling. The constraint in an optical switch is that at each given time (i.e., a time slot) at most one request from input i can be serviced at each output j:

- One assignment of each input per slot is equivalent to the constraint that a job cannot be processed by more than one processor at any given time.

- One assignment of each output per slot is equivalent to the constraint that a processor can perform at most one task at any given time [26].

In the following we describe several solutions for the open shop scheduling problem for different values of τ which have been designed for satellite systems and passive star networks. We develop the algorithms for an $N \times N$ nonblocking optical switch. The first group of the algorithms aim at minimizing the number of switchings for a minimum duration schedule [26, 73]. The second group of the algorithms try to minimize the schedule length when the number of switchings is minimum [26, 67]. The third group of algorithms provide near-optimum solutions with bounds on the number of switchings and the schedule length [28].

3.1.2 Minimal Duration Scheduling

The problem of finding a schedule with minimum duration has been studied in many research areas such as networks, computers, and satellite systems [26, 28, 73, 77], and was shown to be solvable in polynomial time. The algorithms achieve a minimum traffic transmission time, $T_{x_{min}}$, but the minimum schedule length, T_{min}, for non-negligible switching overhead is not guaranteed. In this section we present a simple algorithm which obtains at most $N_s = N^2 - 2N + 2$ switch configurations per port [28]. This algorithm assumes that the assignments for each demand do not need to be continuous, which translates our scheduling problem to solving a *preemptive* $N \mid N \mid$ *openshop* $\mid OFT$ problem.

Given a set of N jobs with task times $D_{ij}, 1 \leq i \leq N$ and $1 \leq j \leq N$, for an N-processor open shop problem, we have:

$$\begin{gathered} length\ of\ job\ i = r_i(D)\ , \\ total\ time\ needed\ on\ processor\ j = c_j(D), \end{gathered} \tag{3.4}$$

It is easily understood that the minimum processing time and $T_{x_{min}}$ are equivalent.

EXACT Covering Algorithm. Here we present the *EXACT* algorithm in more detail since this is one of the basic elements of the frame-based algorithms that will be described in the next chapter. The *EXACT* algorithm, presented in [27, 28], is based on finding the maximum cardinality matching in bipartite graphs. We construct a bipartite graph $G = (V \cup U, E)$, where V is the set of vertices corresponding to the N jobs (inputs of the optical switch), U is the set of vertices corresponding to the N processors (outputs of the optical switch), and E is the set of edges incident to each input-output pair. The algorithm repeatedly performs maximum cardinality matching on the nonzero elements of the traffic matrix D to find a matching (defined by a permutation matrix P). A permutation matrix is a 0/1 matrix that has all row-sums and column-sums of 1. The weight of each matching, which corresponds to each configuration, is equal to the minimum weight of the edges participating in the matching. The weight of each edge (v_i, u_j) is the amount of the requested traffic from input i to output j.

Algorithm EXACT.

```
% initialization
$i = 1, A = D$
create graph $G = (V \cup U, E)$ from $A$
% find a maximum matching $M$ of this graph using algorithm MCB
while $A \neq \bar{0}$;
    $M = MCB(V \cup U, E)$;
    % schedule construction
    $P(k) = M$; % construct a permutation matrix
    $w(k) = min\{w(e) : e \in M\}$;
    % update the graph
    $A = A - w(k)P(k)$;
    k=k+1;
    % finish when all of the elements in $A$ are zero
end
```

In this algorithm $P(k)$ and $w(k)$ are stored to construct the schedule. It has been shown in [28,77] that at most $N_s = N^2 - 2N + 2$ switch configurations are necessary to cover the traffic matrix. However, this approach provides a *pseudo*polynomial-time algorithm, since its running time depends linearly on the weights of G [27] which depend on the input data. In [26,68] a similar algorithm, namely the Complete Matching Algorithm (CMA) has been proposed which obtains the schedule in polynomial time. In this approach the bipartite graph is constructed using $N + M$ real nodes and $N + M$ fictitious nodes. The idea is to force each processor to have the same load equal to $T_{x_{min}}$, and each job to have the same processing requirement equal to $T_{x_{min}}$, hence obtaining a weight-regular graph, that is, a graph whose vertices have equal weights. A weight-regular graph guarantees the existence of a complete matching. With complete matchings it is guaranteed that all processors are being used and all jobs are being processed in each iteration [68]. Therefore it can be proved that the algorithm CMA always produces a minimum length schedule of duration $T_{x_{min}}$ [26,68]. The time complexity of this algorithm is $O(\|V\|\|E\|)$ where $\|V\|$ and $\|E\|$ are the number of vertices and the number of edges in the Bipartite graph respectively. The size of a maximum cardinality matching, and thus the maximum number of iterations required to compute it, is $O(\|V\|)$, and the complexity of a graph search procedure is $O(\|E\|)$.

Using parallel processing methods, the fastest known algorithm has a complexity of $O(\sqrt{\|V\|}\ \|E\|)$ [79]. This algorithm reduces the number of iterations from $O(\|V\|)$ to $O(\sqrt{\|V\|})$ by looking for a set of disjoint M-augmenting paths each iteration, then augmenting along all the discovered paths. In other words, the algorithm runs the search procedure from all unmatched vertices simultaneously rather than one by one.

3.1.3 Minimum Number of Switchings

Recall that the objective function to minimize the overall transmission time is $T(S) = \tau N_s(S) + T_x(S)$, where τ is the switching overhead, T_x is the traffic transmission time, and N_s is the number of switchings. Algorithms described in this section give a schedule with minimum number of switchings. The algorithms aim at minimizing T_x subject to the constraint that N_s is minimum. The problem in this case is formulated as a *nonpreemptive open shop* scheduling problem. Nonpreemptive scheduling guarantees that the minimum number of switchings is always obtained. In [80] it has been proved that minimizing the makespan (total transmission time) in a nonpreemptive open shop scheduling problem when $M > 2$ is NP-complete, where M is the number of processors (for an $N \times N$ switch $M = N$).

The minimum number of switchings is determined by:

$$N_{s_{min}} = \max\{\max_i\{\|r_i(D)\|\}, \max_j\{\|c_j(D)\|\}\}, \tag{3.5}$$

where $\|r_i\|$ is the number of nonzero entries in row i and $\|c_j\|$ is the number of nonzero entries in column j.

Nonpreemptive Open shop Scheduling Algorithm. The algorithm presented in [26] is based on the most number of tasks (demands) first heuristic. The algorithm starts by the output which is requested by the largest number of inputs. At a given time slot if the j-th output is available and the inputs i and k are free, if the total number of tasks of input i is greater than the total number of tasks of input k, then the demand D_{ij} is processed before the demand $d_{k,j}$. Recall that the total number of tasks of input i is the total number of outputs for which input i has requests (the number of nonzero entries in row i of a demand matrix). If the total number of tasks of input i is equal to the total number of tasks of input k, and if the total number of requests by input i is smaller than the total number of requests by input k, then D_{ij} is processed before the demand $d_{k,j}$. The total number of requests by input i is the sum of the total demands requested by input i. This algorithm always produces the minimum number of switching matrices.

3.1.4 Near-Optimum Solutions

The algorithms described in Sections 3.1.2 and 3.1.3 provide solutions for reducing the number of switchings and the transmission time respectively, but there is no guarantee on the performance of the proposed algorithms when τ is neither negligible nor very large. Using the proof described in [72] it can be shown that the algorithm described in 3.1.3 has an unbounded approximation ratio for the transmission time. On the other hand the algorithms described in 3.1.2 (e.g. $EXACT$) provide the minimum traffic transmission time, but there is not a tight bound on the number of switchings. Consequently, the overall transmission time for the case that the switching overhead is not negligible is not close to optimal.

In [27] the problem of minimizing the overall transmission time with non-negligible switching times is described as the *preemptive bipartite scheduling* and shown to be NP-hard using the proof in [67]. Also it is shown that one cannot approximate this problem within a factor less than $\frac{7}{6}$, but the best algorithm proposed in [27] approximates the preemptive bipartite scheduling problem within a factor of two.

In the following we review an approximation algorithm for solving the preemptive bipartite scheduling problem. This algorithm does not restrict the optimal schedule length or the minimum number of switchings. Instead, by allowing twice as many as the minimum number of switchings, it achieves a near-optimum schedule length within a ratio of two.

Graham's List Scheduling. List scheduling (LIST) introduced by R. L. Graham [78], is a greedy algorithm[2] that approximates the optimal open shop problem within a ratio of two. LIST starts by assigning a job to each processor. If multiple jobs are contending for the same processor one of them is chosen arbitrarily. Once a processor is idle, one of the free jobs which has a task for the corresponding processor is chosen arbitrarily. This procedure continues until all of the jobs are processed.

The maximum schedule length produced by LIST is bounded by the sum of the time to process the longest job (equal to the maximum row-sum in the traffic demand), and the time that the most heavily loaded processor needs (equal to the maximum column-sum in the traffic demand) [81]. Therefore this algorithm tends to a delay overhead of at most $2\tau N_{s_{min}}$, though the number of switchings at each port is $N_{s_{min}}$.

[2] A greedy algorithm is an algorithm that optimizes the choice at each stage without regard to previous choices, with the hope of finding the global optimum.

3.2 Slot-based Scheduling for Point-to-Point Transmission

In this section an overview on slot-based scheduling problem for all-photonic networks is presented to provide the background information for performance comparison of the scheduling algorithms in Chapter 4. In the slot-based scheduling algorithms the configuration of the core switch is computed once for each time slot, according to reservation requests from the edge nodes. All of the algorithms proposed for the input-queued switches (e.g. [17, 19, 19–21]) are directly applicable to slot-based scheduling for photonic networks. However, the propagation delay between the edge nodes and the core switch makes direct application of these algorithms less efficient. Therefore care needs to be taken when applying these algorithms to all-photonic networks. The approach proposed in [82] is a good example of the necessary modifications for making the input-queued switch scheduling algorithms such as Parallel Iterative Matching (PIM) specifically applicable to all-photonic networks.

In this approach configuration of the switch is performed by applying a matching algorithm that identifies source-destination node pairs based on the incoming requests. Each source edge node maintains a set of VOQs, one associated with each egress node. At each time slot, every edge node sends a request to the core switch, specifying whether a specific VOQ has traffic to send and hence requires a slot. The central controller applies a matching algorithm to determine a schedule based on the arriving requests, and sends grants back to the edge nodes, indicating which VOQs may transmit during specific time slots. To calculate the schedule for each time-slot, the PIM algorithm [17], an iterative matching algorithm that randomly identifies input-output pairs is used. Each iteration of PIM consists of three steps:

1. Request: Each unmatched input sends a request to every output for which it has queued slots.
2. Grant: If an unmatched output receives any requests, it grants one of them, selecting at random.
3. Accept: If an input receives grants, it accepts one (selecting at random if multiple grants are received).

An edge node may send multiple requests before it has received a single grant. However, an edge node does not send a request immediately upon the arrival of a packet in a VOQ. The number of packets in the VOQ for which a request has not been issued must exceed a specified threshold before a new request is sent. Many packets fit in a time slot, so if this policy is not in place, a lightly-loaded edge node may request more slots than it needs and be granted a disproportionate number of slots. This can lead to poor utilization within slots and blocking of heavily-loaded edge nodes.

Once a request has been issued, the packets associated with that request are "marked" and no second request is issued for them. This avoids the problem of receiving multiple grants for the same set of packets. We must however ensure that every request is eventually granted, although there may be some time delay in the process. To achieve this, the central controller maintains a list of ungranted requests. These ungranted requests have higher priority than requests that have just arrived, and the priority is highest for those requests that have waited longest. The controller applies the PIM algorithm, but instead of each output randomly selecting an input in stage one, it selects the input with highest priority request. If multiple requests have the same priority, one of them is selected at random. Since PIM is an approximate algorithm, and not an optimal algorithm such as the maximum cardinality matching algorithm it may give rise to some unused time slots. As a practical matter, unmatched output ports are randomly assigned to a VOQ and a grant is sent despite the absence of a request.

3.3 Variable-length Frame Scheduling for Broadcast WDM Transmission

Many algorithms have been proposed so far for bandwidth reservation in an all-optical broadcast-and-select network with a star topology [31,63,65,73,83,84]. In these networks a star coupler provides the connections between several nodes, which are equipped with tunable transmitters and/or tunable receivers. In general, the number of nodes is greater than the number of available channels, and therefore the transmitters have to share the channels using a scheduling technique. The schedule can be considered as a two dimensional matrix, with wavelength and time dimensions (see Figure 3.3). At each time instant a given source node can occupy only one wavelength.

The problem of variable-length frame scheduling in these networks has been extensively analyzed [63,65,73,83]. The proposed algorithms try to optimize the schedule while delivering all traffic within the next frame. This problem is usually formalized as follows [31]:

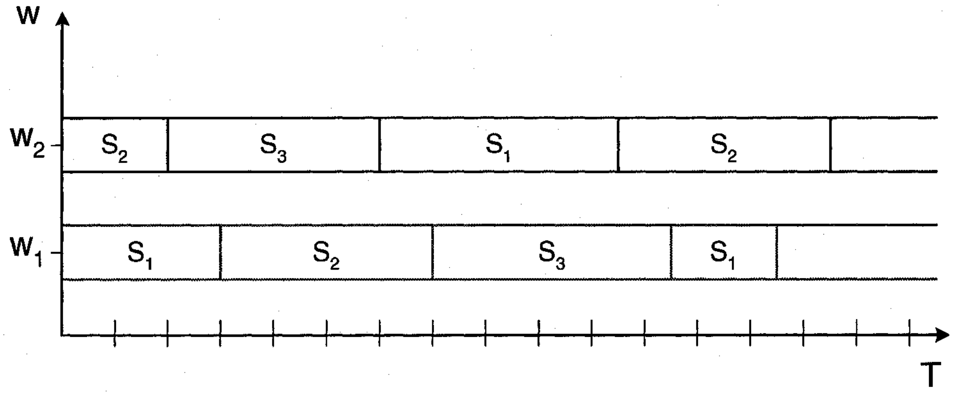


Fig. 3.3 Wavelength-Time scheduling matrix: W, S, and T correspond to wavelength, source and time-slot respectively. In broadcast-and-select networks in which every node is equipped with one tunable transmitter and one tunable receiver, at a given time only one node can transmit on at most one wavelength. The broadcast data is selected by the intended destination.

> "Given a traffic matrix D whose elements D_{ij} are the numbers of (fixed-size) packets that must be transmitted from any source user i to any destination user j, find a time/wavelength assignment that guarantees the delivery of all traffic, while minimizing the time necessary to accommodate all transmissions (the frame duration), subject to tuning delay constraints."

The main objective of a scheduling algorithm is to minimize the computation time while maximizing the utilization of the network resources. In this particular network with a variable frame length, increasing the utilization is equivalent to reducing the schedule length. Choi et al. [85] prove that the lower bound on the schedule length in a broadcast-and-select network of N nodes and w channels, with tuning latency of δ is given by:

$$\max\{\delta + \frac{N^2}{w}, w\delta + \frac{N^2}{w^2} + N - \frac{N}{w}\}. \tag{3.6}$$

This lower bound is achievable if the traffic is an all-to-all transmission, in which every transmitter/ receiver pair has a single packet to be transferred. The algorithms presented in [86] derive an optimal schedule for this case. For the case of an arbitrary traffic demand, the problem of achieving minimum schedule length is NP-hard [67]. Choi et al. [85] show that traffic can be scheduled with a schedule length that is no more than twice the lower bound. If $\delta < 1$, the factor of 2 can be reduced to $1 + \delta$.

In [63, 65, 73, 83] several heuristic approaches have been proposed. In the following we review two simple algorithms for the variable-frame problem in a star-coupled network with tunable transmitters and receivers.

3.3.1 Heuristic Approaches

We now review two reservation-based scheduling techniques for WDM star networks [87]: SEQSAM (SEQuential Scheduling AlgorithM) and BALSAM (BALanced Scheduling AlgorithM). The network architecture is based on a passive star topology composed of M nodes which are equipped with tunable transmitters and receivers capable of operating on C channels. Frame transmission includes a reservation phase, a schedule computation phase, and a data phase. During the TDM-based reservation phase, all the nodes broadcast their information (i.e. a control packet containing the destination identification and the requested packet size is sent to every node in the network). This can be done by tuning all the transmitters and receivers on a single channel. At the end of each reservation phase an $M \times M$ demand matrix D is available. During the schedule computation phase a transmission schedule is computed at every node. The scheduling problem is to specify the durations in which each transmitter and receiver should tune to a specific channel during the data phase.

SEQSAM allows receivers to tune on multiple channels during each transmission phase. But this technique has a very poor performance and is introduced in [34] primarily to show the performance improvement that can be achieved by the other algorithm, BALSAM. BALSAM restricts the receivers to tune on only one channel during each transmission phase. This restriction reduces the effect of tuning latency on the transmission time.

SEQuential Scheduling AlgorithM (SEQSAM)

SEQSAM groups the elements of the $M \times M$ demand matrix into groups of C elements, producing G sub-matrices. Each of the sub-matrices has at most C nonzero elements with no more than one nonzero element on any row or column. Therefore the lower bound on G is given by $(M^2 - M)/C$. SEQSAM implements a simple technique for obtaining the matrix decomposition, but its average time complexity is $O(M^3)$. Scheduling in SEQSAM takes place after grouping the elements and obtaining G sub-matrices:

$$D = D^1 + D^2 + D^3 + \cdots + D^G$$

The length of the transmission phase is the sum of the largest entries in the sub-matrices (D^i_{max}) defined as $\sum_{i=1}^{G} D^i_{max}$.

Example 1. Consider a star network with $M = 4$, $C = 2$, and the following demand matrix and its decomposition:

$$
\begin{aligned}
D = & \begin{bmatrix} 0 & 3 & 2 & 2 \\ 1 & 0 & 4 & 1 \\ 3 & 2 & 0 & 1 \\ 1 & 1 & 3 & 0 \end{bmatrix} \\
= & \begin{bmatrix} 0 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 2 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
+ & \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 3 & 0 \end{bmatrix}
\end{aligned}
$$

In this example the matrix is decomposed into $G = 6$ groups of $C = 2$ nonzero elements. Each sub-matrix is assigned a section of time slots on the transmission schedule. The length of each section is equal to the largest entry of the corresponding sub-matrix. On each sub-matrix the first channel is assigned to the first non-zero entry. The next channel is assigned to the next non-zero entry and so on. The schedule length (as we see in Figure 3.4) is 17 slots.

BALanced Scheduling AlgorithM (BALSAM)

This algorithm converts the $M \times M$ demand matrix to an $M \times C$ matrix by using the Modified Multi-FiT (MMFT) algorithm [88]. Then using an interval-based scheduling algorithm it assigns the channels to the transmitters. BALSAM attempts to reduce the schedule length by reducing the effect of tuning latency and the wasted resources during each frame. It has been shown in [84] that the time complexity of the Interval-Based Scheduling Algo-

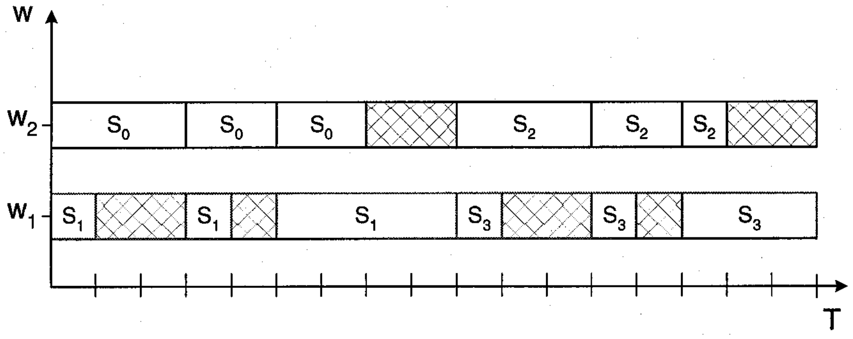


Fig. 3.4 Example 1: Allocation schedule of two available wavelengths (shown by two different colors) achieved by SEQSAM. S_i denotes the source node i. The schedule length in this example is 17 time slots.

rithm (IBS) is $O(MC^2K')$ where K' is the largest element in the $M \times C$ demand matrix. A simple description of this algorithm is as follows. At the first step the MMFT algorithm computes the column sums of the demand matrix and sorts the columns in descending order. Then it assigns each of the first C columns to each of the C channels and starts over with assigning the second C columns to the C channels and so on. So for $C < M$ several columns are given a common channel, and each receiver (corresponding to each column) has to tune on the selected channel. With this technique the load is almost equally distributed on the available channels, and we have an $M \times C$ demand matrix, whose (i, j)-th element represents the number of requested time slots from *node i* on *channel j*. The IBS algorithm for the $M \times C$ demand matrix keeps track of the available intervals on the channels. When a node request for a channel is considered, the algorithm tries to fit the request in the first available interval. If the interval is not sufficient to be allotted to the demand the next available interval is considered. The following example clarifies how this algorithm operates.

Example 2. Consider the demand matrix in the previous example. Using MMFT the 4×4 demand matrix is converted to the following 4×2 demand matrix which shows the demand of each transmitter for each channel. For more information regarding the MMFT algorithm refer to [87].

$$\begin{bmatrix} 4 & 3 \\ 5 & 1 \\ 1 & 5 \\ 3 & 2 \end{bmatrix}$$

Node 0 request: IBS first considers the request of *node* 1 for *channel* 1 which is 4 slots. Initially all of the slots are available. Therefore the first 4 slots are allotted to the first request of *node* 0. The next request to be considered is the *node* 0 request for *channel* 2. Since the transmitter has been tuned to *channel* 1 during the first 4 time slots, the first possibility for transmission on *channel* 2 is the interval [5,7].

Node 1 request: For the next request, *node* 1 on *channel* 1, the available intervals are $[5, \infty]$. Therefore the interval $[5, 9]$ is assigned to this node. Similarly the *node* 1 request for *channel* 2 has to be assigned based on a look up procedure which determines the first fit interval in the scheduling table. In this example the first slot on *channel* 2 suits this request. For the rest a similar procedure is performed. Figure 3.5 shows the final transmission schedule. The schedule length in this example is 15 slots.

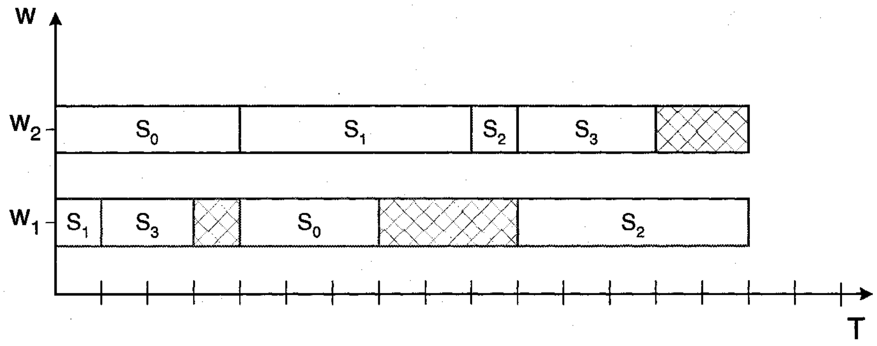


Fig. 3.5 Example 2: Allocation schedule of two available wavelengths (shown by two different colors) achieved by BALSAM. The schedule length in this example is 15 time slots.

Tuning Latency

This section considers the effect of tuning latency on the transmission time for the two proposed algorithms. Let δ_1 denote the number of slots required for tuning the transmitter between the channels, and δ_2 denote the tuning latency of the receivers. We define $d = \max\{\delta_1, \delta_2\}$. For SEQSAM there are G sub-matrices and transceivers tune to their assigned channels between the sub-matrices. Therefore the additional delay introduced by SEQSAM due to tuning latency is GT (Figure 3.6).

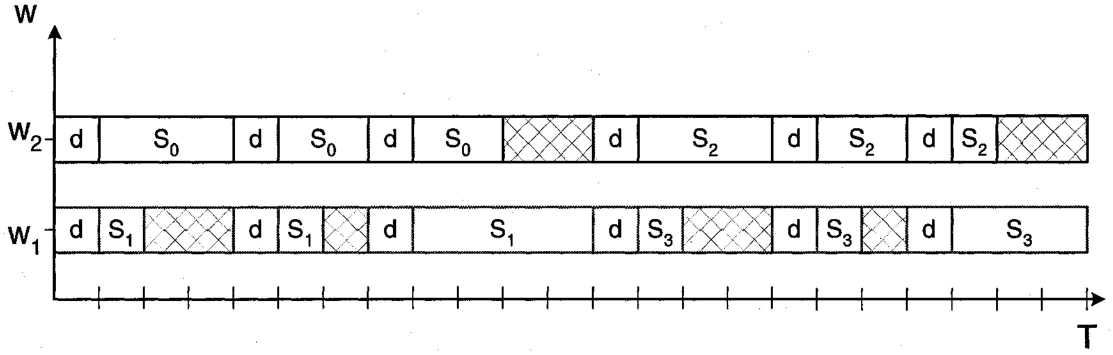


Fig. 3.6 Example 1: Allocation schedule of two available wavelengths (shown by two different colors/shadings) achieved by SEQSAM incorporating the tuning latency for $d = 1$. The schedule length in this example is 23 time slots.

BALSAM introduces less tuning latency to the transmission phase since the receiver assignment to the channels is static during each frame. The receivers can tune to their assigned channels as soon as the MMFT algorithm is performed to balance the requests. Moreover the transmitters may be able to tune their channels ahead of time, during the intervals the other transmitters are sending data. Figure 3.7 shows the effect of tuning latency on the schedule length in the BALSAM algorithm. The schedule length is 17 slots in this case.

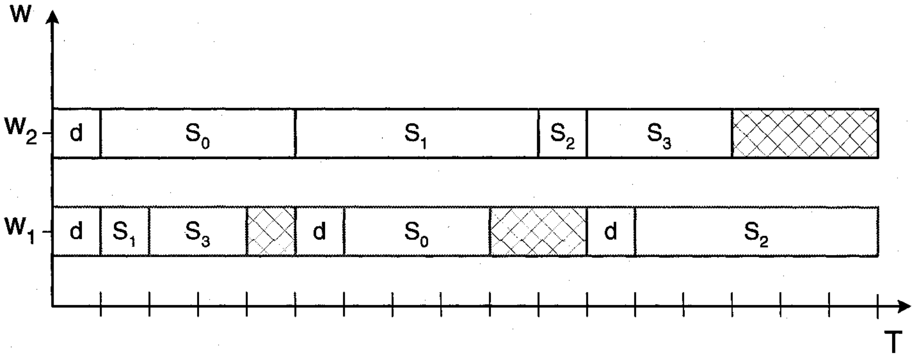


Fig. 3.7 Example 2: Allocation schedule of two available wavelengths (shown by two different colors/shadings) achieved by BALSAM incorporating the tuning latency for $d = 1$. The schedule length in this example is 17 time slots.

3.4 Fixed-Length Frame Scheduling for Broadcast WDM Transmission

In the Broadcast-and-select networks comprising the frames with a fixed length the aim is to minimize the number of packets that can not be transmitted in the scheduled frame. Formalization of the off-line scheduling problem considering a fixed frame length is as follows [31]:

> "Given a traffic matrix D whose elements D_{ij} specify the number of packets that must be transmitted from any source i to any destination j in a pre-specified time frame comprising F slots, find a time/wavelength assignment (satisfying the tuning delay constraints), that minimizes the number of packets that are not accommodated in the frame."

This formalization states that a new scheduling is obtained by re-allocating network resources for all end-to-end user traffic flows, in other words, any change in bandwidth requests results in rescheduling all of the connections.

Even though off-line algorithms can lead to optimal solutions, the time necessary for receiving the whole traffic demand before starting the calculation is high. In order to reduce the time between two consecutive transmission phases, on-line scheduling algorithms have been introduced in which the computation phase is overlapped with the reservation phase by starting calculation as soon as the first request arrives. Then the requests arriving later are assigned the free time-slots without re-allocating the already allocated requests for the current frame.

In order to reduce the complexity and the bandwidth devoted to signalling, the on-line algorithms can accommodate a *transparency constraint* in the scheduling problem: new requests may be accepted only if they do not affect existing allocations, otherwise they are refused. Therefore the on-line scheduling problem imposed for a fixed frame with the transparency constraint is formulated as follows [31]:

> "Given a time frame comprising F slots, in which a number of user-to-user transmission are allocated according to a known schedule, and given a matrix D_n of new requests or modifications of allocated requests, find a time/wavelength assignment, (satisfying the tuning delay constraints), that avoids modification in existing allocations (except for those resulting from D_n) and minimizes the number of packets of D_n that are not accommodated in the frame."

In comparison, the off-line algorithms for dynamic traffic are more efficient in terms of the utilization of the schedule they generate, but the computation time is high. The algorithms proposed in [63,65,73] compute static WDM/TDM schedules, which allocate the resources according to long-term bandwidth requirements. The algorithms are simple, but inefficient in the case of bursty traffic [31]. On-line algorithms usually have less computation time and lower efficiency, since the incremental nature of the algorithms does not provide in general an optimal schedule. In [31] Marsan et al. introduce an on-line algorithm which provides a tradeoff between simplicity and efficiency, assuming a slowly-varying traffic pattern. In the following section we review the scheduling algorithms proposed in [31].

3.4.1 Simple On-line Scheduling Algorithms for All-Optical Broadcast-and-Select Networks

Marsan et al. consider all-optical broadcast-and-select networks with a star topology which provides a number of slotted WDM channels for packet transmission [31]. Each node in the network includes a tunable transmitter and one fixed receiver. The range of the transmitters' tunability is sufficient for a full connectivity between each source/destination pair. A centralized controller provides time-slot assignment in a WDM/TDM frame considering long term bandwidth requests demanded by the users. Different strategies for on-line scheduling are proposed. The algorithms are executed periodically to re-compute the schedule in response to a change in the users' demands (i.e. a new request or a modification of an existing connection). When transparency is enforced, the existing connections in consecutive frames receive their previously assigned time-slots, and the new requests or the modified requests may occupy the free time-slots. The algorithms attempt to allocate the requests for each connection in contiguous slots in order to reduce the overhead due to tuning latency.

The slot allocation process for a new request of k slots to connection (i, j) starts by determining the wavelength on which a connection should be established. In the case of tunable transmitters and fixed receivers, the wavelength for connection (i, j) can be identified from the destination address. Then the algorithm searches for (i, j)-*eligible* slots on the destination wavelength, w_j. The (i, j)-eligible slots are defined as the set of free time-slots on wavelength w_j during which a transmitter is neither tuning nor transmitting on some other wavelength. The three different strategies presented in [31] use different criteria for selecting k slots among the (i, j)-eligible slots. In the first step the algorithms try to assign k consecutive slots to the request based on different criteria. If that is not possible, a second sequential search assigns k eligible slots to the demand on a first-fit basis, even though it is possible to apply a more sophisticated rule for the case that the request must be split. In the following we describe the different criteria for assigning the requests in the first step.

Algorithms Description:

Sequential Search (SS). The first strategy searches for the first k contiguous (i, j)-eligible time-slots, and assigns them to the request. The complexity of this algorithm is linear in the frame size F.

Best Fit Search (BFS). The second strategy searches for all of the sequences consisting of at least k contiguous (i, j)-eligible slots. Among these the shortest sequence is chosen. Similar to the first algorithm, the complexity of this algorithm is linear in the frame size, but it needs more memory for storing the information for selecting the shortest sequence.

Minimum Cost Search (MCS). The third strategy defines a reward function on the space of all free slots for each wavelength. For a demand of k slots on wavelength w_j a search procedure determines the set of k consecutive slots for which the global reward function after this assignment is maximum.

Denote by $C_{t,w_j} = N_{fs}(t) + N_{fw}(t)$ the reward associated with the (i, j)-eligible slot t on wavelength w_j, where N_{fs} is the number of free sources, and N_{fw} is the number of free wavelengths at slot t over all available wavelengths. Let F_{ij} be the number of (i, j)-eligible slots (on wavelength w_j), and S be the set of all contiguous source-free slots, the sequences on which source i is neither tuning nor transmitting. The reward associated with each sequence $s \in S$ of contiguous source-free slots is denoted by $\chi(s)$, an increasing function with s (e.g., $\chi(s) = 1.5\|s\|^{1.2}$, where $\|s\|$ indicates the number of slots in sequence s). Let W be the set of all free sequences on wavelength w_j, denoted as wavelength-free slots at this wavelength. Denote by $\Psi(f)$ the reward associated with each sequence $f \in W$ of contiguous wavelength-free slots. The global reward function associated with source i and wavelength w_j is defined as:

$$M(i, w_j, s, f) = \sum_{t \in F_{ij}} C_{t,w_j} + \sum_{s \in S} \chi(s) + \sum_{f \in W} \Psi(f). \tag{3.7}$$

When a new request must be assigned on wavelength w_j, a sequence of eligible slots is temporarily allocated to the request and M is computed. After computing M for all possible allocations, the MCS algorithm chooses the allocation which provides the maximum value of M. In other words this algorithm implements a search procedure which allocates a contiguous sequence of slots with the lowest impact on the value of the global reward function after the allocation.

3.5 Summary

In this chapter we reviewed the scheduling problem with respect to the type of underlying connections and scheduling frame. Variable-length frame scheduling has been studied in depth for point-to-point satellite systems and broadcast-and-select WDM networks. The main objective in designing a variable-length frame schedule is to minimize the overall transmission time, which is composed of the traffic transmission time and the time spent on switch reconfigurations. It has been proved that the variable-length frame scheduling problem is *NP*-hard, while minimizing the traffic transmission time is solvable in polynomial time. This chapter presented various algorithms for variable-length frame scheduling.

Slot-by-slot scheduling has been studied for input-queued switches and WDM star networks with point-to-point transmission. Fixed-length frame scheduling has been used in broadcast-and-select WDM systems, but there has been limited study of fixed-length frame scheduling for point-to-point transmission. The fixed-length frame scheduling as opposed to the variable-length frame scheduling offers simplicity to the network, and is more manageable with respect to traffic prediction and demand calculations. Estimating the length of the frame for the variable-length frame scheduling adds complexity and error to the network. For large networks synchronization at the frame level will be much easier using fixed-length frame scheduling. In addition, the fixed-length frame scheduling can support long-time transmission of delay sensitive applications. In the following chapter we investigate this problem and propose several scheduling algorithms.

Chapter 4

Frame-based Scheduling Algorithms

This chapter presents the frame-based scheduling algorithms that we designed for the agile-all photonic network (AAPN). Note that these designs are applicable to any kind of single-hop communication network with point-to-point transmission, for example, satellite systems. Recall that the AAPN architecture is an overlaid star-topology of N edge nodes that operates over multiple wavelengths [2]. It permits each node to transmit to one destination node and receive from one source node simultaneously *on each wavelength*. We consider that (flow-based) load balancing has been conducted to divide incoming traffic amongst the various stars. The remaining task is to schedule the traffic for each star.

This chapter is organized as follows: in Section 4.1 we propose a formulation for the fixed-length frame scheduling problem, state its relationship with the variable-length frame scheduling problems for star topologies, and present results concerning NP-hardness of the problem. In Section 4.2 we introduce Minimum Cost Search (MCS), our frame-based greedy algorithm and evaluate its performance. In Section 4.3 we proposed two novel frame-based scheduling algorithms which provide full utilization and fair allocation of time slots. We then examine their properties and evaluate their performance. In Section 4.4 we propose Minimum Rejection Algorithm (MRA) that generates a schedule with minimum total number of rejections for the case of inadmissible traffic. The performance of this algorithm is compared with the previous algorithms. Section 4.5 summarizes this chapter and draws conclusions.

4.1 Fixed-Length Frame Scheduling for Point-to-Point Transmission

We consider a frame of length F time slots with W available wavelengths, such that there are $L = FW$ slots for each destination node available for allocation. Herein we focus on the case where $W = 1$ for clarity, but the algorithms and results are easily extended to the case of $W > 1$. For the latter case, it is possible to schedule all of the wavelengths within a single frame of length FW. However this is not an optimum way of scheduling several wavelengths. To obtain a more balanced schedule, we need to divide the load amongst the available wavelengths and then schedule every wavelength separately. In this case there is not any correlation between the loads assigned to the transmitters/receivers of each wavelength, and the problem is reduced to the case of $W = 1$.

Each edge node signals its bandwidth request for every destination node. The scheduler forms a demand matrix D, where D_{ij} is the number of slots requested by source node i for destination node j during the next fixed-length frame. Recall from chapter 3 we have the following definitions for the *line-sums* of the demand matrix. The *row-sum*, $r_i(D) = \sum_{j=1}^{N} D_{ij}$, is the total demand at source i, and the *column-sum*, $c_j(D) = \sum_{i=1}^{N} D_{ij}$, is the total demand for destination j. It is important to achieve zero rejection if the demand is *admissible.*

Definition 7. *A demand matrix D is* admissible *if*

$$\max\{\max_i\{r_i(D)\}, \max_j\{c_j(D)\}\} \leq L, \tag{4.1}$$

where L is the frame-length, and $r_i(D)$ and $c_j(D)$ are the i-th row-sum and j-th column-sum of the demand matrix, respectively.

Our aim is to devise a schedule S such that the element S_{jk} identifies the source node allocated to the k-th time slot associated with destination j in the frame. The schedule should minimize the number of rejections $REJ(S, D, L)$ whilst also attempting to minimize the number of times that the switch must reconfigure, $N_s(S)$. A switch reconfiguration occurs between two consecutive time slots k and $k + 1$ if the allocated source node to any destination j is altered; $N_s(S)$ counts the number of switch reconfigurations in the entire schedule. Reducing the number of switch reconfigurations is important since the amount of consumed power varies inversely with the switching period defined as:

$$P = (C_{cap}V^2)/T_p, \tag{4.2}$$

where C_{cap} is the capacitance of the switch, V is the voltage and T_p is the switching period [89]. This formula gives the dynamic power dissipation when driving a capacitive load (e.g. a switch modeled as a capacitor). Assuming that the voltage in formula 4.2 is fixed, the power consumption in an optical switch only depends on the switching period. Therefore the more frequent the switching the more the power consumption.

The number of rejections is defined as:

$$REJ(S, D, L) = \sum_i \sum_j \max(0, D_{ij} - \sum_{k=1}^{L} \mathbb{I}[S_{jk} = i]), \tag{4.3}$$

where $\mathbb{I}$ is the indicator function. We can define an objective function (the cost of transmission) as:

$$C(S, D, L) \; = \; REJ(S, D, L) \; + \; g \; . \; N_s(S) \; , \tag{4.4}$$

where g is a constant that determines the relative importance of reducing the number of switch reconfigurations.

We identify a scheduling problem that addresses bandwidth allocation in an AAPN:

PROBLEM 2: Solve the following optimization problem for a frame of fixed length L with $C(S, D, L)$ defined by (4.4) to identify a frame schedule.

$$S_2^* = \arg\min_S C(S, D, L) \tag{4.5}$$

4.1.1 Relationship to Variable-Length Frame Scheduling

The *EXACT* algorithm, presented in Chapter 3, addresses schedule design for variable length frames, primarily for the case of negligible τ, and achieves a minimum traffic transmission time, $T_{x_{min}}$. Thus in the case of admissible demand matrices, the *EXACT* algorithm generates a schedule S that has length less than L. The *EXACT* algorithm is an iterative procedure that repeatedly performs maximum cardinality bipartite matching (MCBM) to obtain the schedule (see Section 3.1.2 for a review). It lies at the heart of the algorithms we present for the case of fixed-length frames.

We establish two results concerning the complexity of *PROBLEM 2*:

Claim 3. *If the demand matrix D is admissible and contains no zero entries (for an $N \times N$ switch and frame of length L) then the EXACT algorithm provides a solution S_E to PROBLEM 2 such that $C(S_E) < C(S_2^*) + g(N^2 - 3N + 2)$.*

Proof. Since the demand matrix is admissible, $T_{x_{min}} < L$. Hence the schedule devised by *EXACT* results in zero rejections, $REJ(S, D, L) = 0$. *EXACT* ensures that the number of switch reconfigurations in this solution is less than $N^2 - 2N + 2$. The minimum number of switch reconfigurations for any schedule under the constraint of no zero-entries in the demand matrix is N [80]. Hence the maximum discrepancy is $N^2 - 3N + 2$. □

Theorem 1. *For large g, such that $g > \max(||D||_1 - L, 0)$, where $||D||_1 = \sum_i \sum_j D_{ij}$, PROBLEM 2 is reduced to the problem of minimizing $REJ(S, D, L)$ subject to the constraint that $N_s(S)$ is minimized. For this range of g, PROBLEM 2 is NP-hard.*

Proof. Consider the set of schedules that achieve minimum $N_s(S) = N_s^*$ and label the schedule within this set that achieves minimum rejection S_a. The minimum achievable rejection is no larger than $REJ(S, D, L) = \max(||D||_1 - L, 0)$, where $||D||_1 = \sum_i \sum_j D_{ij}$ (at least one demand element must be satisfied each time-slot). Thus $C(S_a) \leq \max(||D||_1 - L, 0) + gN_s^*$. Now consider schedules that increase the number of switch reconfigurations to $N_s(S) = N_s^* + 1$ and suppose that one of these, S_b, achieves zero rejection, so that $C(S_b) = g(N_s^* + 1)$. The differential in cost $C(S_b) - C(S_a) \geq g - \max(||D||_1 - L, 0)$. If $g > \max(||D||_1 - L, 0)$, then this difference is strictly positive and any schedule solving *PROBLEM 2* lies within the set of schedules that achieve minimum N_s.

In order to prove that the problem is *NP*-hard for this range of g, we consider *PROBLEM 1*, which for very large values of τ is reduced to minimizing the schedule length subject to the constraint that N_s is minimum. Gopal et al. prove that this problem, which they refer to as the MINSWT problem, is NP-complete [67].

Suppose we had a deterministic polynomial algorithm called *solve-G(D,L)* that could solve *PROBLEM 2* for the identified range of g for demand matrix D and a frame of length L. We could then define the algorithm Solve-MINSWT (Algorithm 2).

Upon termination of this algorithm, the identified schedule S is guaranteed to have the minimum number of switch reconfigurations (as argued above). Since it is also the minimum length schedule that achieves $REJ(S, D, L) = 0$ it is also a solution to *PROBLEM 1* (Chapter 3, p. 35) and hence the MINSWT problem. Algorithm 2 is thus a deterministic

Algorithm 1 Solve-MINSWT

```
$L = 1$;
$S =$ solve-G($D$,$L$);
while $REJ(S, D, L) > 0$ do
  $L = L + 1$;
  $S =$ solve-G($D$,$L$);
end while
```

polynomial algorithm to solve the MINSWT problem. Therefore, solving *PROBLEM 2* for the considered range of g is as hard as solving MINSWT (and any other problem in NP) and hence is NP-hard. □

In a practical scenario, although it is desirable to reduce power expenditure by minimizing the number of switchings, minimizing the number of rejections is far more important. Hence we address the scheduling problem (*PROBLEM 2*) introduced in Section 4.1 when g is small.

In the following we begin by describing a new heuristic algorithm for the AAPN architecture with fixed-length frame schedule. To show the efficiency of the fixed-length frame scheduling in wide-area networks we compare the performance of this algorithm with a slot-based scheduling algorithm. Then we propose several optimal scheduling algorithms with different objectives such as achieving max-min fairness and minimum number of rejections. These algorithms are optimal in the sense that they guarantee there is no rejection as long as the demand is admissible.

4.2 Minimum Cost Search Algorithm

In this section we propose the Minimum Cost Search (MCS) algorithm. In order to reduce signalling overhead and to reduce scheduling complexity, the algorithm satisfies the transparency property defined in Section 3.4. This requires that the scheduling is only modified for new requests or tear-downs (if D_{ij} decreases or increases), so that the already established connections occupy the same slot numbers in a frame as they did in the previous frame.

The Minimum Cost Search algorithm we propose does not achieve optimal utilization, because it does not consider the global allocation problem; it is a greedy algorithm which allocates requests sequentially on a single time slot basis, hoping that the obtained schedule at the end of assignments is close to optimal. The algorithm operates by repeatedly visiting the (i, j) entries in the traffic demand matrix D in a round-robin fashion; at each visit, if the requested number of slots has not yet been assigned, the algorithm attempts to allocate a single time slot to the (i, j) request. If there are unallocated time slots in the schedule they are given to the source-destination pairs using a similar procedure. The round-robin allocation results in an approximately fair assignment of slots to each pair.

In order to determine which slot to allocate to the request, we define a *cost* for the allocation of a (i, j) source-destination pair to a time slot pair t_k for $k \in \{1, \dots L\}$. This cost is determined entirely by the extant, partial frame schedule. The cost function is:

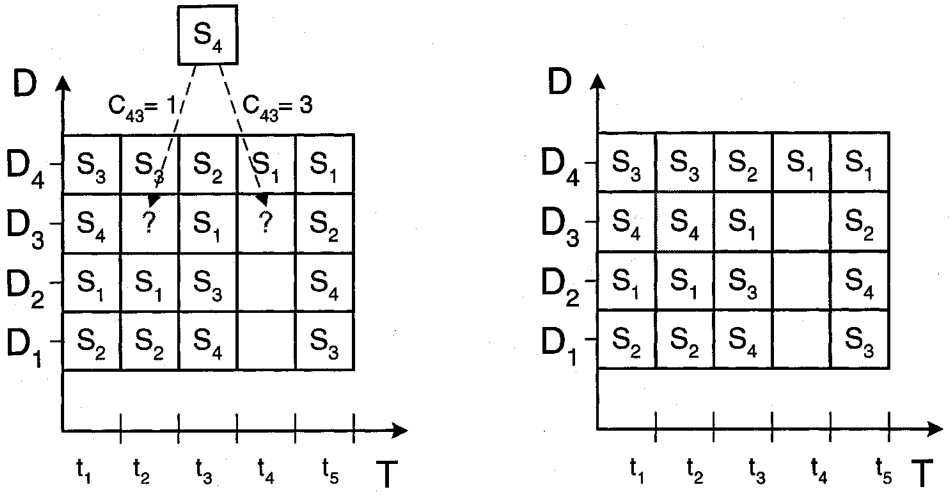
$$C_{ij}(t_k) = N_{fs}(t_k) + \lambda K_{ij}(t_k), \tag{4.6}$$

where $N_{fs}(t_k)$ is the number of free sources at this time slot, i.e., the number of sources not transmitting to any other destinations, λ is a small positive constant, and $K_{ij}(t_k) = \{0, 1, 2\}$ is the number of additional switching operations at the boundaries of each time slot that the core switch must perform to accommodate the allocation. The motivation behind this cost function is as follows:

- The first term represents the current flexibility of that time slot (the number of free sources for future allocation) and reflects the desirability of retaining flexibility by allocating demands to the most constrained slots where possible.

- The second term reflects the desirability of minimizing the power consumption of the optical switch, which is partially determined by the number of switching operations that it must perform during each frame.

The scheduling of a single (i, j) time slot request is performed by first identifying the (i, j)-*eligible* slots in the frame, which are defined as the free time slots during which i is not transmitting to any other destination and j is not receiving from another source. The cost $C_{ij}(t_k)$ of each of these eligible time slots is evaluated, and the demand is assigned to the slot incurring minimum cost. Assigning the least costly slots to a request leaves the free slots with more potential of supporting any possible request in consequent allocations. In the case of ties, the demand is assigned to the earliest slot and the lowest wavelength (assuming wavelengths are ordered in some arbitrary fashion). Deallocation is implemented by a reverse procedure, in which we seek and release the most costly currently-allocated time slot. This algorithm has a worst case time complexity of $O(N.L^2)$, since there are at



	t_1	t_2	t_3	t_4	t_5
D_4	S_3	S_3	S_2	S_1	S_1
D_3	S_4	?	S_1	?	S_2
D_2	S_1	S_1	S_3		S_4
D_1	S_2	S_2	S_4		S_3

	t_1	t_2	t_3	t_4	t_5
D_4	S_3	S_3	S_2	S_1	S_1
D_3	S_4	S_4	S_1		S_2
D_2	S_1	S_1	S_3		S_4
D_1	S_2	S_2	S_4		S_3

Fig. 4.1 Allocation of one time slot to a request from source node 4 for transmission to destination node 3 using MCS algorithm. The costs of the two eligible time slots, t_2 and t_4 are $C_{42}(t_2) = 1$ and $C_{42}(t_4) = 3$ as shown in the left diagram. t_2 is assigned to the request as shown in the right diagram.

most $N.L$ allocations in each frame each of which is obtained by a search procedure over at most L time slots.

Figure 4.1 shows a fixed length frame schedule for a network of 4 edge nodes, and $\lambda = 0$. To assign a time slot to a request from source node 4 to destination node 3, the MCS algorithm finds the eligible time slots on the frame of destination 3, which are slots 2 and 4. Amongst these two time slots the one with the least cost calculated from equation (4.6) is given to source node 4. In this example the least costly time slot is t_2.

4.2.1 Performance of the MCS Algorithm

In this section we examine the performance of our scheduling algorithm in terms of queuing delay and packet loss. We compare the performance of our algorithm with that of the slot-based algorithm described in Section 3.2 that was proposed by Liu et al. in [15]. We perform the experiments for different propagation delays (i.e. metropolitan and wide-area networks) over a wide range of offered loads, 10% to 90% of the link capacity. Then we examine the effect of frame length on queuing delay.

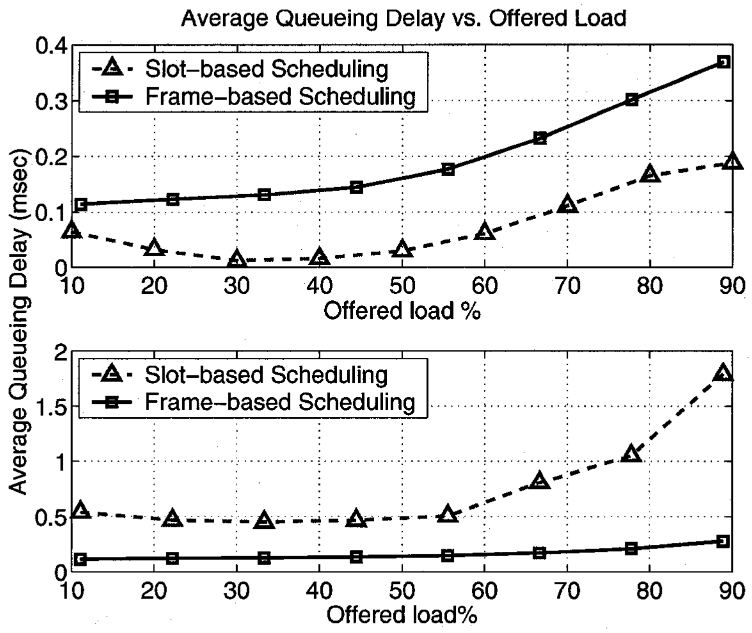


Fig. 4.2 Average service delay as a function of offered load in uniform traffic scenario. Top panel: 1 msec propagation delay. Bottom panel: 5 msec propagation delay. Here service delay is total end to end delay less propagation delay, and the propagation delay is from ingress edge node to egress edge node.

A: Comparison with a Slot-based Algorithm

In this section we show the results of simulations of a joint work with Liu et al. on a comparison of frame-based and slot-based scheduling approaches [15]. The simulations are performed using OPNET Modeler [90]. We performed simulations on a 16 edge-node star topology network. The links in the network have capacity 10 Gbps and the one-way propagation delay between each edge node and the optical switch is 5 msec. A time slot is of length 10 μsec, and a frame has a fixed length of 1 msec (or 100 slots). Every experiment was run for a duration of 0.5 sec (equal to 500 frame durations) and the results were averaged over 5 repetitions of the simulations. The virtual output queues in the simulations have fixed buffer size (90000 packets). Whenever the buffer is full, arriving packets are dropped. We used the average of the traffic arrivals over the past 10 frame durations to form the prediction of the demand matrix D.

In the simulations, traffic sources inject traffic at rates up to 10 Gbps into the edge nodes. The arrival distribution of the data packets is Poisson and the size distribution is exponential with mean size of 1000 bits. Then multiple (approximately 100) packets are wrapped into one optical slot. We investigated two cases of destination distributions: (i)

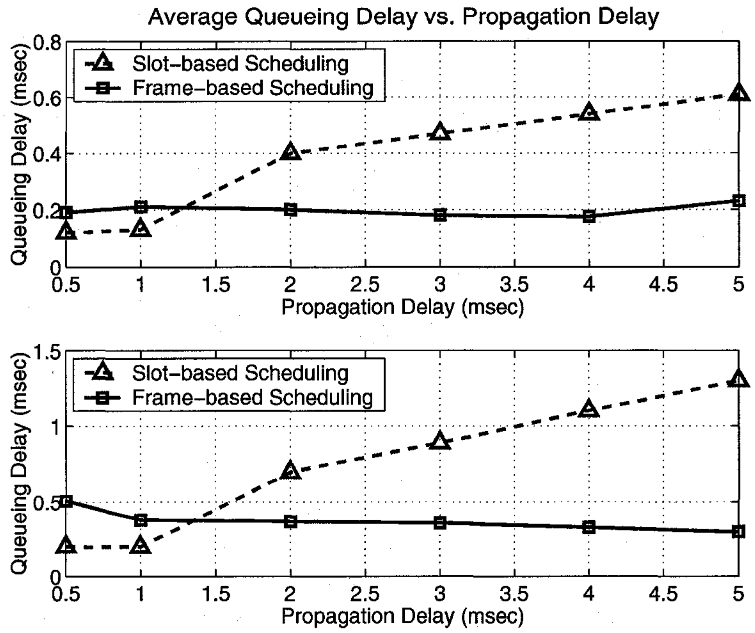


Fig. 4.3 Average service delay as a function of propagation delay in uniform traffic scenario. Top panel: Offered load of 60%. Bottom panel: Offered load of 90%.

a uniform case, where sources send equal amounts of traffic to each destination, and (ii) a non-uniform case, where all destinations receive an equal amount of traffic on average, but each source sends 5 times as much traffic to one destination.

Figure 4.2 shows the average service delay against the offered load, from 10% to 90% link capacity. For the slot-based approach, higher delay is observed at an offered load around 10%. The reason is that, due to the very low offered load, it takes longer to reach the threshold for issuing a request. Figure 4.3 compares average service delay as a function of propagation delay for the frame-based and slot-based scheduling methods. The delay components are propagation delay, transmission delay, and queuing delay. For simplicity, we call the latter two components *service delay.*

The frame-based scheduling method is less sensitive to propagation delay because the round trip time required by the slot-based scheme for the request-grant-transmit process is avoided. In the frame-based scheme the edge nodes send requests for the predicted traffic demand in advance of the traffic arrivals, thereby reducing the delay associated with the grant and request processes. On the other hand the frame-based method may reserve a slot which is unused or under utilized if the actual traffic arriving is less than that forecast. The accuracy of traffic prediction and the resulting efficiency of the frame-based scheme

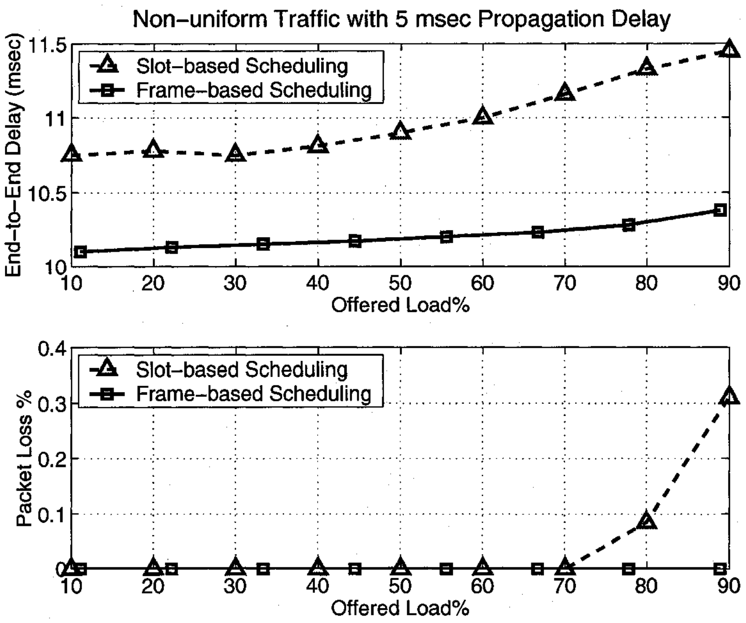


Fig. 4.4 Performance of the scheduling algorithms with non-uniform traffic as a function of offered load with a propagation delay of 5ms. Top panel: End-to-end delay. Bottom panel: Packet loss ratio.

depends upon the stability of the traffic demand. The frame-based method on the other hand incurs a delay associated with transmitting a frame. The scheduler does not take into account the arrival distribution of the packets and schedules the traffic based on the average traffic load during one frame.

Accordingly, one would anticipate a "break-even" distance where the two methods achieve equal mean delay performance. Below this critical distance the slot-based scheme yields lower delays and would appear suitable for metropolitan-area networks (MANs) and perhaps regional networks, while the frame-based scheme is more attractive for networks with a large diameter such as in wide-area networks (WANs). For the specific parameter settings and traffic scenario examined through simulation, Figures 4.2 and 4.3 indicate that this critical network radius is approximately 270 km.

For the uniform traffic demand scenarios no buffer overflow occurred during the simulation time. For the non-uniform traffic scenario, as shown in Figure 4.4, buffer overflow or blocking arises at high traffic loads. Accordingly, by appropriately provisioning link capacity and buffer capacity, high utilization is possible with acceptably low loss and end-to-end mean delay and delay variation or jitter. Note that both scheduling methods adapt to the non-uniform traffic demand with only marginal loss in traffic handling efficiency.

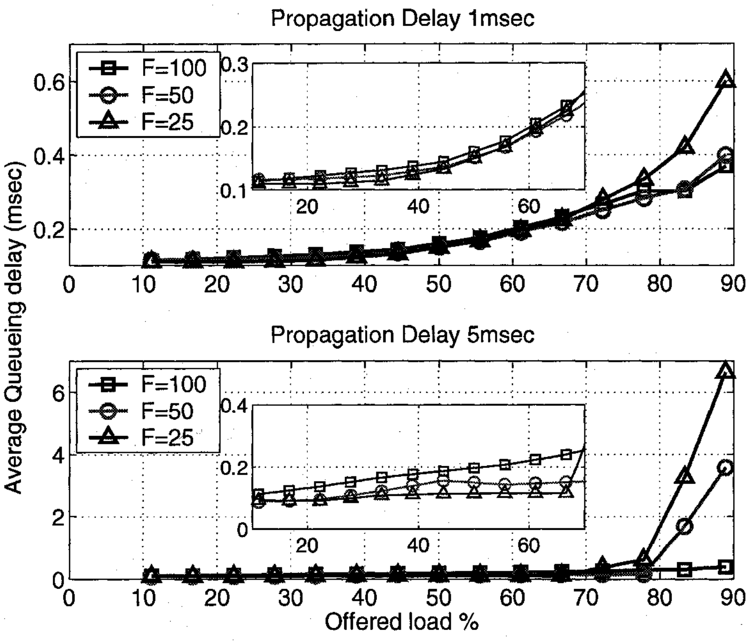


Fig. 4.5 Average service delay as a function of load ranging from 10% to 90% link capacity with different frame durations ranging from 25 to 100 time slots in the uniform traffic scenario. Top panel: Metropolitan-area network with propagation delay of 1 msec between the edge nodes and the core switch. Bottom panel: Wide-area network with propagation delay of 5 msec. The zoomed-in figures show the results for lower loads ranging from 10%-70% of link capacity.

B: Effect of Frame length on Queuing Delay

In this set of experiments we show how the length of the frame can affect the waiting times of the queues. Since there are 16 edge nodes, each of which transmits to at least 15 other edge nodes, we do not consider the case that the frame length is very small (i.e., smaller than 15 time-slots). A scheduler with a very small number of time slots produces too much unfairness, as there are not enough time slots to be distributed amongst all of the requests. If there is not any record of the rejections in a frame the scheduler alone (without memory of rejections) is not able to avoid starvation of some source-destination pairs in the consequent frames.

Figure 4.5 shows the average queuing delay for the network setup explained in experiment A for the case of uniform traffic. The top panel shows the results for a metropolitan-area network, when propagation delay between the edge nodes and the core switch is 1 msec and the bottom panel shows the results for a wide-area network with propagation delay of 5 msec. As the figure shows decreasing the frame length improves the performance

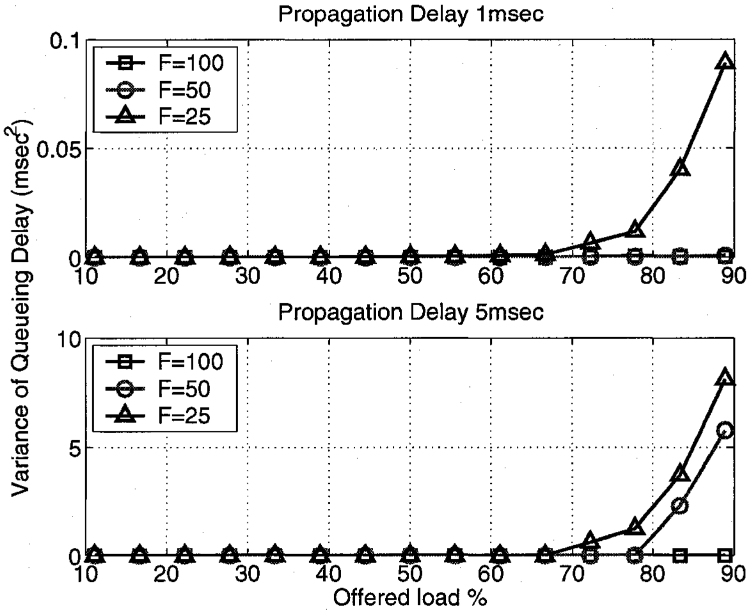


Fig. 4.6 Variance of the queuing delay as a function of load with different frame durations ranging from 25 to 100 time slots in the uniform traffic scenario. Top panel: Metropolitan area network with propagation delay of 1 msec between the edge nodes and the core switch. Bottom panel: Wide-area network with propagation delay of 5 msec.

for low offered loads. However the impact is inverse when the offered load is high. When the load is high the schedulers with smaller buffer lengths have larger queuing delays.

The improvement of waiting times in low loads with small number of time slots is mainly due to the scheduler's ability to adapt to the traffic load faster when the frame length is smaller. On the other hand the performance degradation of a small frame in higher loads arises because the higher loads benefit from extra time slots much more than the lower loads. The higher load connections almost always have packets available for transmission so that extra time slots are usually used to transmit a considerable number of packets. For larger frame lengths there are a large number of extra time slots to be spread amongst source-destination pairs. For a given load the proportion of extra time slots is independent of frame-length, so there are fewer free time slots per frame when the frame is small. When the load is high, this number is small, and the scheduler cannot distribute the slots fairly amongst source-destination pairs. Since there is no control or record of which connection benefited and which did not, the unfairness (i.e., uneven allocation of the extra time slots) can be repeated in several consecutive frames, causing starvation of some queues, and increasing the overall queuing time. Figure 4.6 shows the variance of the waiting times

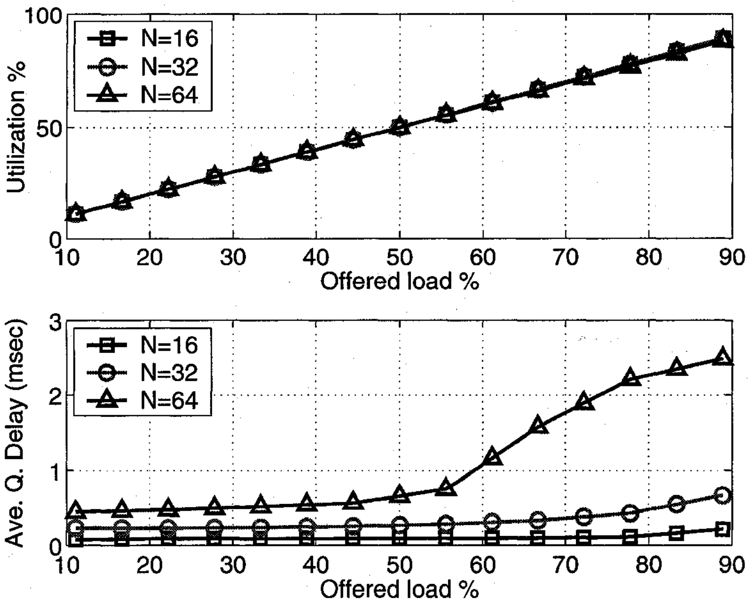


Fig. 4.7 Network performance with uniform traffic as a function of offered load with a propagation delay of 5ms for different number of edge nodes. Top panel: Utilization. Bottom panel: Queuing delay.

of different source nodes. As the figure shows shorter schedules induce higher variance in higher loads compared to the longer schedules.

C: Scalability Analysis

In this experiment we investigate the effect of increasing the number of edge nodes on the performance. We considered a propagation delay of 5 msec and frame length of 100 time slots (1 msec). When the network size increases the resources are shared amongst larger number of edge nodes, and the resultant average waiting time is larger. To explain this in more details consider a small network of 2 edge nodes and a frame of 4 time slots (e.g. every edge node receives two non-consecutive time slots) and compare with a network of 4 nodes and a frame of 4 time slots (every edge node receives one time slot). It is easily understood that the network of 2 edge nodes experiences smaller queuing delay; it is possible for the schedule to transfer the packets of every edge node twice per frame, and so the maximum queuing delay experienced by these packets is half a frame duration. While for the network of 4 edge nodes the maximum queuing delay is 1 frame duration.

Figure 4.7-top panel compares utilization for networks of 16, 32 and 64 edge nodes and uniform traffic. As the figure shows the network achieves similar utilizations for different number of edge nodes. The bottom panel compares the average queuing delays. For lower offered loads the queuing delay multiplies by a factor close to 2 (and 4) for 32 (and 64) edge nodes. Based on the discussion presented above this result is not surprising. For higher loads the queuing delay increases dramatically because the frame length is too small to support the increased number of nodes. Similar to what we observed in experiment B when the extra time slots are not fairly distributed amongst the edge nodes, unfairness occurs and queuing delay grows.

4.3 Fair Algorithms

In this section we introduce a novel scheduling algorithm, Fair Matching Algorithm (FMA), which maximizes the usage of bandwidth and for the case of admissible traffic provides zero rejection. This algorithm minimizes the maximum percentage rejection experienced by any demand, while providing weighted max-min fair allocation of the extra bandwidth. A similar approach can be conducted to achieve max-min fair allocation of the bandwidth, described as Equal Share Matching Algorithm (ESA).

The scheduling problem which will be addressed in this section is defined as:

PROBLEM 3: For an admissible demand matrix D and frame of length L, generate a schedule S that achieves zero rejection, $REJ(S, D, L) = 0$, and allocates spare capacity in the network to the connections in a (weighted) max-min fair manner.

4.3.1 Terminology and Definitions

We now define some terminology that will be used throughout this section and recall some definitions. We denote the line-sum of line ℓ of the demand matrix D by LS_ℓ. Note that line ℓ consists of a set of source-destination demands (connections). Each of these connections belongs to two lines (a row and a column). The i-th row represents a link from source i to the optical switch at the core, and the j-th column represents the link from the core to destination node j.

For an inadmissible demand matrix, we denote the set of overflowing rows of the demand matrix (rows with $r_i(D) > L$) as O_r, and the set of overflowing columns ($c_j(D) > L$) as O_c. The set of overflowing lines, $O_\ell = \{\ell : \ LS_\ell > L\}$ is the union of O_r and O_c. We define a *critical connection*, or critical demand element, as any demand entry D_{hp} such that $h \in O_r$

and $p \in O_c$. The remaining entries constitute *non-critical* connections/demands.

We now recall the definitions of *feasibility* of rate allocation and *weighted max-min fairness* [91,92].

Definition 8. Feasibility*: Consider an arbitrary network as a set of links $\mathcal{L}$ where each link $\ell \in \mathcal{L}$ has a capacity $C_\ell > 0$. Let $\{1, \cdots, \zeta\}$ be the set of connections in the network. Let D_u be the demand (request) of connection u and v_u be its assigned rate. We call a rate allocation $\{v_1, v_2, \cdots, v_\zeta\}$* feasible, *when for every link ℓ we have:*

$$\sum_{u \in H_\ell} v_u \leq C_\ell \quad \forall \ell \in \mathcal{L}. \tag{4.7}$$

Definition 9. Weighted max-min fairness*: Let $\omega_u(v_u)$ be an increasing function representing the weights assigned to connection u at rate v_u. An allocation $\{v_1, v_2, \cdots, v_\zeta\}$ is* weighted max-min fair *if for each connection u any increase in v_u would cause a decrease in transmission rate of connection z satisfying $\omega_z(v_z) \leq \omega_u(v_u)$. The special case of max-min fairness is obtained by $\omega_u(v_u) = v_u$.*

Definition 10. Bottleneck Link*: Given a feasible rate vector v and a weight vector ω, we say that link ℓ is a* bottleneck link *with respect to $(v\ ,\ \omega)$ for a connection u crossing ℓ, if $C_\ell = \sum_k v_k \triangleq F_\ell$ and $\omega_u \geq \omega_k$ for all connections k crossing ℓ.*

Lemma 1. *A feasible rate vector v with weight vector $\omega = \{\frac{v_u}{R_u}\}$ is weighted max-min fair if and only if each connection has a bottleneck link with respect to $(v\ ,\ \omega)$.*

See the Appendix (Section A.1) for a proof.

4.3.2 Fair Matching Algorithm (FMA)

Based on the discussion in Section 4.1.1, when the demand is admissible the *EXACT* algorithm generates a schedule S that has length less than L. This schedule can be used as a part of our fixed-length frame schedule, but the unassigned time slots should be given to the active connections in a fair manner. When the demand matrix is inadmissible, the schedule determined by the EXACT algorithm must be truncated after L time slots. This can lead to starvation of some source-destination traffic, and result in unfairness (such as substantially different average service times for traffic arriving at different nodes). Therefore we need to modify the demand matrix obtained originally from request prediction, such that all of the line-sums of the modified demand matrix equal L. This is done using the FMA algorithm.

If the demand matrix is admissible, FMA incrementally assigns additional demand to all elements until one of the links reaches capacity (its line-sum is equal to L). At that point, the demand elements contributing to that line are clamped. Extra demand is then gradually added to the remaining elements in the matrix until another link (line) reaches its capacity and it too is clamped. This procedure, referred to as the *water-filling* procedure, repeats until all lines have reached capacity. FMA assigns extra capacity *in proportion to the original demand.*

This algorithm can be implemented by processing one line at a time. We first choose the most constrained line (the line that would reach its capacity first under the water-filling procedure) and increase its demand to capacity. Then we choose the next most constrained line and increase its demand to capacity. We repeat until all lines have reached capacity.

A similar procedure can be used for the case of an inadmissible demand matrix. In this case FMA identifies the most overloaded line and reduces the demands on that line such that they sum to capacity (L). Demand reduction is proportional to the original demand, i.e. each adjusted demand experiences the same *percentage reduction.* In subsequent iterations, FMA identifies the next most constrained line, taking into account the effect of any previous adjustments, and clamps its demand to capacity. It repeats the process until no lines exceed capacity. When there are both overloaded and under-utilized lines, the overloaded lines are adjusted first.

Here we describe how FMA treats demands belonging to the adjustable lines in the set $U_\ell = \{\ell : \quad LS_\ell(0) \neq L\}$, where $LS_\ell(0)$ is the line sum of line ℓ at the beginning of calculations. We define $\mathcal{A}_D \subseteq U_\ell$ as the set of unmodified lines and $\mathcal{B}_D \subseteq U_\ell$ as the set of modified lines. Initially $\mathcal{A}_D$ contains all lines in U_ℓ and $\mathcal{B}_D$ is empty. Similarly, we define a_ℓ as the set of unmodified demands in line ℓ and b_ℓ as the set of modified demands. Initially, a_ℓ contains all the demands and b_ℓ is empty. In each iteration we adjust the unmodified demands in line ℓ from the following:

$$D'_{ij} = D_{ij} \times \frac{L - S_{b_\ell}}{S_{a_\ell}} \qquad \forall\ (i,j) \in a_\ell, \tag{4.8}$$

where $S_{a_\ell} \triangleq \sum_{(i,j)\in a_\ell} D_{ij}$ and $S_{b_\ell} \triangleq \sum_{(i,j)\in b_\ell} D'_{ij}$. We always have $S_{a_\ell} + S_{b_\ell} = LS_\ell$. Note that when demand D_{ij} belongs to an overloaded line, $\frac{L-S_{b_\ell}}{S_{a_\ell}} < 1$, and when D_{ij} belongs to an under utilized line $\frac{L-S_{b_\ell}}{S_{a_\ell}} > 1$. Define for each of line in $\mathcal{A}_D$ the value $G_\ell \triangleq \frac{L-LS_\ell}{S_{a_\ell}}$.

Algorithm 2 FMA

```
while $\mathcal{A}_D \neq \emptyset$ do
    Identify the line $\ell^* = \arg\min_{\ell \in \mathcal{A}_D} G_\ell$.
    Apply (4.8) to line $\ell^*$.
    Transfer $\ell^*$ from $\mathcal{A}_D$ to $\mathcal{B}_D$.
    Update $a_\ell$ and $b_\ell$ for all lines $\ell \in \mathcal{A}_D$.
    Re-evaluate $LS_\ell$ for all lines in $\mathcal{A}_D$.
    Transfer lines $\gamma$ with $LS_\gamma = L$ from $\mathcal{A}_D$ to $\mathcal{B}_D$.
end while
Apply $EXACT$ to $\lfloor D' \rfloor$ to generate $S$.
```

The following theorem states that prior to rounding, FMA achieves weighted max-min fair allocation of capacity (weighted relative to the original demand). See the Appendix (Section A.2) for the proof of the theorem.

Theorem 2. *FMA generates an adjusted demand matrix D' with weighted max-min fair allocation, where the weight is $\omega(D'_{ij}) = \frac{D'_{ij}}{D_{ij}}$.*

If the demand matrix contains zero entries, then an algorithm that adjusts requests multiplicatively (such as FMA) cannot always generate full utilization; there can be *natural blocking* because there is no demand. After all of the demands are adjusted FMA uses *EXACT* to allocate the time slots and generate the schedule. We now present some properties of the demand matrix $D' = \{D'_{ij}\}$ obtained by Algorithm 1 prior to rounding.

Property 1 : Algorithm 1 guarantees full allocation of all links provided D contains no zero elements.

Property 2 : If there is no natural blocking the maximum total throughput of the network is obtained:

$$\sum_i \sum_j D'_{ij} = N.L. \tag{4.9}$$

Property 3 : The while-loop in Algorithm 1 has $O(N^2)$ computational complexity in terms of the number of edge nodes ($2N$ iterations with a minimization over N elements in each iteration). The best current implementation of the *EXACT* algorithm has complexity $O(N^{\frac{5}{2}})$, and hence this is also the complexity of Algorithm 1.

Property 4 : Algorithm 1 guarantees minimum rejection if no connections cross two different overloaded links, i.e., if the overloaded links correspond entirely to rows (input links) or entirely to columns (output links) of D. In this case:

$$\min(REJ) = \sum_{\ell \in O}(LS_\ell \; - \; L), \tag{4.10}$$

where O is the set of overflowing lines.

Property 5 : For Poisson arrivals and exponential distribution of the packet lengths, every VOQ_{ij} can be approximated as a $M/M/1$ queuing system, with input rate $\lambda_{ij} = \mathbf{D}_{ij}$ (slots per frame) and output rate $\mu_{ij} = \mathbf{D}'_{ij}$ (slots per frame) [91]. Then the average number of packets in each VOQ_{ij} can be found using the following equation:

$$\mathbf{N}_{ij} = \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}} = \frac{\rho_{ij}}{1 - \rho_{ij}}, \tag{4.11}$$

where $\rho_{ij} = \frac{\lambda_{ij}}{\mu_{ij}}$. Since using the FMA algorithm, ρ_{ij} is the same for all VOQs belonging to a bottleneck link, each queue has the same average number of packets.

Define the *percentage rejection* as $1 - \frac{D'_{ij}}{D_{ij}}$ for the lines which were initially overloaded. Consider the set of demands that experience the highest percentage rejection (i.e., the demands on the most overloaded line). Since the weight ω is a monotonically increasing function of allocated rate D'_{ij}, weighted max-min fairness implies that it is impossible to increase the rate allocated to these demands (or decrease the maximum percentage rejection) without violating feasibility. Decreasing the rejection of any of those demands requires increasing the rejection of another demand on the same line, and hence the maximum percentage rejection increases. We thus have the following corollary of Theorem 2:

Corollary 1. *Subject to the capacity constraints, FMA generates a schedule that minimizes the maximum percentage rejection experienced by the connections.*

$$\max_{ij}\{\frac{D_{ij} - D'_{ij}}{D_{ij}}\}_{FMA} = \min_{CL}\{\max_{ij}\{\frac{D_{ij} - D'_{ij}}{D_{ij}}\}_{CL}\}, \tag{4.12}$$

where CL is any clamping algorithm that clamps the overloaded lines down to L.

4.3.3 Equal Share Algorithm (ESA)

The water-filling procedure can be implemented by assigning equal amounts of extra capacity to the connections traversing underloaded lines. This approach leads to the max-min fair share allocation of extra capacity. Similarly the overloaded lines can be adjusted by reducing equal amounts from the demands of the connections passing these lines. We define for each line the values $H_\ell \triangleq \frac{L-LS_\ell}{|a_\ell|}$, where $|a_\ell|$ is the cardinality of a_ℓ.

The line with minimum H_ℓ is the most constrained line. Repeatedly the most constrained line is defined and the demands of its connections are adjusted. The demand adjustment we perform on each line is:

$$D^{'}_{ij} = D_{ij} + \frac{L - LS_\ell}{|a_\ell|} \quad \forall\ (i,j) \in a_\ell. \tag{4.13}$$

The following theorem states that prior to rounding, ESA achieves max-min fair allocation of capacity. See the Appendix (Section A.3) for the proof of the theorem.

Theorem 3. *ESA generates an adjusted demand matrix D' with max-min fair allocation, where the weight of the connection between source i and destination j is $\omega_{ij} = D_{ij} - D'_{ij}$.*

For the ESA algorithm we have the following corollary:

Corollary 2. *Subject to the capacity constraints, ESA generates a schedule that minimizes the maximum amount of rejection experienced by the connections.*

$$\max_{ij}\{D_{ij} - D^{'}_{ij}\}_{ESA} = \min_{CL}\{\max_{ij}\{D_{ij} - D^{'}_{ij}\}_{CL}\}, \tag{4.14}$$

4.3.4 Distributed Matching Algorithms

FMA and ESA use the *EXACT* algorithm, which collocates most of the allocations for a particular source-destination pair in an attempt to minimize switch reconfigurations. This concentration has the impact of increasing average waiting time of packets. However this effect is considerably reduced if we distribute similar matchings in two different locations in the frame. In next section we show the effect of distribution of the matchings through simulations. In our simulations FMA1 refers to the case that similar matchings are collocated (applying *EXACT* in a standard fashion) and FMA2 refers to the case that similar matchings are separated into two batches, one placed towards the start of the frame and one towards the end. ESA is implemented using the distributed case similar to FMA2.

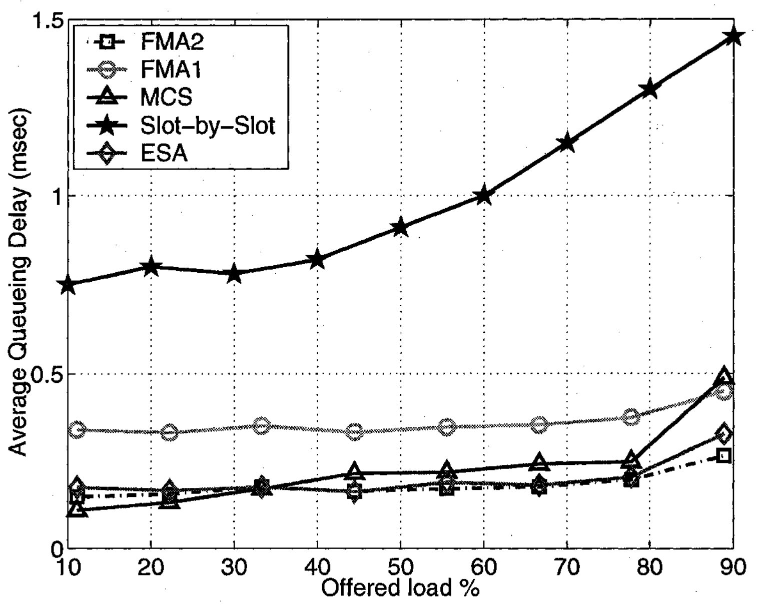


Fig. 4.8 Average queuing delay performance achieved by FMA1, FMA2, ESA (Equal Share Matching), Slot-based and MCS under non-uniform, Poisson traffic.

4.3.5 Simulation Performance

A: Comparison of Scheduling Algorithms under Non-Uniform Traffic

In this section we report the results of simulations of the fair scheduling approaches performed using OPNET Modeler [90]. The network setting is the same as the setting in Section 4.2.1.

We compare performance of FMA1, FMA2, and ESA to two previous algorithms: Minimum Cost Search (MCS) developed in Section 4.2 and a slot-based scheduling approach based on PIM (Parallel Iterative Matching) algorithm discussed in Section 3.2.

Figure 4.8 shows the queuing delays over a wide range of offered load, from 10% to 90% link capacity under non-uniform traffic (uniform traffic gives similar results). The slot-based algorithm has large average queuing delays, since it is more appropriate for metro and local-area networks [15]. FMA1 generates additional average delay compared to FMA2, which is due to the collocation of matchings. ESA, FMA2 and MCS exhibit similar performance, achieving low average delays under all but the highest load. Under higher loads, the performance of MCS deteriorates due to the additional blocking it induces. On average the percentage of blocking generated by MCS is 0.9%. The matching algorithms (FMA and ESA) generate 0.02% blocking (due to natural blocking in the demand matrices).

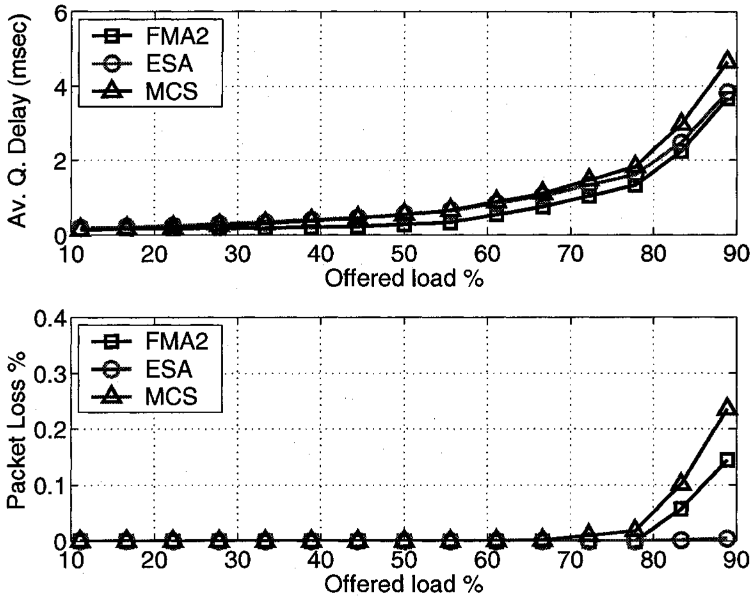


Fig. 4.9 Average queuing delay and packet loss performance for FMA2, ESA and MCS under bursty traffic and non-uniform distribution of the destinations in a wide-area network with 5 msec propagation delay.

When the load is high, FMA2 assigns more time slots to the heavier connections, which can use the extra time slots more efficiently. ESA assigns the same number of extra time slots to each connection irrespective of its load. In this scenario only the slot-based scheduling algorithm experiences packet loss (up to 0.31% for loads exceeding 70% of capacity).

B: Comparison of Scheduling Algorithms under Bursty Traffic

We also performed simulations with bursty traffic using on/off traffic sources. Every edge node is equipped with 6 on/off sources. The "on" and "off" periods have Pareto distributions with $\alpha = 1.9$. The mean of the "off" periods is 5 times greater than the mean of the "on" periods. During "on" periods the sources generate packets with an average rate equal to the full link capacity (10 Gbps). The rate distribution is exponential. Figure 4.9 depicts queuing delays and packet losses for the FMA2, ESA and MCS algorithms. FMA2 demonstrates marginally superior average queuing delay performance compared to the other two algorithms (0.3-0.9 msec less when the load exceeds 50%). Under offered loads greater than 80% of capacity, packet loss occurs as a result of traffic bursts overflowing the network. At 90% load, MCS generates 0.24% loss, FMA2 generates 0.14% loss, and ESA does not generate any packet loss. The loss generated by FMA2 is due to insufficient allocation

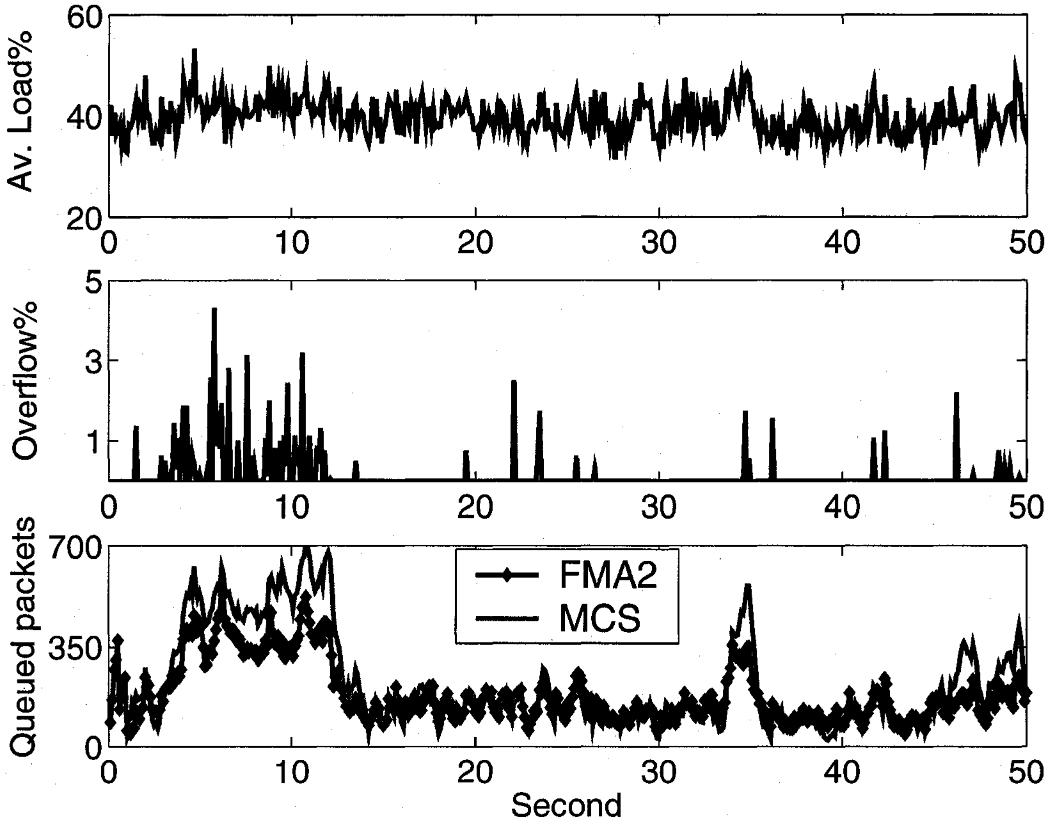


Fig. 4.10 The behavior of FMA2 and MCS in response to traffic loads derived from Internet traces. The upper panel shows the offered load averaged over all source-destination pairs. The middle panel shows the percentage of overflow traffic. The lower panel shows the overall number of queued packets at the edge nodes.

of additional slots to temporarily low-rate connections that experience a sudden increase in traffic arrivals when they enter an "on" period. ESA allocates extra slots irrespective of demand so eliminates this loss at the cost of additional average delay.

C: Comparison of Scheduling Algorithms using Real Traffic

We also explored the performance of our algorithms using traffic derived from empirical Internet measurements. The traffic prediction in this section was performed by Ahmed et al. [93] at McGill university. They used a collection of 50 seconds of packet traces captured from an OC3 link at Colorado State University [94]. The flows were divided into 16 components based on IP source/destination addresses, and each component served as one of the edge nodes. Using auto-regressive flow-based prediction, the traffic demand was

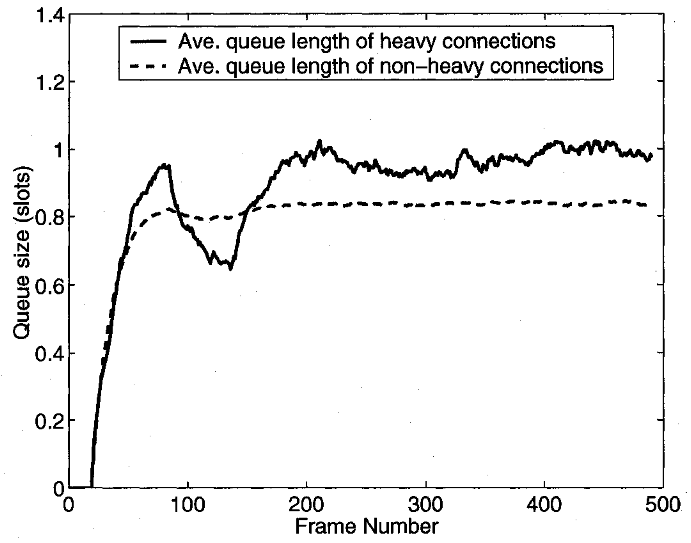


Fig. 4.11 Comparison between the average queue size of the heavy connections and the average queue size of the non-heavy connections for load of 60% in non-uniform traffic scenario. The queue sizes are in the same range which provides support for the approximation in property 5 of the FMA algorithm. We have taken a moving average from frame 20 to remove the effect of transient state on the queues, and also to only show the low frequencies of the queue variations

predicted 1 second ahead (assuming 1 second round-trip and scheduling delay). Then we applied the scheduling algorithm for the predicted traffic demand matrix. We used a more sophisticated prediction technique for this simulation scenario because of the inadequate performance of the simple linear predictor (moving average method) used in the previous simulations. We considered a frame of length 0.1 seconds (equal to 100 time slots of 1 msec.) and for simplicity assumed that each packet fits one time slot completely. We performed simulations for 50 seconds. The average offered load was around 40%; under this load, MCS and FMA are expected to perform similarly if the traffic is admissible. The derived traffic is such that the demand is inadmissible for a duration of 10 seconds (from 2–12 seconds), because one of the edge nodes is overloaded. Growth in the queue sizes is unavoidable during this period. Figure 4.10 shows the total number of queued packets at the edge nodes. FMA2 and MCS adapt to the variations of the arrivals in a very similar fashion, but FMA2 has a lower number of queued packets because it does not induce blocking.

D: Average Queue Length

In Figure 4.11 we examine property 5 (see Section 4.3.2) of the FMA algorithm. We compare the average queue length of heavy connections, and the average queue length of all of the other connections. We have taken a moving average from frame 20 to remove the effect of transient state on the queues, and also to only show the low frequencies of the queue variations. As the figure shows the queue lengths are in the same range, which supports the approximation we made for describing each VOQ as an $M/M/1$ queuing system, with equal ρ-values.

4.4 Minimum Rejection Algorithm (MRA)

In this section we propose an algorithm that generates a schedule with minimum total number of rejections for the case of inadmissible traffic. We first introduce the flow problem and then formulate our minimum rejection (MINREJ) problem as a max-flow problem. Then we propose a heuristic algorithm with a small complexity for solving the MINREJ problem.

Many optimization problems in networks can be formulated as variations of the minimum cost flow problem. The minimum cost flow problem is a special case of a linear programming problem, but it has much more favorable structure and properties than a general linear program [95]. For example, the minimum cost flow problem with integer data can be solved using integer calculations exclusively. Furthermore, some methods (relaxation, auction) are very efficient for some minimum cost flow problems but are less efficient or inapplicable for general linear programs. In practice, minimum cost flow problems can often be solved hundreds and even thousands of times faster than general linear programs of comparable dimensions [95]. The *assignment* problem, *max-flow* problem, and the *transportation* problem are three examples of special cases of the minimum cost flow problem.

Network Flow. A network flow is a vector $\mathbf{f} = (f_{ij})$ where each f_{ij} is a positive real number representing the flow on arc (i, j), i.e., the flow from i to j.

Feasible Flow. A flow $\mathbf{f}$ is feasible if it satisfies the capacity constraints and it is conserved at all nodes (total flow out of a node equals total flow in).

Minimum-Cost Flow Problem. Let $G = (E, \mathcal{A})$ be a directed network defined by a set E of N nodes and a set $\mathcal{A}$ of M directed arcs. Each arc (i, j) has an associated cost c_{ij} that denotes the cost per unit flow. We also associate with each arc $(i, j) \in \mathcal{A}$ a capacity u_{ij} that denotes the maximum amount that can flow on the arc, and a lower bound b_{ij} that denotes the minimum amount that must flow on the arc. We associate with each node $i \in E$ an integer number $k(i)$ representing its supply/demand. If $k(i) > 0$, node i is a supply node; if $k(i) < 0$, node i is a demand nodes with a demand of $-k(i)$; and if $k(i) = 0$, node i is a transshipment node. The decision variables in the minimum cost flow problem are arc flows. The minimum cost flow problem is formulated as follows:

$$Minimize \sum_{(i,j)\in\mathcal{A}} c_{ij} f_{ij}$$

subject to

$$b_{ij} \leq f_{ij} \leq u_{ij} \qquad \forall\ (i, j) \in \mathcal{A}\ ,$$
$$\sum_{\{j:(i,j)\in\mathcal{A}\}} f_{ij} - \sum_{\{j:(j,i)\in\mathcal{A}\}} f_{ji} = k(i) \quad \forall i \in E.$$

Max-Flow Problem. The maximum flow problem seeks a feasible solution that sends the maximum amount of flow from a specified source node s to another specified link t. In general we can formulate this problem as a minimum cost flow problem with $k(i) = 0$ for all $i \in E$, $c_{ij} = 0$ for all $(i, j) \in \mathcal{A}$, and introduce an additional arc (t, s) with cost $c_{ts} = -1$ and flow bound $u_{ts} = \infty$. Then the minimum cost flow problem maximizes the flow on arc (t, s); but since any flow on arc (t, s) travels from node s to t through the arcs in $\mathcal{A}$ (since each $k(i) = 0$), the solution to the minimum cost flow problem will maximize the flow from node s to node t in the original network.

4.4.1 MINREJ Problem

As explained in Section 4.3.2, property 4, when the traffic matrix is not admissible the fair matching algorithm may not achieve minimum total rejection. *PROBLEM 2* for the case of inadmissible demand matrix and negligible g can be rewritten as:

MINREJ(D,L): For a frame of fixed length L with demand matrix D identify a frame schedule S_2^* that satisfies:

$$S_2^* = \arg\min_S REJ(S, D, L) \tag{4.15}$$

In this section we introduce a pruning approach that clamps down the overloaded lines of the demand matrix such that the minimum overall rejection is obtained. Suppose that we have an arbitrary demand matrix $D_{4\times4}$, with 4 overflowing lines. We assume the demand matrix has been decomposed to two matrices D' and D'', where D' is the pruned demand matrix with line sums not greater than the schedule length L, and D'' shows the resulting rejections of every demand after the pruning algorithm.

$$\begin{aligned} D &= \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} \\ \cdots & D_{22} & \cdots & D_{24} \\ D_{31} & D_{32} & D_{33} & D_{34} \\ \cdots & D_{42} & \cdots & D_{44} \end{bmatrix} \\ &= \underbrace{\begin{bmatrix} D'_{11} & D'_{12} & D'_{13} & D'_{14} \\ \cdots & D'_{22} & \cdots & D'_{24} \\ D'_{31} & D'_{32} & D'_{33} & D'_{34} \\ \cdots & D'_{42} & \cdots & D'_{44} \end{bmatrix}}_{D'} + \underbrace{\begin{bmatrix} D''_{11} & D''_{12} & D''_{13} & D''_{14} \\ \cdots & D''_{22} & \cdots & D''_{24} \\ D''_{31} & D''_{32} & D''_{33} & D''_{34} \\ \cdots & D''_{42} & \cdots & D''_{44} \end{bmatrix}}_{D''} \end{aligned}$$

For the rejection matrix D'' and the pruned matrix D' we have the following equations:

$$\begin{aligned} D''_{hp} &= 0, \qquad \text{if } h \notin O_r \text{ or } p \notin O_c\ , \\ 0 \le D''_{hp} &\le D_{hp}, \quad \forall\ (h,p) \text{ s.t. } h \in O_r \text{ or } p \in O_c\ , \\ \sum_p D'_{hp} \le L, \ &or\ r_h(D) - L \le \sum_p D''_{hp} \quad \forall\, h \in\ O_r\ , \\ \sum_h D'_{hp} \le L, \ &or\ c_p(D) - L \le \sum_h D''_{hp} \quad \forall\, p \in\ O_c\ , \end{aligned}$$

where O_r and O_c are the set of overflowing input and output links of the photonic network respectively. We define the following set:

$$\mathcal{B} \triangleq \{(h,p) : h \in O_r \text{ or } p \in O_c\}$$

We wish to obtain the minimum overall rejection of the demand matrix such that the line sums of the overflowing lines equal L in the pruned matrix D', but this is not always feasible. Thus we formulate the following optimization problem:

$$\begin{aligned} & \textit{Minimize} \sum_{(h,p)\in\mathcal{B}} D''_{hp} \\ & \textit{subject to} \\ & 0 \le D''_{hp} \le D_{hp} \qquad \forall\ (h,p) \text{ s.t. } h \in O_r \text{ or } p \in O_c\ , \\ & r_h(D) - L \le \sum_p D''_{hp} \quad \forall\, h \in\ O_r\ , \\ & c_p(D) - L \le \sum_h D''_{hp} \quad \forall\, p \in\ O_c\ . \end{aligned} \tag{4.16}$$

This problem is a variant of the minimum cost flow problem. The cost on each arc is either 1 or 0. We can use the algorithms for solving the minimum-cost flow problem to solve this optimization problem. However, translating this problem to a max-flow problem introduces more straight-forward solutions.

Transforming the optimization problem in (4.16) to a max-flow problem. Using $D'_{ij} = D_{ij} - D''_{ij}$ the problem in (4.16) for a demand matrix $D_{n\times n}$ is transformed to the following max-flow problem:

$$\begin{aligned} & \textit{Maximize} \sum_{(h,p)\in\mathcal{B}} D'_{hp} \\ & \textit{subject to} \\ & 0 \le D'_{hp} \le D_{hp} \quad \forall\ (h,p) \text{ s.t. } h \in O_r \text{ or } p \in O_c\ , \\ & \sum_p D'_{hp} \le L \quad \forall\, h \in\ O_r\ , \\ & \sum_h D'_{hp} \le L \quad \forall\, p \in\ O_c\ . \end{aligned} \tag{4.17}$$

In this problem the flow of each arc is defined as:

$$f_{ij} = D'_{ij}.$$

Figure 4.12 shows the corresponding network flow of matrix $D_{4\times 4}$ when rows 1 and 3, and columns 2 and 4 are overloaded.

The numbers over the arcs show the arc capacities which correspond to the lower and upper bounds of flows in our maximization problem. Ford and Fulkerson presented a solu-

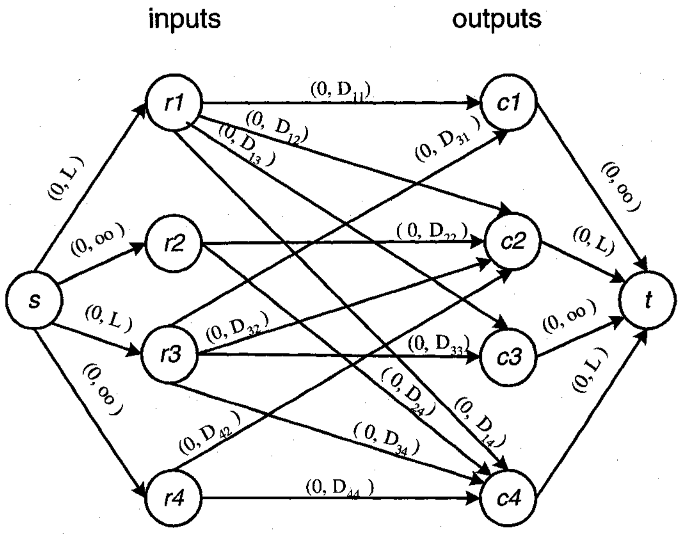


Fig. 4.12 Network flow of matrix $D_{4\times4}$ for the max-flow problem (4.17). Every node corresponds to a row or a column of the demand matrix. Every arc begins and/or ends at an overloaded node, so there is no connection between two nodes which are not overloaded. The numbers over each arc correspond to the lower and upper bounds of the flow of each arc respectively.

tion to the max-flow problem in 1954 [96]. The algorithm starts from an arbitrary feasible flow. In subsequent iterations, the Ford-Fulkerson method identifies an augmenting path, and augments the flow. If the augmenting path is denoted as a set of arcs $\{a_1, a_2, ..., a_k\}$, then the flow augmentation is $\delta = \min_{1\leq i\leq k} \delta(a_i)$, where $\delta(a_i) = \kappa_{a_i} - f_{a_i}$ for forward arcs and $\delta(a_i) = f_{a_i}$ for backward arcs. The flow is adjusted using $f_{a_i} \leftarrow f_{a_i} + \delta$ on forward arcs and on backward arcs using $f_{a_i} \leftarrow f_{a_i} - \delta$. The algorithm iterates until no augmenting path exists, upon which the maximum flow is obtained, as specified by the following theorem:

Theorem 4. *Ford-Fulkerson [96]: Flow* $\mathbf{f}$ *is maximum in graph* $\mathcal{G}$ *if and only if there is no augmenting path in* $\mathcal{G}$ *bearing flow* $\mathbf{f}$.

When there are no lower bounds on capacity, the flow $\mathbf{f}$ defined by $f_{ij} = 0 \;\; \forall(i,j) \in \mathcal{A}$ (the set of arcs in the network) is feasible and can be used to initialize the Ford-Fulkerson method. There are numerous methods for searching for augmenting paths; techniques include shortest path (fewest number of arcs) and fattest path (maximum bottleneck capacity along the path) algorithms [97]. Note that the solution to the maximum flow problem is in general not unique.

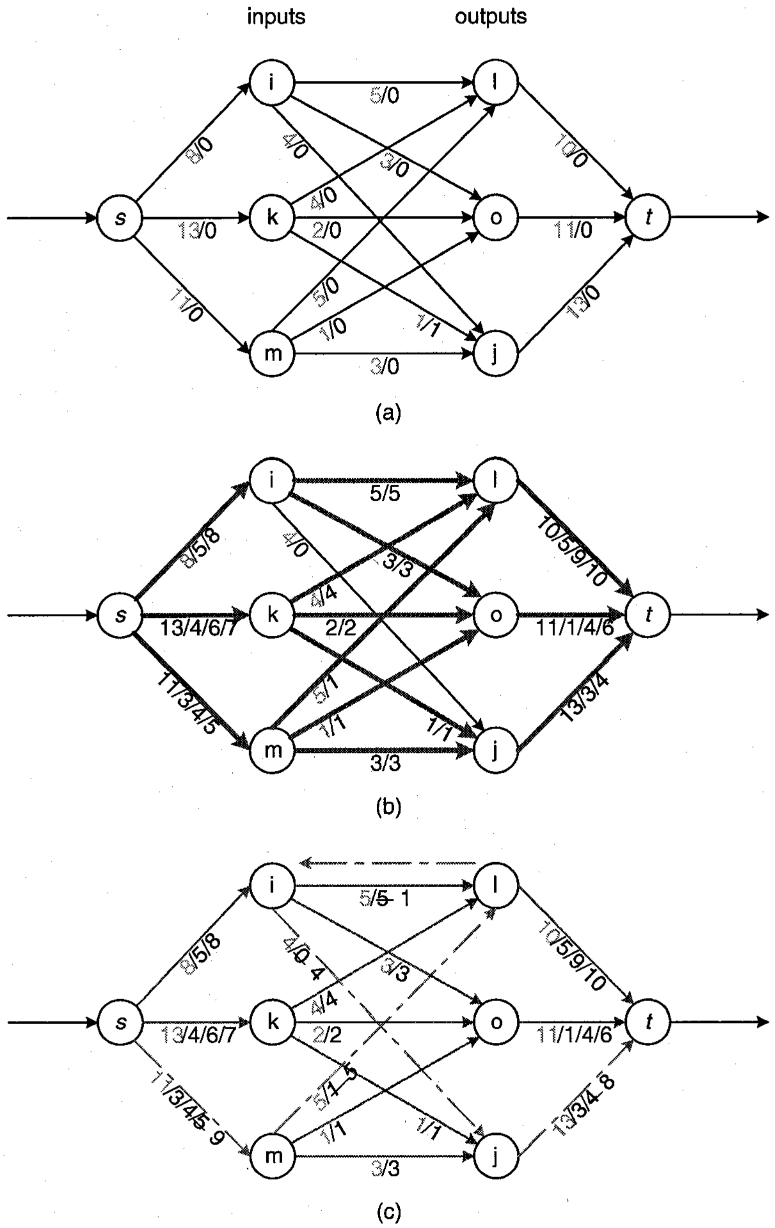


Fig. 4.13 Ford-Fulkerson approach for finding maximum flow in the graph: the capacity of each arc is shown with green (light) numbers. The black (dark) numbers show the progress of the flow of each arc which is increased or decreased as a result of the new augmenting path; (a) shows the initial step where the flows are set to zero, (b) shows the progress after finding several augmenting paths without any backward arcs, and (c) shows an augmenting path with one backward arc. The resulting flow at this step is the maximum flow, since augmentation is not possible anymore. The crossed-out numbers are the previously assigned flows which are updated by the new augmenting path.

Example 1: Consider the network flow of Figure 4.13-(a). The lower bounds are all zero, and the upper bounds are shown with green numbers. We want to find the amount of maximum flow in this network.

Using the Ford-Fulkerson method (see Appendix B for pseudo code of this approach) we maximize the rejections of these connections. The flow graph of this problem is shown in Figure 4.13- (a). The capacity of each arc is the maximum amount of flow on that arc. At the beginning the flow of every arc is set to zero. Then the augmenting paths are chosen one after another, and their flows are increased until one of the arcs is saturated. Recall that the flow of an augmenting path is the minimum of the residuals of the forward arcs and the flows of the backward arcs.

Figure 4.13-(b) shows the augmenting paths (red paths / thick paths) which are chosen based on a simple shortest path first criterion. The numbers beside the capacities show the progress of the flows at each step of finding another augmenting path. At this point we cannot increase the flow anymore without changing the flows of the other paths. In Figure 4.13-(c) we show an augmenting path (olive path / dashed path) with flow of 4 units. This augmenting path consists of four forward and one backward arcs. Augmentation is done by deducting the flow of the augmenting path from those of the backward arcs, and adding it to those of the forward arcs. The resulting overall flow in the graph is higher than the previous step. Since any additional augmentation is not possible, based on the Ford-Fulkerson theorem this flow is the maximum flow. In this example the maximum flow is 24 units (time slots).

Complexity of the Max-flow Algorithms

The Complexity of the Ford-Fulkerson algorithm depends on the complexity of the search procedure which finds the augmenting path for each iteration. In the approach proposed by Edmonds-Karp the flow is always augmented along a "shortest path" from the source to the sink [98]. A shortest path in this case is defined as a directed path in the residual network consisting of the fewest number of arcs. If we augment flow along a shortest path, the length of any shortest path either stays the same or increases. Moreover, within M augmentations, the length of the shortest path is guaranteed to increase, where M is the number of arcs in the network. Since no path contains more than $N-1$ arcs (N is the number of nodes in the flow network), this result guarantees that the number of augmentations is at most $(N-1)M$. One simple approach for implementing the shortest path search procedure is to look for the shortest path by performing a breadth-first search in the residual network. A labeling

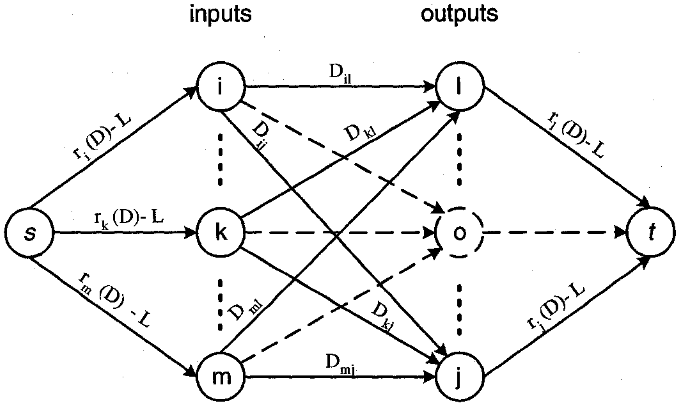


Fig. 4.14 $s \rightarrow t$ network: In this example the input vertices correspond to the overflowing rows of an arbitrary demand matrix D $(i, k, m \in O_r)$, and the output vertices correspond to the overflowing columns of D $(l, o, j \in O_c)$. The numbers over the arcs show the arc capacities which correspond to the upper bounds of flows in our maximization problem. The capacity of each arc (not connected to the source or sink) is equal to the upper bound on the amount of rejection that can be assigned to the corresponding critical connection.

algorithm maintains a set of labeled nodes as a queue, then by examining the labeled nodes in a first-in, first-out order, it would obtain a shortest path in residual network. Each of these iterations would require $O(M)$ steps in the worst case. Therefore Edmonds-Karp is a polynomial time algorithm for max-flow with computation time of $O(NM^2)$ [98]. This computation is excessive, but on average the number of iterations per augmentation is $O(N)$, which results an average time complexity of $O(N^2M)$.

The fastest maximum flow algorithms to date are preflow-push algorithms. Other flow problems, such as the minimum-cost flow problem, can be solved efficiently by preflow-push methods. Preflow-push algorithms work in a more localized manner than the Ford-Fulkerson method. Rather than examine the entire residual network $G = (E, \mathcal{A})$ to find an augmenting path, preflow-push algorithms work on one vertex at a time, looking only at the vertex's neighbors in the residual network. The complexity of the preflow-push algorithm for finding max-flow is $O(N^3)$ [98].

4.4.2 A Heuristic Algorithm for Solving the MINREJ problem

First we develop a theorem that helps to identify a procedure for solving *MINREJ(D,L)*. We commence by defining *MAXFLOW(D,X,L)*, a max-flow linear programming problem.

Problem Y = MAXFLOW(D,X,L): D is a demand matrix, X is a non-negative matrix that specifies capacity bounds, and L is the frame-length (available capacity on each row/column). Matrices D, X and Y are all of size $N \times N$. Identify a nonnegative matrix Y such that $\sum_{h \in O_r} \sum_{p \in O_c} Y_{hp}$ is maximized, subject to the following constraints:

$$\begin{aligned}
Y_{hp} &= 0 \quad \text{if } h \notin O_r \text{ or } p \notin O_c, \\
Y_{hp} &\leq X_{hp} \quad \forall\ (h,p) \text{ s.t. } h \in O_r \text{ and } p \in O_c, \\
\sum_{p \in\ O_c} Y_{hp} &\leq r_h(D) - L \quad \forall\ h \in\ O_r\ , \\
\sum_{h \in\ O_r} Y_{hp} &\leq c_p(D) - L \quad \forall\ p \in\ O_c\ ,
\end{aligned}$$

The following theorem establishes a relationship between a solution to the problem *MAXFLOW(D,D,L)* and a solution to the minimum rejection problem *MINREJ(D,L)*. The proof is in the Appendix (Section A.4).

Theorem 5. *Set A = MAXFLOW(D,D,L). Construct a rejection matrix $D'' = A + Q$, where Q is an arbitrary non-negative matrix such that $Q_{hp} \leq D - A\ \ \forall\ \ (h,p)$, $r_h(Q) = r_h(D) - L - r_h(A)\ \ \forall\ \ h \in O_r$, and $c_p(Q) = c_p(D) - L - c_p(A)\ \ \forall\ \ p \in O_c$. Then if S is a schedule that generates the decomposition $D = D' + D''$, it is a solution to the problem MINREJ(S,D,L).*

We now describe an algorithm to identify a solution A to *MAXFLOW(D,D,L)*. The corresponding maximum flow problem is depicted in Figure 4.14. We define a network with a source s and a sink t and try to maximize the flow between them. In our problem, the total amount of flow emitted from source s (and therefore arriving at sink t) is equal to the total amount of rejection contributed by A at the critical connections. The rejection at any specific critical connection (A_{hp}) is equal to the flow on arc (h,p). The capacities of the arcs (upper bounds) are dictated by the constraints in *MAXFLOW(D,D,L)*. We denote the upper bound on arc (i,j) by $\kappa(i,j)$. So we have:

$$\kappa(s,h) = r_h(D) - L \quad \forall\ h \in\ O_r$$

$$\kappa(p,t) = c_p(D) - L \quad \forall\ p \in\ O_c$$

For a feasible flow vector $\mathbf{f}$, an *augmenting path* is a simple path from s to t that can be used to increase flow from s to t. Note that this path is not necessarily directed. On forward arcs in this path ((i,j) points in the direction $s \rightarrow t$) the flow f_{ij} must satisfy $0 \leq f_{ij} < \kappa_(i,j)$, and on backward arcs, i.e. (i,j) is reverse, the flow must satisfy $0 < f_{ij} \leq \kappa(i,j)$. Note that the solution to *MAXFLOW(D,D,L)* is in general not unique.

To form a Minimum Rejection Algorithm, we first use the Ford-Fulkerson algorithm to identify A. Subsequently we set $D \leftarrow D - A$ and apply FMA to the resultant D. As described in Section 4.3.2, FMA processes overflowing lines sequentially, adjusting the demand on the line so that it sums to L (thereby identify a line of the rejection matrix). Since we have constructed A so that after modification $D(h,p) = 0$ at any intersection point of overflowing lines h and p, when FMA adjusts one of the overflowing lines it does not affect any other overflowing line. This means that after FMA has been applied, it has generated a Q that satisfies the requirements of Theorem 1. In the process, FMA has developed a schedule S that performs the decomposition $D = D' + D''$, where $D'' = A + Q$. The combined Minimum Rejection Algorithm is specified in Algorithm 2. This algorithm has a worse case complexity of $O(N^3)$, but in average it has a much smaller complexity. The number of edges in MINREJ algorithm specified by (4.18) is smaller than the number of edges in optimization problem specified by (4.17).

Algorithm 3 Minimum Rejection Algorithm

```
Apply the Ford-Fulkerson algorithm to solve $A$ =MAXFLOW(D,D,L).
Set $D \leftarrow D - A$.
Apply FMA to $D$ to generate $Q$ and a schedule $S$.
```

4.4.3 Simulation Performance

In this section we report the results of simulations of the scheduling approaches performed using OPNET Modeler [90] under Bursty Traffic. In this set of experiments the network setting is the same as the setting in Section 4.3.5-B. Since the behavior of MRA and $FMA2$ only differs when there are critical elements in the demand matrix, we investigate scenarios where critical demands are likely to exist. In order to do this, in each frame we choose one arbitrary source i and one arbitrary destination j. Each source generates z times as many packets for destination j compared to other destinations. Similarly source i generates z times as many packets (to all destinations) as any other source. As z increases, the elements of the demand matrix corresponding to these two edge nodes are more likely to be critical connections; the demand element D_{ij} has even higher likelihood of being critical.

Figure 4.15 compares the percentage of rejected demand achieved by FMA2 and MRA

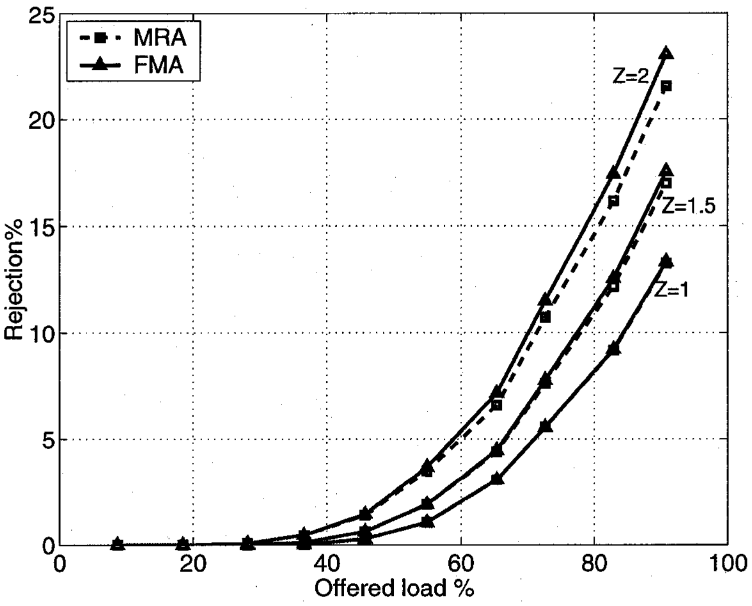


Fig. 4.15 Comparison between the rejection obtained by FMA and MRA under varying offered load for different factors of imbalanced load (z). Traffic is bursty (generated by on-off sources) and has uniform distribution, aside from the impact of z.

as the offered load changes for various values of z. At high load (greater than 70%) with $z = 2$, there are numerous critical elements and MRA begins to achieve less rejection than FMA2. The discrepancy is still only 2 percent at 90% load. Figure 4.16 compares the maximum percentage rejection experienced by any demand when scheduling is performed by FMA2 and MRA. As the offered load increases, MRA concentrates rejection on the critical elements; the maximum percentage rejection is thus much (up to 25 percent) higher than that achieved by FMA2, which distributes rejection fairly amongst all competing connections. Figure 4.17 compares the average end-to-end delay experienced by packets when scheduling is performed using FMA and MRA; the approaches yield similar average delay.

4.5 Summary

In this chapter we concentrated on establishing a formulation for fixed-length frame scheduling for point-to-point transmission. Prior to this research, state-of-the-art scheduling algorithms for fixed-length frames were limited by their inability to address the scenarios of inadmissible or low demands [29–32]. In this chapter, we described our designed algorithms

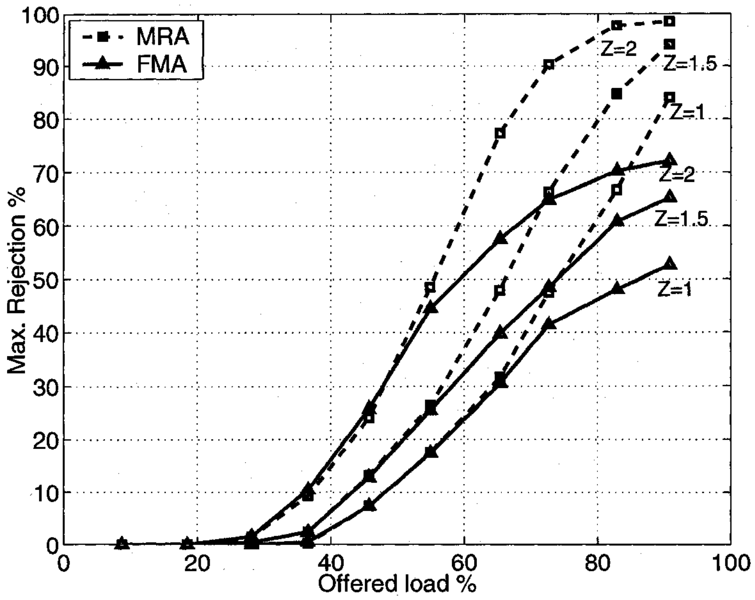


Fig. 4.16 Comparison between the maximum percentage rejection experienced by any demand after scheduling by MRA and FMA for different values of z and varying offered load.

that particularly addressed the shortcomings of fixed-length frames without introducing additional complexity to the network.

We defined a cost function with two parameters, the number of switchings that the core switch must perform, and the number of rejected requests in time slots during one frame. We proved that this problem in general is NP-hard. When minimizing the number of switchings has a very low importance, the problem is reduced to minimizing the number of rejections which can be solved in polynomial time. For the case of admissible demand matrix, the maximum cardinality bipartite matching algorithm provides zero rejection. For the case of inadmissible traffic the algorithms proposed in Section 4.4.1 provide minimum rejection.

FMA is an algorithm based on MCBM, which not only provides zero rejection for admissible demand, but also establishes a weighted max-min fair strategy to share the extra capacity of the underloaded links, and reject the extra requests of overloaded links fairly. We showed that this algorithm minimizes the maximum percentage rejection experienced by any connection. Similarly, ESA establishes the max-min fair criterion and minimizes the maximum amount of rejection experienced by any connection.

We also proposed MCS, a heuristic algorithm which performs close to FMA under nonbursty traffic. For every allocation it performs a simple search procedure to find an eligible

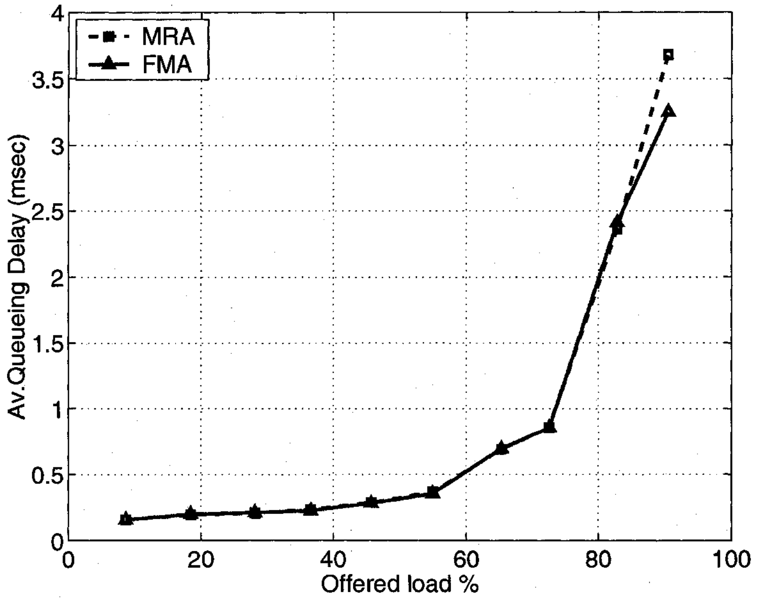


Fig. 4.17 Average queuing delay performance achieved by MRA and FMA for varying offered load and $z = 2$.

time slot with the least number of free transmitters. This strategy leaves free time slots with more potential of supporting any possible request in consequent allocations.

In this chapter we also investigated the effect of frame length on queuing delay. Simulations show that decreasing the frame length improves the performance for low offered loads. However the impact is inverse when the offered load is high. When the load is high the schedulers with smaller buffer lengths have larger queuing delays.

It was shown that FMA minimizes the proportional percentage of the rejection experienced in the network while it does not provide the total minimum rejection. We proposed the Minimum Rejection Algorithm (MRA), a novel algorithm that generates a schedule that minimizes the total rejection. Simulations showed that the discrepancy in total rejection achieved by MRA and FMA is relatively minor, whereas there is a major difference in the fairness of the allocation of rejection. In addition, MRA appears to be less robust to demand prediction errors (when traffic arrivals differ substantially from the demand matrix used for scheduling). While it is expected that we achieve smaller queuing delay with minimum rejection algorithms, the results of simulations show that FMA and MRA perform similarly.

Chapter 5

Scheduling and Control in the Agile-All Photonic Network

In Chapter 4 we described three algorithms for time-slot allocation in AAPNs [99,100] with large propagation delay. Scheduling in wide-area networks must be based on predictions of traffic demand and the resultant errors can lead to instability and unfairness. The Fair Matching Algorithm (FMA), described in Chapter 4, achieves no rejection if the demand does not exceed capacity. It also allocates spare capacity in a weighted max-min fair manner (subject to capacity constraints, each source-destination connection receives a share of the free capacity that is proportional to its original request). If the demand exceeds capacity, then FMA rejects demand in a weighted max-min fair manner (source-destination connections experience a reduction in their request that is proportional to the size of the request). These two properties of FMA represent *clamping:* resource allocation is clamped to the full capacity available in a frame irrespective of the demand from the edge-nodes. The scheduling algorithms we described in Chapter 4 are memoryless; the algorithms do not record how many rejections occur for each connection in a frame if the demand is too great. In addition, there is a rounding procedure embedded in the scheduler, since time slots must be allocated in an integral fashion. The memoryless clamping and rounding act as further sources of imperfection in the resource allocation system. In this chapter we introduce a feedback control system based on Smith's principle to reduce the effect of prediction errors, increase the speed of the response to the sudden changes in traffic arrival rates and improve the fairness in the network through equalization of queue-lengths.

This chapter is organized as follows: Section 5.1 provides background material on congestion control in data networks. Section 5.2 describes the resource allocation task in agile all-photonic networks as a control system, and explains why we cannot use common controllers such as PI in control systems with long dead-time. Section 5.3 introduces the Smith principle and its modifications. Section 5.4 describes the design of a modified Smith controller that interacts with our scheduling approach to produce a stable bandwidth allocation mechanism. Section 5.5 describes the simulation experiments we have executed to assess the performance of the scheduling algorithm and the Smith controller and discusses the results. Finally, Section 5.6 summarizes this chapter and indicates future research directions.

5.1 Background

Bandwidth sharing methods are usually classified into two basic groups: resource allocation (scheduling) and congestion control techniques [101]. Resource allocation techniques avoid congestion at the network by scheduling the explicit share of bandwidth for every connection before transmission [45, 102]. The congestion control approaches attempt to perform online adaptation of the rates of connections to control or avoid congestion that might occur at the network (switch) level [103, 104] using the feedback information received from the network.

Feedback congestion control has been examined from a control theoretic perspective by many authors, with the primary focus being controlling the rates at which sources inject best-effort traffic into a network in order to reduce the congestion at bottleneck queues whilst maintaining high utilization. In [105], Zhao et al. formulate the available bit-rate (ABR) resource allocation problem as a variant of the classical disturbance rejection problem. In a slightly different approach, Altman et al. pose the same task as a stochastic control problem, modelling the disturbance as an autoregressive process that is estimated by the controller using recursive least squares [106]. In [107], Holot et al. analyze the combination of TCP and Active Queue Management (AQM) model from a control theoretic standpoint. They examine an AQM system implementing random early detection (RED) and present design guidelines for choosing parameters that lead to stable operation of the linear feedback control system. Similar approaches using linear control theory are presented in [108–111] for congestion control and queue management of Internet traffic. There are many other examples of the application of linear control theory; see [108, 112] for surveys.

Network congestion control mechanisms employ two classes of algorithms: explicit and implicit rate control algorithms [104]. In explicit rate control approaches, the controller receives signals from the transmitters detailing their current rates and service requirements and explicit messages reflecting the congestion state of the network; the controller uses this information to calculate rates for the connections to regulate network queue lengths, and it transmits these rates to the sources. The controller can be distributed or centralized.

In implicit congestion control approaches, there are no explicit messages about the network congestion state; the controller infers the state based on end-to-end performance metrics such as loss and delay. One of the most widespread implicit congestion control algorithms is the additive-increase and multiplicative-decrease algorithm [113]. In this approach, in the absence of congestion, users increase their rates linearly until congestion occurs (as indicated by packet loss or excessive delay) and then begin to decrease their rates exponentially. This principle is widely implemented in standardized protocols such as the congestion avoidance algorithms of TCP and the congestion control algorithms of ATM/ABR [114,115].

In contrast to congestion control approaches, which respond to the state of the network, resource allocation techniques act proactively, assigning bandwidth shares to every connection prior to transmission [45,102]. Depending on the topology of the network, and in particular the propagation delay between source nodes and scheduler, the scheduling can either be based on explicit demands from the sources that report the lengths of current (virtual) output queues, or the requests can be predictions of future demand.

In the work most closely related to the controller design presented in this chapter, Mascolo combines classical control theory and Smith's principle to design a simple congestion control law that guarantees no packet loss and efficient use of bandwidth [25]. Bauer et al. propose a new class of time-variant Smith predictors using time-variant network delay models for forward and backward paths [116]. The proposed model features better tracking and faster rise and settling time. In both of these designs, the dynamic behavior of each network queue in response to data input is modelled as an integrator followed by a time delay. The use of Smith's principle, which alleviates the stability difficulties of control systems with large delays, makes Mascolo's design applicable to network paths with a wide range of propagation delays. Although the theoretical techniques we adopt in our design are similar to those used by Mascolo, the problem we address differs significantly. We assume that we have no control over arrival rates; instead we can adjust, through scheduling, the resources allocated within the network. This results in an inverted version of the standard congestion control problem: switch resources are controlled rather than source rates.

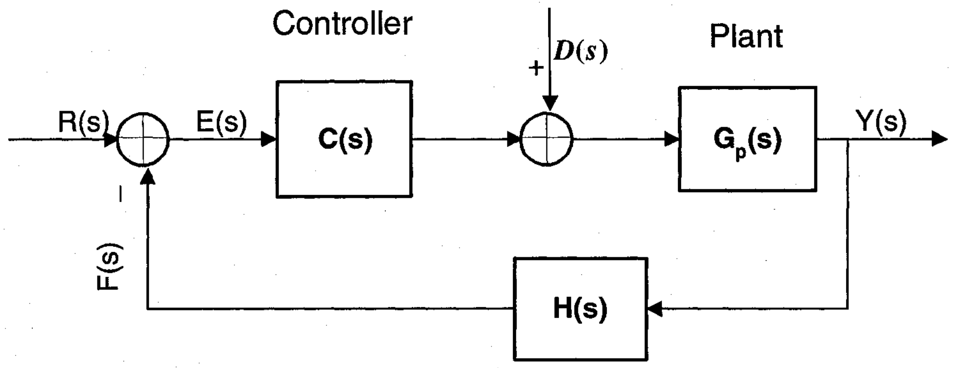


Fig. 5.1 A typical control system with controller $C(s)$, plant dynamic $G_p(s)$, and feedback transfer function $H(s)$. In this figure $R(s)$ is the reference signal, $F(s)$ the feedback, $D(s)$ the disturbance, and $Y(s)$ the output of the control system. $E(s)$ shows the error, which is the difference between the feedback and the reference signals.

5.2 Queue Control and Stability

Figure 5.1 shows a typical closed-loop control system. In this system, the plant, $G_p(s)$, is controlled by a feedback system and a controller $C(s)$. The four basic transfer functions are defined as:

1. *Feedforward transfer function:* $\frac{Y(s)}{E(s)} = C(s)G_p(s)$,
2. *Feedback transfer function:* $\frac{F(s)}{Y(s)} = H(s)$,
3. *Open-loop transfer function:* $L(s) \triangleq \frac{F(s)}{E(s)} = C(s)G_p(s)H(s)$,
4. *Close-loop transfer function:* $H_r(s) \triangleq \frac{C(s)G_p(s)}{1+C(s)G_p(s)H(S)} = \frac{C(s)G_p(s)}{1+L(s)}$.

We usually refer to the denominator of the closed loop transfer function as the *characteristic equation* of a closed loop system, since this function defines characteristics of the system in terms of stability and speed of the response. The above transfer functions are obtained assuming that the disturbance $d(t) = 0$ and the only input to our system is $r(t)$. We can follow a similar procedure for the disturbance to obtain $H_d(s) \triangleq \frac{Y(s)}{D(s)}$. Then the overall output of the control system is:

$$Y(s) = H_r(s)R(s) + H_d(s)D(s).$$

In this section, we develop a control system model for resource allocation in an agile all-photonic network. Recall from Chapter 1 that the AAPN architecture consists of edge nodes, where the optical electronic conversion takes place, connected via selec-

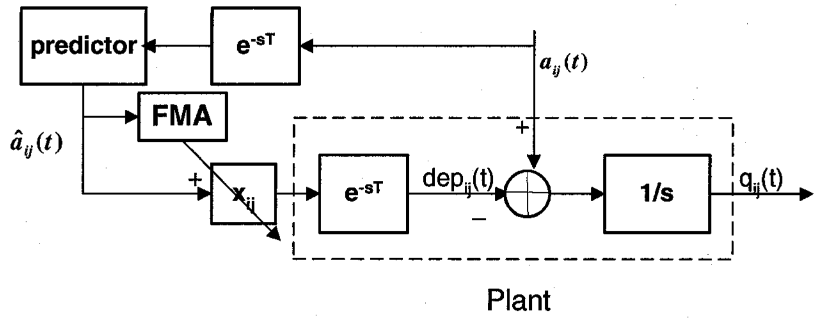


Fig. 5.2 Representation of the bandwidth allocation mechanism of an agile all-photonic network employing the FMA scheduling algorithm as a simple open-loop control system. The inputs to the system are the predicted arrival rate $\hat{a}_{ij}$ and the actual arrival rate a_{ij}, and the output is the information from each VOQ shown by q_{ij}. This representation is applicable for each (i,j) source-destination pair, and the FMA block receives input from each of these open-loop systems. The controller gain $\mathbf{x}_{ij}$ is calculated as $\mathbf{x}_{ij} = \mathbf{D}'_{ij}/\mathbf{D}_{ij}$.

tor/multiplexor devices to photonic core crossbar switches, which act independently of one another. It permits each node to transmit to one destination node and receive from one source node simultaneously on each wavelength. Each edge node constructs a separate queue for the traffic destined to each of the other edge nodes referred to as a virtual output queue (VOQ).

The scheduling algorithms described in Chapter 4 are solely based on the predicted demands arriving as signals from the edge nodes, and so can be described as feedforward or open-loop systems. Figure 5.2 shows the schematic of the open-loop system and its relationship with the FMA scheduler. Following the procedure in [25], we use a simple integrator as the dynamic model for a VOQ. Let a_{ij} be the arrival rate to VOQ_{ij}, and let $\hat{a}_{ij}$ be the predicted arrival rate. Let $q_{ij}(t)$ be the length of the virtual queue of packets at edge node i destined to edge node j and let dep_{ij} be the depletion rate of this queue. Although the depletion rate varies within a frame due to the specific allocation of time slots, we model it as constant throughout the duration of a frame. Under this model, the depletion rate dep_{ij} has the following relationship with the number of time slots given to source-destination (i,j) by the FMA scheduler:

$$dep_{ij}(t+T) = \frac{\mathbf{D}'_{ij}(k)C}{L} \qquad kT_s \leq t \leq (k+1)T_s. \tag{5.1}$$

The demand signal $d_{ij}(t)$ is equal to the predicted arrivals:

$$\hat{a}_{ij}(t) = d_{ij}(t) = \frac{\mathbf{D}_{ij}(k)C}{L} \qquad kT_s \leq t \leq (k+1)T_s. \tag{5.2}$$

Here $\mathbf{D}_{ij}(k)$ is the number of time slots demanded for a source-destination pair (i, j) during one frame duration, T_s, $\mathbf{D}^{'}_{ij}$ is the adjusted number of allocations based on the FMA algorithm, C is the line rate in bits-per-second, L is the frame-length in time slots, and T is the signalling delay between the controller and the plant (the VOQs).

Demand matrix adjustment is performed by a clamping algorithm (e.g., FMA) which clamps the line sums of the demand matrix up or down to L. FMA multiplies the predicted arrival rate $\hat{a}_{ij}$ by a factor, $\mathbf{x}_{ij} = \frac{\mathbf{D}^{'}_{ij}}{\mathbf{D}_{ij}}$. Since this factor changes with the overall arrival rates the gain of the controller is tuned each frame.

If the system relies on only the feedforward information then the effect of errors is not taken into account in future calculations and this can lead to instability and unfairness in the network. Therefore a closed loop control system is needed to achieve stability (i.e., steady state queue size variation) by controlling the states of the queues.

Before we can generate a schedule to allocate the available switch resources to the various edge nodes, the central controller must construct a predicted demand matrix $\mathbf{D}$, where $\mathbf{D}_{ij}$ is the anticipated number of slots required by source node i for destination j during the fixed-length frame occurring T seconds into the future. The signalling delay, T, exceeds the largest propagation delay between any edge node and the core. Many approaches can be adopted for performing this prediction, ranging from a naive predictor, where the estimate equals the number of slots required to accommodate the traffic that arrived in the current frame, to more elaborate techniques based on sophisticated traffic models such as those presented in [117]. The edge nodes can generate their own estimates and send these to a central controller collocated with the photonic core switch, or they can send the raw measurements of traffic volumes and the central controller can form the estimates.

Figure 5.3 shows a simple control model for an agile all-photonic network (AAPN) with a central controller. In the control model the length of each VOQ is compared with a reference signal, $r(t)$, and the difference is the input to the controller, which calculates the necessary adjustment rate, $ac_{ij}(t)$, for the predicted traffic arrival rate $a^{'}_{ij}(t)$. Provided that the queue does not empty ($q_{ij} > 0$), the depletion rate can be expressed as the sum of the predicted arrival rate $a^{'}_{ij}$ and the adjustment due to the feedback ac_{ij}, suitably delayed in

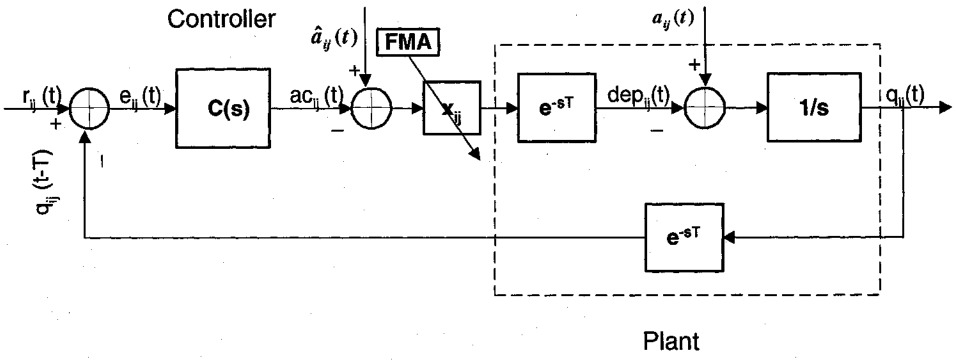


Fig. 5.3 The provision of a feedback signal results in bandwidth allocation in an agile all-photonic network becoming a simple closed loop control system. Inputs to the system are a reference signal r_{ij}, the estimated arrival rate $\hat{a}_{ij}$ and the true arrival rate a_{ij}, and the feedback is the information from the VOQ, indicated by q_{ij}. The propagation delay from the controller to the plant (core node to edge node) is T.

time, i.e., $dep_{ij}(t) = a'_{ij}(t-T) - ac_{ij}(t-T)$. Note that this adjustment is *subtracted* from the predicted arrival rate because it is proportional to the discrepancy between the desired state r_{ij} and the current state q_{ij}. When this discrepancy is negative, the depletion rate should be *increased* to address the additional packets in the queue.

The queue length based on the flow conservation equation [25] with initial condition $q_{ij}(0) = 0$, is:

$$q_{ij}(t) = \int_0^t [a_{ij}(\tau) - dep_{ij}(\tau)]d\tau. \tag{5.3}$$

This model does not consider the case when the queue is empty: according to (5.3), departures from an empty queue result in a negative queue length. A more precise model for the network requires the inclusion of the operation "$q_{ij} = \max(0, q_{ij})$" immediately prior to transmission of the feedback signal. Since this function can only be realized with a non-linear component we do not incorporate it, choosing instead to model the queues as always-occupied.

For this control system we aim to minimize the error between the queue length and the reference signal. The reference signal is interpreted as the desired queue length and may be calculated based on the state of the network. For example, if the desired state is equal queue lengths for all of the VOQs, then the reference signal should be the average of the VOQ lengths. In the absence of the control system, the scheduler would set a depletion rate that is equal to the predicted arrival rate. The error signal is used to adjust the estimated arrival rate through the use of a controller.

5.2.1 PI controller and Stability Conditions

The choice of a controller is very important in controlling systems with dead-time. A PI (proportional-integral) controller is usually used to improve the dynamic response as well as to reduce or eliminate steady state error. Figure 5.4 shows a PI controller with $C(s) = K_p + K_I/s$. The K_p term arises because the adjustment should be proportional to the error; the K_I/s adds the integral of the error to eliminate residuals. The objective is to select K_p and K_I such that the linear control system in Figure 5.3 is stabilized. A linear feedback control system is stable if bounded input produces only bounded output. The theoretical approach for obtaining a stable design is to set the control parameters such that $\Phi M > 0$ and $GM > 0$ are achieved, where $\Phi M = \angle L(j\omega_g) - 180°$ and $GM = -20\log|L(j\omega_c)|$ are the phase margin and the gain margin of the system, respectively. This in turn requires the two following conditions to hold:

$$|L(j\omega_c)| \leq 1 \tag{5.4}$$

$$\angle L(j\omega_g) > 180°. \tag{5.5}$$

Here ω_c is the frequency at which the phase is -180 (the cut-off frequency), ω_g is the frequency at which the gain of the system is 1 (the crossover frequency), and $L(s) = C(s)e^{-sT}/s$ is the open-loop transfer function.

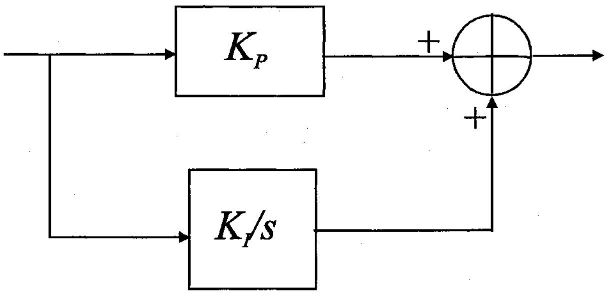


Fig. 5.4 Block diagram of a PI controller - K_p and K_I are the proportional and the integral coefficients respectively (see Section 5.2).

For delayed systems, one can use the Nyquist diagram to investigate the stability conditions, as the inequalities (5.4) and (5.5) are in general difficult to simplify. The Nyquist diagram is the plot of the magnitude and phase of the transfer function evaluated along the

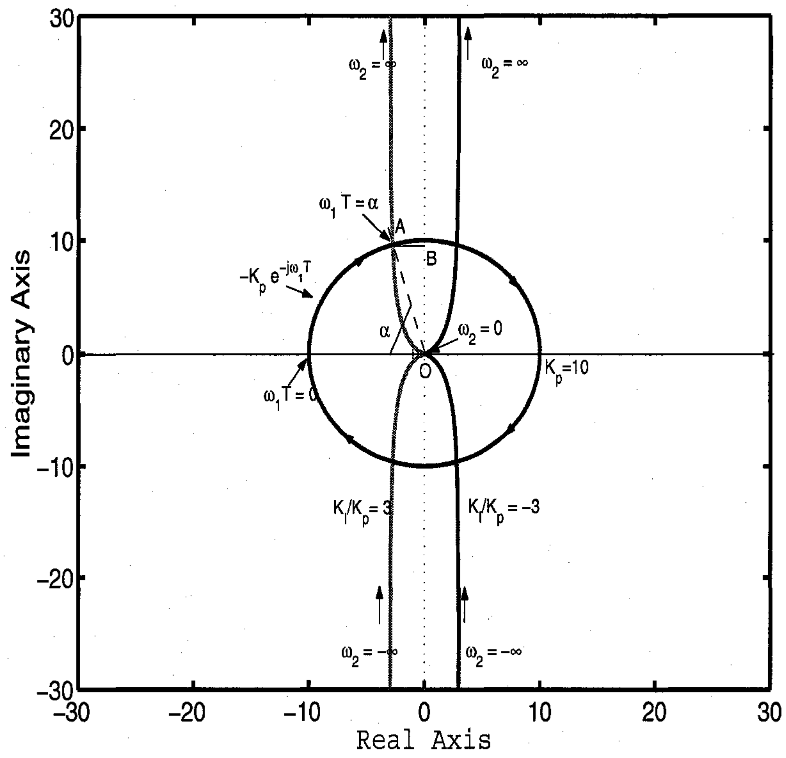


Fig. 5.5 The Nyquist plots of $[G(s)]^{-1}$ and $K_p e^{-sT}$, with $K_I = \pm 30$ and $K_p = 10$, for the system in Figure 5.3. Using this figure we examine the stability of the control system described by equation (5.6). For other values of K_I andK_p similar results are obtained: the two Nyquist plots intersect, so the system is not always stable.

$j\omega$ axis (for a transfer function whose poles are not located on the left side of the imaginary axis in the s plane). Based on the Nyquist criterion, if we plot the Nyquist diagram for the open-loop transfer function of our system, then the number of times that the Nyquist plot circles the $-1 + j0$ point gives the number of poles of the closed-loop system (zeros or roots of the characteristic equation) which are in the right-half plane. The stability condition for a closed-loop system requires that all of the poles of the system have a negative real part. Therefore for stability to hold, the Nyquist plot must not circle the $-1 + j0$ point. The characteristic equation of our system is: $1 + L(s) = 0$, with open-loop transfer function $L(s) = (K_p + K_I/s)e^{-sT}/s$.

For a delayed-system it is not useful to plot the Nyquist diagram of the open-loop transfer function directly, so instead we use Satche's method, as described in [118, 119].

The characteristic equation of our system can be written as:

$$1 + K_pG(s)e^{-sT} = 0, \tag{5.6}$$

where $G(s) = (1 + \frac{K_I}{K_p.s})/s$. The roots of the characteristic equation must lie in the left-half plane for stability. In order to apply the Nyquist criterion we rewrite the characteristic equation as:

$$[G(s)]^{-1} = -K_pe^{-sT}. \tag{5.7}$$

We plot the Nyquist diagrams of the two functions $[G(s)]^{-1}$ and $-K_pe^{-sT}$ and check whether their plots overlap. If the two plots do not overlap then there are no roots of the characteristic equation in the right-half plane and the system is stable. If the two plots overlap then the system is not always stable. In the latter case, it is possible that the system is stable for some restricted range of the system parameters. Figure 5.5 shows the Nyquist diagrams. Note that the plots of e^{-sT} and K_pe^{-sT} are the unit circle and a circle of radius K_p, respectively, regardless of the delay T. The circles are centered at the origin. From this figure we can see that for all values of K_I and K_p the two plots overlap and hence the proposed control system is not always stable. If we assume that K_p and K_I are fixed, then for a restricted set of time delay values T the system is stable, but in general it is unstable.

If at the intersection point of the Nyquist diagrams of $[G(s)]^{-1}$ and $-K_pe^{-sT}$ the values of ω are the same for each function, then there will be a closed-loop pole on the $j\omega$ axis. Here we consider only the values $\omega > 0$. For the Nyquist plot of $[G(s)]^{-1}$ the value of ω at A (the point of intersection of the K_p circle and $G(j\omega)$ plots) is given by $\omega_2^A = (\frac{1}{2} + \sqrt{1 + 4K_I{}^2})^{\frac{1}{2}}$, which is obtained from $|[G(j\omega_2^A)]^{-1}| = K_p$. On the delay contour $(-K_pe^{-T.s})$ the value of ω at A is given by $\omega_1^AT = \alpha$, where $\alpha = \arcsin\frac{OB}{K_p}$ and $OB = Im\{G(j\omega)^{-1}\}|_{\omega=\omega_2^A} = \frac{K_p(\omega_2^A)^3}{K_I{}^2+(\omega_2^A)^2}$. If $\omega_1^A = \omega_2^A$, there is a root of the characteristic equation on the $j\omega$ axis. This case corresponds to the oscillatory situation.

If $\omega_1^A < \omega_2^A$ then the plot of the $[G(s)]^{-1}$ lags behind the delay contour, i.e., the delay contour passes A before the G^{-1} plot. This case corresponds to the unstable situation where the Nyquist plot of the entire system encircles $-1 + j0$ point. If $\omega_1^A > \omega_2^A$ then the delay contour lags behind the plot of $[G(s)]^{-1}$, i.e., the delay contour passes A after the G^{-1} plot. This case corresponds to the stable situation where the Nyquist plot of the entire system does not encircle $-1 + j0$ point (for more detailed discussion of Satche's method see [118]).

5.3 Smith's Principle

Instability is a common problem in delayed systems, since the addition of delays in a system introduces an additional phase lag, resulting in a less stable system (delay decreases the phase margin). To overcome this dilemma a common approach is the use of the Smith predictor, as introduced by Smith in [120]. Using Smith's principle the effect of delay is eliminated from the characteristic equation of a closed-loop control system. However, the original design of the Smith predictor requires a perfect representation of the "actual" plant, which raises the problem of insufficient robustness to modelling errors. In addition to this problem, the original design does not address the problem of disturbance compensation.

During the past twenty years, researchers have analyzed properties of the Smith predictor including response time, disturbance compensation characteristics and robustness; improvements and extensions have been proposed, particularly addressing the case of integrative or unstable plants [8,9,121–127]. In some of these works only an appropriate tuning of the parameters is proposed. In other cases the structure of the controller is different from Smith's original idea. Aström et al. propose a new Smith predictor in [8] that overcomes some of the shortcomings of the original design. The main contribution of this modified Smith predictor is the decoupling of the disturbance response from the setpoint response. Aström's approach also improves the setpoint response time. Several other modifications and simplifications have been introduced [9,121,125,127] to decouple setpoint response from the disturbance response, primarily for the case of integral processes. Liu et al. proposed a technique for controlling unstable plants (poles in the right half s plane) with long dead-time [128]. In order to use the Smith predictor (or the modified versions) in industrial applications, simple tuning procedures must be developed. Some papers have been written addressing this task: Hagglund [129] proposed a three parameter Smith predictor controller with a simple tuning procedure; Normey-Rico et al. have proposed techniques to improve robustness whilst maintaining a simple structure and tuning procedure [124,126]. In this section we introduce the Smith predictor [120] and its modifications with respect to disturbance decoupling and compensation [8,9,125] for the case of integral processes.

5.3.1 The Original Design of Smith's controller

Figure 5.6 shows a typical time-delay control system, with reference signal $R(s)$ and disturbance $D(s)$, in which the backward delay is pushed to the forward path. The transfer

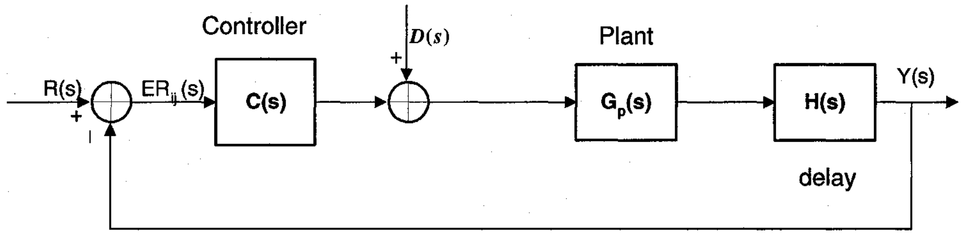


Fig. 5.6 Schematic of a typical time-delay system, with backward delay implied in the forward path. $R(s)$ is the reference signal, $ER_{ij}(s)$ is the error signal, $D(s)$ is the disturbance, and $Y(s)$ is the output. The transfer function of this system is given by (5.8).

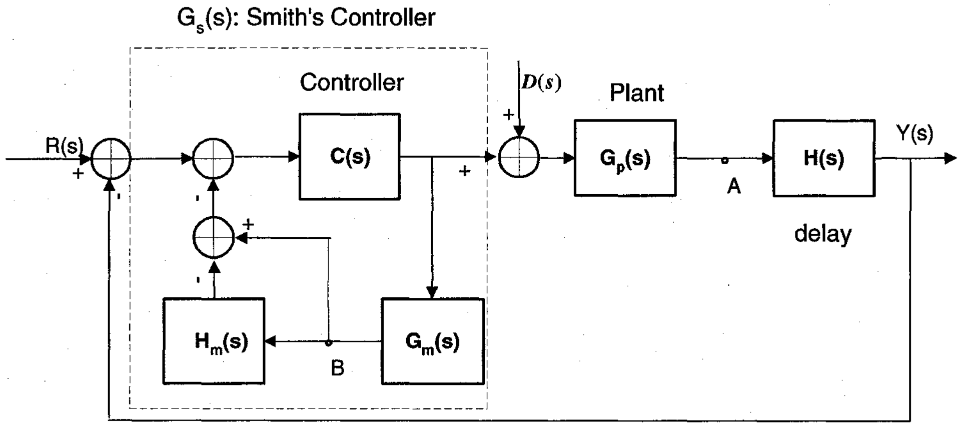


Fig. 5.7 A Smith predictor for a time-delay system. G_m represents the plant's dynamic model and H_m represents the system's delay model. The controller $G_s(s)$ has removed the effect of time delay from the characteristic equation of the system (see (5.10)).

function of this system is

$$\frac{Y(s)}{R(s)} = \frac{C(s)G_p(s)H(s)}{1 + C(s)G_p(s)H(s)}. \tag{5.8}$$

The goal of the Smith predictor is to remove the time delay term from the characteristic equation (the denominator of the transfer function). We wish to modify $C(s)$ so that the transfer function of the system becomes (under ideal conditions):

$$\frac{Y(s)}{R(s)} = \frac{C(s)G_p(s)H(s)}{1 + C(s)G_p(s)}. \tag{5.9}$$

Suppose that $G(s)$ is the modification of $C(s)$ based on the Smith principle. We need to design $G(s)$ as a combination of $C(s)$ and other elements such that (5.9) holds. If in (5.8) we replace $C(s)$ by $G_s(s)$, then the transfer function is:

$$\frac{Y(s)}{R(s)} = \frac{G_s(s)G_p(s)H(s)}{1 + G_s(s)G_p(s)H(s)}. \tag{5.10}$$

Equating (5.9) and (5.10), we have

$$\frac{G_s(s)G_p(s)H(s)}{1 + G_s(s)G_p(s)H(s)} = \frac{C(s)G_p(s)H(s)}{1 + C(s)G_p(s)}, \tag{5.11}$$

which leads to

$$G_s(s) = \frac{C(s)}{1 + C(s)G_p(s)(1 - H(s))}. \tag{5.12}$$

In constructing this controller, we do not, strictly speaking, have true knowledge of the plant $G_p(s)$ or the delay $H(s)$, so we use models G_m and H_m, respectively (in the ideal case $H_m = H$ and $G_m = G_p$). Figure 5.7 shows the resultant original design of the Smith predictor.

Smith's principal can be used to realize any form of controller that we wish to obtain, using the following steps:

- *perform the delay-free design to obtain $C(s)$,*
- *realize the controller G_s.*

In practice it is very common for the plant's behavior (G_p) to change over time, and this necessitates a subsequent change in the controller. Landau et al. proposed an adaptive control scheme that adapts the Smith predictor to the variations in the delay and the plant transfer function [130]. For the moment we assume that G_p is fixed and known and $G_m = G_p$. We also assume that the forward and backward delays are fixed.

An Interpretation of Smith's Method as a Predictor

For an ideal system realization the signal at point B in Figure 5.7, $S_B(t)$, is a copy of the signal at point A, since the input to both G_p and G_m in the absence of disturbance are the same and $G_p = G_m$. Also the signal at point B is a prediction of the output $Y(t)$, which is a time-delayed version of the signal at A. From this point of view, Smith's technique can be regarded as a predictor, with $S_B(t) = Y(t + T)$ [118].

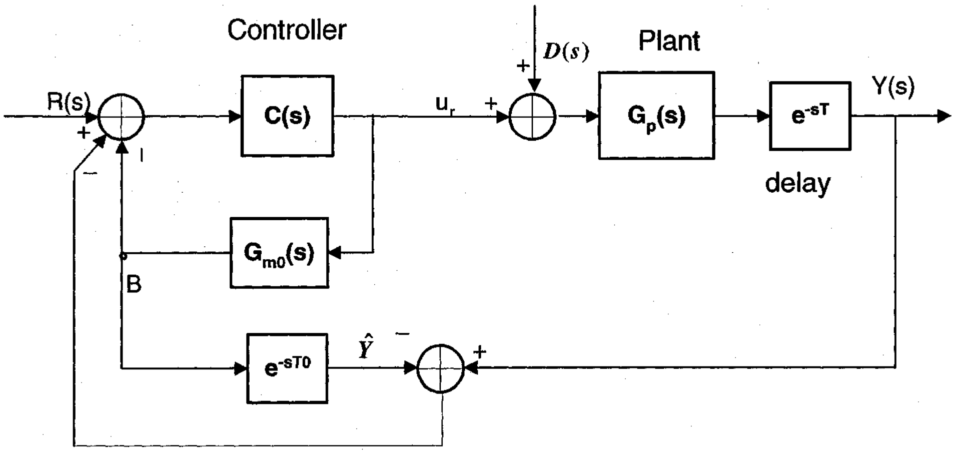


Fig. 5.8 An equivalent representation for the Smith predictor. The transfer function of this system is identical to that of the system in Figure 5.7.

Reduction of Output Disturbance

The Smith predictor removes the effect of delay from the characteristic equation of the setpoint response. There has been a considerable effort to modify the Smith predictor's load disturbance response. Figure 5.8 shows an equivalent representation of the Smith predictor that is frequently used as the basis for the modifications.

5.3.2 Aström's Smith Predictor

The controller proposed by Aström et al. provides faster response time and better load disturbance response [8] compared to the Smith's controller. This approach was specifically proposed for controlling a process with an integrator and long dead-time (delay), but the idea can be extended to any form of process. Figure 5.9 shows the structure of Aström's Smith predictor, where $C(s) = k$ is the gain, $G_p(s) = G_{m0}(s) = 1/s$, $T_0 = T$, and $M(s)$ is the compensator. The setpoint transfer function is given by:

$$H_r(s) \triangleq \left.\frac{Y(s)}{R(s)}\right|_{d=0} = \frac{ke^{-sT}(1 + M(s)\frac{1}{s}e^{-sT})}{s + k(1 + M(s)\frac{1}{s}e^{-sT})} = \frac{k}{s+k}e^{-sT}. \tag{5.13}$$

The setpoint response time can be improved by choosing an appropriate value for the controller gain k. The P controller can also be replaced by other types of controller, for

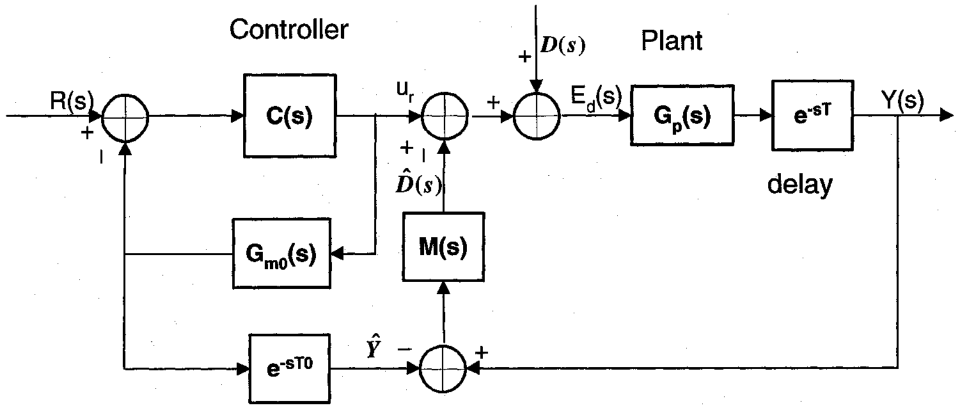


Fig. 5.9 The structure of Aström's Smith predictor; by choosing an appropriate compensator, $M(s)$, the effect of disturbance is eliminated from the setpoint response. The compensator described by (5.15) was introduced by Aström et al. [8].

example, a PI controller. The disturbance response is given by

$$H_d(s) \triangleq \left.\frac{Y(s)}{R(s)}\right|_{r=0} = \frac{\frac{1}{s}e^{-sT}}{1+M(s)\frac{1}{s}e^{-sT}}. \tag{5.14}$$

The setpoint response can be controlled by the gain k; we can control the disturbance response by choosing an appropriate compensator $M(s)$. Thus, the scheme has decoupled the disturbance response from the setpoint response (i.e., the setpoint response and the disturbance response can be optimized independently, resulting in a system with two degrees-of-freedom). Aström et al. proposed the following transfer function for $M(s)$, deriving it through the application of robust control techniques:

$$M(s) = \frac{k_4 + \frac{k_3}{s}}{1 + k_1 + \frac{k_2}{s} + \frac{k_3}{s^2} - (\frac{k_4}{s} + \frac{k_3}{s^2}) - (\frac{k_4}{s} + \frac{k_3}{s^2})e^{-sT}}\ , \tag{5.15}$$

where $k_4 = k_2 + k_3T$. With this choice the load disturbance response is given by

$$H_d(s) = \frac{e^{-sT}(s^2(1+k_1) + k_2s + k_3 - (k_4s + k3)e^{-sT})}{s(s^2(1+k_1) + sk_2 + k_3)}\ . \tag{5.16}$$

The choice of $M(s)$ in (5.15) eliminates the effect of delay from the denominator of the disturbance transfer function and also decouples the load disturbance response from the setpoint response. However, this design involves too many tuning parameters; setting these to appropriate values is not simple to achieve in practice. Moreover, the use of a robust controller for the entire system is too restrictive [131].

5.3.3 Modified Smith Predictor

We now review two simple modified Smith predictors. The first, proposed by Zhang and Sun in [125], extends Aström's method to the general integrator/time delay process. In this design the assumption is that $G_p(s) = \frac{1}{Ls}$. If the model is exact we have $G_{m0} = G_p$ and $T_0 = T$, and the setpoint response, for a controller $C(s)$, is

$$H_r(s) = \frac{C(s)}{Ls + C(s)} e^{-sT}. \tag{5.17}$$

The disturbance response is:

$$H_d(s) = \frac{G_p(s)e^{-sT}}{1 + M(s)G_p(s)e^{-sT}} = \frac{e^{-sT}}{Ls + M(s)e^{-sT}}. \tag{5.18}$$

In an ideal case $C(s)$ does not have any effect on the disturbance response. Whereas the goal of the Smith predictor is to eliminate the effect of the time delay on the setpoint response, Zhang and Sun designed $M(s)$ in order to eliminate the time delay from the characteristic equation of the disturbance response, choosing

$$M(s) = \frac{sM_0(s)}{1 - sM_0(s)G_{m0}(s)e^{-sT}}, \tag{5.19}$$

where $M_0(s)$ is a rational function. Substituting (5.19) into (5.18), we obtain

$$H_d(s) = \left(1 - \frac{M_0(s)}{L} e^{-sT}\right) \frac{1}{Ls} e^{-sT}. \tag{5.20}$$

The time delay is thus eliminated from the characteristic equation of $H_d(s)$ and the poles of $H_d(s)$ are those of $M_0(s)$. Hence, $M_0(s)$ should be a stable transfer function (i.e. a transfer functions with all of its poles in the left-half plane). In order to be physically realizable it must be a strictly proper rational transfer function (i.e., it must have form $\frac{P(s)}{Q(s)}$, with $P(s)$ and $Q(s)$ polynomials and the degree of P less than that of Q). The steady-state performance is an important characteristic of a control system; to obtain a zero steady-state disturbance error, two constraints on M_0 can be developed.

First, we wish to eliminate the effect of the disturbance from the steady-state (zero frequency) output. The output $Y(s) = H_r(s)R(s) + H_d(s)D(s)$, so this implies that we must have

$$\lim_{s\to 0} H_d(s) = 0,$$

which is equivalent to (using l'Hospital's rule)

$$\lim_{s\to 0} \frac{d}{ds}\left(\left(1 - \frac{M_0(s)}{L}e^{-sT}\right)\frac{1}{L}e^{-sT}\right) = 0. \tag{5.21}$$

This equation leads to the following constraint:

$$2TM_0(0) - TL - M_0'(0) = 0, \tag{5.22}$$

Second, we wish to set the steady-state error signal ($E_d(s) = D(s) - \hat{D}(s)$) to zero for the case of a "step input" disturbance ($D(s) = \frac{1}{s}$). This requires that $e_d(\infty) = 0$, which, based on the final value theorem, implies that $\lim_{s\to 0} sE_d(s) = 0$. From Figure 5.9 we have:

$$Y(s) = E_d(s)G_p(s)e^{-sT}.$$

The output (to the disturbance input only) is given by:

$$Y(s) = H_d(s)D(s)$$

Equating these two relationships, we have:

$$H_d(s)D(s) = E_d(s)G_p(s)e^{-sT},$$

For the step disturbance input, $D(s) = \frac{1}{s}$, this implies that

$$E_d(s) = H_d(s)e^{sT},$$

inducing the following relationship

$$\lim_{s\to 0} sE_d(s) = \lim_{s\to 0}\left(1 - \frac{M_0(s)}{L}e^{-sT}\right)\frac{1}{L} = 0,$$

and finally, resulting in the constraint

$$M_0(0) = L. \tag{5.23}$$

So $M_0(s)$ should satisfy (5.22) and (5.23) and be a strictly proper function,

$$M_0(s) = \frac{\beta_m s^m + \ldots + \beta_1 s + \beta_0}{(\lambda_2 s + 1)^n}, \quad m < n, \quad \lambda_2 > 0 \tag{5.24}$$

for some set of constants $\beta_0, ..., \beta_m$ and λ_2. From (5.22) we conclude that the minimum order of the transfer function $M_0(s)$ is 2, and based on (5.22) and (5.23) we have:

$$M_0(s) = \frac{(LT + 2\lambda_2 L)s + L}{(\lambda_2 s + 1)^2}, \tag{5.25}$$

where λ_2 remains as a tuning parameter that can be used to adjust the disturbance response. The disturbance transfer function becomes:

$$H_d(s) = \frac{(\lambda_2 s + 1)^2 - ((T + 2\lambda_2)s + 1)e^{-sT}}{(\lambda_2 s + 1)^2 Ls} e^{-sT} \tag{5.26}$$

By using a model for the compensator $M(s)$ and a minimum order $M_0(s)$, the time delay is removed from the disturbance characteristic equation and a type I system is obtained. A type I system is one that integrates the output (s in the denominator); for such a system the steady-state error to a step input is zero. $M_0(s)$ is in fact a low pass transfer function, and its bandwidth is determined by λ_2. For the setpoint controller the assumption is that $C(s) = \lambda_1$. So in this design λ_1 and λ_2 play the central roles for the setpoint and disturbance responses respectively. Large values of these two factors correspond to a high speed of response and poor robustness. Smaller values of these factors correspond to a low speed of response and good robustness.

Matausek et al. describe an alternative modified Smith predictor in [9]. This design chooses $M(S) = K_0$ and $C(s) = K_r$, and similar to [125], $G_p(s) = \frac{K_p}{s}$. The main goal in this system is to eliminate the effect of disturbance on the output using a stable controller, so the disturbance controller is designed based on stability criteria. Note that unlike the design of Zhang and Sun [125], Matausek et al. do not attempt to remove the delay term from the disturbance characteristic equation. There is also an additional feedback from the output to the main controller (see Figure 5.10). This feedback was removed in Aström's original modification of the Smith predictor. The setpoint and disturbance transfer function of the Matausek design are:

$$H_r(s) = \frac{K_p K_r e^{-sT}}{s + K_p K_r}. \tag{5.27}$$

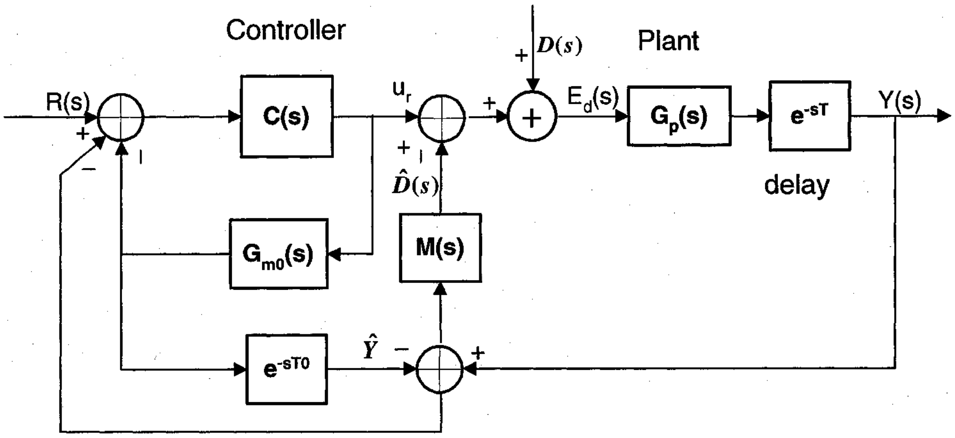


Fig. 5.10 Smith predictor with a disturbance compensator $M(s)$ proposed by Matausek et al. [9]; compared to Figure 5.9, this design has a feedback from output to the input.

$$H_d(s) = \frac{K_p[s + K_pK_r(1 - e^{-sT})]e^{-sT}}{(s + K_pK_r)(s + K_0K_pe^{-sT})}. \tag{5.28}$$

In order to eliminate the load disturbance steady-state response it is necessary to have $\lim_{s\to 0} H_d(s) = 0$, which is possible if $K_0 \neq 0$. Under this condition we have

$$\lim_{t\to\infty} y(t) = r. \tag{5.29}$$

From Figure 5.10 and (5.37), we obtain

$$u_r = K_r(r - y - u_r\frac{K_p}{s})$$

which implies that $\lim_{t\to\infty} u_r(t) = 0$, and consequently

$$\lim_{t\to\infty} \hat{d}(t) = d. \tag{5.30}$$

Thus the signal $\hat{d}(t)$ is an estimate of the constant input load disturbance d. From (5.28) it follows that the stability of the modified Smith predictor depends on the roots of the characteristic equation

$$(s + K_pK_r)(s + K_0K_pe^{-sT}) = 0. \tag{5.31}$$

The first term implies that $K_pK_r > 0$ must be satisfied. In order to find the roots of the second term and the ultimate gains for which the roots are located at the left-half plane we assume that the controllable parameter is K_0 and that we have a closed loop system whose characteristic equation is described by:

$$(s + K_0 K_p e^{-sT}). \tag{5.32}$$

This is purely an "imaginary" or hypothetical system. For the analysis, (5.32) is rewritten in the form $1 + W(s)$ where

$$W(s) = \frac{K_0 K_p}{s} e^{-sT}.$$

The Nyquist criterion can then be applied to find the ultimate gain K_{0u}, which indicates the supremum of K_0 for which the system described by the characteristic equation (5.32) is stable. K_{0u} is obtained by setting the phase margin and gain margin of the "imaginary" system described by (5.32) to zero. Note that these margins are not those of the actual control system; the construction of a control system described by (5.32) simply permits the application of standard techniques for finding the locations of the poles of a closed loop system. We have

$$W(j\omega) = \frac{K_0 K_p}{j\omega} e^{-jT\omega} = \frac{K_0 K_p}{\omega} e^{-j(T\omega + \pi/2)}$$

$$\phi M = \pi + \angle W(j\omega) = \pi/2 + T\omega = 0$$

$$GM = -20 \log |W(j\omega)| = -20 \log \frac{K_0 K_p}{\omega} = 0.$$

Thus the ultimate gain K_{0u} is

$$K_{0u} = \frac{\pi}{2 K_p T}$$

And for all $K_0 < K_{0u}$ the system is stable ($\phi M > 0$). Matausek et al. provide a concrete example by analysing the case where $K_0 = \frac{1}{2K_pT}$ (corresponding to $\phi M = 61.3065°$); this choice provides satisfactory closed loop system performance both in terms of setpoint and load disturbance responses.

5.4 AAPN Smith Controller

Our controller design is based on the modified Smith predictor by Matausek et al. [9]. Figure 5.11 shows our controller for an AAPN. The inputs to the system are $r_{ij}(t), a(t)$ and $\hat{a}(t)$, and the output is $q_{p_{ij}}(t)$. We consider the arrival rate $a(t)$ and its prediction $\hat{a}(t)$ as disturbances, since it is desirable to reduce the effect of arrival rate variations from the queue size, and make the queue size follow the reference signal, r_{ij}. It is also assumed that

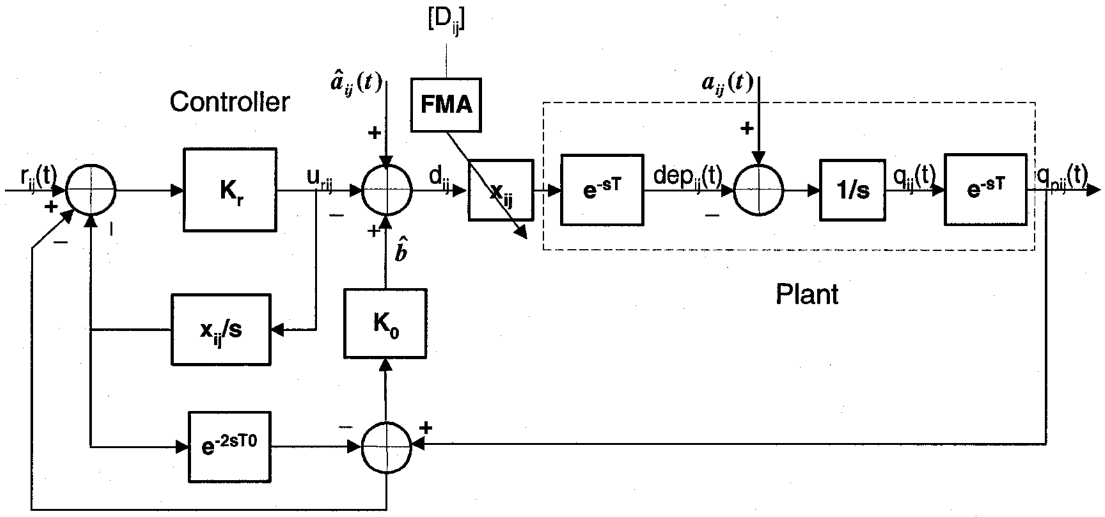


Fig. 5.11 Schematic of a control system for AAPN with long propagation delay using the Smith principal. This model is based on the modified Smith predictor proposed by Matausek et al. [9].

the delay T and its model T_0 are equal. In this design the response of the system is given by:

$$Q_{p_{ij}}(s) = H_r(s)R(s) + H_d(s)A(s) + \hat{H}_d(s)\hat{A}(s), \tag{5.33}$$

where:

$$H_r(s) \triangleq \left.\frac{Q_{p_{ij}}(s)}{R(s)}\right|_{a=0,\hat{a}=0} = \frac{\mathrm{x}_{ij}K_r e^{-sT}}{s+\mathrm{x}_{ij}K_r}. \tag{5.34}$$

$$H_d(s) \triangleq \left.\frac{Q_{p_{ij}}(s)}{R(s)}\right|_{r=0,\hat{a}=0} = \frac{e^{-sT}[s-\mathrm{x}_{ij}K_r(1-e^{-2sT})]}{(s+\mathrm{x}_{ij}K_r)(s+K_0\mathrm{x}_{ij}e^{-2sT})}. \tag{5.35}$$

$$\hat{H}_d(s) \triangleq \left.\frac{Q_{p_{ij}}(s)}{R(s)}\right|_{r=0,a=0} = \frac{\mathrm{x}_{ij}e^{-2sT}[s-\mathrm{x}_{ij}K_r(1-e^{-2sT})]}{(s+\mathrm{x}_{ij}K_r)(s+K_0\mathrm{x}_{ij}e^{-2sT})} = \mathrm{x}_{ij}e^{-sT}H_d(s). \tag{5.36}$$

We aim to obtain equal steady-state queue lengths for all of the VOQs, and therefore we consider the arrival rate and its prediction as the disturbance loads. In order to eliminate the load disturbance steady-state response it is necessary to have $\lim_{s\to 0} H_d(s) = 0$, which is possible if $K_0 \neq 0$. Under this condition and based on the final value theorem we have:

$$\lim_{t\to\infty} q_{p_{ij}}(t) = \lim_{s\to 0} R_{ij}(s)H_r(s) = r_{ij}. \tag{5.37}$$

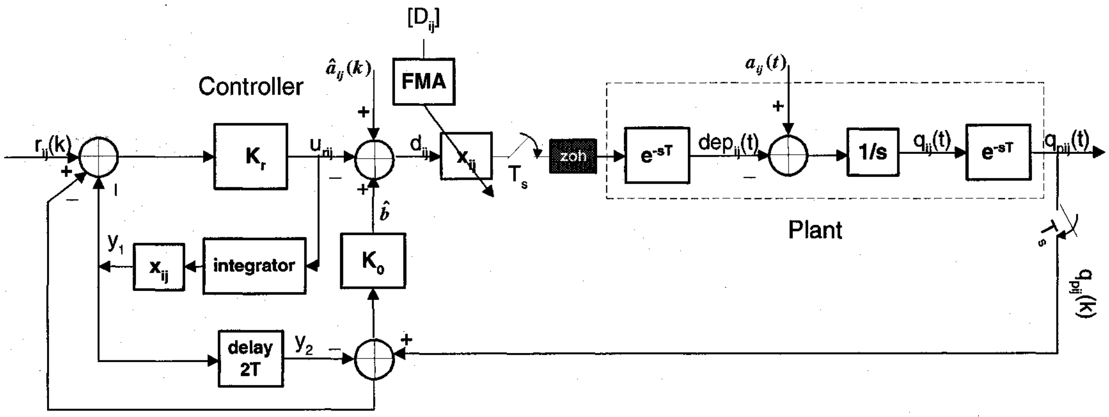


Fig. 5.12 Discrete time representation of the continuous control system (Figure 5.11). The discrete time system is approximated from the continuous-time system using the Delta transformation.

Following the approach presented in Section 5.3.3 we obtain the ultimate gain K_{0u} as:

$$K_{0u} = \frac{\pi}{4\mathbf{x}_{ij}T}, \tag{5.38}$$

and for all $K_0 < K_{0u}$ the system is stable ($\phi M > 0$).

5.4.1 Discrete-Time System Equations

To obtain the equivalent discrete-time system equations one simple approach is to design a digital control system using the *Delta transform*. Then the input to the plant is converted to continuous form with zero-order-hold [132]. In the *Delta transform* approach we approximate the differential equation ($\frac{dy}{dt}$) with $\frac{y(t+\Delta)-y(t)}{\Delta}$ [132]. For the control system presented in Figure 5.12 the discrete time equations (for $T_0 = T$) are approximated from the continuous form as:

$$\begin{aligned}
\mathsf{d}_{ij}(k) &= \hat{a}_{ij}(k) - u_{rij}(k) + K_0 q_{pij}(k) - K_0 y_2(k),\\
y_1(k) &= y_1(k-1) + \mathbf{x}_{ij}(k-1)u_{rij}(k-1)T_s,\\
y_2(k) &= y_1(k - \frac{2T}{T_s}),\\
u_{rij}(k) &= K_r(-y_1(k) + y_2(k) - q_{pij}(k) + r_{ij}(k)).
\end{aligned} \tag{5.39}$$

Given that $\lambda = \frac{T}{T_s}$, then we have:

$$u_{rij}(k) = K_r \left(- \sum_{p=1}^{\lambda} \mathbf{x}_{ij}(k-p) u_{rij}(k-p) T_s - q_{pij}(k) + r_{ij}(k) \right). \tag{5.40}$$

This rate adjustment involves the divergence of each queue length from the average queue length, r_{ij}, as well as the amount of the queue backlog $q_{pij}(k)$ through the controller parameters K_0 and K_r. The role of the Smith controller is to take into account the effect of rate adjustment on the queues during the λ previous frames for which there is no feedback available to the controller.

5.4.2 Control Parameters Design

To avoid unwanted fluctuations in the queue lengths due to Smith controller we set the controller such that it affects the system only when there is a queue length increase greater than several time slots (depending on the frame length and the number of nodes) during one frame. This limitation serves as an anti-aliasing for our control system, preventing overcorrection to small high frequency fluctuations of the queue length.

The gain of the controller K_r should be chosen such that the equivalent discrete-time system is stable. Standard digital control theory suggests that the sampling period should be at most half the time constant of the continuous system $(1/\mathbf{x}_{ij}K_r)$. Since our sampling time is the frame duration (T_s), we set:

$$K_r < \frac{1}{2\mathbf{x}_{ij}T_s}. \tag{5.41}$$

Using a fixed controller gain can result in undesirable behavior. A small gain does not provide sufficiently fast response to traffic changes, but a large gain results in overreaction to minor fluctuations. An adaptive gain can provide a good compromise. We design the controller such that the gain K_r adapts to the size of the queue variations:

$$K_r(k) = \min\{A \exp(C\Delta q_p), \frac{1}{2\mathbf{x}_{ij}T_s}\}, \tag{5.42}$$

where $\Delta q_p = q_p(k) - q_p(k-1)$. The choice of the constants A and C determines how fast the system reacts to traffic changes and whether there are residual oscillations. Simulations are used to determine a suitable range of values. Also to avoid overcompensation due to

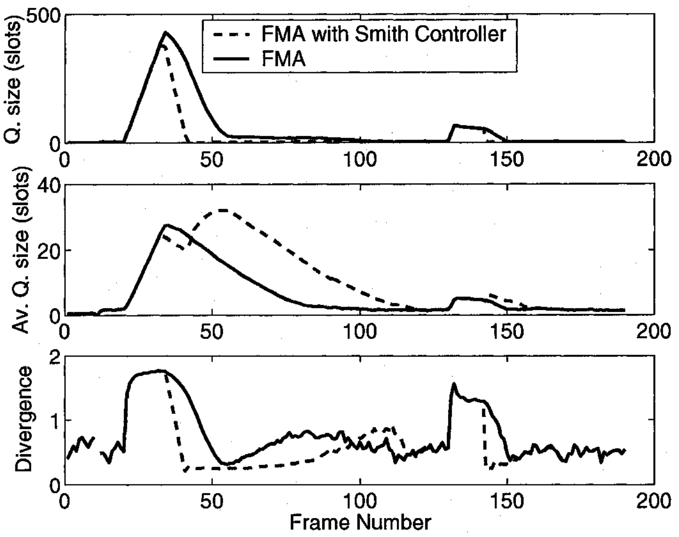


Fig. 5.13 *Adaptive gains*: The impact of the feedback controller for the simulation conditions in Section V (Scenario A). Top panel: Average queue length for VOQ experiencing the heavy connection. Middle panel: Average queue lengths of all VOQs. Bottom panel: Relative fairness factor (divergence) as defined by (5.43). During the periods of increased load the queue length of the heavy connection increases (top panel). The response to this sudden change is faster when the Smith controller is used. The rapid reaction of the Smith controller affects the average queue length more than the system without the Smith controller (middle panel), and so the average divergence increases around frame 100 (bottom panel).

large control gains we use an ad-hoc fast-start slow-finish compensation procedure in which we reduce the gains of the controller by a factor of 0.05 two frames after activation of the Smith controller. This approach ensures that the heavy connections receive a considerable amount of bandwidth through very high gains during two frames to empty their queue. This period has an inverse relationship with the gain factors K_r and K_0.

5.5 Simulation Performance

In this section we report the results of simulations of the scheduling approaches performed using OPNET Modeler [90]. We use simulations to investigate how the incorporation of the Smith controller impacts the response time of our system when there is a sudden change in traffic arrival rates. Faster response to a sudden change avoids buffer overflow as well as expiration of packets due to long waiting times. We are also interested in exploring the effect on the fairness in the system.

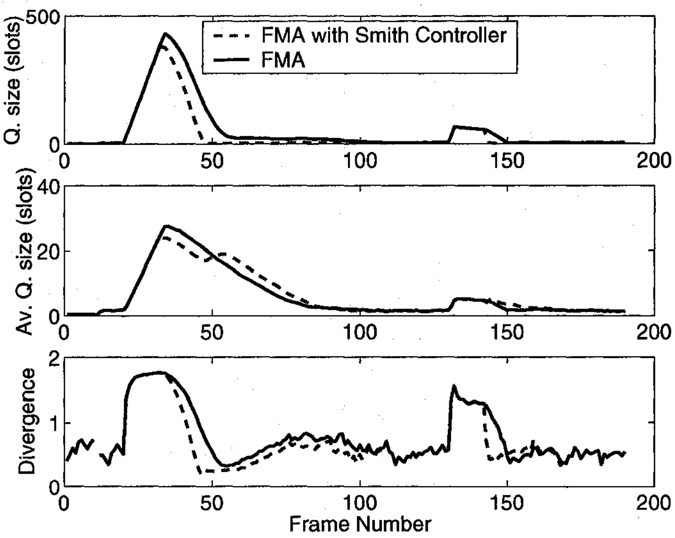


Fig. 5.14 *Fast-start slow-finish compensation*: The impact of the feedback controller with fast-start slow-finish compensation for the simulation conditions in Section V (Scenario A). Top panel: Average queue length for VOQ experiencing the heavy load. Middle panel: Average queue lengths of all VOQs. Bottom panel: Relative fairness factor (divergence) as defined by (5.43). The initial response is very fast but the queue of the heavy connection then drains slower (top panel), which smooths the effect on the queues of the non-heavy connections (middle panel). The average divergence has been improved compared to Figure 5.13 (bottom panel).

We measure an average relative fairness factor (*divergence*), which we define for source node j as:

$$\delta_j = \frac{\sum_{i,i\neq j} |q_{p_{ji}} - \frac{\sum_{i,i\neq j} q_{p_{ji}}}{(n-1)}|}{\sum_{i,i\neq j} q_{p_{ji}}} \tag{5.43}$$

This factor measures the average divergence of the queue lengths of all VOQs at source node j from the overall average. It thus provides a good indication of the degree of equality of waiting times for packets in different queues (a value closer to zero indicates better fairness).

We performed simulations on a 16 edge-node star topology network. The links in the network have capacity 10 Gbps and the distance between each edge node and the photonic switch is 5 msec. A time slot is of length 10 μsec, and a frame has a fixed length of 1 msec (or 100 slots). Every experiment was run for a duration of 0.5 sec (equal to 500 frame durations). We investigate two traffic scenarios. In both scenarios, the average arrival rates to the VOQs are equal except for two periods (frames 20-32 and frames 130-132) during which the arrival rate of traffic from one source to one destination increases by a factor of 10. The two traffic scenarios are as follows.

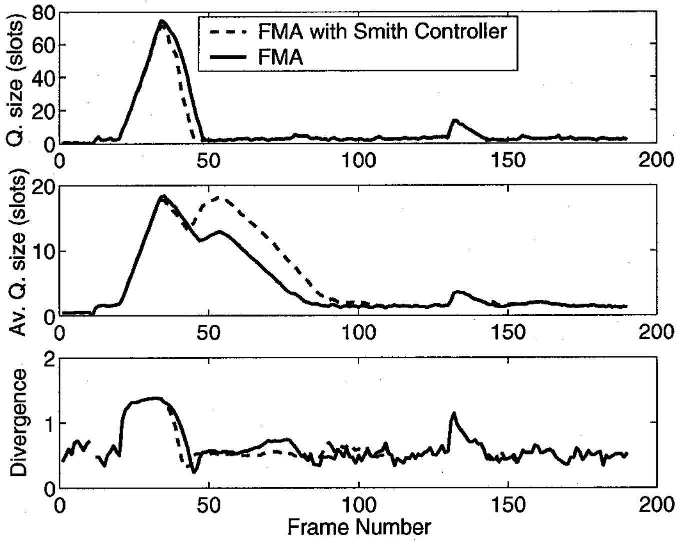


Fig. 5.15 The effect of the Smith controller for the simulation conditions in Section 5.5 with four heavy connections. The distribution of the destinations is uniform except for two periods, from frames 20-32 and 130-132, during which four of the connections experience a sudden high load, three times greater than their regular rates. The Smith controller provides a faster response to this change (the top panel shows the queue length of one of the heavy connections). The middle panel shows the average queue length of the connections. The bottom panel shows the average divergence from the mean of the queue lengths.

Scenario A: The arrival distribution of the data packets is Poisson with average arrival rate of 9 Gbps during the baseline periods; the packet size distribution is exponential with mean size of 1000 bits.

Scenario B: Six Pareto ($\alpha = 1.9$) on-off sources are connected to each edge node. The mean on-period is 0.33 msec and mean off-period is 1.6 msec. The average rates are 9 Gbps during the on-period.

The top panel of Figure 5.13 compares the queue lengths of the VOQ carrying the heavy connection when using FMA with and without the Smith controller for the case of adaptive gains with $A = 63/\mathbf{x}_{ij}$ and $C = 0.08$ in (5.42). The Smith controller decreases the response time substantially, reducing the queue length of the heavy connection much faster than the pure feed-forward controller derived by applying only FMA. This rapid response in the following frames empties the queue of the heavy connection too fast, causing the other connections to starve and develop (relatively) large queue lengths, as shown in the middle panel. The bottom panel of Figure 5.13 compares the average divergences. During the initial period of heavy traffic, the fast draining of the long queue improves fairness,

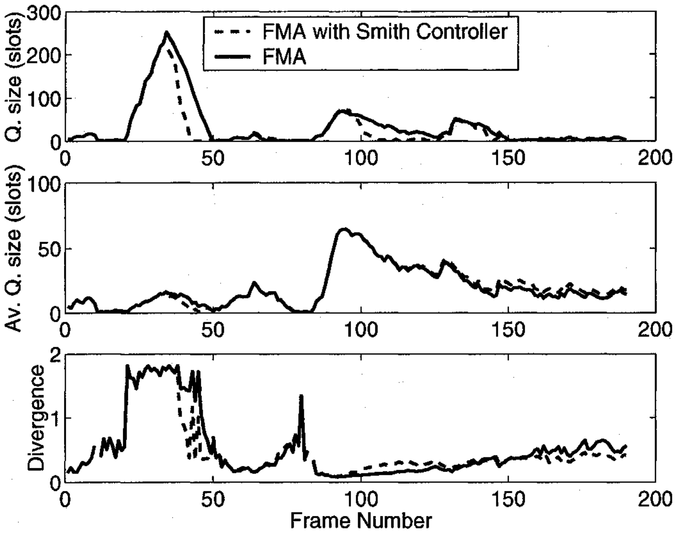


Fig. 5.16 The effect of the Smith controller under the bursty traffic conditions in Section V (Scenario B). Top panel: Average queue length for VOQ experiencing the heavy load. Middle panel: Average queue lengths of all VOQs. Bottom panel: Relative fairness factor (divergence) as defined by (5.43). As for Poisson arrivals, the Smith controller provides a faster response, draining the heavy VOQs at a faster rate (top panel). There is minimal negative effect on the other queues (middle panel) or the fairness, as measured by the average divergence (bottom panel) even when a large burst arrives (frame 83).

but later (around frame 100) the overcompensation results in a slight increase in average divergence. As outlined in Section 5.4.2, the use of fast-start slow-finish compensation can improve the performance of the controller. Figure 5.14 shows the performance of the Smith controller with this gain adjustment. The response to large changes in the queue length remains fast, but there is no starvation of the other queues, so the divergence remains low throughout the simulation.

We also performed simulations for the case that 4 connections experience a sudden high load for 12 frames. Around frame 20 the arrival rates of these 4 connections increase suddenly to 3 times of their regular rates. As Figure 5.15 shows the queues do not change as fast as those in the example of a single heavy connection. Therefore the speed of the response is not dramatically improved. However there are improvements in both the response times as well as the average queue length and divergence from the mean of the queues.

Figure 5.16 examines the performance in response to bursty traffic and one heavy connection as described in Scenario B , which is more unpredictable and thus poses a greater challenge for the Smith controller. The simulations indicate that the Smith controller still provides better drainage of the queues experiencing severe congestion. There is minimal negative effect on other queues or long-term fairness.

5.6 Summary

In Chapter 5, we proposed a feedback control system that compensates for scheduling errors due to mispredictions and rejection in single-hop communication networks with large propagation delays. Scheduling in wide-area networks is based on prediction of future traffic arrivals. If the system is not designed such that the unallocated requests are recorded, or the prediction errors are compensated, there is a high chance that some of the buffers will overflow. This is an unstable situation in control jargon. Despite the large volume of research devoted to scheduling, there has been no research on incorporating the theory of control to address the traffic prediction errors and unfair rejection. This is mainly due to the fact that the scheduling algorithms have usually considered small networks or delay-free systems, in which the need for error compensation is not evident. In this thesis we applied for the first time the theory of linear control to the systems employing scheduling as their bandwidth sharing method with large delays.

The controller design is based on the Smith principle, which removes the destabilizing delays from the feedback loop by using a "loop cancelation" technique. To be able to use the controller for a wide range of queue length variations we adopted adaptive gains and some ad-hoc approaches to reduce the controller side effects. We have shown through simulations that our controller reduces the response time to a sudden change in a queue length and imparts fairness by controlling the divergence from the average queue length. This work can be extended to develop simple procedures for tuning the control system parameters that are insensitive to changes in traffic.

The model developed in this study was based on the exact knowledge about the delay. While this assumption is violated if the queueing delay is more than 1 frame duration. Future work can focus on exploring the incorporation of methods from robust control to address the scenario where there are delay variations over time due to queueing packets for a relatively long time.

Chapter 6

Conclusion

6.1 Summary

In this thesis we investigated bandwidth allocation and scheduling problem in single-hop all-photonic networks with cross-connect switches and large propagation delays. We formulated fixed-length frame scheduling of point-to-point transmissions as an optimization problem, studied its NP-hardness and proposed algorithms that addressed fairness. This is the first time that a this scheduling problem has been studied in depth. Most of the frame-based scheduling algorithms proposed in the literature [26–28] have focused on *variable-length* frames. There has been some study on designing schedules with fixed lengths, but there has been no discussion of the NP-hardness of an optimum solution for this problem. Moreover, the solutions did not address the problem of utilizing the frame capacity in case of admissible traffic or the problem of rejecting some of the requests based on an appropriate criterion when the traffic is inadmissible. When the predicted demand is insufficient to fill the schedule completely, we need a policy to divide the extra time slots amongst active connections to ensure the network resources are utilized. Similarly when the overall request is more than the capacity, some of the predicted demand must be rejected. In very recent work, Peng et al. [133] addressed this problem, proposing an algorithm that attempted to maximize the "similarity" between the original traffic matrix and the modified version, but this work did not discuss or solve the fairness issue.

The second major problem that we addressed in this thesis was instability. Regardless of which open-loop scheduling policy we adopt, the network can become unstable or unfair due to traffic prediction error. We proposed a control architecture based on the Smith principle to compensate for errors, imposing queue stability and fairness.

In Chapter 1 we introduced the proposed network architecture for AAPN and our research objectives. In Chapter 2 we discussed photonic networks and different aspects of data communication in these networks such as routing and wavelength assignment, and switching schemes. This chapter provided background information on photonic networking which is the basis for our research. Chapter 3 presented a comprehensive literature review on scheduling. The chapter discussed variable-length and fixed-length frame scheduling for broadcast transmissions. It also examined variable-length frame and slot-based scheduling for point-to-point transmissions.

In Chapter 4 we formulated the bandwidth allocation problem in AAPN as a scheduling problem with the objective of minimizing rejection whilst reducing the number of switch reconfigurations. We proposed several scheduling algorithms, which address different objectives. Our first design, Minimum Cost Search algorithm, is a simple greedy algorithm that tries to minimize the blocking probability of time slots while controlling the number of switchings using a connectivity factor. Simulation results conveyed that this algorithm performs much better that the slot-based algorithms for wide area networks. However this algorithm does not guarantee zero rejection when the traffic demand is admissible. The average percentage of blocking for this algorithm is 0.9%, which generates some rejection when the load is high. To address this deficiency we proposed the Fair Matching Algorithm (FMA), a novel scheduling algorithm that achieves zero rejection for admissible demands and provides weighted max-min fair allocation of free capacity. We showed through network simulations that the resulting queueing delay when using this algorithm is almost constant over a wide range of offered load. The max-min fairness criterion does not in general achieve minimum rejection when the traffic is not admissible.

We demonstrated that when the demand matrix is inadmissible, the Fair Matching Algorithm minimizes the maximum percentage rejection experienced by any connection. We also proposed the Minimum Rejection Algorithm (MRA), a novel algorithm that generates a schedule that minimizes the total rejection. Simulations showed that the discrepancy in total rejection achieved by MRA and FMA is relatively minor, whereas there is a major difference in the fairness of the allocation of rejection. In addition, MRA appears to be less robust to demand prediction errors (when traffic arrivals differ substantially from the demand matrix used for scheduling). Thus it appears that whilst MRA achieves minimum rejection schedules, FMA is a better choice for scheduling in practice.

In wide area networks due to the large propagation delay the scheduling algorithms may result in unfairness and inefficiency. In Chapter5 we proposed a feedback control system that compensates for scheduling errors due to mispredictions and rejection in single-hop communication networks with large propagation delays. The controller design is based on the Smith principle, which removes the destabilizing delays from the feedback loop by using a "loop cancelation" technique. We have shown through simulations that our controller reduces the response time to a sudden change in a queue length. However, the use of Smith controller can negatively affect other parts of the system, such as queue length of the connections with small offered load. To overcome these sort of limitations we adopted adaptive controller gains and demonstrated using simulations that under different traffic scenarios the queue lengths are controlled and fairness is achieved by controlling the divergence from the average queue length.

6.2 Future Work

Here we present a number of potential topics for future research.

- The network architecture that we assumed in this research considered equal propagation delays for every connection. However, this is not a realistic assumption. The reason behind this assumption was that we were looking for general results regarding our proposed algorithms, which needed abstract or simplified network models. To extend this study, one should incorporate different propagation delays for edge nodes. This will determine the impact of significant variation of the distances between core and edge nodes on the performance of our bandwidth allocation strategies.

- In designing a fixed-length frame schedule we mainly focused on obtaining a fair share of capacity. Our definition of fairness was (weighted) max-min fairness, which is the most natural one. However, max-min fairness gives priority to small demands. On the other hand *proportional fairness* maximizes the sum of the logarithmic values of the rate allocations over all links. It has been proved that TCP with the additive increase/multiplicative decrease algorithms converges to proportional fairness, where shorter flows (i.e., with smaller RTT) have higher priority (or the fair share of bandwidth is proportional to the flow's response time). Implementing proportional fairness approximates TCP rate allocation in Internet, which has proved to achieve a good trade-off between efficiency and fairness.

- Utility max-min fairness can be used along with our scheduling framework to support different classes of applications. This fairness criterion maximizes the minimum performance experienced by any application.

- As shown in Section 4.2.1 decreasing the number of time slots per frame does not improve the queueing delay substantially at low offered loads, while it has a negative impact on the performance at high loads. However, we did not investigate the case where we are able to increase or decrease the frame length by changing the duration of a time slot.

- Even though we argued that using a variable-length frame scheduling is not a practical option for an AAPN, we did not investigate the effect of using a variable-length frame schedule on performance as the offered load varies. As shown in Figure 4.5 at low offered loads small frame lengths and at high offered loads large frame lengths are more desirable, however it is not clear how the performance of a variable-length frame schedule is affected by a large propagation delay and bursty traffic.

- The scheduling methods that we proposed are applicable to any single-hop communication system such as satellite systems, since there is no specific assumption in the scheduling design with respect to the type of network (wired or wireless). For multi-hop photonic networks the problem is more complicated; a global demand matrix clamping algorithm is needed to define the fair share of bandwidth for a request which passes several photonic switches. Aside from clamping, the scheduling algorithm will run for every photonic switch separately which is not different from a single-hop scheduling algorithm.

- The scheduling process in this research is based on predicting deterministic values for the number of requested time slots for each connection. However, the demands can be identified as random variables drawn from a probability distribution which shows the number of necessary time slots for a given connection with respect to its probability of occurrence. The notion of *probabilistic scheduling* has been explored in some articles [134, 135], but has not been investigated with respect to fairness.

- Our scheduling algorithm for an AAPN is performed for every wavelength independently. This necessitates employing a load balancing technique which distributes the load on several wavelengths. Perhaps the most straightforward solution is to divide the traffic equally amongst the wavelengths, but this is possible only for short flows.

When the traffic is bursty and contains delay sensitive media we prefer to assign a single wavelength to a burst, which needs a global load balancing strategy to avoid overloading of some wavelengths, whilst achieving almost equal performances per wavelength.

- As we showed in Section 4.2.1 increasing the number of edge nodes and the frame duration increase the queueing delay. In wide-area networks where we have around 1000 edge nodes, increasing the frame length to incorporate at least 1000 time slots is not a practical solution. One suggestion is to dynamically group the edge nodes such that every group operates on only a group of wavelengths. Since it is necessary to provide full connectivity between the edge nodes, both the group of edge nodes and the wavelengths should be overlapping. However this should be done such that we assign minimum number of edge nodes to every wavelength to achieve smaller frame duration. This problem is a combination of wavelength assignment described in Section 2 and scheduling.

- The controller design proposed in this research can be extended to developing simple procedures for tuning the control system parameters that are insensitive to changes in traffic. It is also worthwhile to explore the incorporation of methods from robust control to address the scenario where there are delay variations over time. As discussed in Chapter 5 we assume that all of the packets experience almost equal queueing delay (i.e., less than one frame duration), and hence the digital control system does not incorporate the delay variations.

Appendix A

A.1 Proof of Lemma 1

Proof of Lemma 1. It has been proved that a feasible rate vector v is max-min fair if and only if each connection has a bottleneck link with respect to v (see [91]- p. 527). Lemma 1 considers the extension to weighted max-min fairness, for the case that every connection u is associated with a weight $\omega_u(v_u)$. Similar to the proof of max-min fairness we proof this lemma by contradiction.

Suppose that v is weighted max-min fair with the weight vector ω. To arrive at a contradiction, assume that there exists a connection u with no bottleneck link. Then for each link ℓ crossed by u for which $C_\ell = F_\ell$, there must exist a connection $x \neq u$ such that $\omega_x > \omega_u$; thus the quantity

$$\delta_\ell = \begin{cases} C_\ell - F_\ell & if \quad F_\ell < C_\ell \\ (\omega_x - \omega_u) \times R_x & if \quad F_\ell = C_\ell \end{cases} \tag{A.1}$$

is positive. Therefore, by increasing v_u by the minimum δ_ℓ over all links ℓ crossed by u, while decreasing by the same amount the rates of the connections x of the links ℓ crossed by u with $F_\ell = C_\ell$, we maintain feasibility without decreasing the rate of any connection k with $\omega_k \leq \omega_u$; this contradicts the weighted max-min fairness property of (v, ω). Note that $v_x - \min(\delta_\ell)$ is always positive.

Conversely, assume that each connection has a bottleneck link with respect to the feasible set (v, ω). Then to increase the rate of any connection u while maintaining feasibility, we must decrease the rate of some connection k crossing bottleneck link ℓ of u (because we have $F_\ell = C_\ell$ by the definition of a bottleneck link). Since $\omega_k \leq \omega_u$ for all k crossing ℓ (by the definition of a bottleneck link), the feasible set (v, ω) satisfies the requirement for weighted max-min fairness. □

A.2 Proof of Theorem 2

Proof. Let $u \in \{(i,j), 1 \leq i,j \leq N\}$ index the source-destination connections specified by the demand matrix. We focus on the properties of the modified demand matrix and associated sets at various iterations of the while loop in Algorithm 1, so we index entities by iteration number and note that this indicates the value of the entity at the *start* of the iteration. For example, $\mathcal{A}_D(h)$ denotes the set of unmodified overloaded lines at the start of iteration h of the algorithm.

We prove that FMA achieves weighted max-min fair allocation of the demands. During each iteration h of the while-loop, FMA identifies the line $\gamma \in \mathcal{A}_D(h)$ such that $G_\gamma(h) = \min\{G_\ell(h); \ell \in \mathcal{A}_D(h)\}$. It alters the demands in $a_\gamma(h)$ according to (4.8) and after this modification, there is no subsequent modification of these demands. Substituting (4.8) into the definition of the weight, we have $\omega_u = 1 + G_\gamma(h)$ for all $u \in a_\gamma(h)$.

We demonstrate that the adjustment at iteration h leads to γ being a bottleneck link (line) for $u \in a_\gamma(h)$, i.e., after this adjustment it holds that $\omega_z \leq \omega_u$ for $u \in a_\gamma(h)$ and $z \in b_\gamma(h)$. Equivalently, we prove that $\min\{G\}$ is monotonically increasing with respect to the iteration number, i.e., $\min\{G(h)\} \leq \min\{G(h+1)\}$. The equivalence follows since the ω_z are obtained from adjustments prior to iteration h.

Suppose that line β has minimum G at iteration $h+1$. Lines γ and β have at most one connection (demand) in common. If there is no common connection, then $G_\beta(h+1) = G_\beta(h) \geq G_\gamma(h)$. If there is a common connection k, then:

$$\begin{aligned} LS_\beta(h+1) &= LS_\beta(h) + D_k(\omega_k - 1) && \text{(A.2)} \\ S_{a_\beta}(h+1) &= S_{a_\beta}(h) - D_k && \text{(A.3)} \end{aligned}$$

and hence

$$\begin{aligned} G_\beta(h+1) &= \frac{L - LS_\beta(h) - D_k(\omega_k - 1)}{S_{a_\beta}(h) - D_k} \\ &= \frac{S_{a_\beta}(h)G_\beta(h) - D_k(\omega_k - 1)}{S_{a_\beta}(h) - D_k} \\ &\geq G_\gamma(h) && \text{(A.4)} \end{aligned}$$

where the last inequality follows from substitution based on $G_\beta(h) \geq G_\gamma(h) = \omega_k - 1$.

Thus the application of FMA upon an inadmissible demand matrix D leads to the generation of a bottleneck link for each connection u with weight $\omega_u = \frac{D'_u}{D_u}$. By Lemma 1, this establishes that FMA achieves weighted max-min fair allocation of adjusted demands D'.

□

A.3 Proof of Theorem 3

Proof. The proof is similar to the proof of Theorem 2. In this case, we have $\omega_u = D_u - D'_u$ and $\omega_u = H_\gamma(h)$ for all $u \in a_\gamma(h)$, where γ is the line with minimum H at iteration h. If β is the line with minimum H at iteration $h+1$, then if γ and β share an element k, $LS_\beta(h+1) = LS_\beta(h) + \omega_k$ and $a_\beta(h+1) = a_\beta(h) - 1$. Hence:

$$\begin{aligned} H_\beta(h+1) &= \frac{L - LS_\beta(h) - \omega_k}{|a_\beta(h)| - 1} & \text{(A.5)} \\ &= \frac{|a_\beta(h)|H_\beta(h) - \omega_k}{|a_\beta(h)| - 1} & \text{(A.6)} \\ &\geq H_\gamma(h) & \text{(A.7)} \end{aligned}$$

where the last inequality follows from substitution $H_\beta(h) \geq H_\gamma(h) = \omega_k$. Hence, ESA leads to a bottleneck link with weight $\omega_u = D_u - D'_u$ and hence, by Lemma 1, achieves max-min fair allocation of capacity. □

A.4 Proof of Theorem 4

Proof. Consider an arbitrary rejection matrix D^w and set $B = MAXFLOW(D, D^w, L)$. Then we can write $D^w = B + Q$ where Q is a non-negative matrix. Now consider the conditions necessary for D^w to achieve minimum rejection. First, $D^w_{hp} = 0$ if $h \notin O_r$ and $p \notin O_c$ (any non-zero values constitute unnecessary rejection).

Now consider a node pair $h \in O_r$, and $p \in O_c$ in Figure 4.14, and the edges (S, h), (h, p) and (p, O). Since B achieves maximum flow, then the flow of at least one of these edges is at full capacity. Therefore, at least one of the following holds:

1. $B_{hp} = D^w_{hp}$
2. $\sum_{j \in O_c} B_{hj} = r_h(D) - L$

3. $\sum_{i \in O_r} B_{ip} = r_p(D) - L$.

If the first equation is true, then $Q_{hp} = 0$. The second equation implies that B has provided the necessary rejection at row h, but $\sum_{j \in O_c} Q_{hj} = 0$ does not necessarily hold; the other overflowing columns may enforce additional rejections on D^w_{hp} which causes $Q_{hj} > 0$ for some $j \in O_c$. We have a similar property for the third equation. Therefore Q is composed of two distinct types of lines which cover all of its nonzero elements:

Type I: The lines composed of $Q_{hp} \geq 0$, and $Q_{hj} \geq 0$ or $Q_{ip} \geq 0$, for $h \in O_r, p \in O_c, i \notin O_r$, and $j \notin O_c$; these lines correspond to the lines in D^w with $r_h(D^w) = r_h(D) - L$, or $c_p(D^w) = c_p(D) - L$, which impose additional rejections to (h,p) elements after obtaining B =*MAXFLOW*(D, D^w, L). Consequently we have $r_h(Q) = r_h(D) - L - r_h(B)$, or $c_p(Q) = c_p(D) - L - c_p(B)$.

Type II: The lines composed of $Q_{hp} = 0$, and $Q_{hj} \geq 0$ or $Q_{ip} \geq 0$, for $h \in O_r, p \in O_c, i \notin O_r, j \notin O_c$; for these lines $B_{hp} = D^w_{hp}$ $\forall h \in O_r, p \in O_c$ holds. Therefore additional rejection on these lines is calculated from: $r_h(Q) = r_h(D) - L - r_h(B)$, or $c_p(Q) = c_p(D) - L - c_p(B)$.

Based on this discussion, we can express the total number of rejections, $|D^w|$ as:

$$\begin{aligned} |D^w| &= \sum_h \sum_p (B + Q) \\ &= |B| + \sum_{h \in O_r} (r_h(D) - L - r_h(B)) \\ &\quad + \sum_{p \in O_c} (c_p(D) - L - c_p(B)) \\ &= \sum_{h \in O_r} (r_h(D) - L) + \sum_{p \in O_c} (c_p(D) - L) - |B| \end{aligned} \tag{A.8}$$

Therefore, in order for D^w to achieve minimum rejection, $|B|$ must be maximized (the first two terms are functions solely of D and L). Compare the solutions B = *MAXFLOW*(D, D^w, L) and A = *MAXFLOW*(D, D, L). Since $D^w_{hp} \leq D_{hp}$ for any (h, p), the constraints in the second problem are looser, which implies that $|A| \geq |B|$, irrespective of the particular values in D^w. Note that A is also a solution to *MAXFLOW*(D, A, L).

Hence if we ensure that $D^w_{hp} \geq A_{hp}$ for all (h, p), we derive $|B| = |A|$, which implies that $|B|$ attains its maximum value (and hence $|D^w|$ is the minimum rejection). We can thus construct a rejection matrix that achieves minimum rejection by solving A = $\mathit{MAXFLOW}(D, D, L)$, and setting $D'' = A + Q$, where Q satisfies the constraints specified in the theorem. If a schedule S decomposes the demand into an allocated matrix D' and this rejection matrix D'', then it achieves minimum rejection. □

Appendix B

Ford-Fulkerson Algorithm

1. Construct a graph G(V,E,U), where V is the set of vertices, E the set of edges and U the set of capacities of the edges.
2. Set all flows, f_{ij} of the edges to zero.
3. Find an augmenting path P, with flow $\delta = \min\{\delta_{ij}, \forall i, j \in P\} > 0$, where $\delta_{ij} = u_{ij} - f_{ij}$ for forward edges, and $\delta_{ij} = f_{ij}$ for backward edges. If there is not an augmenting path go to step 5.
4. Augment the path by setting $f_{ij} = f_{ij} + \delta$ for forward edges, and $f_{ij} = f_{ij} - \delta$ for backward edges. Go to step 3.
5. End.

References

[1] G. Bochmann, M. Coates, T. Hall, L. Mason, R. Vickers, and O. Yang, "The agile all-photonic network: An architectural outline," in *Proc. Queens' Biennial Symp. Comm.*, (Kingston, Canada), pp. 217–218, June 2004.

[2] L. Mason, A. Vinokurov, N. Zhao, and D. Plant, "Topological design and dimensioning of agile all photonic networks," *Computer Networks*, vol. 50, pp. 268–287, Feb. 2006.

[3] H. Zang, J. Jue, and B. Mukherjee, "Photonic slot routing in all-optical WDM mesh networks," in *Proc. IEEE Globecom*, (Rio de Janeiro, Brazil), Dec. 1999.

[4] C. Guillmot, M. Renaud, and P. Gambini, "Transparent optical packet switching: The European ACTS KEOPS project approach," *Lightwave Tech.*, vol. 16, pp. 729–375, May 1998.

[5] W. D. Zhong and R. S. Tucker, "Wavelength routing based photonic packet buffers and their applications in photonic packet switching systems," *Lightwave Technology*, vol. 16, pp. 1737–1745, Oct. 1998.

[6] M. Yoo, C. Qiao, and S. Dixit, "Optical burst switching for service differentiation in the next-generation optical Internet," *IEEE Comm. Mag.*, pp. 98–104, Feb. 2001.

[7] J. Ramamirtham and J. Turner, "Time-sliced optical burst switching," in *Proc. IEEE INFOCOM*, (San Francisco, CA), Mar. 2003.

[8] K. J. Aström, C. C. Hang, and B. C. Lim, "A new Smith predictor for controlling a process with an integrator and long dead-time," *IEEE Trans. Automatic Cont.*, vol. 39, pp. 343–345, Feb. 1994.

[9] M. Mataušek and A. Micić, "A modified Smith predictor for controlling a process with an integrator and long dead-time," *IEEE Trans. Automatic Cont.*, vol. 41, pp. 1199–1203, Aug. 1996.

[10] R. Izmailov, S. Ganguly, T. Wang, Y. Suemura, Y. Maeno, and S. Araki, "Hybrid hierarchical optical networks," *IEEE Comm. Mag.*, vol. 40, pp. 88–95, Nov. 2002.

[11] L. Xu, H. Perros, and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Comm. Mag.*, vol. 39, pp. 136–142, Jan. 2001.

[12] R. Ramaswami and K. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Trans. Networking*, vol. 3, pp. 489–500, Oct. 1995.

[13] T. J. Hall, S. A. Paredes, and G. v. Bochmann, "An agile all-photonic network," in *Proc. Int. Conf. on Optical Comm. and Networks*, (Bangkok, Thailand), pp. 365–368, Dec. 2005.

[14] I. Keslassy, M. Kodialam, T. Lakshman, and D. Stiliadis, "Scheduling schemes for delay graphs with applications to optical packet networks," in *Proc. IEEE Work. High Perf. Switch. and Routing*, (Phoenix, AZ), Apr. 2003.

[15] X. Liu, N. Saberi, M. Coates, and L. Mason, "A comparison between time-slot scheduling approaches for all-photonic networks," in *Proc. Int. Conf. Inf., Comm. and Signal Processing (ICICS)*, (Bangkok, Thailand), Dec. 2005.

[16] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space division switch," *IEEE Trans. Comm.*, vol. 35, pp. 1347–1356, Dec. 1987.

[17] T. Anderson, S. Owicki, J. Saxe, and C. Thacker, "High-speed switch scheduling for local-area networks," *ACE Trans. Comp. Sys.*, vol. 11, pp. 319–352, Nov. 1993.

[18] N. McKeown, *Scheduling Algorithms for Input- Queued Cell Switches.* PhD thesis, University of California at Berkeley, 1995.

[19] N. McKeown, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," in *Proc. IEEE INFOCOM*, (San Francisco, CA), pp. 296–302, Mar. 1996.

[20] A. Mekkittikul and N. McKeown, "A starvation-free slgorithm for schieving 100% throughput in an input-queued switch," in *Proc. IEEE ICCCN*, (Rockville, MD), pp. 226–231, Oct. 1996.

[21] Y. Tamir and G. Frazier, "High performance multiqueue buffers for VLSI communication switches," in *Proc. of 15th Ann. Symp. on Comp. Arch.*, (Honolulu, HI), pp. 343–354, June 1988.

[22] N. McKeown and A. Mekkittikul, "A practical scheduling algorithm to achieve 100% throughput in input-queued switches," in *Proc. IEEE INFOCOM*, (San Francisco, CA), pp. 792–799, Mar. 1998.

[23] A. Mekkittikul and N. McKeown, "Scheduling VOQ switches under non-uniform traffic," CSL Technical report CSL-TR-97-747, Stanford University, Stanford, CA, Sept. 1997.

[24] A. Bianco, E. Leonardo, F. Neri, J. Solé-Pareta, and S. Spadaro, "A framework for differential frame-based matching algorithms in input-queued switches," in *Proc. INFOCOM*, (Hong Kong, China), pp. 1147–1157, Mar. 2004.

[25] S.Mascolo, "Congestion control in high-speed communication networks using the Smith principle," *Automatica*, vol. 35, pp. 1921–1935, Dec. 1999.

[26] A. Ganz and Y. Gao, "Efficient algorithms for SS/TDMA scheduling," *IEEE Trans. Comm.*, vol. 40, pp. 1367–1374, Aug. 1992.

[27] P. Crescenzi, X. Deng, and C. H. Papadimitriou, "On approximating a scheduling problem," *J. Combinatorial Optimization*, vol. 5, pp. 287–297, Sept. 2001.

[28] B. Towles and W. J. Dally, "Guaranteed scheduling for switches with configuration overhead," *IEEE/ACM Trans. Networking*, vol. 11, pp. 835–847, October 2003.

[29] K. Bogineni, K. M. Sivalingham, and P. W. Dowd, "Low-complexity multiple access protocols for wavelength-division multiplexed photonic networks," *IEEE J. Sel. Areas Comm.*, vol. 11, pp. 590–604, May 1993.

[30] G. N. Rouskas and M. H. Ammar, "Analysis and optimization of transmission schedules for single-hop WDM networks," in *Proc. IEEE INFOCOM*, (San Francisco, CA), pp. 1342–1349, May 1993.

[31] M. Marsan, A. Bianco, E. Leonardi, F. Neri, and A. Nucci, "Simple on-line scheduling algorithms for all-optical broadcast-and select networks," *IEEE European Trans. Telecom.*, vol. 11, pp. 109–116, Jan. 2000.

[32] A. Bianco, D. Careglio, J. Finochietto, G. Galante, E. Leonardo, F. Neri, J. Solé-Pareta, and S. Spadaro, "Multiclass scheduling algorithms for the DAVID metro network," *IEEE J. Sel. Areas Comm.*, vol. 22, pp. 1483–1496, Oct. 2004.

[33] N. Saberi and M. Coates, "Minimum rejection scheduling in all-photonic networks," in *Proc. IEEE BROADNETS*, (San Jose, CA), Oct. 2006.

[34] R. Ramaswami and K. Sivarajan, *Optical Networks: A practical Perspective.* Morgan Kaufmann, 1998.

[35] G. Xiao and Y. W. Leung, *Advances in Optical Networks*, ch. Allocation of Wavelength Converters in All-Optical Networks, pp. 299–345. Kluwer Academic Publishers, 2001.

[36] H. Zang, J. P. Juet, and B. Mukherjee, "A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks," *Optical Network Mag.*, vol. 1, Jan. 2000.

[37] G. Wedzinga, *Photonic Slot Routing in Optical Transport Networks.* Kluwer Academic Publishers, Nov. 2002.

[38] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Co., June 1979.

[39] I. Chlamtac, A. Ganz, and G. Karmi, "Light path communications: An approach to high-bandwidth optical wans," *IEEE Trans. Comm.*, vol. 40, pp. 1171–1182, July 1992.

[40] D. Mitra and J. B. Seery, "Comparative evaluations of randomized and dynamic routing strategies for Circuit-Switched networks," *IEEE Trans. on Comm.*, vol. 39, no. 1, pp. 102–116, 1991.

[41] S. Ramamurthy, *Optical Design of WDM Network Architectures.* PhD thesis, University of California, Davis, 1998.

[42] S. Ramamurthy and B. Mukherjee, "Fixed-alternate routing and wavelength conversion in wavelength-routed optical networks," in *Proc. IEEE GLOBECOM*, (Sydney, Australia), Nov. 1998.

[43] K. Chan and T. P. Yum, "Analysis of least congested path routing in WDM lightwave networks," in *Proc. IEEE INFOCOM*, (Toronto, Canada), Jun. 1994.

[44] L. Li and A. Somani, "Dynamic wavelength routing using congestion and neighborhood information," *IEEE/ACM Trans. Networking*, vol. 7, Oct. 1999.

[45] B. Mukherjee, *Optical Communication Networks.* New York: McGraw-Hill, 1997.

[46] M. T. Jones and P. E. Plassmann, "A parallel graph coloring heuristic," *SIAM J. Scientific Computing*, vol. 14, pp. 654–669, May 1993.

[47] S. Subramaniam and R. A. Barry, "Wavelength assignment in fixed routing WDM networks," in *Proc. IEEE ICC*, (Montreal, Canada), pp. 406–410, June 1997.

[48] E. Karasan and E. Ayanoglu, "Effect of wavelength routing and selection algorithms on wavelength conversion gain in WDM optical networks," *IEEE/ACM Trans. Networking*, vol. 6, pp. 186–196, April 1998.

[49] R. A. Barry and S. Subramaniam, "The max-sum wavelength assignment algorithm for WDM ring networks," in *Proc. OFC*, (Dallas, TX), pp. 121–122, Feb. 1997.

[50] I. Chlamtac, A. Farago, and T. Zhang, "Lightpath (wavelength) routing in large WDM networks," *IEEE J. on Selected Areas in Comm.*, vol. 14, pp. 909–913, June 1996.

[51] D. J. Blumenthal, "All-optical label swapping for the future Internet," *Optics and Photonics News*, vol. 13, pp. 26–29, March 2002.

[52] Y. Shun, B. Mukherjee, and S. Dixit, "Advances in photonic packet switching: an overview," *IEEE Comm. Magazine*, vol. 38, pp. 84–94, Feb. 2000.

[53] H. Harai, N. Wada, F. Kubota, and W. Chujo, "Contention resolution using multistage fiber delay line buffer in a photonic packet switch," in *Proc. IEEE Int. Conf. Comm./ICC*, (New York, USA), pp. 2843–2847, April 2002.

[54] A. Acampora and I. Shah, "Multihop lightwave networks: A comparison of store-and-forward and hot-potato routing,"

[55] G. Bendelli, M. Burzio, P. Gambini, and M. Puleo, "Performance assessment of a photonic ATM switch based on a wavelength controlled fiber loop buffer," *Optical Fiber Comm.*, pp. 106–107, 1996.

[56] J. M. Gabriagues and J. B. Jacob, "Photonic ATM switching matrix based on wavelength routing," in *Proc. SPIE Photonic Switching*, (Minsk, Belarus), pp. 355–359, July 1992.

[57] C. Qiao and M. Yoo, "Just-Enough-Time(JET): A high speed protocol for bursty traffic in optical networks," in *Proc. IEEE/LEOS Conf. Technologies for Global Information Infrastructure*, (Montreal, CA), pp. 26–27, Aug. 1997.

[58] M. Yoo, C. Qiao, and S. Dixit, "QoS performance of optical burst switching in IP-over-WDM networks," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 2062–2070, Oct. 2000.

[59] K. Sivalingam, J. Wang, X. Wu, and M. Mishra, "An interval-based scheduling algorithm for optical WDM star networks," *J. Photonic Network Comm.*, Jan. 2002.

[60] A. Bianco, G. Galante, E. Leonardo, F. Neri, and A. Nucci, "Scheduling algorithms for multicast traffic in TDM/WDM networks with arbitrary tuning latencies," *Computer Networks*, vol. 41, pp. 727–742, Apr. 2003.

[61] E. Johnson, M. Mishra, and K. Sivalingam, "Scheduling in optical WDM networks using hidden Markov chain based traffic prediction," *J. Photonic Network Comm.*, vol. 3, pp. 271–286, July 2001.

[62] W. Schmidt, "An on-board switched multiple-access system for millimetre-wave satellites," in *Proc. Digital Satellite Comm. Conf.*, (London, UK), 1969.

[63] M. S. Borella and B. Mukherjee, "Efficient scheduling of nonuniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies," *IEEE J. Sel. Areas. Comm.*, vol. 14, pp. 923–934, Sept. 1996.

[64] M. Azizoglu, R. A. Barry, and A. Mokhtar, "Impact of tuning delay on the performance of bandwidth limited optical broadcast networks with uniform traffic," *IEEE J. Sel. Areas. Comm.*, vol. 14, pp. 935–944, Jun. 1996.

[65] M. Ajmone Marsan, A. Bianco, E. Leonardi, F. Neri, and A. Nucci, "Efficient multi-hop scheduling algorithms for all optical WDM broadcast-and-select networks with arbitrary transceiver tuning latencies," in *Proc. IEEE Globecom*, (Sydney, Australia), 1998.

[66] R. L. Graham, E. L. Lawler, J. K. Lenstra, and K. R. Kan, "Optimization and approximation in deterministic scheduling: A survey," *Ann. Disc. Math.*, vol. 5, pp. 287–326, 1979.

[67] I. S. Gopal and C. K. Wong, "Minimizing the number of switchings in an SS/TDMA system," *IEEE Trans. Comm.*, vol. 33, pp. 1497–1501, June 1985.

[68] T. Gonzalez and S. Sahni, "Open shop scheduling to minimize finish time," *J. ACM*, vol. 23, pp. 665–679, Oct. 1976.

[69] D. McLaughlin, S. Sardesai, and P. Dasgupta, "Preemptive scheduling for distributed systems," in *Proc.11th Int. Conf. Parallel and Distributed Computing Systems*, (Chicago, Illinois USA), Sept. 1998.

[70] K. Jansen and M. I. Sviridenko, "Polynomial time approximation schemes for the multiprocessor open and flow shop scheduling problem," *Lecture Notes in Computer Science*, vol. 1770, pp. 455–465, 2000.

[71] K. Jansen, R. Solis-Oba, and M. Sviridenko, "A linear time approximation scheme for the job shop scheduling problem," in *Proc. Second Int. Workshop Approximation Algorithms for Combinatorial Optimization Problems*, (Berkeley, Ca), pp. 177–188, Aug. 1999.

[72] C. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity.* Prentice-Hall, 1982.

[73] A. Ganz and Y. Gao, "A time-wavelength assignment algorithm for a WDM star network," in *Proc. IEEE INFOCOM*, (Florence, Italy), May 1992.

[74] C. A. Pomalaza-Raez, "A note on efficient SS/TDMA assignment algorithms," *IEEE Trans. Comm.*, vol. 36, pp. 1078–1082, 1988.

[75] G. Bongiovanni, D. Coppersmith, and C. Wong, "An optimal time slot assignment algorithm for an SS/TDMA system with variable number of transponders," *IEEE Trans. Comm.*, vol. 29, pp. 721–726, Oct. 1981.

[76] I. Gopal, G. Bongiovanni, M. A. Bonuccelli, D. T. Tang, and C. K. Wang, "An optimal switching algorithm for multibeam satellite systems with variable bandwidth beams," *IEEE Trans. Comm.*, vol. 30, pp. 2475–2481, Nov. 1982.

[77] T. Inukai, "An efficient SS/TDMA time slot assignment algorithm," *IEEE Trans. Comm.*, vol. 27, pp. 1449–1455, May 1979.

[78] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Applied Mathematics*, vol. 17, pp. 416–429, March 1969.

[79] S. Micali and V. V. Vazirani, "An $O(\sqrt{\|V\|}\|E\|)$ algorithm for finding maximum matching in general graphs," in *Proc. IEEE Symp. Found. Comp. Sci.*, (Syracuse, NY), pp. 17–27, 1980.

[80] R. Karp, "Reducibility among combinatorial problems," in *Proc. Complexity of computer computations* (R. Miller and J. Thatcher, eds.), (New York, NY), pp. 85–103, Plenum Press, 1972.

[81] E. D. Hochbaum, *Approximation Algorithms for NP-hard Problems.* Boston, MA: PWS Publishing Company, 1996.

[82] X. Liu, A. Vinokurov, and L. Mason, "Performance comparison of OTDM and OBS scheduling for agile all-photonic network," in *Proc. IFIP Metropolitan Area Network Conference*, (Ho Chi Minh City, Vietnam), Apr. 2005.

[83] G. N. Rouskas and V. Sivaraman, "Packet scheduling in broadcast WDM networks with arbitrary transceiver tuning latencies," *IEEE/ACM Trans. Networking*, vol. 5, pp. 359–370, June 1997.

[84] K. Sivalingam, J. Wang, X. Wu, and M. Mishra, "Improved on-line scheduling algorithm for optical WDM nertworks," in *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, (New Brunswick, NJ), pp. 43–61, 1998.

[85] H. Choi, H.-A. Choi, and M. Azizoglu, "Efficient scheduling of transmissions in optical broadcast networks," *IEEE/ACM Trans. Networking*, vol. 4, pp. 913–920, Dec. 1996.

[86] G. R. Pieris and G. H. Sasaki, "Scheduling transmissions in WDM broadcast-and-select networks," *IEEE/ACM Trans. Networking*, vol. 2, pp. 105–110, Apr 1994.

[87] N. M. Bhide, M. Mishra, and K. M. Sivalingam, "Scheduling algorithms for star-coupled WDM networks with tunable transmitter and tunable receiver architecture," *Photonic Network Comm.*, pp. 219–234, Nov. 1999.

[88] E. Coffman, M. Garey, and D. Johnson, "An application for bin-packing to multiprocessor scheduling," *SIAM J. of Computing*, vol. 7, pp. 1–17, Feb. 1978.

[89] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design.* Addison-Wesley, 1993.

[90] "OPNET modeler 10.5." http://www.opnet.com.

[91] D. Bertsekas and R. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice Hall, 1992.

[92] P. Marbach, "Priority service and max-min fairness," *IEEE/ACM Trans. Networking*, vol. 11, pp. 733–746, Oct. 2003.

[93] T. Ahmed and M. Coates, "Predicting flow vectors," tech. rep., McGill University, Montreal, Canada, Sept. 2005.

[94] "Passive measurement and analysis (PMA) project." http://pma.nlanr.net.

[95] D. Bertsekas, *Linear Network Optimization, Algorithms and Codes.* Cambridge, MA: MIT Press, 1991.

[96] L. R. Ford, Jr., and D. R. Fulkerson, "Maximal flow through a network," *Canadian. J. Math.*, vol. 8, pp. 399–404, 1956.

[97] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. Assoc. Comput. Mach.*, vol. 19, pp. 248–264, 1972.

[98] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms.* Cambridge, MA: MIT Press, 2nd ed., 2001.

[99] N. Saberi and M. Coates, "Bandwidth reservation in optical WDM/TDM star networks," in *Proc. Queens' Biennial Symp. Comm.*, (Kingston, Canada), pp. 219–221, June 2004.

[100] N. Saberi and M. Coates, "Fair matching algorithm: Fixed-length frame scheduling in all-photonic networks," in *Proc. IASTED Int. Conf. Optical Comm. Sys. and Networks*, (Alberta, Canada), pp. 213–218, July 2006.

[101] A. Leon-Garcia and I. Widjaja, *Communication Networks: Fundamental Concepts and Key Architectures.* Boston: McGraw-Hill, 2nd ed., 2004.

[102] G. Rouskas and H. Perros, *A tutorial on optical networks*, vol. 2497 of *Lecture Notes in Computer Science*, pp. 155–193. New York, NY: Springer-Verlag, 2002.

[103] S. Ryu, C. Rump, and C. Qiao, "Advances in Internet congestion control," *IEEE Communication Survey and Tutorial*, vol. 5, no. 1, pp. 28–39, 2003.

[104] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Trans. Networking*, vol. 7, no. 4, pp. 458–472, 1999.

[105] Y. Zhao, S. Q. Li, and S. Sigarto, "A linear dynamic model for design of stable explicit-rate ABR control schemes," in *Proc. IEEE INFOCOM*, (Kobe, Japan), pp. 283–292, April 1997.

[106] E. Altmann, T. Bassar, and R. Srikant, "Robust rate control for ABR sources," in *Proc. IEEE INFOCOM*, (San-Francisco, CA), pp. 166–173, Mar. 1998.

[107] C. Hollot, V. Misra, D. Towsley, and W. Gong, "A control theoretic analysis of RED," in *Proc. IEEE INFOCOM*, (Anchorage, AK), pp. 1510–1519, April 2001.

[108] S. H. Low, F. Paganini, and J. C. Doyle, "Internet congestion control," *IEEE Cont. Syst. Mag.*, vol. 22, pp. 28–43, Feb. 2002.

[109] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for future high bandwidth-delay product environments," in *Proc. ACM SIGCOMM*, (Pittsburgh, PA), Aug. 2002.

[110] F. Paganini, J. C. Doyle, and S. H. Low, "Scalable laws for stable network congestion control," in *Proc. IEEE Decision and Cont.*, (Orlando, FL), Dec. 2001.

[111] S. H. Low, F. Paganini, J. Wang, and J. C. Doyle, "Linear stability of TCP/RED and a scalable control," *J. of Comput. Networks*, vol. 42, pp. 633–647, Dec. 2003.

[112] N. Bonmariage and G. Leduc, "A survey of optimal network congestion control for unicast and multicast transmission," *Computer Networks*, vol. 50, pp. 448–468, Feb. 2006.

[113] D. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Computer Networks and ISDN Systems*, vol. 17(1), pp. 1–14, 1989.

[114] A. Arulambalam and X. Chen, "Allocating fair rates for available bit rate service in ATM networks," *IEEE Comm. Mag.*, vol. 34, pp. 92–100, Nov. 1996.

[115] V. Jacobson, "Congestion avoidance and control," in *Proc. of SIGCOMM '88*, (CA, USA), Aug. 1988.

[116] P. H. Bauer, M. L. Sichitiu, R. Ernst, and K. Premaratne, "A new class of Smith predictors for network congestion control," in *Proc. Int. IEEE Conf. on Electronics, Circuits, and Systems (ICECS)*, (St. Julian's, Malta), pp. 685–688, Sept. 2001.

[117] A. Sang and S. Q. Li, "A predictability analysis of network traffic," in *Proc. IEEE INFOCOM*, (Tel Aviv, Israel), pp. 342–351, Mar. 2000.

[118] J. E. Marshall, *Control of Time-Delay Systems.* New York: Stevenage, 1979.

[119] M. Malek-Zavarei and M. Jamshidi, *Time Delay Systems: Analysis, Optimization, and Applications.* Amsterdam: North-Holland, 1987.

[120] O. Smith, "A controller to overcome dead-time," *J. ISA*, vol. 6, pp. 28–33, Feb. 1959.

[121] K. Watanabe and M. Ito, "A process-model control for linear systems with delay," *IEEE Trans. Automatic Cont.*, vol. AC-26, pp. 1261–1269, Dec. 1981.

[122] H. Huang, C. Chen, Y. Chao, and P. Chen, "A modified Smith predictor with an approximated inverse of dead time," *J. AICHE*, vol. 36, no. 7, pp. 1025–1031, 1990.

[123] Z. Palmor, *The control handbook, Chapter 10.* CRC Press and IEEE Press, USA, 1996.

[124] J. Normey-Rico, , and E. Camacho, "Robust tuning of dead-time compensators for processes with an integrator and long dead-time," *IEEE Trans. on Automatic Cont.*, vol. 44, no. 8, pp. 1598–1603, 1999.

[125] W. D. Zhang and Y. X. Sun, "Modified Smith predictor for controlling integrator/time delay processes," *Industrial Eng. Chemistry Research*, vol. 35, pp. 2769–2772, Aug. 1996.

[126] J. Normey-Rico and E. Camacho, "A unified approach to design dead-time compensators for stable and integrative processes with dead-time," *IEEE Trans. on Automatic Cont.*, vol. 47, pp. 299–305, Feb. 2002.

[127] T. Hägglund, "A predictive PI controller for processes with long dead-times," *IEEE Cont. Syst. Mag.*, vol. 12, p. 5740, Feb. 1992.

[128] T. Liu, Y. Cai, D. Gu, and W. Zhang, "New modified Smith predictor scheme for integrating and unstable processes with time delay," in *Proc. IEE Cont. Theory and Appl.*, pp. 238–246, Mar. 2005.

[129] T. Hägglund, "An industrial dead-time compensating PI controller," *Cont. Eng. Practice*, vol. 4, pp. 749–756, June 1996.

[130] I. Landau, *Adaptive Control: The Model Reference Approach.* New York: Marcel Dekker, Inc., 1979.

[131] M. Huzmezan, G. Dumont, W. Gough, and S. Kovac, "Adaptive control of integrating time delay systems: A PVC batch reactor," *IEEE Trans. on Cont. Sys. Tech.*, vol. 11, no. 3, pp. 390–398, 2003.

[132] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, *Control System Design.* Prentice Hall, Sept. 2000.

[133] C. Peng, S. A. Paredes, T. J. Hall, and G. v. Bochmann, "Constructing service matrices for agile all-optical cores," in *Proc. IEEE Symp. Comp. and Comm.*, (Pula-Cagliari, Italy), June 2006.

[134] A. Burns, S. Punnekkat, L. Stringini, and D. Wright, "Probabilistic scheduling guarantees for fault-tolerant real-time systems," in *Proc. of the 7th Int. Working Conf. on Dependable Computing for Critical Appl.*, pp. 361 – 378, Nov. 1999.

[135] T. Som and R. Sargent, "Providing bandwidth guarantees in an input-buffered crossbar switch," in *Proc. of the 12th Workshop on Parallel and Distributed Simulation*, pp. 56–63, May 1998.